Project Number: LD16

Arthritis Care
Internal Communications System

An Interactive Qualifying Project Report

submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfilment of the requirements for the

Degree of Bachelor of Science

by

_____
Helene Gwizdak

_____
Peter Vitello

_____
Nicholas Williams

_____
Stephen Worsham

Date: April 27, 2001
Approved:

_____
Professor James Demetry, Ph.D

_____
Professor Ruth Smith, Ph.D

Abstract

The purpose of this project was to design and implement an internal communications system, known as an intranet, and produce a training manual to accompany the intranet for Arthritis Care, one of the United Kingdom's largest non-profit organisations. The dispersed structure of Arthritis Care poses communication challenges for the organisation, creating a need for an effective internal communications system. We used a workshop, a survey and focus groups to gather information about the content and structure of the intranet and training manual. The intranet was designed for easy maintenance and expandability to accommidate for any future needs of theorganisation

Acknowledgements

There are many individuals that we would like to thank for their contributions to our project. We would like to extend our gratitude to the employees and volunteers in Arthritis Care, for taking their time to answer our many questions. We would especially like to thank our liaison David Wright for his continued suggestions and support. We would also like to express our deepest gratitude to our advisors, Professors James Demetry and Ruth Smith, for their efforts in correcting, rewording, and suggesting improvements in our report.

Executive Summary

Arthritis Care is one of the United Kingdom's largest voluntary organisations working with and for people with arthritis from all areas of society. Their mission statement contains goals to promote health, well-being, and independence through the various services of support, self-help, and informational packets they provide. There are seven Regional/National offices, with 620 branches and more than 7,000 volunteers donating their time. Due to the dispersed structure, the internal communications within the organisation has been inhibited. Therefore, the goal of our project was to design and implement an internal communications system that would help in reducing some of the communication problems as well as helping in the reduction of expenditures and wasted resources. Through our Interactive Qualifying Project (IQP), we hope that the Arthritis Care's sense of community will be enhanced.

Arthritis Care is a large organisation, and in order for the organisation to be successful, it must have an efficient and timely communication system. Currently, many of Arthritis Care's employees are not able to access the organisation's sources of operational and managerial information as efficiently as needed. Therefore, it was recognised that there existed a need for a more accessible and efficient internal communications system, known as an intranet. The design was based on information offered by the staff through a workshop, a survey, and focus groups about the content and structure of the intranet. There was active participation by all staff and volunteers at Arthritis Care. We relied on the survey to obtain data that pertained to general information that would aid us in the construction of the training manual that accompanies the intranet.

Focus groups were only held at the central office. They allowed us to gather a large number of opinions and perceptions of the intranet. The focus groups consisted of a number of staff representing all of the departments in the central office. During the first set of focus groups the prototype was shown and explained, and the staff was then encouraged to stay for a trial of the internal communications system. This allowed the staff to give us feedback on what had been constructed thus far. The second set of focus groups was with a more in-depth and complete prototype that had been constructed up to that point. This consisted of a fully functional directory and document repository. From the last set of focus groups we received feedback on the training manual and quick reference sheet that was created to aid the staff in their use of the intranet. At this last focus group session, the staff commented on what they felt needed greater explanation within the training manual, as well as any corrections that had to be made.

When designing the intranet it was important to first clarify the exact forms of content that it would contain. Then in designing its structure, the various forms of content that were to be included were thoroughly evaluated but did not dictate the final structure of the intranet. The intranet was designed for easy maintenance and expandability for the future needs that the organisation may encounter. The IQP is a project that provides students with the opportunity to see how technology and society are linked. We hope that this project, which included designing and implementing an intranet, will be used as a mechanism to help establish a sense of community for the entire Arthritis Care organisation.

Table of Contents

List of Figures

List of Tables

Authorship Page

Helene Gwizdak – HG
Peter Vitello – PV
Nicholas Williams – NW
Stephen Worsham – SW
David Wright - DW

Editors Page

1.0 Introduction

The success of any organisation relies heavily on efficient and timely communications. Arthritis Care, the sponsor of this project, is one of the largest non-profit organisations in the United Kingdom. The organisation has for some time recognised the need for a more accessible and efficient internal communication system so that its staff and volunteers could access the organisation's sources of operational and managerial information.

Arthritis Care was looking for improvements in their current internal communications system. The current organisational structure of Arthritis Care consists of a central office with many satellite branch offices. Because of the distributed nature of this organisation, internal communications between managers, staff, and volunteers has been inhibited. A better communications system would provide easy and efficient access to information for all staff and volunteers, and reduce wasted time and resources. The goal of our project was to implement an internal communications system, which was accomplished by the creation of an intranet.

By implementing an appropriate information architecture, an intranet was constructed to address the communication problems. Concurrent with the intranet implementation, a training and maintenance guide for the current and future staff was created.

The implementation of this project relied heavily on the intranet having a proper design. To create this design we researched the known methods of a good information architecture and investigated successful implementations of intranets. In the design we also allowed for easy maintenance and expandability. Once the design was completed,

we began constructing a prototype, and at the same time we developed a training guide to enhance the ease of staff and volunteers in using the intranet. We received their feedback from this initial prototype to lead in the development of the training and maintenance guide, while tailoring the intranet to the organisation's special needs.

With the implementation of this project we hope that the sense of community for the dispersed structure of Arthritis Care will be enhanced. By implementing this intranet, we were able to help in reducing some of the communication problems of Arthritis Care as well as aid in the reduction of expenditures and wasted resources for the organisation.

The Interactive Qualifying Project at Worcester Polytechnic Institute provides students with the opportunity to see just how technology and society are linked. This project, though it may seem internally bound, is in actuality, a mechanism to establish a sense of community for the entire Arthritis Care organisation.

2.0 Background

The goal of our project was to help Arthritis Care develop an intranet to improve the internal communications within the organisation. In order to complete our task we needed to have a variety of background information that included: Arthritis Care as an organisation; social and organisational aspects of information flow; information architecture; networking; and training techniques and methods. Researching these topics was crucial in the completion of our project.

2.1 Arthritis Care

Arthritis Care is one of the United Kingdom's largest voluntary organisations working with and for people with arthritis from every section of the community (Arthritis Care, 1999, p.1). Arthritis Care has worked with and for the people with arthritis to promote their health, well being, and independence through services, support, self-help, and informational packets. The organisation prides itself on its user-led approach. Arthur Mainwaring-Bowen, who had arthritis, founded Arthritis Care in 1947. Currently, more than half of the trustees of Arthritis Care are people suffering from arthritis. There are more than 620 Arthritis Care branches and groups in the United Kingdom and over 7,000 volunteers (Arthritis Care, 1999, p.6). The organisation receives little government funding, so it is these 7,000 volunteers who are the secret to Arthritis Care's success. Arthritis Care provides a number of services to help people with arthritis and to raise a greater awareness of the needs of all people with arthritis. These services include: help lines; training courses; hotels; national home visiting services; and a number of publications.

Since 1997, Arthritis Care's Mission has been "empowering people to take control of their arthritis, their lives and their organisation" (Arthritis Care, 2000, p.6). This mission describes three areas where people with arthritis need to take control. "Empowering people to take control of their arthritis," means Arthritis Care providing quality service tools, such as help lines, publications, and courses, to help people self-manage the disease process, get access to information about treatments and aids in daily living. "Empowering people to take control of their lives," enables people with arthritis to regain a sense of perspective in their own lives. "Empowering people to take control of their organisation," is the understanding that ensures Arthritis Care's strategy and outputs are determined by the primary customer (Arthritis Care, 2000b, p.12).

The "KEY" is a collection of handbooks that are designed to provide the staff and volunteers with consistent messages about what Arthritis Care sets out to do and their role in making its mission a reality (Arthritis Care, 2000, p.5). The "KEY" provides the Arthritis Care staff with access to existing policies and procedures, team structures, and guidance notes. There are six main themes that run throughout the material in the "KEY" (Arthritis Care, 2001, p.6):

1. Being accountable for your role and responsibilities
2. Empowerment – of yourself and others – as a way of maximising Arthritis Care's mission
3. Continuous improvement of what the staff and volunteers do and how they do it
4. Equality of opportunity
5. Good communication

6. Support and development

The handbooks included in the "KEY" aim to empower Arthritis Care's staff. They are intended to enable the staff and volunteers to share a sense of pride in and commitment to Arthritis Care's mission.

2.2 Social and Organisational Aspects of Information Flow

Information flow within an organisation is a function of the organisation's formal structure, and of it social dynamics and operational culture. The understanding of the methods by which information flow could be studied and improved was deemed to be essential to our project.

The external organisational scheme, as shown in Figure 1, is able to emphasise many things about an organisation such as its division of labour, the chain of command and the areas of an organisation that would need certain types of information in order for it to function properly (Lakey, 1995, p.71).

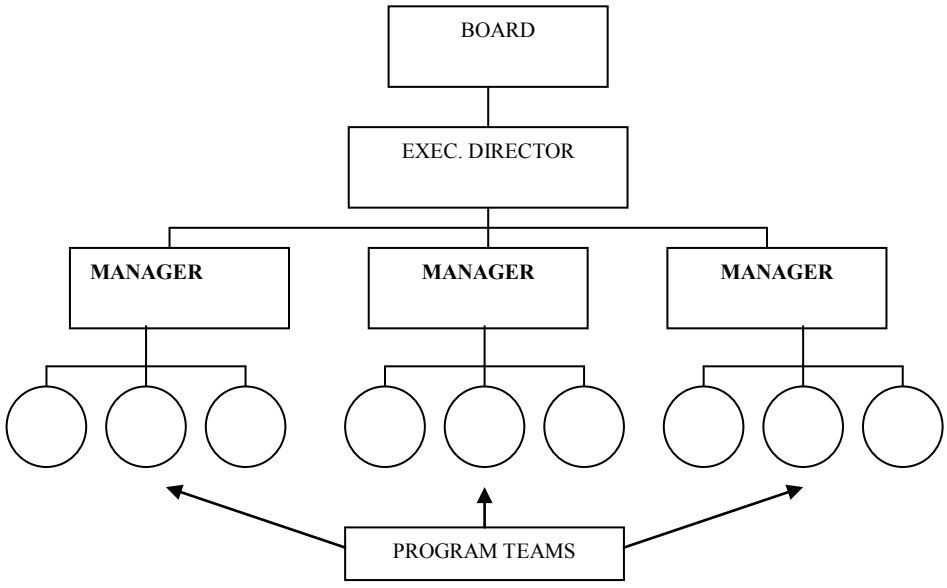


Figure 1 - Managerial Team Hierarchy

The external schematic structure of an organisation can therefore hold a lot of information as to how internal communications are suppose to flow within it. An external organisation, however, is limited in its ability to properly display the internal dynamics of the various parts of the organisation, most importantly the social organisation and informal processes within the organisation. Similarly, as the structural schematics are able to describe the formal division of labour, chain of command and communication of an organisation, ethnography is able to address the informal structure and social arrangements of an organisation. Ethnography therefore allows for a deeper understanding of how communication flows within an organisation, not on the basis of its formal structuring but rather as an organisation with its own unique social dynamics.

The definition of ethnography varies between its understanding as a conceptual term and in its application. The conceptual meaning has been referred to by Spradley as: "ethnography is the work of describing a culture" (Berg, 2001, p.133). Werner & Schoepfle referred to the application of ethnography as: "ethnography becomes the process of gathering systematic observations partly through participation and partly through various types of conversational interviews" (Berg, 2001, p.139). This process is further detailed as entering an environment with an accepting, appreciative, interested and most importantly sensitive attitude toward the customs and practices of the establishment.

The internal organisation of a non-profit organisation like Arthritis Care included volunteers and staff with a wide range of backgrounds, skills and education levels. The understanding of internal structuring and the uniqueness of the people who make up the organisation is explained and obtained through ethnography. In terms of this project

ethnography presented itself as a valuable tool, in addition to surveys and focus groups to understand Arthritis Care as a unique and diverse organisation.

2.2.1 Surveys

There is always a purpose to a survey (Newman, 1998, p.1). The purpose is dependent upon the audience wanting the information. It is crucial to focus the attention on the amount of time needed to satisfy the needs or requirements that are expected out of the survey. More people generally need to be surveyed than what is originally estimated because of the low return rates of the survey. Also, data gathering often requires more time than anticipated, resulting in data analysis and report writing taking longer.

After the purpose of a survey is developed, the next step is to define the population of your survey (Newman, 1998, p.11). When defining the population, one must consider such variables as age, gender, race, socio-economic status, religion, occupation, and education.

There are various procedures to which a survey can be administered (Newman, 1998, pp.25-28). The four basic ways to collect survey information are: (1) mail, (2) direct administration, (3) telephone, and (4) interview. The developer(s) of a survey must take into account many factors, such as, length, pertinence, clarity, and specifically the types of responses the survey is intended to receive. The mail survey is the most frequently used survey, likely to believe that it can be easily created. Often with the mail survey, the outcome results in either the survey being thrown away or the responses are biased, therefore not representative. The procedures for developing a directly administered survey are identical to those suggested for a mail survey. An advantage

with the directly administered surveys over the mail survey is that a poor return rate does not pose a problem.

The way one develops a survey is important in obtaining useful information. There are four major ways that items can be presented to respondents (Newman, 1998, pp.37-38). An item can be open-ended, which means that the respondent is asked a general question and might respond in various ways. This allows respondents to give the amount of detail that they choose. The opposite of an open-ended question is a dichotomous-choice item. Dichotomous-choice items divide the answers into two categories, usually true-false or yes-no. Another item is a forced-choice item, which is used when or if socially desirable responses might occur. This kind of item demands that the respondent chooses between two or more options. The best use of this item is where the options can be viewed as equally positive options. The last item format is the scaled-choice format, which allows more than two possible choices. Although respondents usually respond favourably, this allows the data to be used to compare various samples or to compare different items within one survey.

2.2.2 Focus Group

A commonly used way to effectively gather information in a social setting without the use of surveys is known as focus groups. Focus groups can either be a guided or un-guided discussion, within a small group, on a topic that is relevant to the group and of interest to the researcher. If a focus group is guided, the person guiding it is known as the facilitator or moderator. The moderator's job is to draw information from the members of the focus group, which is of interest to the research, by encouraging the members to speak freely about their opinions and feelings toward the issue. The size of a

focus group is very important to the success of the exercise. The group should not be too large because the moderator will not be able to obtain a well-represented opinion from each member. Too small of a group will result in not obtaining a good distribution of the representative opinions and perspectives that are available.

Focus groups are effective tools in obtaining attitudes, opinions and feelings on the topic at hand. The reason focus groups are so effective at obtaining these responses is because the responses are spontaneous and serve as insights on a topic that are socially constructed and not individually thought responses. Rubin and Rubin explain the goal of a focus group, " …the goal is to let people spark off one another, suggesting dimensions and nuances of the original problem that any one individual might not have thought of. Sometimes a totally different understanding of a problem emerges from the group discussion" (Berg, 2001, p.115). Although focus groups may not reach the level of depth that a personal interview can attain, they are instrumental in obtaining a breadth of opinions from representative people on a social setting. The use of focus groups by social scientists has risen dramatically over the past ten years. Having existed since the beginning of World War II the use of group interviews has proven to be a valuable technique in gathering qualitative data in social science and will prove to be effective into the next century as the opinions and attitudes towards technological advancements increases.

2.3 Information Architecture

Information Architecture (IA) is a fundamental part of creating any efficient website. The same fundamentals that apply to regular architecture apply in IA. The way to think about this is to think about different types of buildings, e.g. a gas station, pizza

shop, and coffee shop; each has a different architecture and a different meaning. Most people would not be comfortable sitting down at a gas station for a cup of coffee and reading a book. The same principles hold true for web site design. A web site must be designed in a way that will work for its purpose, just like a building has to do the same. Understanding this shows the necessity for a proper IA in any web site.

Now that we know why an information architect is necessary, we must understand what his/her primary goals are (Rosenfeld & Morvile, 1998, p.11). An information architect must:

- Clarify the mission and vision for the site, balancing the needs of its sponsoring organisation and the needs of its audience
- Determine what the content and functionality the site will contain
- Specify how users will find information in the site by defining its organisation, navigation, labeling, and searching system
- Map out how the site will accommodate change and growth over time

These goals may seem obvious, but the information architecture is about what is not obvious. No one notices when architecture is done correctly, only when it is done poorly. And if someone were to notice the good architecture, it usually is attributed to a good layout or graphics. Truly understanding the information architecture as a consumer takes a long time and a lot of navigation through the site, but these are necessary parts of a site. If the information architecture does not work, then it negatively affects consumers and producers of the site.

Consumers of the site are usually the users; they can be customers or employees. A user does not want to get lost in a site looking for the information that he/she needs. This is important on the outside to keep customers happy and on the inside as well to reduce the time employees' waste looking for and using internal communications (Rosenfeld & Morvile, 1998, p.12). Users have varying needs when it comes to a site. Some know exactly what they are looking for, what it is called, and where it should be. Others though, might not know quite as well what they are looking for or where to find it. A site should be able to handle both of these users, ones that have a known item they are searching for and those who are just browsing.

Producers of the site are an important factor in the design as well. Any producer is looking for a return on his/her investment. In the case of IA, it is hard to show the tangible results of the work, but the results are evident (Rosenfeld & Morvile, 1998, p.13). Having a well-defined site that lets people find what they need quickly and easily will create a better customer base. This also applies if the site is internal because it will create easier and better communications.

Knowing the goals of the information architecture and what is expected, the actual information architect's requirements should be discussed. The information architect must be able to see the project as an outsider but be able to understand it like an insider. The architect should be a committed employee with various skills. Specialised skills include: graphic design; information and library science; journalism; usability engineering; marketing; and computer science (Rosenfeld & Morvile, 1998, p.17). Knowing whom the information architect is and what he/she has to do, we must now look at the structure of information architecture starting with organisation.

2.3.1 Organisation

Understanding information is based largely on the ability to organise it. Answer any question about where you live or how to get in touch with someone, and this reveals a system of classification that forms the foundation of understanding (Rosenfeld & Morvile, 1998, p.23). In our ability to organise, we can explain and control. Information architects must use the ability to organise to guide people. In a flexible environment like the web, organisation is a powerful tool to create successful pages. In an attempt to organise information there are many challenges that one must be aware of.

Organisational challenges have always existed and are nothing new. There are ways to deal with the common recurring challenges. These challenges include: ambiguity; heterogeneity; differences in perspective and internal politics.

Ambiguity is often a problem encountered in language and is then carried into classification systems (Rosenfeld & Morvile, 1998, p.23). It is when a word or phrase can have multiple meanings, and the user may understand the word in the wrong context.

Heterogeneity is a collection of unrelated or unlike parts. In a website this is a very common occurrence, a website is a collection of links that can lead to documents, articles, pictures, and even more links (Rosenfeld & Morvile, 1998, p.24). Because of this, these websites are very hard to impose a structured organisation system on.

Difference in perspective is another common problem with web site organisation (Rosenfeld & Morvile, 1998, p.25). When a website is created it must be done so from the point of view of a person external to the organisation. If this does not happen, the site is usually created to represent the organisational chart of the company, and that is not always intuitive to the customer.

Internal politics is another common ground for internal organisational problems. This usually can be avoided if the information architect is an outsider to the firm (Rosenfeld & Morvile, 1998, p.26). But the fact is that internal politics will always play a role in an organisation. Groups always feel that their information is the most important and should be the first displayed. Therefore, the structure of intranets and Web Sites specifically will be discussed.

Organisation of information in Web Sites and intranets is an example of an organisational system (Rosenfeld & Morvile, 1998, p.261). Most organisational systems can be broken down into two different topics, organisational schemes and organisational structures. A scheme defines the shared characteristics of content items and the grouping of those items. A structure defines the types of relationships between content items and groups. Organisational schemes are used daily, such as searching a phone book, supermarket or on the web all reveal different types of organisation. These types of schemes are either exact or task-orientated designs (Rosenfeld & Morvile, 1998, p.261). The first of the two is alphabetical organisation, the type used in phone books. It is also the primary organisational scheme for most encyclopaedias and dictionaries, and for most non-fictional books provided with an alphabetical index. The next exact organisation scheme is chronological. This scheme orders things by date. Usual items include history books, magazine archives and diaries. Another exact organisational scheme is geographical. Geographical organisation is a rather straightforward design and is usually used by maps and directions.

Opposing the exact-based organisational structure is the ambiguous organisational scheme. These types of schemes divide information into categories that defy exact

definition. Though these methods may be a bit more confusing to map out, they are generally more useful in organising websites.

The first example of an ambiguous organisational scheme is topical (Rosenfeld & Morvile, 1998, p.29). A topical design is where all information is organised by topic, such as in the Yellow Pages. In general, web sites should not be solely organised as a topic structure but should be included in the navigation.

Another ambiguous scheme is task oriented. Task oriented schemes are generally used in applications, usually in ordering menus and goals to be accomplished. It is not common to see task oriented web sites because most sites are content driven, not application driven.

The third type of ambiguous scheme is that of audience specific (Rosenfeld & Morvile, 1998, p.30). This method refers to organising the information according to the group that will be using it. This method is useful on the web for creating externally accessed intranets and portals. Another scheme of the same type is metaphor-driven organisation. This method uses common terms to refer to items of information, such as the desktop, with files and folders. This can be hard to fully utilise in the web though. Using metaphors can sometimes imply that more is available under a certain topic than is actually there.

The last type of ambiguous organisational scheme is a hybrid. (Rosenfeld & Morvile, 1998, p.34). The hybrid scheme is the most common in use on the web but in most cases does not work well. A hybrid scheme uses parts of all the previously listed schemes and jumbles them up into one organisation. Sometimes this can work if the structures are kept separate, but it is usually hard to accomplish.

Organisational structures are another intangible yet priceless part of the design of a website. The structure of the organisation is rarely thought about but never missed. There are three main type of organisational structure that should be discussed, hierarchy, hypertext, and database models.

The hierarchy organisational structure is usually the foundation for almost all-good information architecture (Rosenfeld & Morvile, 1998, p.37). This may seem not to coincide with the normal layout of the web, but the hierarchal structure has been used since information was first starting to be organised. This makes it easy to understand and follow. The normal design of a hierarchal structure is a top-down approach. In designing a hierarchal structure it is important to the keep the structure in mind; the depth and span of the menus should be kept balanced. In newer sites that are leaving room for expansion it is better to create a wide shallow structure as opposed to a deep narrow one. The hierarchal model is a good starting place for organisation, but it is not the only model available.

The hypertext structure is relatively new and a highly non-linear way of organising data (Rosesnfeld & Morvile, 1998, p.37). The system consists of two components, the information and a link to the information. A combination of these creates many interconnected nodes with no structure. A hypertext structure is very hard to navigate and can be confusing. It is usually not a good structure for a front page but does have its uses in side of documents and other pages for a quick useful link.

The last type of structure is the relational database model (Rosesnfeld & Morvile, 1998, p.39). A database is a tool for organising information about a particular item. A good example of a database is an address book or an inventory. Databases allow for

collections of data to be organised and accessible across many platforms including the web. Usually databases are not a good design idea for the main site. They are, however, very useful for data intense sub-sites within the site.

As was seen through the discussion of organisation, there are many things that need to be kept in mind when an organisation for a structure is being developed. All of these factors will contribute to well planned and effective organisation. Now knowing how to organise information in a web site, we must look at how to navigate through it.

2.3.2 Navigation

Navigation is life and death on the web (Rosenfeld & Morvile, 1998, p.47). Having a poor navigation system accompanying a good organisational structure will negatively affect the performance of the site. A good organisational structure will do a lot for the navigation of a site, but it is not the only thing. There needs to be a good accompanying navigation system to help customers learn about products, services, or topics associated to the specific content.

Navigation systems consist of a variety of elements (Rosenfeld & Morvile, 1998, p.48). Some are graphical like navigation bars and pull down menus that are integrated into the documents, where others feature a site map or index that provides remote access to the content. The successful merger of these items creates different types of navigation systems.

The first navigation system to look at is the hierarchical system (Rosenfeld & Morvile, 1998, p.54). The hierarchal navigation system is primarily taken from the information organisation. The organisation creates the main and sub pages and links

between them. A hierarchal system is very important but often limiting, so it usually requires additional elements.

A global navigation system is a site-wide navigation system that encompasses the entire site (Rosenfeld & Morvile, 1998, p.55). The easiest way to accomplish this is with a navigation bar. The bar would be available on all pages and change based on the where it links to. A global navigation system can usually be applied to an entire site keeping everything in context. If a site is really large and complex, it is often beneficial to include a local navigation tool that provides for more localized redirection, such as in a catalogue. The global navigation could take you to the sections of a catalogue, but a local tool would itemise the products and link to them.

The last type of navigation is ad hoc (Rosenfeld & Morvile, 1998, p.55). Ad hoc navigation is usually used where neither of the previous systems would suffice. This usually happens within the text of a document itself. If the author wants to link to another section or a reference document from inside the text, it is easy to do so with simple hypertext links.

In global and local navigation the tools that are used to accomplish the navigation are really integrated elements. These are elements that show up again and again on all the pages of the site and can usually be categorised into two types, navigation bars and pull-down menus.

Navigation bars, as mentioned in global navigation, are very useful tools for providing a single point of navigation throughout the entire website (Rosenfeld & Morvile, 1998, p.56). In its simplest form, the navigation bar is a collection of hyperlinks at the same point on a page or can be more advanced as graphics with image maps or

combinations of graphics. Whatever the design is, navigation bars provide for a centralised location to navigate through most of the site.

Another common way to do navigation on a site is with frames (Rosenfeld & Morvile, 1998, p.58). A frame allows the screen to be divided into different sections that can remain while others change. Frames are somewhat more complex and harder to implement. When done well they are not obvious to the user and a natural part of the navigation, though frames have become less and less accepted in recent times.

The last common integrated navigation system is pull-down menus (Rosenfeld & Morvile, 1998, p.59). Pull-down menus usually provide a list of options with only one of them available at first. The user would select the menu and a list of elements would become available, the user would then select the item he/she wants and then usually would click a "go" or "submit" button beside the list. Pull-down menus can be useful when implemented correctly but often are not a good solution to the problem.

Along with integrated navigation elements there also exists remote navigation elements. Remote navigation tools are supposed to be supplemental to the site. They are an external view to the site that should be provided but not relied upon. A few examples of remote navigation elements are: table of contents, indices, and site maps.

The table of contents and index are usually the sole element of navigation in print. They are well refined and usable and this should not be overlooked on the electronic media. A table of contents is a good fit on a web site that has a strong hierarchical organisation. It provides good reinforcement and quick navigation for people that know what they are looking for. An index would be a better fit for a site with a weak organisational hierarchy. An index provides a list of elements on the web page usually in

alphabetical order. Determining the elements to index can sometimes be more of a challenge, should it be: web pages; concepts; or even individual paragraphs that need to be indexed. This decision does not have to be set in stone. Sometimes the best way to determine the index is to track the usage of the site and determine what the majority of people are looking for and then index those elements.

The last common remote navigation element is the site map. A site map provides a graphical representation of the architecture of the web site. The site map, unlike the table of contents and index, is used to navigate physical rather than intellectual space. When creating a site it is good to keep the level of abstraction high, and this tends to keep the map more intuitive, making the map more useful.

Designing a navigation system that works well is a challenge. It requires a combination of global and remote navigation tools and a lot of testing. Navigation is an important part of any site and must be done properly to create an easy-to-use and well-designed site.

2.3.3 Labeling

In order to represent large pieces of information quickly and effectively we must use some type of labeling system. Just as spoken words are used to represent thought, labels are used to represent larger pieces of information in print and web sites. Labeling is usually a normal part of the design of the web site and not often thought about, but if it is a conscious part of the design, the labels will work much better. Labels are very important because they cannot change to an audience's reaction as a speaker can; they are constant and must be well planned.

Before discussing the actual labels and how to make them effective, it is important to define our discussion in terms of labeling systems and not labels (Rosenfeld & Morvile, 1998, p.74). Like the other systems that have been discussed, labeling is a system itself. A good labeling system must cover subject areas and not just key words. By covering subject areas the labels will be more comprehensive to the user. Looking at a consistent labeling system the user understands how the system works, and this will make the entire site seem familiar. Knowing that labels are part of a system, the next thing to discuss is the type of labeling systems.

In web sites, labels are usually either textual or iconic and are generally used in two ways: as links to larger pieces of information on other pages, or as headings that break up and identify larger pieces of information.

The first major type is a label within navigation systems (Rosenfeld & Morvile, 1998, p.76). Just as navigation system must remain constant, so should a labeling system. Familiarity and consistency keep navigation easy and smooth, and labels have a large part to do with this. Keeping labels consistent within the local web is important and even through the entire web helps. This can be achieved by using common terms to represent home, searching, contact, help, and other information. This also can cause some problems with users not being exactly sure what a label might lead to. The best solution to this problem is to provide some scope notes, or brief descriptions of the label.

The next type of labeling system is labels as indexing terms (Rosenfeld & Morvile, 1998, p.77). This type of labeling is two fold. The first type is external searching of the site and the second type is internally supporting browsing of the site. The external aspect is using tags in the web page to provide keywords that browsers

search for. This is also where the description usually gets placed and helps for searching for the site. The internal aspect is very similar. This type is when keywords are defined for sections of the web page that makes it easier to search for sections and information.

Another type of labeling system is labels as headings (Rosenfeld & Morvile, 1998, p.82). This is much like how a report is set up. There are different levels of organisation and information below each level. This can happen in two forms on the web. Usually on a front page the sections of information are very small but on pages with a lot of content it is similar to reports.

The last major type of labeling is iconic labeling system (Rosenfeld & Morvile, 1998, p.84). Iconic labels, though seemingly fitting the common phrase "a picture is worth a thousand words," are often not that useful on the web. It is very hard to come up with common icons that represent the meaning that you desire. Icons can represent a thousand words when you only want them to represent one, and this is a problem for consistent and easy navigation. The best time to use icons is when they can contribute to the graphical layout of the site or when they are very familiar and common.

Now knowing what some of the common labeling systems are, the next thing to discuss is the creation of labels. When creating labels the first thing to figure out is where the labels are going to come from. The most common sources are: the site itself; other web sites; vocabularies and thesauri; content; and users and experts.

Taking labels from the site that is already in place is an easy way to figure out the labeling system (Rosenfeld & Morvile, 1998, p.88). When the site was created and the content compiled, there were labels that naturally came into place. These labels should be gathered and then organised as a good starting place for the labeling system. Another

good way to create the labeling system of the site is to take a look at other sites and use their ideas. Take a look at other similar sites and see what works for their site and what does not work and adapt that to your own.

The next place to look at a source of labels is in controlled vocabularies and thesauri (Rosenfeld & Morvile, 1998, p.90). These resources are a good place to look when you do not have any other place to start. Controlled vocabularies are simply lists of predetermined terms that describe a topic. A thesaurus is a controlled vocabulary that also includes relationships to those words. The relationships that a thesaurus can provide are a really good way of thinking of labels for your system.

Another source to consider for creating labeling systems is content (Rosenfeld & Morvile, 1998, p.94). The labels can form from the document and information on the site itself. You can use the titles in the document, or the general subject concepts that the author uses. There can be problems with using the terms that the author uses. The author may use terms that do not exactly fit the true meaning of the subject. This can skew the labels' meaning and confuse the user.

The last place to look for sources of labels is users and experts (Rosenfeld & Morvile, 1998, p.96). This is often the hardest but best place to take the labels from. One way to get labels from users is to analyse the search strings, from searches on the web site. The other place to go is to experts. People like librarians are a good resource for common labeling words and sources.

Bringing attention to labels before the implementation of the website will help to reduce any problems that might have come up when users are using the site. Labels are an important system in the structure of a website and must be looked at for clarity. They

are constant representations of the site and do not change, thus they must be as clear and effective as possible.

2.4 Information Technology

An easy and widely used method of connecting two computers together is known as the client/server method (Lowe, 1997, pp. 8-10). This networking system makes use of a high-powered computer, known as the mainframe that is equipped with ample space and processing power to store large amounts of data. The data stored in the mainframe computer (server) is made available to the smaller personal computer (client) so that it does not overwhelm its storage capacity and processing power. The advantages to the client/server method mainly revolve around its centralisation, secured access and ability for its users to log on (Wagner, 1999, p.37). The disadvantages are that if the mainframe server goes down, the client computers suffer as well. Without expanding to another network, the client/server gets increasingly harder to configure and fix as it grows in size. The hardware and software for the server is also expensive. When the PC first started to come out in 1981, the users of these new personalized computers found that they needed a more efficient way of sharing information (Wagner, 1999, pp.37-39). This developed into the use of Local Area Networks (LAN), allowing the personal computers to talk to each other. With this more efficient and cheaper means of inter-PC communication, the use of mainframe computers became obsolete, but it did not mark the end of client/server based operations.

There are several other ways network designs are able to facilitate the communication of information (Wagner, 1999, p.8). As mentioned above, Local Area Networks are especially useful for the efficient transport of data between departments of

an organisation. These networks are usually confined to a single building or office and allow users to efficiently share network resources. The necessary equipment to set up a LAN is inexpensive and readily available. The essential components of a LAN are: servers, workstations, a network operating system and communication links (Wagner, 1999, pp.8-9). Servers appear at the top of the pyramid that centralise the security, management and data of the network. The operating system is the second component that is specifically designed for networking and creates the arena through which information can actually travel over a network. Software and hardware that are designed to facilitate data over a network are the third component. Applications and hardware components that are found in a personal computer or computer workstation are the fourth component. This set-up seems quite versatile but is deficient in the following areas: lack of centralisation, integration between those of a different design, and networking area limitations (Wagner, 1999, pp. 8-9).

Metropolitan Area Networks (MAN) are designed to transport data over large but localised land areas (Wagner, 1999, p.11). They are ideal for campuses, towns and cities utilising digital media and fibber optic wires to communicate information. These networks are ideal in connecting not only large areas (campus to campus, building to building) but also are able to connect various LAN's together. The downside of using Metropolitan Area Networks is that they are not often open for public access, and their large size of coverage makes maintenance difficult.

Wide Area Networks (WAN) are a step up from Metropolitan Area Networks in that they are able to connect together organisations, campuses, institutions, cities and countries. WAN's are unique in that they are able to effectively centralise the data that

must also be sent to other distant locations (Wagner, 1999, pp.12-13). The data are sent over telephone lines, which allows for enormous amounts of data being sent to thousands of machines on a global level. The downside to this is expensive hard and software, extensive and costly maintenance and high connectivity charges.

The previous network designs acted as stepping-stones to what is now known as the World Wide Web or Internet (Wagner, 1999, pp.13-14). This network, which is also known as the "Network of Networks" (Wagner, 1999, pp.13), transfers information over the international telephone system and connects thousands of personal computers that are on local, metropolitan and wide area networks. The Internet that we are most familiar with first came to light between 1982 and 1985. During the mid 1980's the number of hosts started to increase greatly, and the first available and free public server was available. In 1990 the number of Internet hosts climbed to 300,000, and the US government relinquished its control over the prodigious network, handing it over to the National Science Foundation (NSF), which in 1991 let the public take it over (Wagner, 1999, p.13).

In contrast to the Internet, a smaller, more self-contained version called the intranet allows all the communication features of the Internet (i.e. web browsers, network administration tools, and collaboration programs) but is not accessible to the outside world (Wagner, 1999, p.15). This private network is controlled by a business or organisation and is intended for its use only. Another benefit of this type of networking is that data can be accessed with any type of computer, whether it is an IBM, Macintosh or UNIX system, because it still operates through the World Wide Web (Internet) (Wagner, 1999; p.270). Conversely, an Extranet is a privately maintained intranet that is

not as centralised as an intranet, allowing other distant locations to collaborate over secure connections of the Internet (Wagner, 1999, p.270). Together, intranets and Extranets are the two main divisions of the Internet that allow information to be securely shared without relinquishing the many benefits of being connected to the Internet.

The various networks previously mentioned have proven themselves as a great way to streamline a company, interconnect colleges and universities, and generally make knowledge readily available to anyone. There are various levels that information is made available to people. Secured access, therefore, is a very important concern for anyone who relies on a small network or Internet. The first issue for tightening security to a network (on an internal level) is within the operating system that the network runs on (Wagner, 1999, p.191). Factors such as encryption, logging and access control can prevent security breaches. If the network is heterogeneous (meaning systems accessing it use different operating systems), security issues become harder to maintain. Ideally a secured network is one where all the workstations use the same operating systems (Wagner, 1999, p.192).

A breach in security can basically occur at four levels: human, hardware, software and networking (Wagner, 1999, pp.193-197). On the human level, it is basically common sense to avoid offering network password information on the phone, via e-mail, written on paper. In regards to hardware, data are always prone to theft and destruction. Splicing is also a concern to the security of a network if an unauthorised individual succeeds in connecting his/her laptop or computer to the shared network wire. On the software level, some risks are flawed applications, viruses and weak password protection. On the network level, passwords should never contain less then five characters,

commonly used words, names, birth-dates, social security numbers and all the same numbers, letters or combinations of them (Wagner, 1999, pp.193-197).

Once Internet connectivity is available external security issues must be taken into consideration. Some easy ways to remedy this are by using sacrificial hosts, firewalls, security checklists, security mailing lists and security news-groups (Wagner, 1999, pp. 201-205). Sacrificial hosts and firewalls circumvent an intruder by either leading them to a server storing minimal information that can be recovered or leading the intruder to a dummy computer, router or hardware device which is the single and only entry point to one's site. This computer can then effectively grant accessibility requests to the authorized ones and discard the others.

2.5 Hypertext Mark-up Language

The Internet is assuredly the most extensive and comprehensive reference tool to access various forms of information throughout the world. With this communication tool in place the next problem is how to organise and present the information on a computer screen so that it is easy to navigate through and directly access other sites that are referenced in the previous one. One method to do this is by using a language called Hypertext Mark-up Language (HTML) (McFedries, 1997, pp.11-12). HTML is used to create hypertext links and documents. This rather complex title is given to a language that is primarily composed of directional tags that organise and present text on a website. It is a small library of two and three letter combinations and words are used to specify styles, positions, orientations, listings, and tables of text that are viewed on a website (McFedries, 1997, pp12-13). Pictures can also be positioned and viewed using HTML. The word "Hypertext" refers to the feature of the browser system (i.e. Outlook Explorer)

immediately transporting the user to another web page when the hypertext link, which is usually an underlined word or phrase in the text, is selected on the present page.

The use of short tags to make text bold or italic is one reason why HTML is so effective. HTML improves the attractiveness of a website and makes navigation much easier than if it were all in plain text. An HTML document can be created using any text program such as Microsoft Word or Notepad. Internet browser programs do the hard work of deciphering the HTML marker tags so that the text will appear on the web page as directed by the tags (McFedries, 1997, pp.27-28). The tags, which are sometimes words or phrases, are always surrounded by angle brackets < > so that the web browser understands that it is HTML and not extraneous text (McFedries, 1997, pp.33-34). The fundamental design of an HTML sentence will begin and end with the same tag containing text in the middle, for example, <TAG> text </TAG>. WebPages begin and end with the tag <HTML> (McFedries, 1997, pp.28-29). The document is then divided into two sections known as the head and the body. The head tag is defined as <HEAD> and </HEAD> that is typed below the <HTML> tag. The main body of the text in the website is then further divided under the head tag by using <BODY> and </BODY>. Another main component that is needed to load a web page into a browser is the <TITLE> tag. This tag gives the website a title that appears in the upper title bar of the browser's window. The title tag is always placed after the <HEAD> tag in the main backbone of the HTML document (McFedries, 1997, pp 29-30).

With this basic understanding of how HTML tagging works and the main structure of an HTML document, an attractive and easy to navigate web page can be easily constructed. Additional text features such as: paragraph alignment </P>, bulleted

lists <UL TYPE=type>, background colour <BODY BG COLOR="nnnn">, bold type <B> and italic <I> organise the text for easy reading and navigation (McFedries, 1997, pp. 36-37). The Internet has created an arena with boundless limits for efficient information exchange throughout the world. Without coding languages like HTML the navigation and organisation of information available on the Internet would be nothing like we know it today.

2.6 Training

Information technology concepts are constantly changing, which is why employees must be trained to stay up to date with the most recent changes. Arthritis Care is composed of volunteers as well as staff members, which brings about different levels of education. In training a co-worker or even a relative at home, there are some simple guidelines to follow (Goldsborough, 2000, p.31):

1. Keep in mind the trainee's experience level.
2. Use language the trainee will understand.
3. Reassure him/her that it is difficult to break the computer.
4. Be aware that adults learn best when they are self-motivated, not pressured.
5. Appeal to trainee's self-interest.
6. Demonstrate simple tasks first.
7. Provide trainees with "cheat sheets" that show step-by step methods.
8. Give trainees time to practice.
9. Have patience.
10. Have fun.

If an organisation is to be viable, it needs to have an ongoing training plan for self-renewal. An ideal continuous training plan contains five phases (Connors, 1988, sec19.1):

1. Pre-service Training
2. Start-up Support
3. Maintenance-of-effort Training
4. Periodic Review and Feedback
5. Transition Training

Pre-service training helps the new volunteer take a look at self and skills, at the job that needs to be done and at the organisation's philosophy and services (Connors, 1988, sec19.1). A tour of the organisation's headquarters should definitely be included in the pre-service training because it gives the volunteer an understanding of the operation of the organisation through structured observation. Group meetings, role-playing, and informal conversations with other employees are also vital to this training.

Start-up support is when volunteers, having some initial training, begin actual work for the organisation; they enter a period in which much support is needed (Connors, 1988, sec19.3). It is during this time that volunteers need to feel recognised and accepted among co-workers because their skills and resources are probably least developed. It is during this critical time that supervisors, trainers or co-workers need to reach out to the volunteer and communicate what they want and need from the volunteer.

The purposes of maintenance-of-effort training are to increase the skill of volunteers, to get them out of the ruts they might have fallen into, to answer questions

and deal with concerns, and to refine practices (Connors, 1988, sec19.4). On-the-job training helps build and maintain good morale, which gives the volunteers the feeling that the organisation is committed to growth on the job.

Periodic review and feedback are important to any kind of training. The implementation plan ought to include some checkpoints or stop-action periods when discussion can be held about how well the training is working out and what new problems have come up (Connors, 1988; sec19.5). The feedback from all sources should be taken into consideration in deciding whether to continue or redesign any training programme.

Transition training is important when a volunteer prepares to enter into a new role and more responsibilities. Transition training, whether individual or group forms, should recapitulate the first four phases of the training framework, in a somewhat different manner (Connors, 1988, sec19.5). There should be a pre-transition training, which is a period of referral if the volunteer is going to be working with different people, a start-up training, maintenance-of-effort training, and periodic review and feedback sessions. Using the continuous training plan a number of training models can also assist in the training procedure.

There are a variety of training models, but each one is a package unto itself, designed for a particular group or individual at a particular time for a specified purpose (Connors, 1988, sec19.6). One model for training is a workshop on a particular topic. Such workshops could be conducted for a group of volunteers with some specialised resources to work on a particular series of content items. Another method of training is to conduct "simulation hours," in which participants simulate situations that have

occurred or that are anticipated. Role-playing is another method that can be used as part of a workshop or total training piece. A conference, which brings together volunteers and staff members, is another training method. During a conference volunteers and staff members can get together to discuss problems, ideas, and innovative practices. While using these training models there are a number of training techniques and training devices that can accompany the training procedure to enhance the learning experience.

Training techniques are learning activities that the instructor selects as being best suited to the needs of the learners and are often employed in combination with training devices (Ilsley, 1981, p.122). Such techniques include lectures, panels, group discussions, buzz groups, role-plays, process demonstrations, field trips and case studies.

The primary purpose with using a lecture technique is to convey information. The instructor may introduce a new subject or relate new material to content that has been taught. This is an appropriate technique when the particular information is not available by other means.

A panel is like a lecture, but it is primarily designed to present specific knowledge. However, this knowledge is a composite of information possessed by experts or by persons whose experience qualifies them to speak with authority (Ilsley, 1981, p.124). The advantage of this technique is that a variety of views held by qualified people can be communicated in an informal manner to learners, who will thereby enjoy a new learning experience.

The group discussion technique requires the instructor to guide a group of learners to engage in purposeful dialogue on a mutually selected topic. The purpose of a group discussion provides learners with an opportunity to learn from each other by sharing

information and opinions, and by identifying and exploring problems and possible solutions (Ilsley, 1981, p.126). Group discussions encourage active participation and allow participants to develop their ability to work in a group, which will help them acquire self-confidence.

Smaller, more intimate buzz groups allow learners to hold brief, informal discussions of assigned topics, or concentrate as a group on a specific topic (Ilsley, 1981, p.127). These buzz groups allow for a number of questions and recommendations to be formulated. They also allow for shy people to find it easier to participate and allow for the instructor to observe individual learners more closely.

Role-play helps in understanding the behaviour of individuals when confronted with a certain problem (Ilsley, 1981, p.129). A role-play provides the learner with an actual problem situation to act out, analyse and discuss.

Demonstration permits the learner to gain a new activity or skill (Ilsley, 1981, p.130). By showing clearly how an activity or skill should be conducted, the demonstration can arouse the learner's interest and help the learner develop confidence by performing the activity or skill.

The field trip provides information to learners through a wide variety of experiences, such as education and visits to a learning centre (Ilsley, 1981, p.131). The informal nature of a field trip gives it a flexible structure, which makes it possible for the learners to become better acquainted with each other.

By analysing data pertaining to a real situation or case study, learners find ways to deal with similar problems (Ilsley, 1981, p.132). The case study technique develops

analytical skills involved in problem solving and enables learners to tackle a "real" situation rather than just a hypothetical one.

A training device always accompanies a training technique. The purpose of using a training device is to reinforce the knowledge or skills being taught through more than one sense, for instance sight, sound, or touch (Ilsley, 1981, p.136). Such devices include printed materials, audio and visual aids.

Printed materials include books and pamphlets. These types of materials allow the learners to study at their own rate. The information is also presented in a systematic manner, which allows learners to follow the instructor as well as get a better understanding of the material being presented by the instructor.

Audio and visual aids include videotapes, audiotapes, films, transparencies, models, chalkboards and many more. These devices allow the instructor to present material in a systematic sequence, highlight major ideas, and reinforce verbal explanation.

3.0 Methodology

3.1 Introduction

The goal of our project was to create a comprehensive intranet design containing all necessary information that would serve to facilitate the internal communications between staff, volunteers and managers of the Arthritis Care Foundation. A training manual was also created to accompany the intranet. Creating the intranet was accomplished by designing an information architecture to structure the content which it held and then implemented the design. The training guide was created by the feedback that we obtained from the survey and focus groups with the Arthritis Care staff, volunteers and managers.

3.2 Information Architecture Design

In designing this intranet it was important to clarify the exact forms of content it would contain. The intranet contains a directory and document repository, but is designed such that it can be easily expanded. The directory contains various types of organisational and contact information, and the document repository is a location where many different types of documents can be stored and then manipulated. The document repository was designed to be customisable such that it can store any type of data in any form of organisation, this was done so that it does not limit the content and can be easily expanded.

3.3 Intranet Implementation

To implement the intranet design we used HyperText Mark-up Language (HTML), JAVA script and Perl. HTML is the language in which the majority of web pages are written. JAVA script and Perl are tools to accomplish more tasks that would otherwise be impossible with only HTML. Perl was used to produce many Common Gateway Interface (CGI) scripts to allow the user to interact with the intranet. This constitutes the background to the intranet design. The implementation began with the main intranet web page. This single page contained the links to other various parts of the internal website. One of the main links directed the user to the Arthritis Care document repository. In the repository the training manuals and orientation handbooks for staff and volunteers are kept. The repository has the ability to have documents downloaded and also uploaded by privileged users. Other links on the main page provide easy navigation to different sections of the intranet. Some of these sections range from recent Arthritis Care publications, to a directory and scheduling tools. These various links are part of the navigation system that in part defines the functionality of the intranet. Another part that was implemented is the search tool to allow for quick and easy access to information on the entire site.

Secured access was also of primary importance. Secured access allows for external entry to the intranet from satellite Arthritis Care locations outside of the central office. This allows anyone with a password to view all internal information. The method of entry consists of a user name and password. The goal was for synchronisation of the intranet with the currently designed user management system so that the user name and password would work for both systems. Unfortunately, this was not possible because of

the complexities of the systems. Instead a user management system was implemented and the passwords were generated.

3.3.1 Workshop

We conducted a workshop that included showing the staff of Arthritis Care what an actual intranet looks like. To do this we used the Worcester Polytechnic Institute (WPI) home page (http://www.wpi.edu), where we logged into the WPI Web Information System with our user name and password. We showed the staff how to move around through the site, how to go backwards, forwards, click on icons, as well as how to actually log into the site. We explained the different password access ability by demonstrating how a student can change his/her own schedule but cannot change the class lists whereas a faculty member can. The workshop was intended to just familiarise the staff with what an intranet can do and contain.

3.3.2 Focus Groups

We conducted focus group sessions to learn what the staff and volunteers thought of the prototype of the intranet. These sessions were conducted in the London Central Office. By meeting with focus groups we were able to interact with the staff and volunteers more openly to find out what they thought worked and did not work for the intranet. We also took into consideration some methods of ethnography, such as: being sensitive, open-minded, interested, and appreciative in obtaining the feedback and suggestions from the organisation's members. By using these methods we were able to ease ourselves into the social structure of Arthritis Care, to understand its unique social structuring, and to present our intranet prototype as something that they could change and

not necessarily something that would change them, or affect them adversely. By using focus groups we were able to stimulate discussion on the advantages and disadvantages of the intranet based on the prototype. The focus groups generated a large number of ideas, opinions, and topics, as well as solutions to problems that were identified during the discussion. This feedback from the staff and volunteers allowed us to construct an intranet that was beneficial to Arthritis Care.

3.4 Training

3.4.1 Survey Questionnaire and Focus Groups

We distributed a survey questionnaire to the entire staff at the Central Office. This survey questionnaire contained questions on how one prefers to learn and what types of learning techniques and devices he/she prefers to use. Distributing the questionnaire before conducting the focus groups allowed the participants to think about what types of techniques and devices they usually use to help them learn. Allowing the participants to think about the questions before hand gave them time to think about what they might want to discuss in the session and prepare any questions that they might have to start with.

We conducted focus groups to generate ideas on how the people of Arthritis Care prefer to learn. The focus groups stimulated discussions on the structure of the intranet and on different training devices and techniques that were helpful to the staff of Arthritis Care, as well as establish the training techniques and devices that were not helpful. The focus groups and the survey questionnaire helped us generate common training devices and techniques that the majority of the staff, volunteers and managers can use.

3.4.2 Training Manual and Quick Reference Sheet

All people learn differently. With this idea in mind we created training mechanisms for all types of learners at Arthritis Care. Using the feedback we received from the focus groups and the survey questionnaire we produced training methods that were able to enhance the learning process for Arthritis Care. A written training manual was produced so that Arthritis Care would have a written reference of how the intranet works and how it can be updated. We also created a written one-sided quick reference sheet for the staff. This sheet can be kept in or on an individual's desk so whenever they need help or a quick question answered they can just look at the reference sheet and find out what they need to know.

A practical, user-friendly training manual for the intranet was produced. It contains step-by-step methods on how to use and update the intranet. It provides information about the intranet that not everyone needs to know all the time but contains everything about the intranet and makes for a good reference when something needs to be changed or fixed. The training guide was written so the people at Arthritis Care would be able to read and understand it. This training guide was important for the implementation of the intranet; without it only certain people and certain training techniques would be accommodated.

The quick reference sheet was produced to give the staff of Arthritis Care an easy way to look up how to do simple tasks on the intranet. If an individual forgets how to get to a directory or how to change their profile this sheet provides them with quick, simple steps to help them solve their problem. This sheet does not go into a lot of detail, that is what the training manual is for, but does provide for concise but thorough directions.

4.0 Results and Analysis

4.1 Data

We used three different methods to gather data: a workshop, a survey and focus groups. The questions used in each method were different and the data were combined for use in constructing the training manual. The workshop gave us a sense of where the Arthritis Care staff stood on the understanding of what an intranet entails. The surveys allowed the staff and volunteers at Arthritis Care to learn what was going to be discussed in the focus groups. The data were analysed simultaneously for the three methods. We combined the data to allow us to compare the different needs and wants of each staff member for the intranet. The data also gave us suggestions on training methods and techniques we could include in the training manual to accommodate everyone at Arthritis Care.

4.2 Workshop

In conducting workshops we were able to show the distinction between an intranet and a regular home page and obtained feedback from the staff and volunteers of Arthritis Care. We accomplished this by using the WPI home page (http://www.wpi.edu), where we logged into the WPI Web Information System with our user name and password. This was a good example to show how the intranet we have designed for Arthritis Care has secured access features. We also showed the staff and volunteers how we have the ability to edit our class schedules with our password access ability. In comparison, we showed them that class lists and grades cannot be changed because our password access ability does not allow us to perform that function. Our goal

was to show the staff that documents they post could be updated and edited at any time as long as they had the proper password access ability. We received many comments and feedback from this workshop on what the staff would like their intranet to contain. The main issues are listed below.

- Clarifying the distinction between a Corporate Information Database (CID) and an intranet
- Deciding how much to include in the training packet (i.e. using Internet Explorer)
- Using a site map
- Having employee pictures included in the directory
- Understanding how regions view organisations
- Getting to other regional offices from the site
- Jumping around to get to sections that apply to them
- Linking to Internet

We conducted two workshops with three people in each. The relatively low attendance was due to conflicts with previously scheduled staff meetings. Even with the sparse attendance the demonstration worked well in showing the participants what an intranet is and how they can use it. The questions and suggestions of the participants helped the flow of the workshops and resulted in a lot of positive feedback to help us with our design of Arthritis Care's intranet.

4.3 Surveys

We distributed a survey to all the staff and volunteers at the Central Office of Arthritis Care. The survey was distributed via email that had a Microsoft Word attachment file that was readily accessible. The survey was designed so that it could be filled out quickly by giving the respondent choices to check off, see Appendix C. We sent out sixty-seven surveys through email, only forty-five opened the attachment and we received twenty-one surveys back. The results of the survey questions were organised into a bar chart and pie graphs.

The first sets of questions on the survey were related to an individuals ability to eagerly use a computer. After collecting the survey we placed the results onto a bar chart, shown in Figure 2, to illuminate the results.

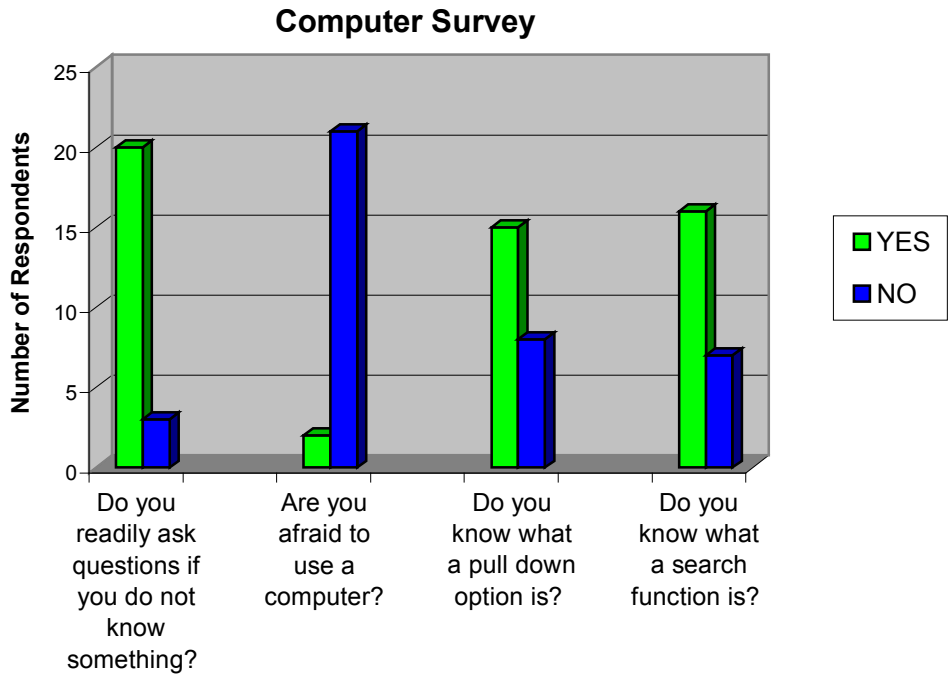


Figure 2 - The staffs' skill level and understanding of a computer

This chart shows that the majority of the staff of Arthritis Care are not afraid to use a computer and most know what the different tools are. It was important to find out if the staff felt comfortable around a computer and if they knew what the basic tools are because it provided us with the information necessary to begin creating the training manual. The majority of the staff also replied that they are not afraid to ask questions, which was vital when we are asking for feedback during the focus groups.

The second sets of questions on the survey were related to training. The questions were about what kinds of training methods and training devices an individual prefers to use. The pie graph in figure 3 displays the results of the different types of training methods the staff at Arthritis Care prefer.

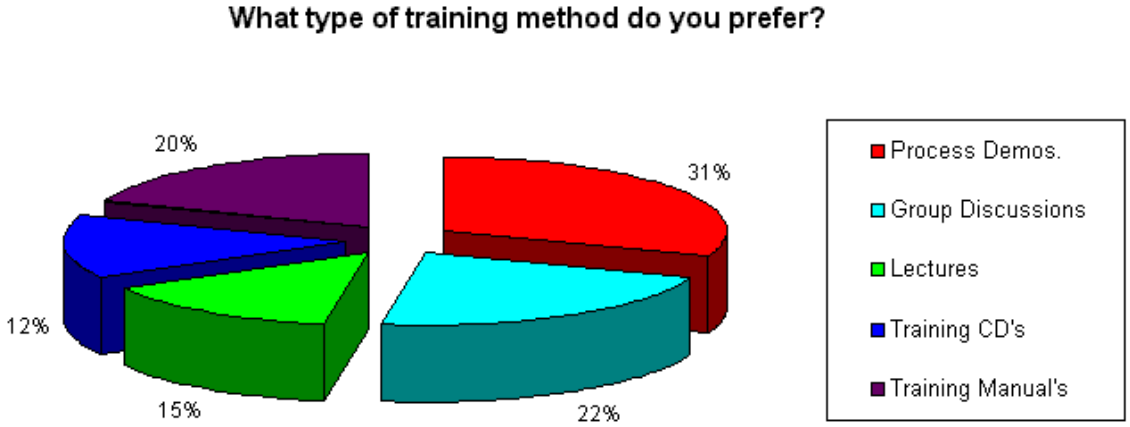


Figure 3 - Training methods the staff prefer to use

The survey did allow an individual to check off more than one training method and as shown in Figure 3, the majority of the staff that returned the survey prefer to use process demonstrations and group discussions as the chosen way of training. Although, the results about the process demonstrations and groups discussions are a little biased. We found out from the focus groups that were conducted that the reason why the staff did

not chose a training manual as a preferred method of training is because they have had bad experiences with training manuals in the past. The training manuals used in the past were too long and not incredibly user friendly. This was important information to know when designing our training manual, to make sure it was user friendly and concise so that the it would be a productive training method.

A training device accompanies a training method to enhance the learning process of the information being taught. The pie graph in figure 4 shows what types of training devices the staff of Arthritis Care like to use.

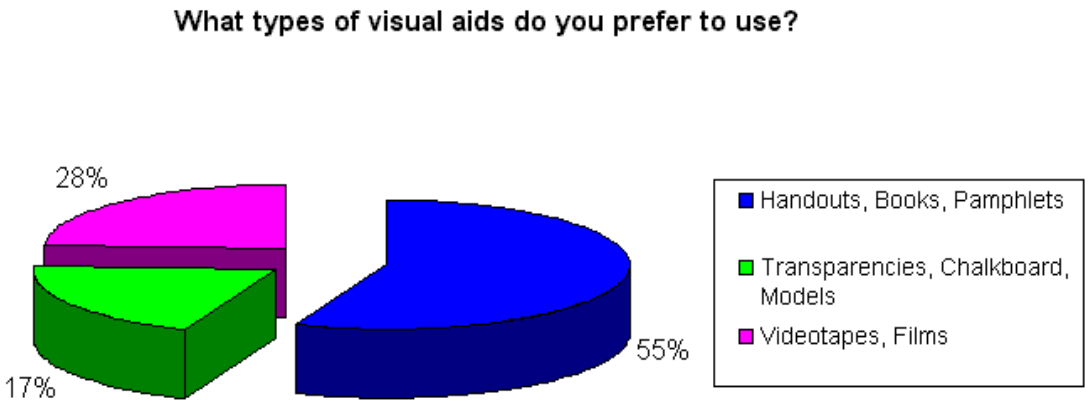


Figure 4 - Visual aids the staff prefers to use

As seen by the graph most people prefer to have a written handout or pamphlet to use as a training device to go along with a training method. From these results we have realised that the majority of the staff are visual learners. The staff not only likes to see what we are doing or showing through demonstrations, but also likes to follow along themselves, whether actually doing the demonstration with us or by looking at a handout or pamphlet. Finding out the staff likes to use a handout, it gave us the idea of creating a one-sided quick reference sheet. This quick reference sheet allows for simple questions

for instance, how to create a directory or how to upload a file, to be answered quickly without having to look it up in the training manual. By finding what works for the staff of Arthritis Care we were able to design a training manual and a quick reference sheet that was concise and user friendly.

4.4 Focus Groups

The focus groups provided us with a great deal of qualitative information that proved to be extremely useful. The focus groups we conducted gave us insight from individuals in various departments of Arthritis Care's Central Office on the prototype of the intranet. Their opinions, suggestions, and questions of various issues pertaining to the intranet helped us in refining the intranet, as well as constructing the training manual.

The first set of focus groups contained a total of twenty-three Arthritis Care staff and volunteers in four different sessions. The document repository was completed and served as the topic for demonstration at the beginning of each session. At the end of the demonstration of the prototype we preceded into the focus group discussion, which brought up several interesting issues and concerns, listed below:

INTRANET	TRAINING
Features…	
* Internal new stories	* Sheet of notes to follow (Cheat sheet)
* Clarification of departments	* Individual learning
* Minimise the number of clicks	
* Search option on the intranet to get to the Internet	* Talk in their computer language in the training manual
* Departments having linkable web pages	

* Add links to get to other pages whether on The intranet or internet
* Discussion board
* Linking addresses to MapQuest

Documents…

* Publications online
* Capacity of storage
* Sorting documents alphabetically
* Accessibility to commonly used forms at A.C.
* Documents displayed as HTML
* Index of documents

Directory…

* Master list of departments
* Updateable directory
* Include nicknames and photos
* Search ability within the directory
* Directory-post routes (zip code)
* Regional information pages
* Regional maps

Security…

* Internet hackers
* Levels of security
* Restrictions on content being uploaded
* Permission, authorization, and access to the intranet

* Fear of large training manual
* Training demonstration with manual
* Availability of what/who after training sessions
* Break up training sessions by skill level
* Small training group sessions
* Difficulties of uploading and maintenance

Table 1- Topics of discussion in focus group section 1

The second sets of focus groups were divided up into two sessions with a total of twenty-seven Arthritis Care staff and volunteers. This demonstration not only went through the document repository to show the changes that were made, but also introduced the directory. After the demonstration we broke into discussion about the intranet. These sessions discussed the intranet structure and not just what more the staff wanted the

intranet to contain. There was more discussion on the colour scheme and design of the intranet, contents of which are listed below:

INTRANET

Features on Main Page…

* Colour-code sections
* Font size
* Search engine to find out who deals with what
* Click down on topics in green side bar
* Regional departments information
* Internal job advertisement
* Trade union
* Advance search allowing to search any field
* Break down in side bar
* Regional staff access
* Colour --> instead of green use pale yellow inside the side bar
* Searchability of the entire web site
* More room between icons of directory and documents

Documents and Directory…

* Photographs in directory
* Send automatic update message
* Place date on screen when document or or directory was last updated
* Pictures in profile
* Branch office directory
* Link to email through the directory

Later improvements...

* Upgrade speed
* Audio

Table 2 - Topics of discussion in focus group section 2

After organising the feedback from both sections of focus groups, we constructed a pie graph that shows what percentage each section was talked about. The pie graph is shown below in Figure 5.

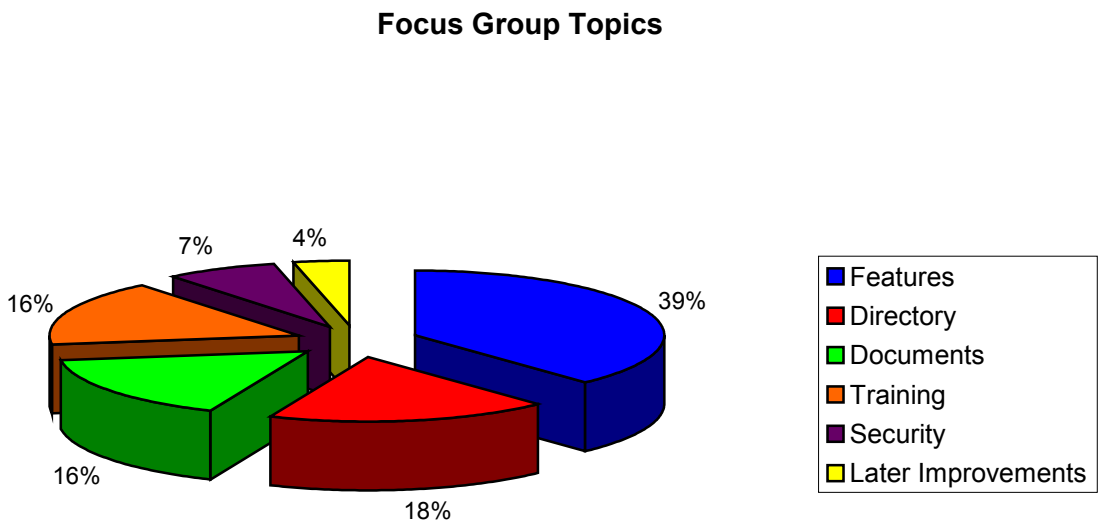


Figure 5 - Combined topic percentages of focus groups

Conducting focus groups provided us with a great deal of qualitative information that was useful in the refinement of the intranet. We were able to show the prototype and allow the staff to use it to get a better feel of how to navigate through the intranet. Conducting focus groups brought out suggestions and questions we did not think of as well as things that we used to refine the intranet. The sessions were a way for the staff to voice any concerns or thoughts about the intranet. The staff members were willing and open to participating in the sessions, but a lot of them were not sure of what actually to say. Because this type of intranet was new to the staff they were not sure what type of

feedback we were looking for in the begining. The atmospheres during the sessions were friendly and sociable and after the first sessions of focus groups the staff seemed to be more comfortable talking candidly about their personal concerns and questions. The feeling of being comfortable was important to us in conducting these sessions. If the staff did not feel comfortable talking with us openly then we would not have been able to produce an intranet that fit Arthritis Care's needs.

4.5 Training

The training manual and quick reference sheet were placed on the local central office server and in the document repository on the intranet. An email was sent out to all staff of the central office with detailed instructions on how to view the intranet and training documents. This email also explained that the topic of discussion at the focus group would be on how affective the training manual and the quick reference sheet are. The training documents were put on both the local server and the intranet, which allowed the staff easy access to the manual at any time and the ability to print out their own personal copies. Allowing the staff the opportunity to view the training manual or quick reference sheet, enabled them to use and test the intranet at their own pace prior to the meeting of the focus group.

At the beginning of the meeting we gave a demonstration on the intranet and how to get to the training manual and quick reference sheet before breaking into open discussion. This demonstration was given just in case some of the attendees did not have time to test the intranet or training manual and quick reference sheet on their own before the discussion.

The purpose of this group meeting was to receive feedback on the training manual and quick reference sheet. We found out during the meeting that a number of the staff had not yet looked at these materials, hence the quality and number of responses were not as good as we had anticipated. Busy work schedules and perhaps apprehensiveness about starting to use the intranet on their own may explain why so few of the staff had not "done their homework." Some of the them mentioned that they would rather have a full workshop or process demonstration given during the workday with someone showing them exactly how to use it. We would have conducted these workshops or process demonstrations but we did not have sufficient time or resources. The training manual and quick reference sheet were designed so that a process demonstration on how to use the intranet would not be needed. However, workshops and process demonstrations in the future will only enhance the learning of the staff of Arthritis Care.

4.6 Summary of Analysis

In any type of organisation, one of the most influential factors that determine its success is communication. It is quite important that Arthritis Care, being such a large organisation, has an easy and efficient way of communication between their staff and volunteers. The intranet was designed to improve the communication within the organisation. Many people in our focus groups stated that the communication and information exchange within the organisation needs to be improved. People expressed that too often employee information is out of date or just not readily available. Several staff members also have expressed the need for a system, which cuts back on the excess number of memos and emails they receive. In response to this, the intranet will allow for individuals to maintain their own personal information directory. This allows individuals

to change their contact, or organisational information at any time. The intranet also helps improve communication by allowing people to post upcoming meeting dates, latest news, or just any types of notes to their own department or for the entire organisation. The improved communication will allow everyone to become more productive and better serve people with arthritis.

By allowing the staff to see and be involved in the development of the intranet we were able to design the intranet to fit their needs. The analysis of our data allowed us to refine the intranet to meet these needs of the Arthritis Care organisation and the future implementation of the intranet. The data analysis also helped us make recommendations for the future maintenance and training of the staff on using the intranet.

5.0 Recommendations

Our recommendations are concerned with the future of the intranet at Arthritis Care. The recommendations are divided into five sections where each section contains our specific recommendations on the various topics. The last section is a prioritised list of our recommendations. The sections of our recommendations are listed below:

- Training
- Departmental participation
- Regional Offices
- Hosting
- Continuing development

5. 1 Training

The most important thing we feel Arthritis Care should address is to be able to maintain and update this intranet with the training of its staff. Training of Arthritis Care's staff is going to be essential to the success of this intranet. The staff needs to have the knowledge of how to use and update the intranet, so the information it contains does not become obsolete. The staff also has to be able to maintain the intranet in case something fails or breaks. The training manual and maintenance guide produced to accompany the intranet needs to be easily accessible for everyone to use. A hard copy should be on file in each department in the central office in addition to having them on the web.

5.2 Departmental Participation

Each department in the central office has an area on the intranet in which to post information about their department. It is important that each department uses this allotted space on the intranet to its full potential. This entails appointing someone from the respective departments to post, update and maintain information from their department so that it can be of use to all the staff of Arthritis Care. This responsibility must also be maintained if and when the appointee leaves the department or the organisation. Active participation is needed by all the staff members of Arthritis Care to provide the content in the intranet structure.

5.3 Regional offices

A benefit the intranet offers to the organisation is the enhanced ability to communicate information amongst the regional offices and the central office. With the intranet in place, Arthritis Care's staff from the central office and regional offices will have the ability to access confidential documents and information anywhere in the world. It is important to involve the regional offices as much as possible with the training, updating, and maintenance of the intranet. Active participation by all is needed to carry the resource to its fullest potential. Therefore a staff member should be appointed at each regional office to organise the training, updating and maintenance of the intranet.

5.4 Hosting

The hosting of the site refers to where the site is actually located. The usage of the intranet is going to be high, so the best place for housing the system should be at the central office. Arthritis Care should consider purchasing another server on which to host

the site. The server does not have to be a top of the line machine; it could be a current desktop that is being replaced. Once that machine is secured, Apache and Perl will have to be installed and configured. The system should be given a dedicated Intranet Protocol (IP) address that is not in the internal range. This can be purchased through the Internet Provider, and easily configured at the router. By doing this, the intranet will be externally accessible but will not saturate the current connection that the central office has at the web, and will greatly increase the central office's access to the intranet because the machine will be local.

5.5 Development

When we are done with the project we hope that Arthritis Care we will be able to continue to work on the intranet. Our focus groups proved that people have a desire for new features and this will require continued work on the development of the intranet. The next parts that should be implemented are: investigating the ability of linking to GroupWise and specifically the calendar feature; adding a more advanced news and discussion board section; expanding the regional section with more information; improving the search engine; and other small changes such as photos in the directory and implementing forms for orders. These are just some examples of where this intranet can be taken from this point on and we hope that Arthritis Care has the opportunity to continue this for themselves.

5.6 Prioritised Recommendation List

- Print and distribute at least one copy of the training manual and the maintenance guide to each department in the central office
- Make available copies of the training manual and maintenance guide to each regional office of Arthritis Care
- Contact department heads, at the central office and regional offices, to appoint people in their department to create, up-date and maintain their department's space on the intranet
- Encourage all staff members to take active participation in using the intranet and updating their contact information in the directory by holding workshops throughout the year
- Move hosting of intranet to the Central Office
- Continue to develop and expand the intranet

6.0 Conclusion

For our Interactive Qualifying Project (IQP), we designed an internal communications system through an intranet. This intranet was designed to help improve the internal communications not only within the central office of Arthritis Care but within the entire Arthritis Care Organisation. The IQP is a project that emphasises technology and its interaction with society. In our project we specifically designed the intranet to fit the Arthritis Care needs.

In order for us to improve Arthritis Care's communications through an intranet, we needed to get a better understanding of what kind of skill level the staff of Arthritis Care was at for using and understanding a computer. To familiarise ourselves with the skill level of the staff we conducted a workshop, distributed survey and held focus groups. The workshop, survey and focus groups supplied us with valuable data pertaining to computer skill level, information needs and wants, and training methods and techniques. We analysed the data using charts, and tables. From the charts and tables we were able to see that the staff of Arthritis Care had more computer skills than what we had anticipated and also found the training methods that the staff preferred to use which helped us implement the intranet.

Arthritis Care provides services that are extremely valuable to society. We hope that this project helps Arthritis Care improve communications within the organisation, and helps regional offices feel more connected to the central office to better serve its members. We realise that this is just the beginning of a long process of change, and we are thankful for the opportunity to help provide an intranet that will help all the Arthritis Care staff and volunteers.

Glossary

AC: Arthritis Care

Arthritis: Inflammation of the joints.

Client-server networks: Computer networks in which some nodes (servers) are dedicated to performing certain tasks in behalf of the nodes (clients) accessed by the users.

Database: A generic term referring to any conceivable medium for the storage of information and maintenance of data relationships.

Directory: A listing that includes staff and member contact information as well as job descriptions.

Document Repository: Location where various forms of information can be stored and organised

Extranet: A privately maintained intranet that is not as centralised as an intranet but allows other distant locations to collaborate over secured connections of the Internet.

Hierarchy: The classification of a group of people according to ability or economic, social, or professional standing.

Hypertext Mark – up Language (HTML): Computer language that web pages are written in.

Informantion Architecture (IA): The structure and organisation of information

Information Technology (IT): Any type of technology that organises and processes information.

Intranet: A smaller more self contained version of the internet which allows all the communication features of the internet but is not accessible to the outside world.

Internet: Transfers information over the international telephone system and connects thousands of personal computers that are on local, metropolitan, and wide area networks.

Interactive Qualifying Project (IQP): The IQP challenges students to identify, investigate, and report on a topic examining how science or technology interacts with societal structures and values. The objective of the IQP is to enable WPI graduates to understand, as citizens and as professionals, how their careers will affect the larger society of which they are a part

Local Area Network: Are useful in the transport of data between departments of an organisation.

Metropolitan Area Network (MAN): Is designed to transport data over large but localised land areas.

Qualitative: Nature and description of things, i.e. feelings, characteristics, and meanings.

Server: A networked computer usually dedicated to serving resources on a network.

Staff members: Volunteers and employees of Arthritis Care.

Wide A rea Network (WAN): It i s a ble t o c onnect t ogether l arger or ganisations, campuses, institutions, cities and countries through telephone lines.

Worcester Polytechnic Institute (WPI): College assicated with this project

References

Arthritis Care. (July 2000b). *Empowering Our Staff Across the UK: Arthritis Care's Mission in Practice*. Euston London, England: Athritis Care.

Arthritis Care. (August 2000). *Arthritis Care's UK Staff Support Handbook*. Euston London, England: Athritis Care.

Arthritis Care. (September 2000). *Arthritis Care's UK Staff Policy Directory*. Euston London, England: Athritis Care.

Arthritis Care. (July 2001). *Arthritis Care's UK Staff Induction Handbook*. Euston London, England: Athritis Care.

Berg, Bruce L. (2001). *Qualitative Research Methods for the Social Scientists*. Boston, Massachusetts: Allyn and Bacon.

Connors, Tracy Daniels (Ed). (1988*). The Nonprofit Organization Handbook*. New York, New York: McGraw-Hill Book Co.

Goldsborough, Reid. (2000). Management: Personal computing: How to teach a computer novice. *Commercial Law Bulletin* 4, 30-31.

Ilsley, Paul J., and Niemi, John A. (1981). *Recruiting and Training Volunteers*. New York, New York: McGraw-Hill Book Co.

Lakey, B., Lakey, G., Napier, R., and Robinson, J. (1995). *Grassroots and Nonprofit Leadership*. Philadelphia, Pennsylvania: New Society Publishers.

Lowe, D. (1997). *Client/Server Computing for Dummies (2nd Edition).* San Diego, California: IDG Books Worldwide, Inc.

McFedries, Paul. (1997). *The Complete Idiot's Guide to Creating an HTML Web Page*

(2nd Edition). Indianapolis, Indiana: Que Corporation.

Morville, Peter & Rosenfeld, Luis. (1998). *Information Architecture for the World Wide Web*. Sebastopol, California: O'Reilly.

Newman, I., McNeil, K., (1998). *Conducting Survey Research in the Social Sciences*. Lanham, Maryland: University Press of America

Wagner, M., and Negus, R. (1999). *The Complete Idiot's Guide to Networking (2nd Edition)*. Indianapolis, Indiana: Que Corporation.

Bibliography

Allen, Kerry Kenn. (1980). Worker Volunteering: A New Resource for the 1980's. New York, NY: AMACON.

Arthritis Care. (July 2000a). Arthritis Care's UK Staff Who's Who. Euston London, England: Athritis Care.

Arthritis Care. (July 2000b). Empowering Our Staff Across the UK: Arthritis Care's Mission in Practice. Euston London, England: Athritis Care.

Arthritis Care. (August 2000). Arthritis Care's UK Staff Support Handbook. Euston London, England: Athritis Care.

Arthritis Care. (September 2000). Arthritis Care's UK Staff Policy Directory. Euston London, England: Athritis Care.

Arthritis Care. (July 2001). Arthritis Care's UK Staff Induction Handbook. Euston London, England: Athritis Care.

Berg, Bruce L. (2001). Qualitative Research Methods for the Social Scientists. Boston, Massachusetts: Allyn and Bacon.

Bruce, Ian. (1999). In My Opinion. Management Today, 14.

Cohen, Allan R., et al. (1995). Effective Behaviour in Organisations: Cases, Concepts, and Student Experiences. Chicago, Illinois: IRWIN.

Connors, Tracy Daniels, (Ed). (1988). The Nonprofit Organization Handbook. New York, New York: McGraw-Hill Book Co.

Coronel, Carlos. (2000). Database Systems: Design, Implementation & Management (4th Edition). Cambrigde, Massachusetts: Course Technologies.

Fidishun, Dolores. (2000). Teaching adult students to use computerized resources: Utilizing Lawler's keys to adult learning to make instruction more effective. Information Technology and Libraries 3, 157-158.

Goldsborough, Reid. (2000). Management: Personal computing: How to teach a computer novice. Commercial Law Bulletin 4, 30-31.

Ilsley, Paul J., and Niemi, John A. (1981). Recruiting and Training Volunteers. New York, NY: McGraw-Hill Book Co.

Jennings, R. (1997). Using Microsoft Access 97. Indianapolis, Indiana: Que Corporation.

Janczewski, Lech. (2000). Internet & Intranet Security Manegment: Risks and Solutions. Hershy, Pennsylvania: Idea Group Publishing.

Korten, David C. (1989). Getting to the 21st Century. West Hartford, Connenticut: Kumarian Press, Inc.

Lakey, B., Lakey, G., Napier, R., and Robinson, J. (1995). Grassroots and Nonprofit Leadership. Philadelphia, Pennsylvania: New Society Publishers.

Lowe, D. (1997). Client/Server Computing for Dummies (2nd Edition). San Diego, California: IDG Books Worldwide, Inc.

Maynard, H.B., (Ed). (1960). Top Management Handbook. New York, New York : McGraw-Hill Book Co.

McFedries, Paul. (1997). The Complete Idiot's Guide to Creating an HTML Web Page (2nd Edition). Indianapolis, Indiana: Que Corporation.

Morville, Peter & Rosenfeld, Luis. (1998). Information Architecture for the World Wide Web. Sebastapol, California: O'Reilly.

Nelson, Rosemarie. (2000). Staff, this is a computer. Medical Economics 23, 27-28.

Newman, I., McNeil, K., (1998). Conducting Survey Research in the Social Sciences. Lanham, Maryland: University Press of America.

O'Hare, Frank, & Funk, Robert. (2000). The Modern Writer's Handbook. Boston, Massachusetts: Allyn and Bacon.

Plew, R., & Stephens, R. (2001). Database Design. Indianapolis, Indiana: Sam's Publishing.

Rodely, J. (1997). Developing Databases for the Web and Intranets. New York, New York: The Coriolis Group.

Spatz, Bruce M. (1992). Building an Object Orientated Database. The Story of O_2. San Mateo, CA: Morgan Kaufmann Publishers.

Thuraisingham, B. (1997). Data Management Systems: Evolution and Interoperation. San Diego, California: CRC Press.

Wagner, M., and Negus, R. (1999). The Complete Idiot's Guide to Networking (2^{nd} Edition). Indianapolis, Indiana: Que Corporation.

Yannakoudakis, E.J. (1998). The Architectural Logic Design of Database Systems. New York, New York: Springer-Verlag.

Appendix A - Contact Information

Our sponsor:

Arthritis Care

18 Stephenson Way

London NW1 2HD

Tel: 0171 916 1500
Fax: 0171 916 1505

Our liaison:

David Wright
Dave@arthritiscare.org.uk

Appendix B - Organisation Information

Arthritis Care

Mission of Agency:

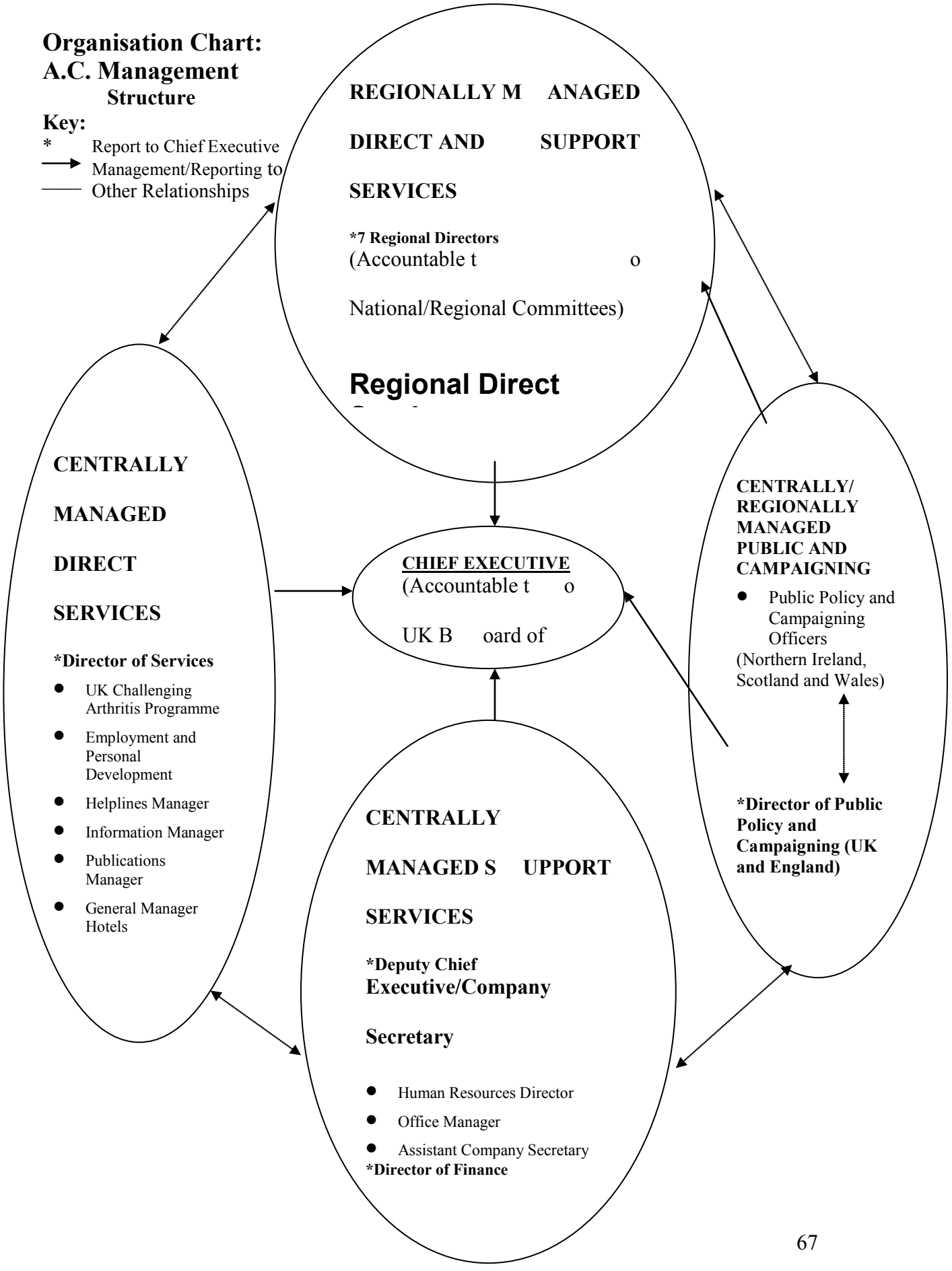
Since 1997, Arthritis Care's mission has been empowering people to take control of their arthritis, their lives and their organisation.

Size and Budget:

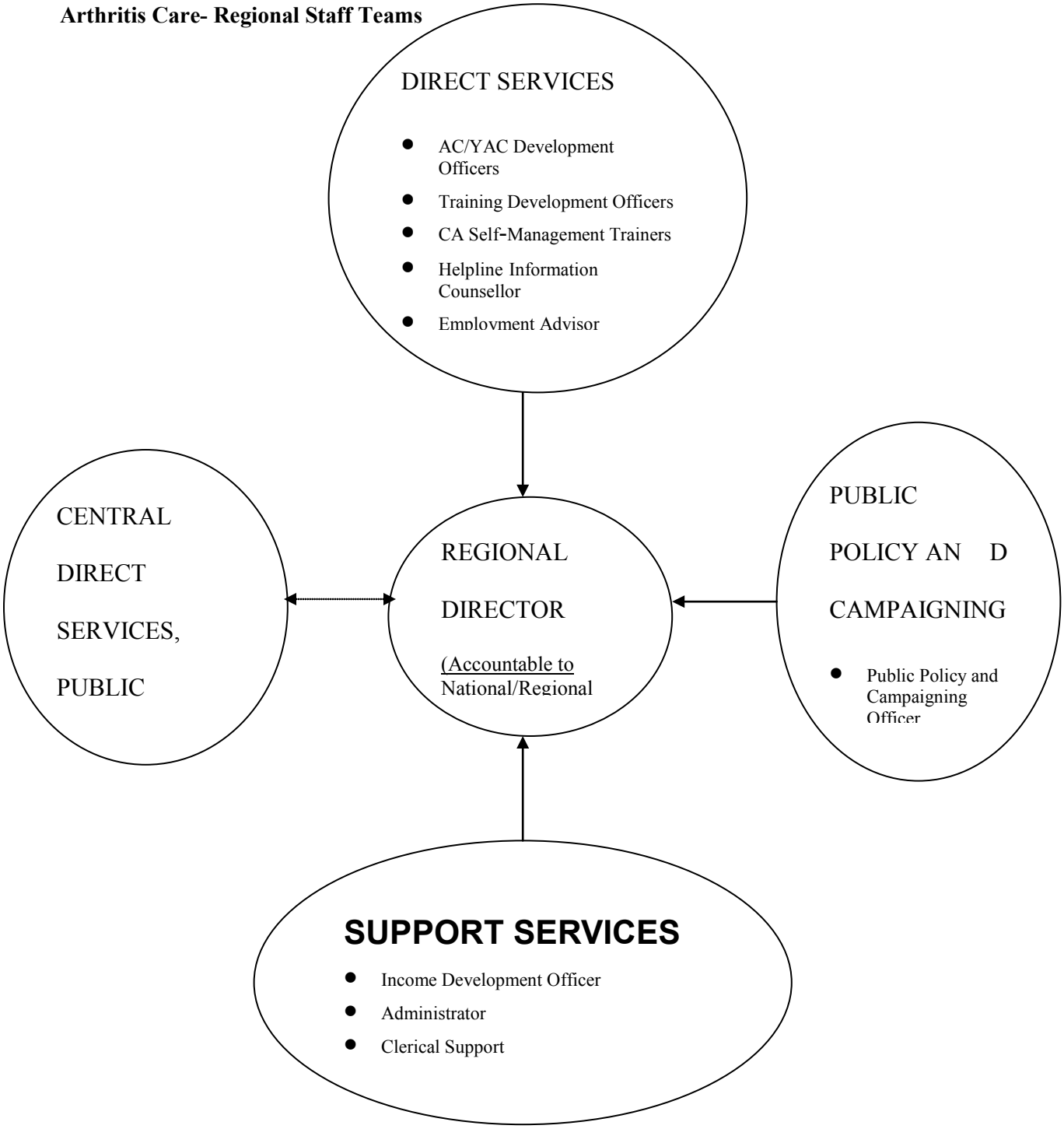
Arthritis Care is the UK's largest voluntary organisation with over 7,000 volunteers and over 620 branches throughout England, Scotland, Wales and Northern Ireland. The organisation receives very little government funding, so the survival of the organisation depends on voluntary income.

Who they serve:

Arthritis Care works with and for all people with arthritis from every section of the community.


Organisation Chart:
A.C. Management
Structure
Key:
* Report to Chief Executive
Management/Reporting to
Other Relationships
REGIONALLY M ANAGED
DIRECT AND SUPPORT
SERVICES
*7 Regional Directors
(Accountable t o
National/Regional Committees)
Regional Direct
CENTRALLY
MANAGED
DIRECT
SERVICES
*Director of Services
UK Challenging Arthritis Programme
Employment and Personal Development
Helplines Manager
Information Manager
Publications Manager
General Manager Hotels
CHIEF EXECUTIVE
(Accountable t o
UK B oard of
CENTRALLY/
REGIONALLY
MANAGED
PUBLIC AND
CAMPAIGNING
Public Policy and Campaigning Officers
(Northern Ireland, Scotland and Wales)
*Director of Public Policy and Campaigning (UK and England)
CENTRALLY
MANAGED S UPPORT
SERVICES
*Deputy Chief
Executive/Company
Secretary
Human Resources Director
Office Manager
Assistant Company Secretary
*Director of Finance

Arthritis Care- Regional Staff Teams



Appendix C – Interface News Article

interface

The bi-monthly bulletin for Arthritis Care's staff and trustees
Issue no.10 spring/summer 2001

ARTHRITIS CARE
Empowering people with arthritis.

This is IT...coming to AC

***Camilla Francis** asks what difference the internet and email have made to Arthritis Care employees' daily working life*

In this issue...

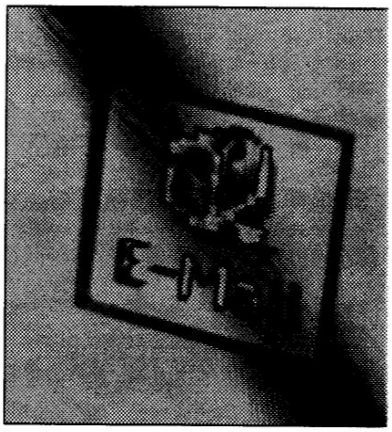
Nearly everyone had something to say about the new information technology developments at Arthritis Care, and it really was a question of extremes. For many staff these changes have revolutionised the way they work. Emily Butler in public policy and campaigning says that using the internet and email has made 'a huge difference' to her job. For Sheila Benneyworth, PA to the chief executive, email has been 'brilliant'. 'I can set up meetings a lot easier, communicate more easily and cut down on time wasting,' she says.

However, all advancements are not without their cynics or their difficulties in practical application. For Stuart Cantle, information assistant, the advent of email has made things 'a lot more messy.' He believes it will never replace pen and paper and gives people the misguided notion that you are immediately contactable: 'Emails may sit unread and people wonder why things have not been actioned', he says.

More training needed

Jo Johnson, former PA to the deputy chief executive, feels that training is needed on using internet search techniques as well as a good introduction to Groupwise, our email package, so that it can be used to its full potential.

Angela Sibbit, administrator, North England office, says the team's arrival online has been something of an IT nightmare. Since Christmas they have been beset with problems. In spite of this, Angela maintains they are keeping their 'email sense of humour' and admits that, when things work, 'it is brilliant.'



Dave Wright, information and office systen manager, acknowledges that there has not bee sufficient IT support but stresses that the depar ment needs to know when there is a problem. H says an evaluation process is necessary to find ou where people are with their IT skills – and whe they want to be.

There are interactive CDs and books in centr and regional offices but staff need to know the are available and make use of them to make th most of the new developments.

In terms of future developments, one projec that is currently underway includes a corpora information database or 'intranet,' organised fc Arthritis Care by a group of American consu tants (see information, page 7).

Monitoring staff use of email and the interne is not highly developed yet according to Dav but let's just say that big brother is watching yo

Information page

US visitors provide intranet

A project to develop an intranet or corporate information database (CID) is currently underway at Arthritis Care. The intranet will be used to distribute internal documents and will work in much the same way as the internet, but using a local network which is not publically accessible.

Arthritis Care is hosting four consultants from the Worcester Polytechnic Institute, Massachusetts, in the United States, to develop the product. The aim will be to improve and promote communication and information flow throughout the organisation. The intranet will be password protected and accessible to all key staff, trustees and, in time, appointed volunteers.

Its success will be the ease at which users can access and maintain the data which forms the core of the intranet or CID. The consultants have been hosting focus groups with both users and stakeholders, to explore how the intranet will be of maximum benefit and, crucially, how they can learn how to use and develop the database.

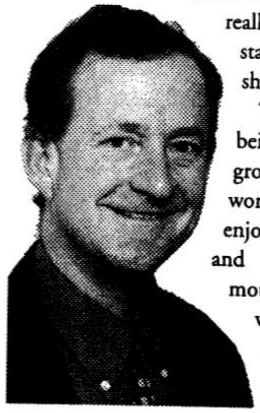
An induction training pack focused to the needs of new users, and providing guidance for developmental work will be produced.

If you have not yet been involved in the project, and would like to be, please contact them by email or by telephone on 020 7380 6507.

Farewell Richard

This is my last issue of *Interface*. I remember soon after I started at Arthritis Care in 1992 the staff very daringly produced a newsletter called *Not Arthritis News*. Apparently it was the first time the staff felt they had permission to express their own views.

Well, I'm glad to say that is not longer an issue. Interface provides a really good vehicle for staff and trustees to share ideas and views.



Thank you for being such a great group of colleagues to work with. I have enjoyed your support and friendship enormously and I know I will miss you. Good luck to each and every one of you.

MSF membership

MSF represents over 400,000 skilled and professional people. They spend their time representing members in areas such as health and safety, unfair treatment and redundancy.

As a member of MSF you are entitled to free legal advice on any issue. You can get discounts on insurance and holidays, credit cards with low annual percentage rate and a full range of services from the trade union bank, Unity Trust.

Your MSF representatives in central office are Christine Stewart and Emily Butler, and in the nations, John Bissel and Tom Scott.

This year, MSF will look towards updating the Job Security Agreement between MSF and Arthritis Care and will continue to represent our members on a range of issues including health and safety, working from home, and pay issues.

If you would like more information please contact: Emily Butler: 020 7380 6548, Christine Stewart: 020 7380 6551, John Bissel: 01384 895 242, Tom Scott: 01555 751 120.

Information alert

A reminder to you all that Stuart and I staff the information department in central office. We maintain a library of literature to help everyone and anyone who has a question concerning arthritis. If we do not have a certain document in the collection we can search for it in other collections and get copies... so there should be no reason not to ask us to try.

We also work closely with all departments that have projects requiring help with organisation or generation of information. For example, we have just finished a project for the training department researching and creating UK bibliographies for the two main US-based books used in their work, so that attendees of the sessions can buy books available in this country to help them with topics.

Another on-going project is the *Current Awareness Bulletin* that has been reinstated after a brief hiatus. This is the information department's monthly opportunity to let staff know what newsworthy and recent literature has been published on and around arthritis.

Project work is run concurrently with our day-to-day information support role to help enquiries. Don't forget we are also here to support your personal information needs so do not hesitate to contact us if you need some information on arthritis for yourselves.

Lizzie Eastwood, information manager

Appendix D – Survey Questionnaire

1. Do you readily ask questions if you do not know something?
   - ❑ Yes
   - ❑ No

2. Are you afraid to use a computer?
   - ❑ Yes
   - ❑ No

3. Do you know what a pull down option is?
   - ❑ Yes
   - ❑ No

4. Do you know what a search function is?
   - ❑ Yes
   - ❑ No

5. What type of training method do you prefer to use? (Tick as many as you want)
   - ❑ Process demonstration
   - ❑ Group discussion
   - ❑ Lecture
   - ❑ Training CD
   - ❑ Training Manual

6. What types of visual aid do you prefer to use? (Tick as many as you want)
   - ❑ Handout, Book, Pamphlet
   - ❑ Transparency, Chalkboard, Model
   - ❑ Videotape, Film

7. Do you prefer to participate in the training session or just listen?
   - ❑ Participate
   - ❑ Listen

8. Would you rather have someone show you how to use the intranet or would you prefer to try it on your own with an instructor there just in case you have a question?
   - ❑ Someone show me how
   - ❑ Try it on my own

9. Do you like to have an instruction sheet to follow along with?
   - ❑ Yes
   - ❑ No

10. Please write what you feel is the most the important information that needs to be included in the intranet.

_____
_____
_____
_____
_____

Appendix E – Screen Shots of Intranet

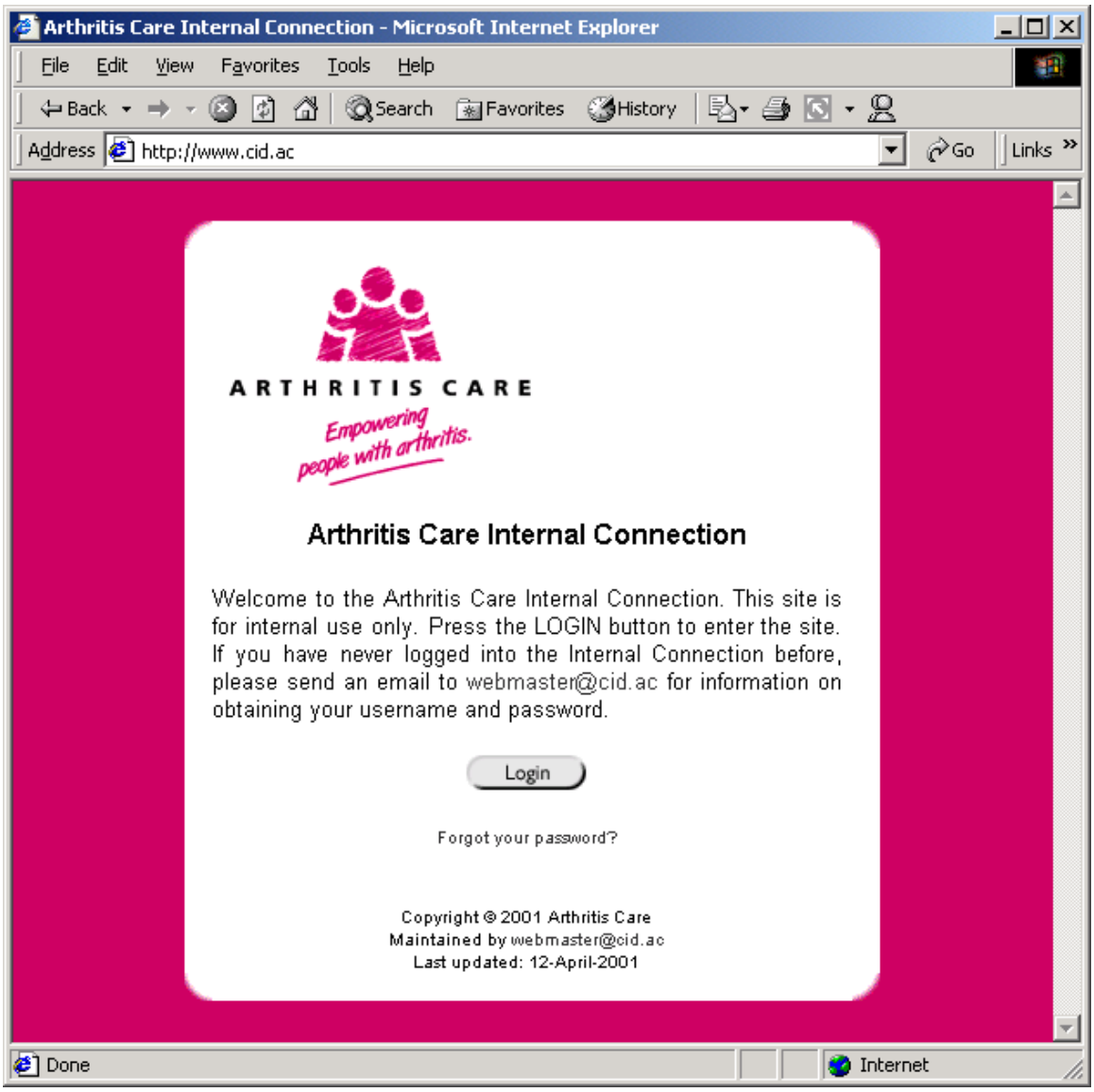


Figure 1 - Internal Connection Login Page

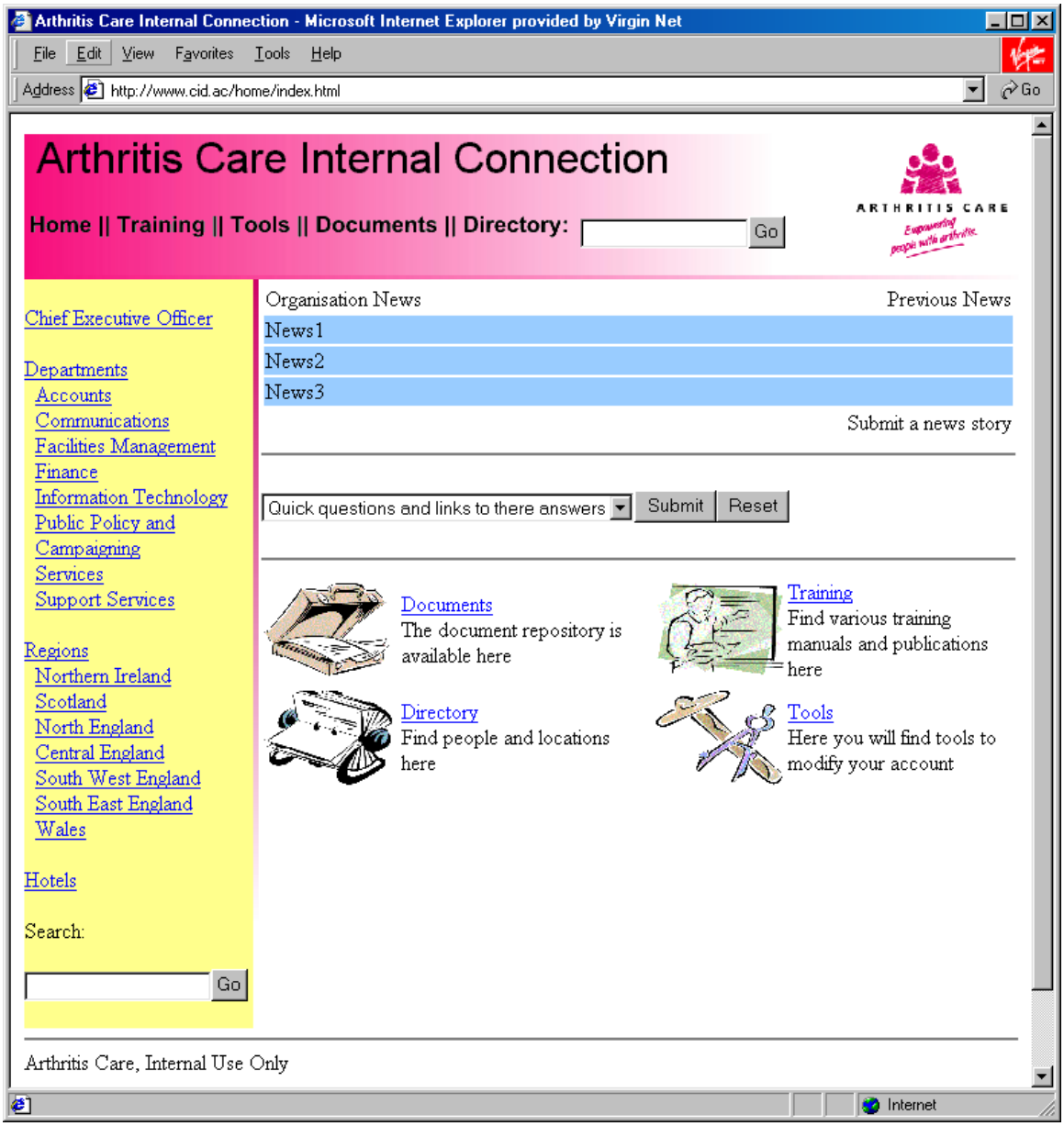


Figure 2 - Internal Connection Main Page

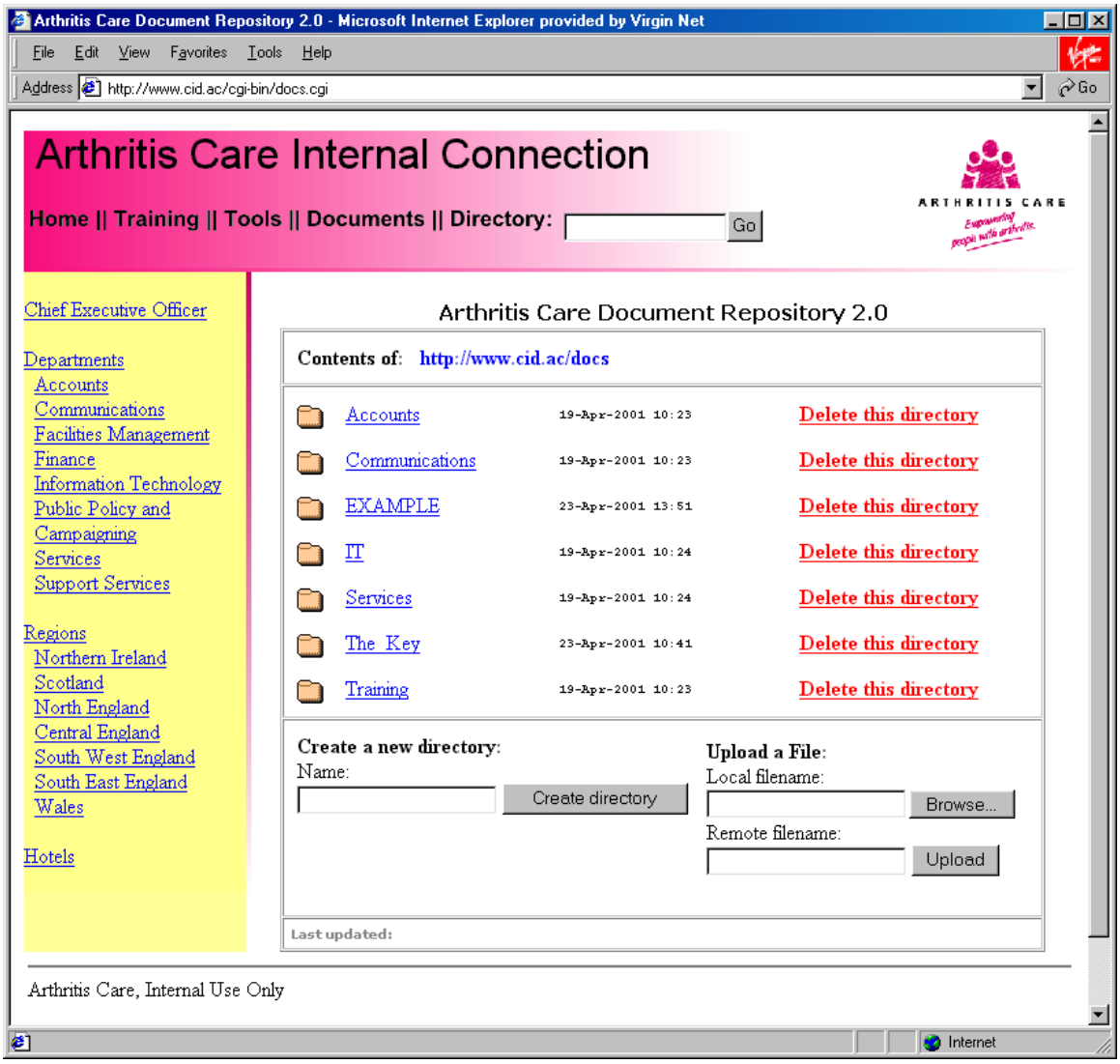


Figure 3 - Document Repository

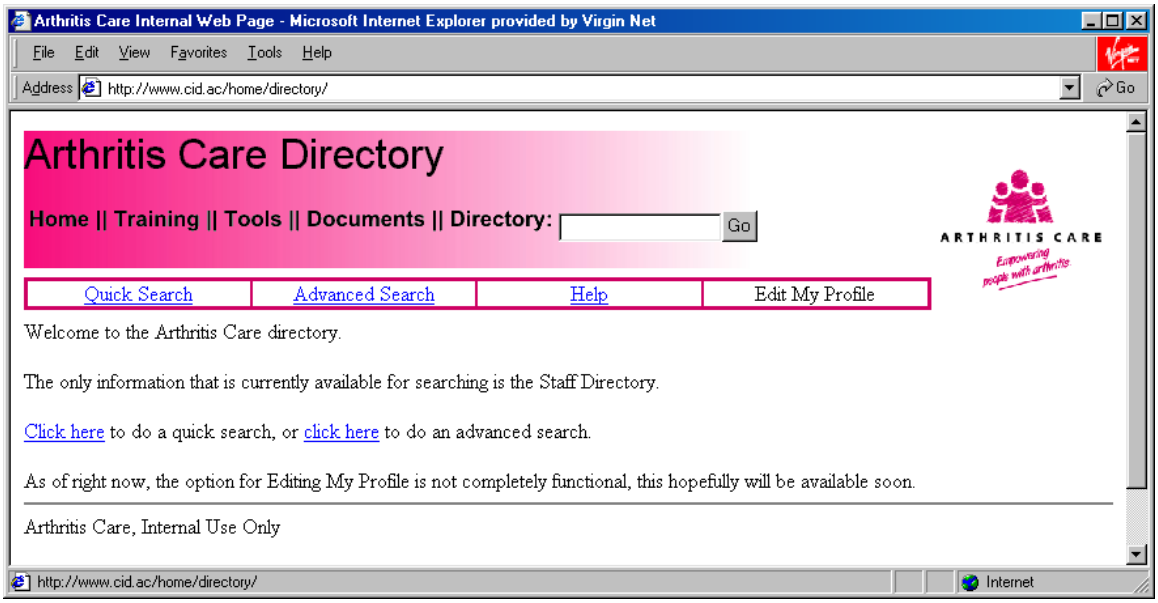


Figure 4 – First Page of the directory

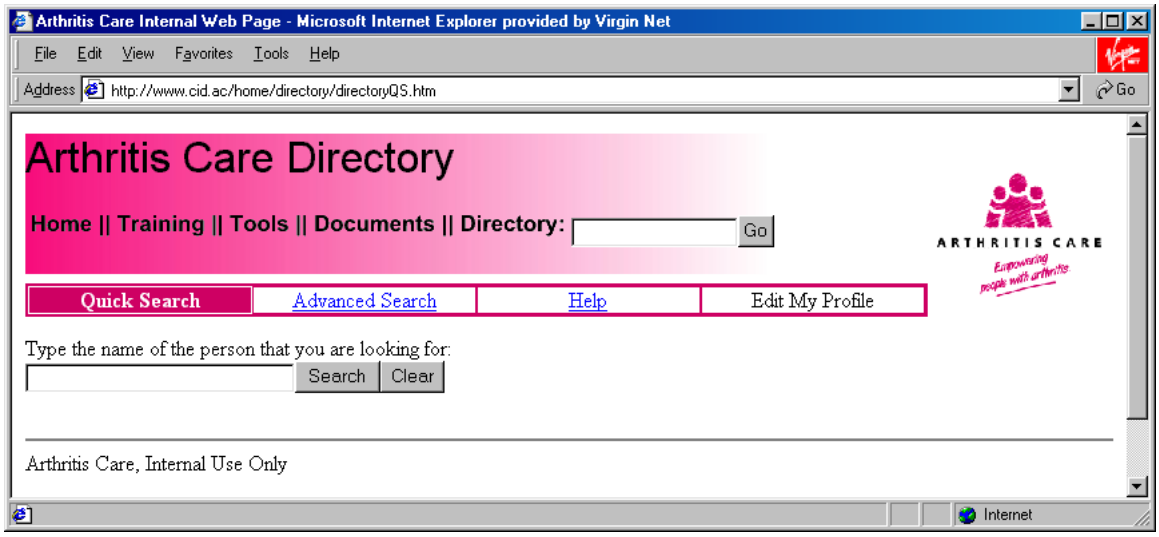


Figure 5 - Directory quick search

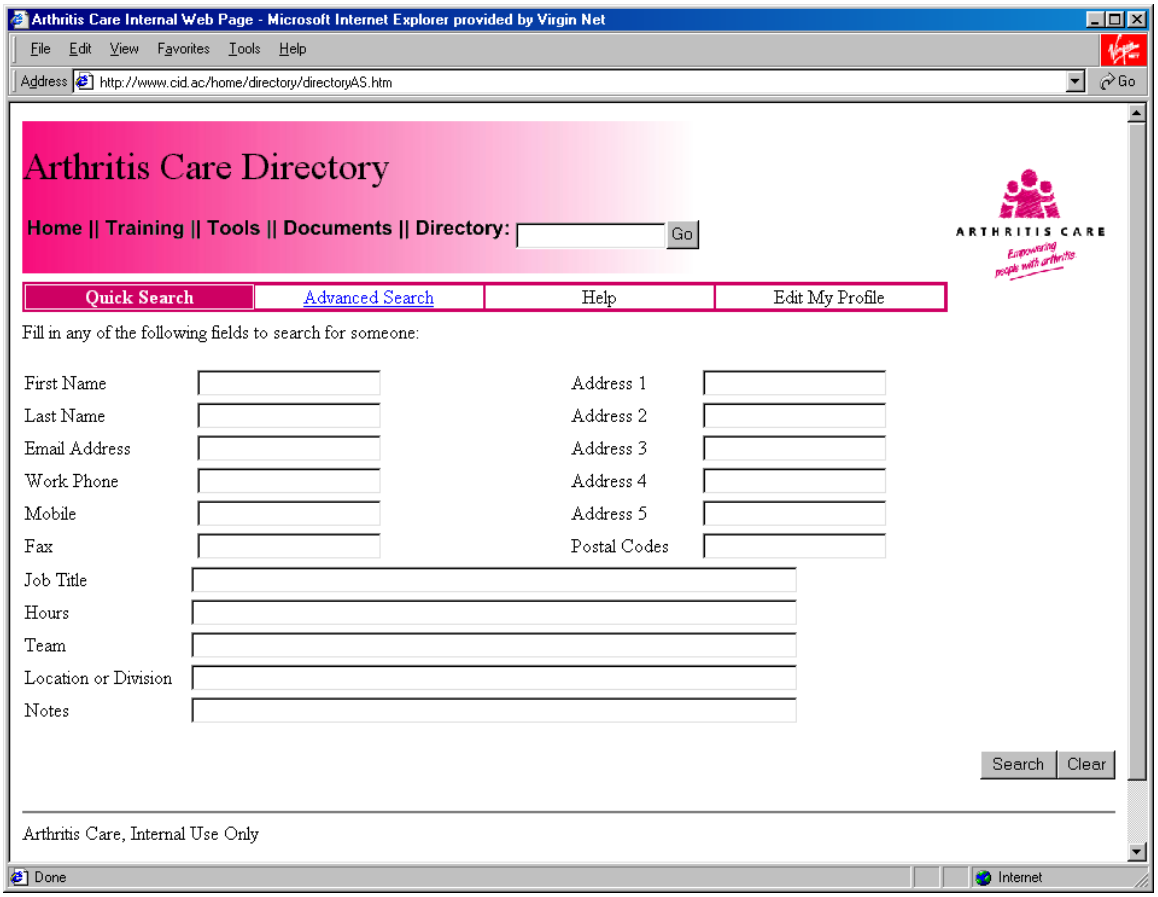


Figure 6 - Directory advanced search

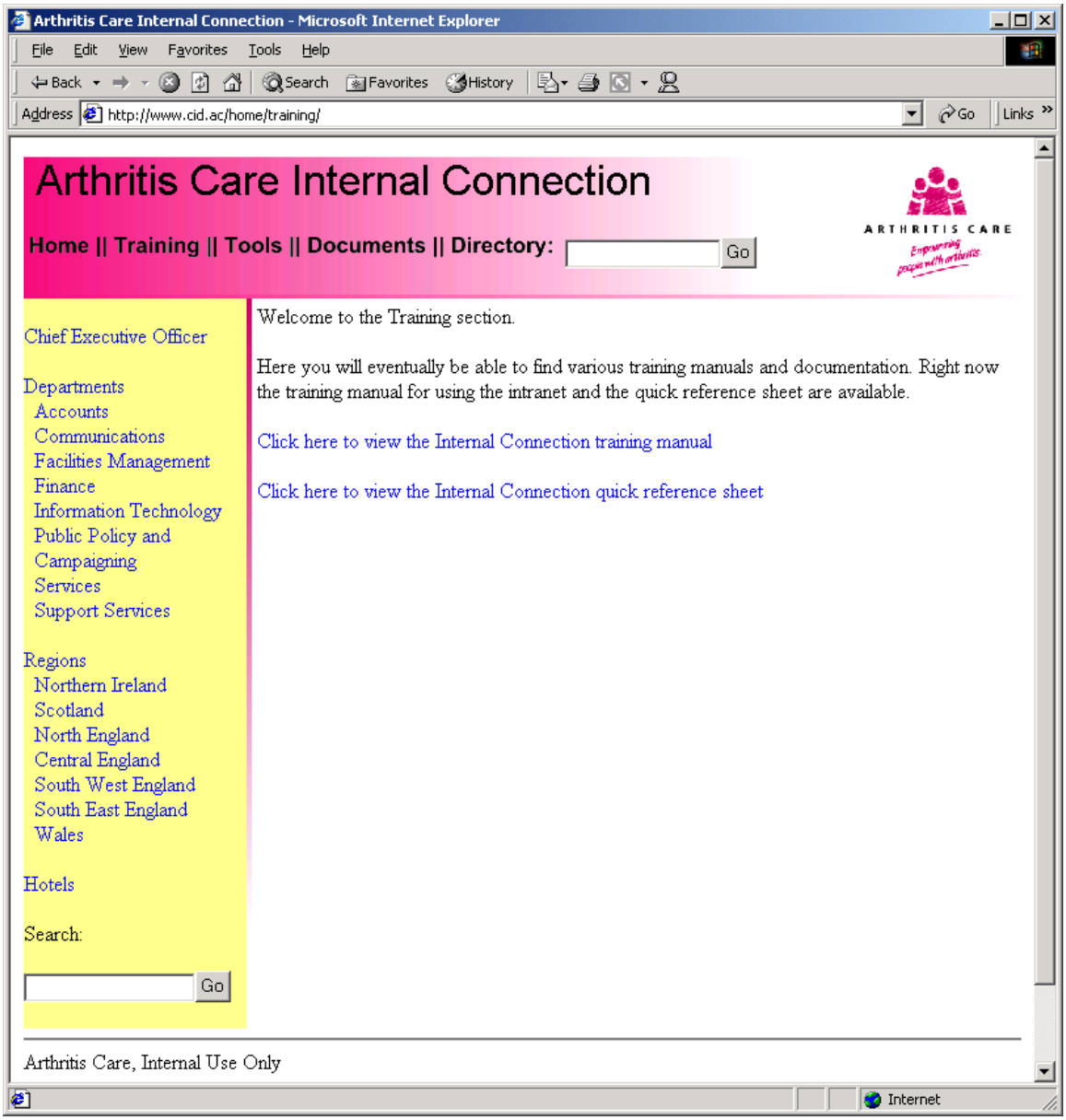


Figure 7 – First page of training

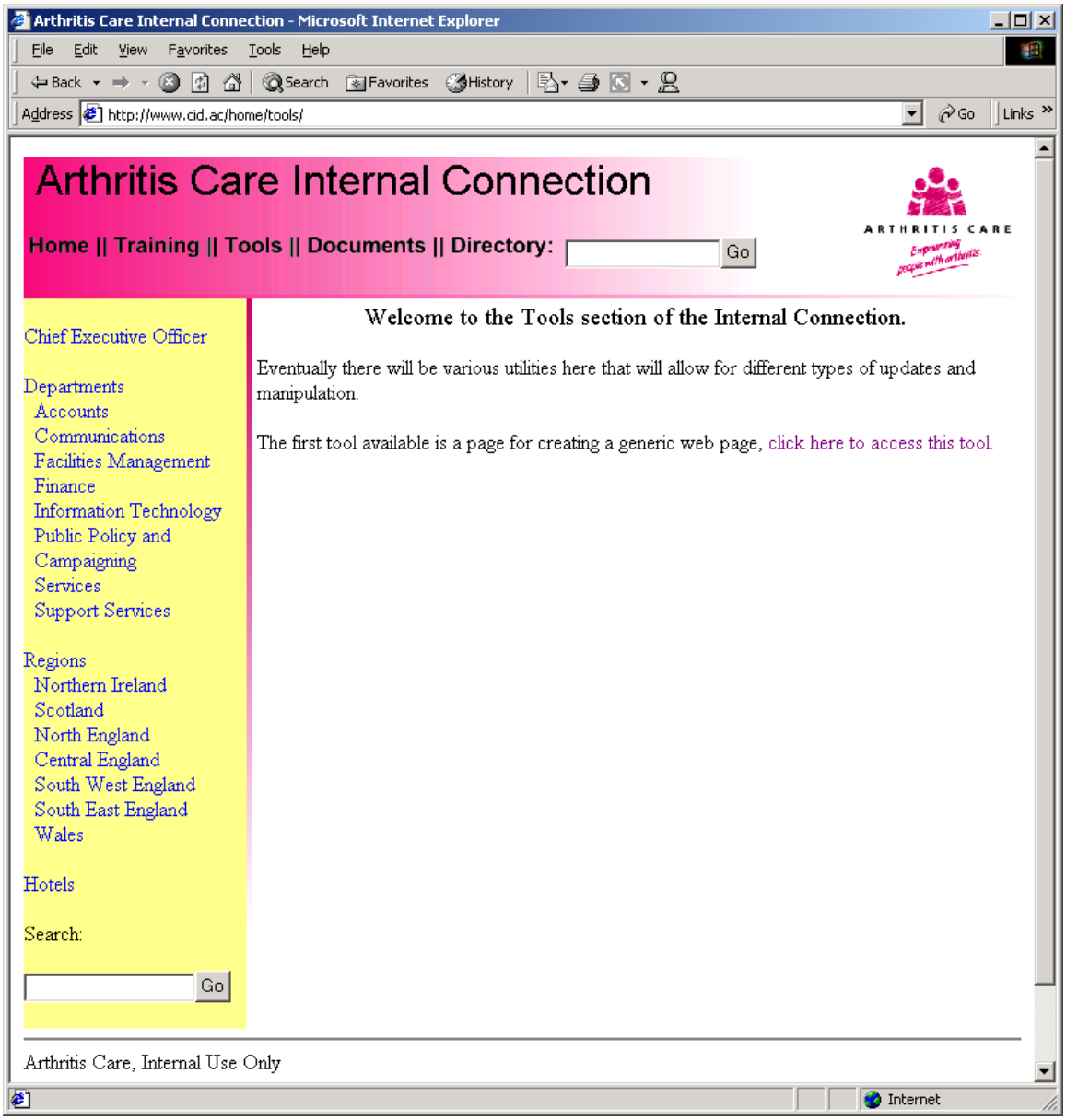


Figure 8 - First page of tools section

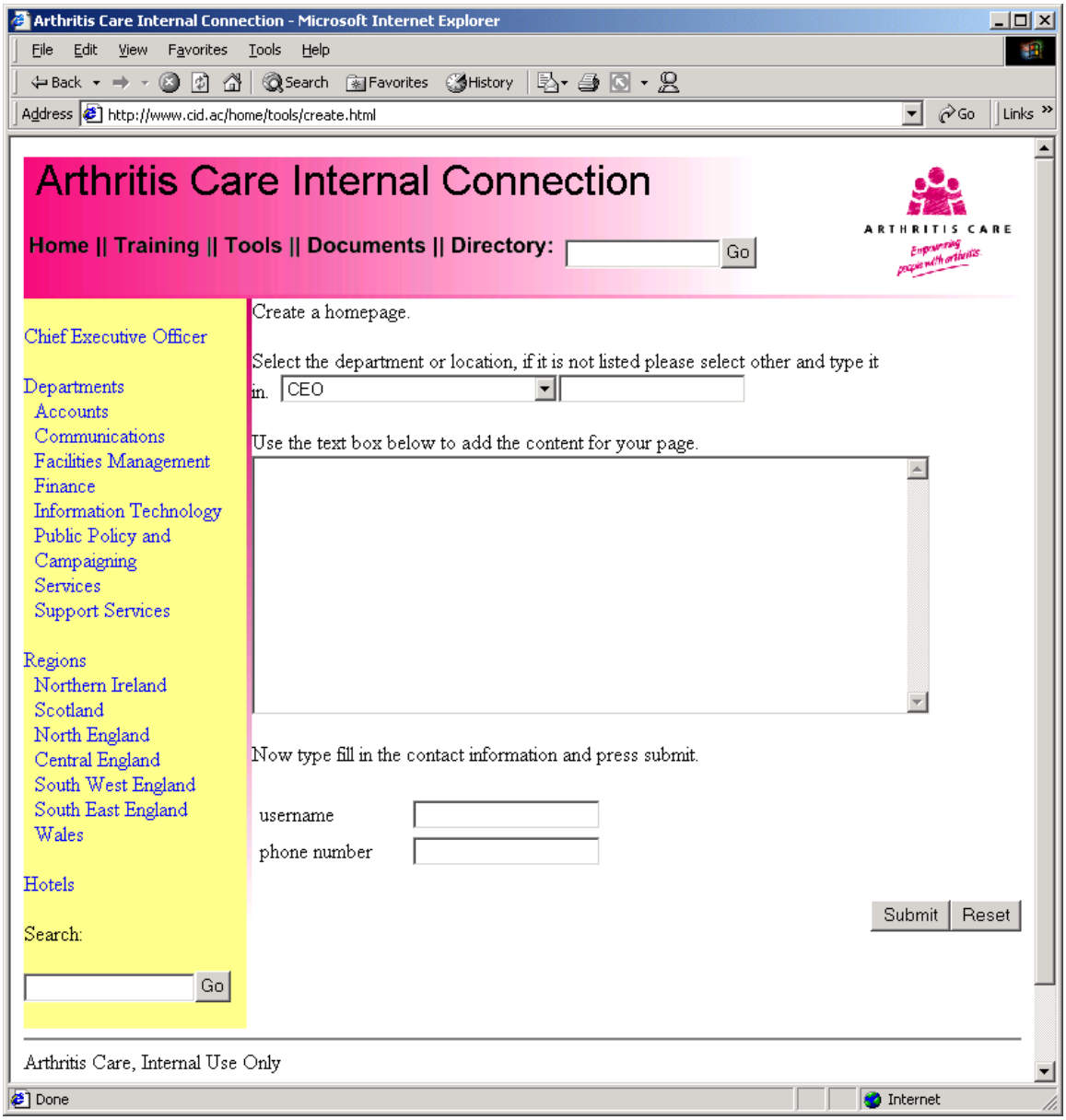


Figure 9 - Tool to create a web page

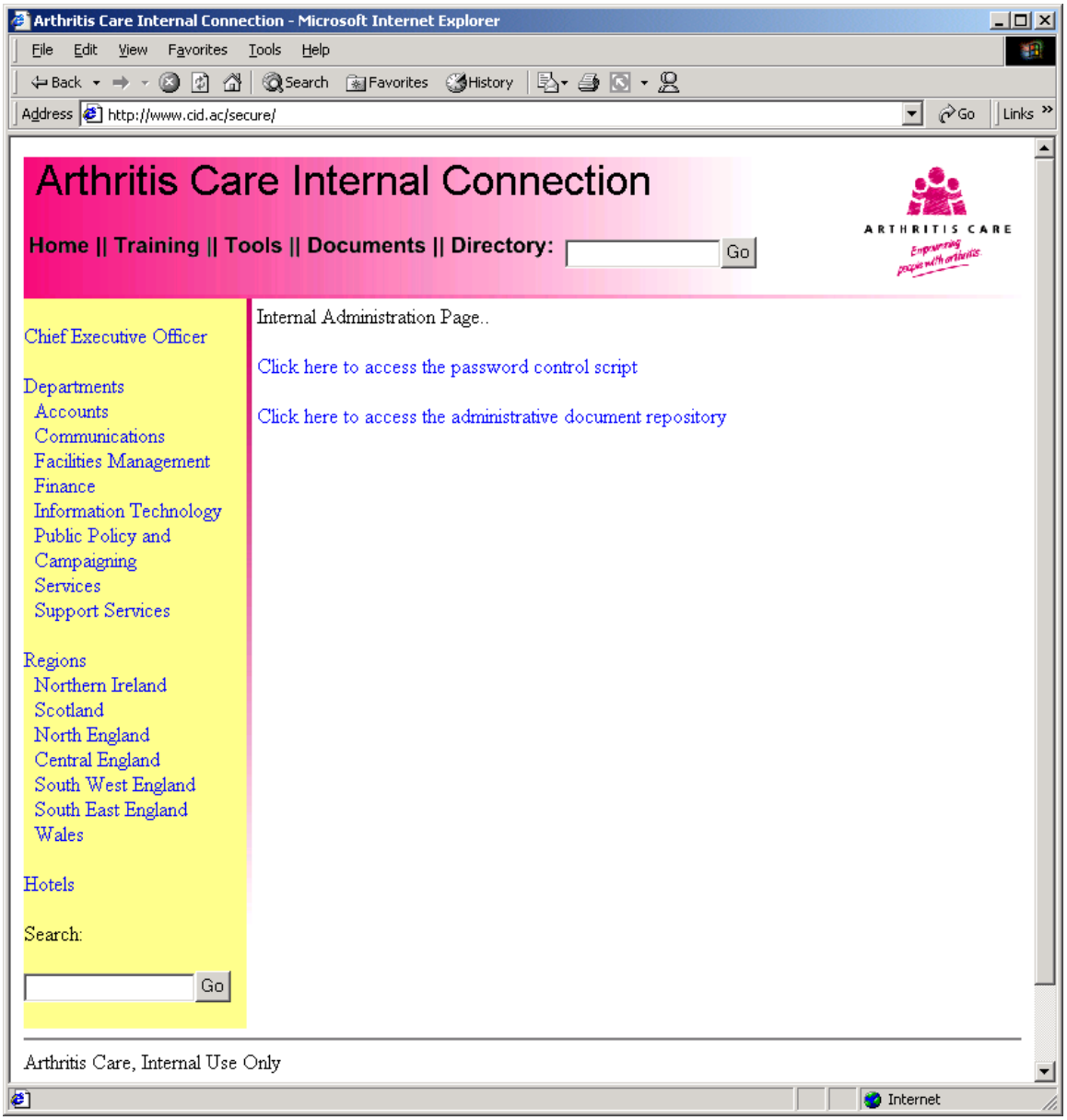


Figure 10 - Administrative main page

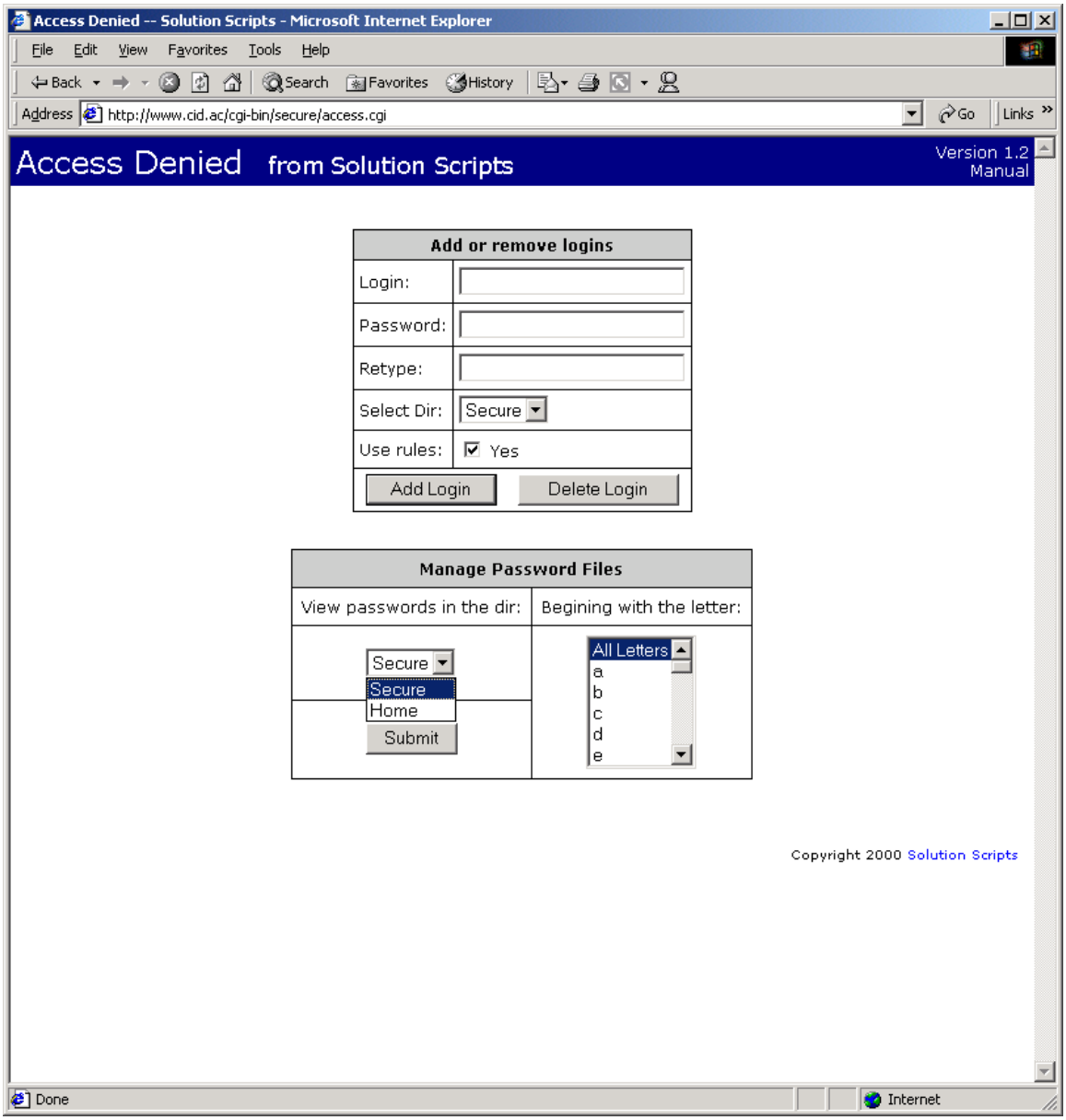


Figure 11- Password management system

Figure 12 - Administrative document repository

Appendix F- Quick Reference Sheet

ARTHRITIS CARE INTERNAL CONNECTION

QUICK REFERENCE SHEET

How to access it:
Open your web browser (Internet Explorer) and type http://www.cid.ac into the text box below then click the Go button or press the enter key.

How to log in:
When prompted type in your user name and password, then press the enter key to log in.

What can be found in it:

Document Repository Directory Training Tools

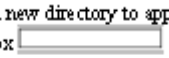
Central office departmental information Regional office information Current Arthritis Care news

DOCUMENT REPOSITORY

To access it click on **Documents** or its icon  found on the bottom of the main page.

After clicking, the main page of the Document Repository will appear with a list of directories that have been created.

 EXAMPLE

To create a new directory to appear in this list, enter its name in the text box  (that is on the bottom left of the page) and click on **Create Directory**  .

To upload a file to a directory, click on the chosen directory from the list and click on  . Choose the file you wish to upload from the screen that immediately appears and click on **Open**. The **Local filename** and **Remote file name** will automatically appear in the text box. (At this point the Remote file name can be changed to your preference. Be sure to include the appropriate extension to the filename, such as **.doc** for a word document) Click on  and the file will be uploaded and appear in the chosen directory.

To create a directory within a directory (sub-directory) click on the directory from the main list and enter the name of the new directory in the text box. Click on the **Create Directory** button beside the text box and the new directory will appear within the chosen directory.

To upload a file to the sub-directory, click on the sub-directory and follow the above steps for uploading a file to a directory.

Files and their extensions:

 Word file **.doc**  Access file **.mdb**

 Excel file **.xls**  PowerPoint file **.ppt**

Delete / Rename click on:

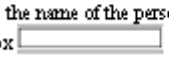
Delete this directory

Rename this file

DIRECTORY

To access it click on **Directory** or its icon  found on the bottom of the main page.

After clicking here, the main page of the Directory will appear with the **Quick Search**  and **Advanced Search**  features available to click on.

To do a Quick Search type the name of the person that you are looking for into the text box  and click on the **Search** button  . If the search was successful the name of the person will appear and can be clicked on to view that persons contact and organisational information.

NOTE: The **Quick Search** option allows you to search the directory by first name, sir name, misspelled name, or exact spelling. However, the quick search will generate a larger list of names when only part of the name is entered. This is shown below:

Results for typing in the letters Rac.	**Results for typing in the name Tracy.**	**Results for typing Charles Rackett.**
Rachel Johnson	Tracy Dudley	Charles Rackett
John Puract	Tracy Billings	
Charles Rackett		
Tracy Billings		
Rachel Wickett		
Tracy Dudley		

The **Advanced Search** allows you to search the Directory for a person based on any form of contact and organisational information that is on their profile. A search can be done using one, or more than one form of the available contact/ organisational information.

To search for staff members at the central office, enter Central office in the **Location or Division** text box. This will generate a list of everyone who works at the central office. To search for staff at the central office whose sir name is **Rackett** type both the name and location into the appropriate text boxes and click on **Search**.

ADDITIONAL MAIN PAGE FEATURES:

Click on **Training** or its icon  for tutorial information.

Click on **Tools** or its icon  for tools to modify your account.

Appendix G – Training Manual

ARTHRITIS CARE

INTERNAL CONNECTION

ARTHRITIS CARE

Empowering people with arthritis.

TRAINING MANUAL

Arthritis Care Internal Connection Training Manual

1st Edition

Edited by:

Helene Gwizdak

Peter Vitello

Nicholas Williams

Stephen Worsham

Worcester Polytechnic Institute

Worcester, Massachusetts

April 2001

Table of Contents

1.0 INTRODUCTION

This Training Manual for the internal website, also known as an intranet, is aimed towards people who have not used or dealt with any type of intranet before. Arthritis Care's internal communication system has been inhibited due to its large size and complex structure. This intranet is going to help in reducing some of the communication problems of Arthritis Care as well as help in the reduction of expenditures and wasted resources for the organisation. The training manual introduces the fundamental tools and knowledge of an intranet. It will answer questions of how to use it and how to update it. Upon completion of this training manual, learners should be able to use the intranet effectively. This manual is designed to optimise the quality of learning by:

- Enabling learners to learn at their own pace through comprehensive practical experience
- Adapting to learners with different experience levels.
- Making learner's feel their time on the induction pack is well spent most importantly well worthwhile.

After completion of the training manual, learners should feel that they have learned a great deal, that their time has been worthwhile and that they will now be able to use the intranet efficiently and effectively.

2.0 INTRANET

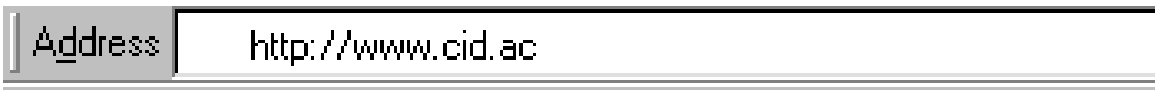
Information can be transferred from one computer to another by means of a connection. This is known as networking. As the number of networked computers and area networks became connected to each other throughout the world, the internet or "World Wide Web" emerged. The internet connects thousands of personal computers in offices, homes and schools, allowing for information to be shared between them. With the understanding of what an INTERnet is, an INTRAnet is only slightly different. An INTRAnet is different from an INTERnet in that it is a smaller, private network that is controlled by a business, school or organisation and its use is intended only for those authorized to use it. An INTRAnet is not a program or application, like the Group Wise e-mail system, and does not have to be installed on a computer. An authorized user can connect to the INTRAnet from any computer in the world provided it has access to the INTERnet. Aside from restricted access, the INTRAnet is almost identical to the INTERnet. All the communication features of the INTERnet such as: worldwide accessibility, web browsers, and exchange of information are the same for an INTRAnet.

3.0 HOW TO USE A SEARCH ENGINE

One of the ways you can get to the Arthritis Care Internal Communication system is by using a search engine. This section will briefly explain how to get to the internal system and move within it using the search engine called Internet Explorer as an example. For a more comprehensive understanding of search engines and web browsers refer to the Microsoft Internet Explorer training manual**: ECDL Syllabus3.0 Microsoft Internet Explorer 5.5 for Windows**.

3.1 ENTERING A WEB ADDRESS

Below is the space where a web address can be entered. Click on the space with the mouse and use the keyboard to enter in the web address.



3.2 TOOL BAR

Shown below is the Internet Explorer tool bar. Basic buttons like: search, forward and backward arrows and the stop button can be used to move within the Arthritis Care Internal Communications system.

Search button

Pressing the search button will go to the web address that is typed in at the address box.

Forward and Backward arrows

To the far left, there are two large arrows. The arrow pointing to the left is to go back to the previous page and the arrow pointing to the right is to go to the next page.

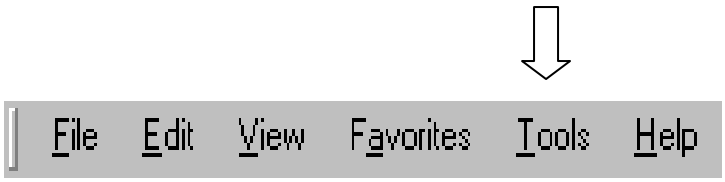
Stop button The circle with an X in the centre is to stop the browser from loading a new page or information.

3.3 HOME PAGE

Web browsers, such as Internet Explorer, have the option of setting a particular web site to automatically open once the web browser has been opened. This is known as your **Homepage**. The following steps will show you how to set the Arthritis Care Internal Connection web site as the **Homepage** on your computer.

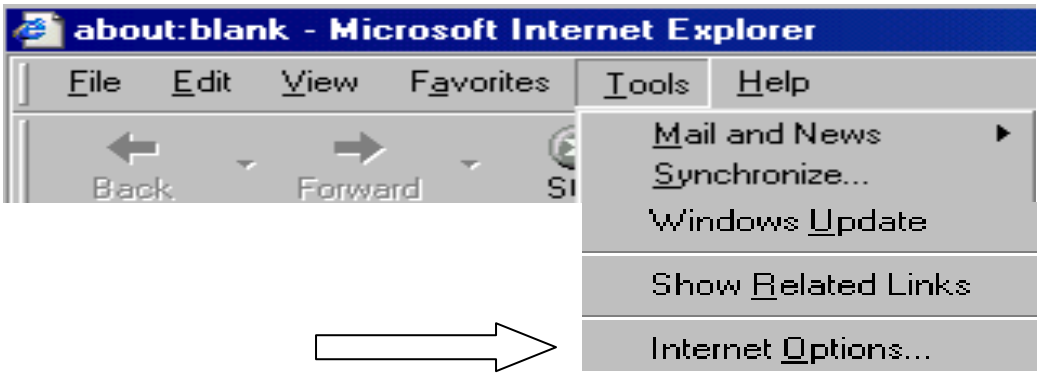
The first step is to open Internet Explorer. Shown below is the heading that appears at the top of the page when the Internet Explorer web browser has been opened.

Under the blue band there are six options. Click on **Tools** as shown below.

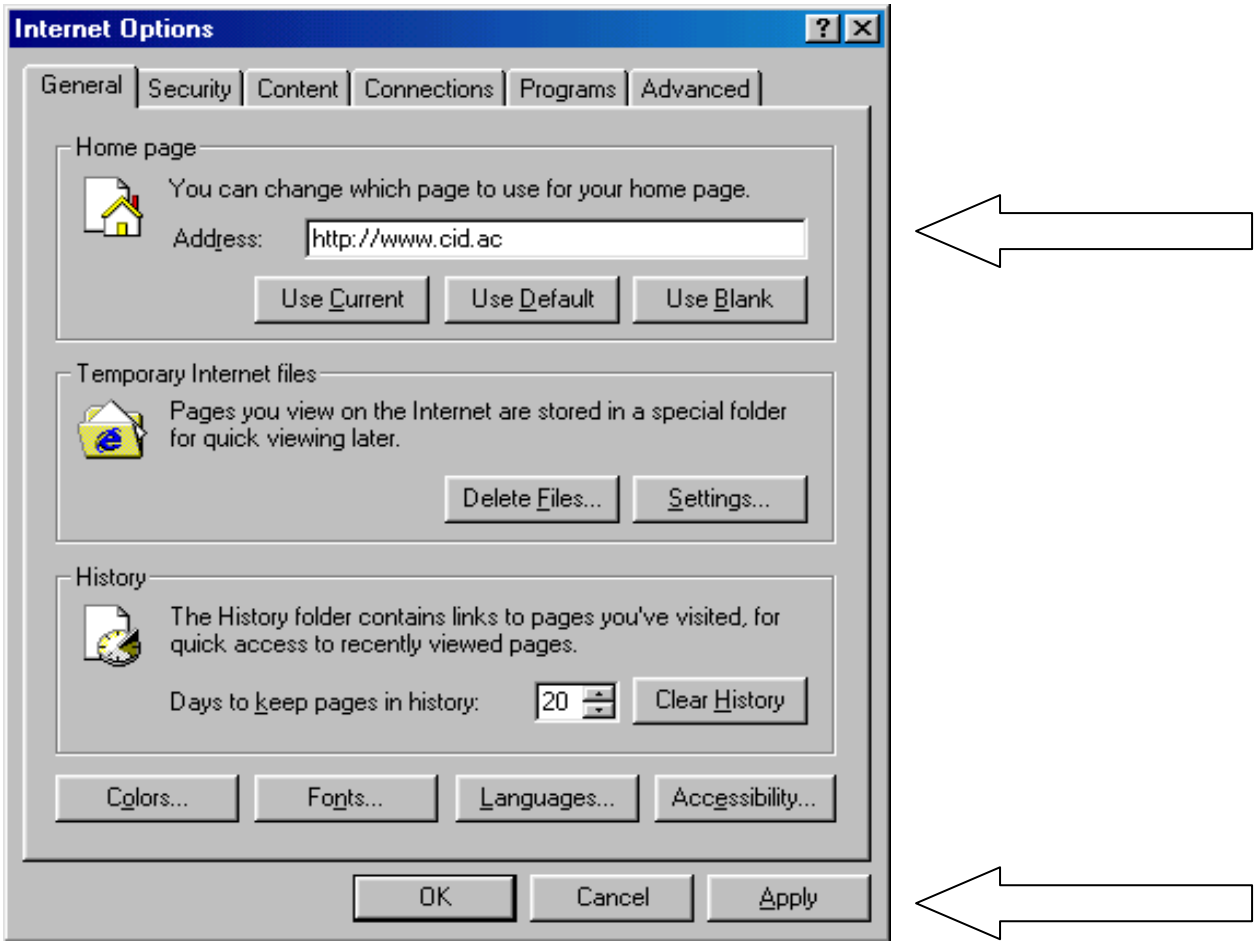


A new list of options will appear directly under **Tools** when clicked as shown below.

Click on **Internet Options** from this list.



Clicking on **Internet Options** will open a new box as shown below. On the top there is a heading titled **Home page** with a **house** icon. To set the Arthritis care Internal Connetion as your home page click on the white space across the word **Address** and type in the web address **http://www.cid.ac**



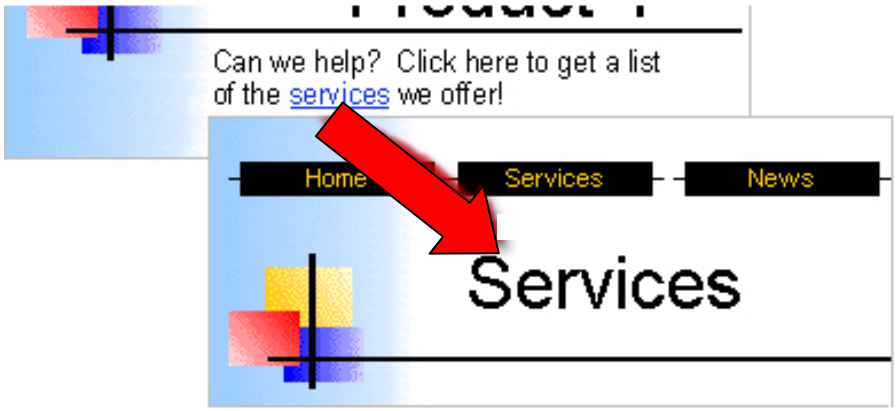
Once the web address has been entered click on the **Apply** button for the changes to occur on your web browser. Close Internet Explorer and then reopen it to verify that the change has been made.

4.0 LINKS AND ICONS

Links and icons are important features that enhance the usability of the internet. Both links and icons are visual items that serve as gateways from one place to another on the internet. Section 4.0 describes what links and icons are and details the ones used in the Arthritis Care Internal Connection.

4.1 LINKS

A link is a word or phrase on a web page that is usually in a different color, bold print, and/or underlined. As seen here, the word services is blue and underlined. This link serves as a connection from one page to another destination. This new destination can be another web page, another location on the same page, a picture, e-mail address, a document or a program. The picture below shows how clicking on the services link opens a new web page titled services.



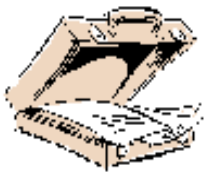
Many links can be placed on a single web page and are very easy to recognise. Links allow the user to access information on the web quickly and easily from the original web page without the effort of accessing another page entirely. Links make searches through the web easy and efficient.

4.2 ICONS

An icon on a computer screen is a picture or graphic that represents an available function. Icons work the same way links do. Using the mouse to click on an icon, the user can get to a new destination, whether it is a new page or document. Sections 4.2.1 and 4.2.2 discuss the icons that are used in this intranet and the purpose they serve.

4.2.1 MAIN PAGE ICONS

Listed below are the icons on the main page that connect to the **Document Repository**, **Directory**, **Tools** and **Training** areas of the intranet.



This icon connects to the **Document Repository**.

This icon connects to the **Directory**.

This icon connects to **Training.**

 This icon connects to **Tools.**

4.2.2 DOCUMENENT REPOSITORY ICONS

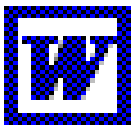
Each directory has an icon that appears next to its name. The icon is an orange file as shown below. By clicking on either the orange file or the name the user can view the contents of the directory.

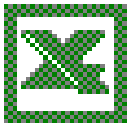
 This icon represents a Directory.

EXAMPLE:

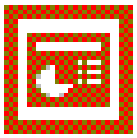
 DIRECTORY

Directories in the **Document Repository** contain files. These files may contain different types of documents. Below are **icons** that appear next to the file name that represent the type of document that is in the file. By clicking on the icon the user can view the document.

 This icon represents a Microsoft Word document. (Text)

 This icon represents a Microsoft Excel document (Spreadsheet).

 This icon represents a Microsoft Access document. (Database)

 This icon represents a Microsoft PowerPoint document. (Presentation)

EXAMPLE:

 trainingpart1.doc

 ukstaff.csv

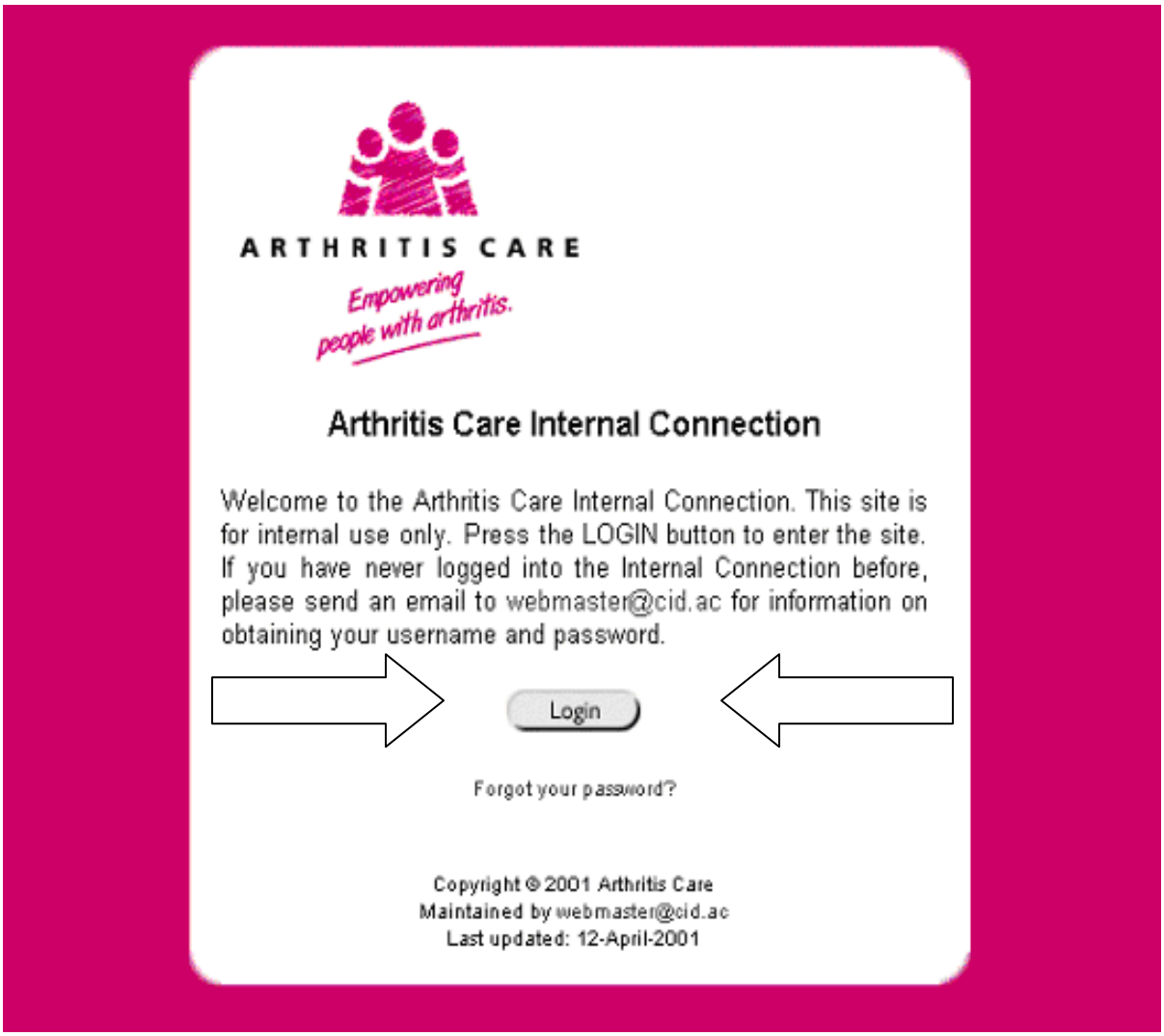
 contacts.mdb

 power .ppt

5.0 ARTHRITIS CARE INTERNAL CONNECTION

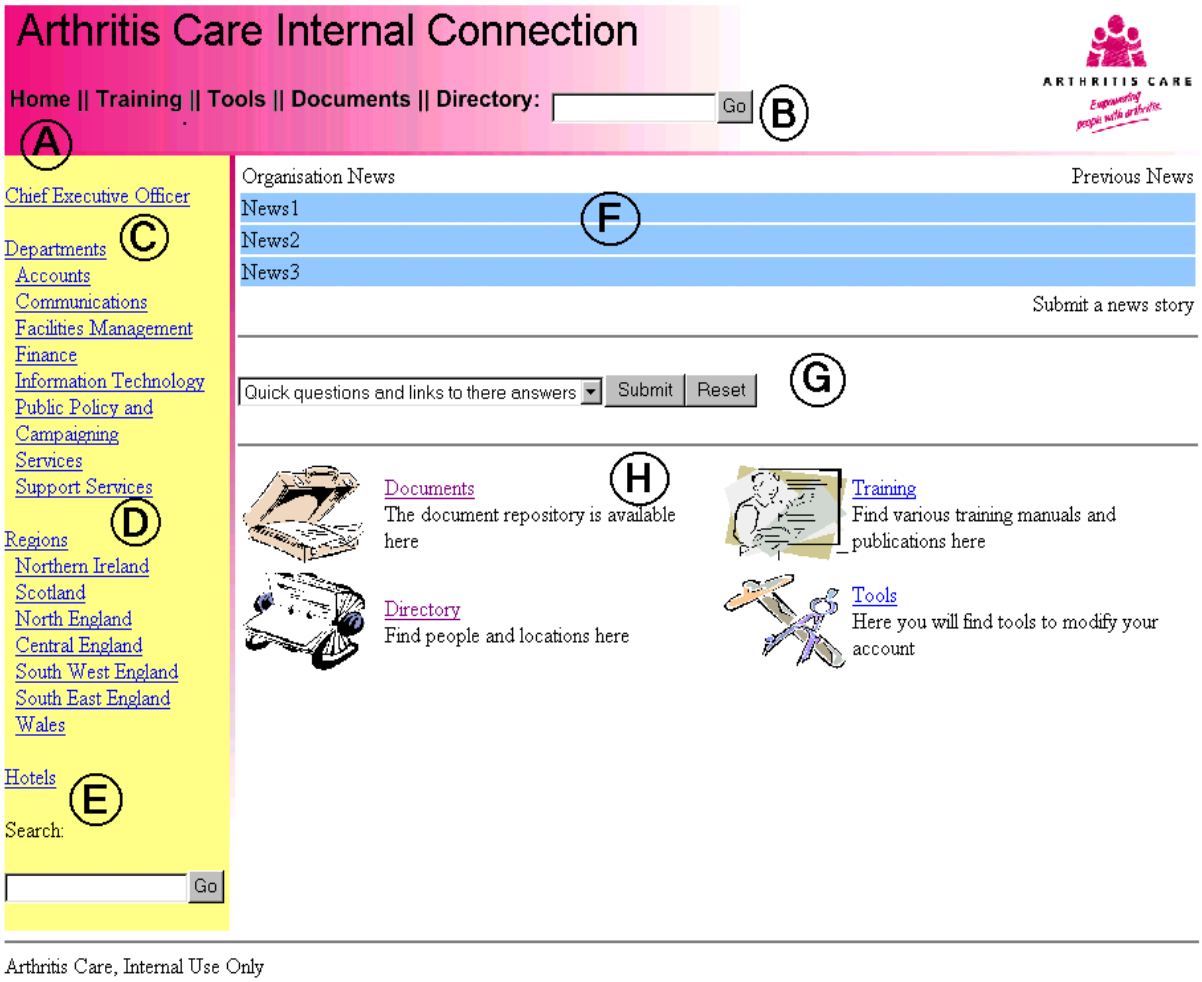
5.1 LOGIN

The Arthritis Care Internal Connection has a login page that will appear after the user types in the web address to access the site. On the login page there is a login button that the user must click on to enter their password and user name. The login page is shown below.



5.2 MAIN PAGE

The main page of the Arthritis Care Internal Connection is on the following page. From this page the user has access all the parts of the intranet quickly and easily by clicking on a link or icon. This page is designed so that all the parts of the intranet are clearly viewed on one page and can be accessed within three clicks of the mouse button.



Features

Feature **A**

On the top of the page, the **Home**, **Training**, **Tools**, **Documents** and **Directory** are listed as links for the user to click on and directly go to.

Feature **B**

On the top right is a search option that allows the user to type in a name and search the Arthritis Care staff **Directory** right from the main page.

Feature **C**

On the far left side of this main page there is a list of **Departments** in the central office. Each department can be visited by clicking on it with the mouse.

Feature **D**

Below the central office department list there is a list of regional offices that can be visited by clicking on it with the mouse.

Feature **E**

Below the Regional office list there is a **Search** option. This allows the user to search for anything in the intranet by simply typing in a query and clicking on the **Go** button.

Feature **F**

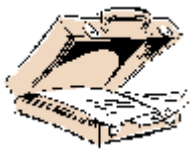
On the top of the main page is **Organisational News**. Here, authorized intranet users can view recent news and events of Arthritis care.

Feature **G**

In the middle of the main page there is box where the user can get answers to commonly asked queries about the intranet.

Feature **H**

Below Feature G are connections to the **Document Repository**, **Directory**, **Training** and **Tools**.



6.0 DOCUMENT REPOSITORY

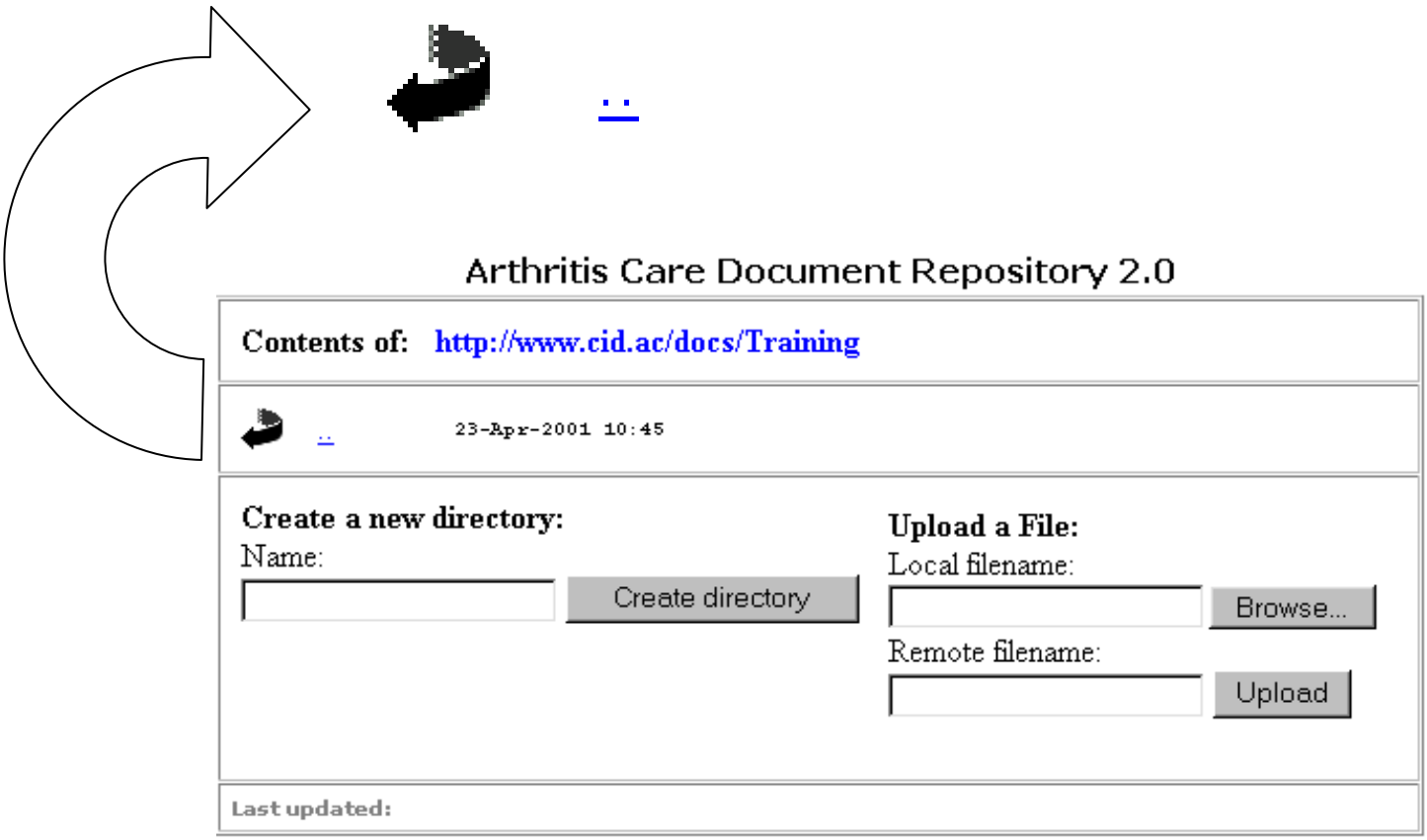
The document repository is a feature of the intranet that allows you to create a directory and upload / download documents of your choice to it. Any authorised intranet user is able to view the documents but only certain authorised users will have the ability to edit and revise them. Documents can be put on the web for quick and easy viewing. An added feature of the document repository is the date and time stamp. It shows the latest posting or edition of a particular directory or document and

appears next to its name in the document repository. This section explains how to create a directory, create additional directories, upload files to the document repository, and change the name of a file.

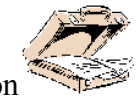
6.1 BUTTONS

These buttons are found in the **Document Repository**. Clicking on either one of the buttons will immediately take you up one level in the document repository from the previous page.

Document Repository buttons are found here:



6.2 CREATE A DIRECTORY

To create a directory in the document repository first open the main page of Arthritis Care Internal Connection and click on the Documents link or the icon  . This will take you to the first page of the Document Repository where all the directories are listed on the left side and the date and time that they were created or last updated are listed in the middle as shown below:

Arthritis Care Document Repository 2.0

Contents of: http://www.cid.ac/docs

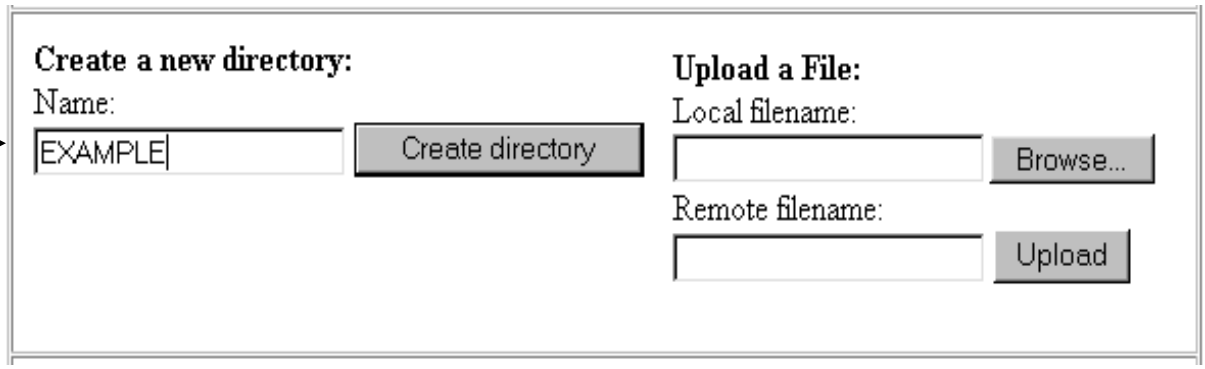
EXAMPLE	23-Apr-2001 12:22	Delete this directory
Accounts	19-Apr-2001 10:23	Delete this directory
Communications	19-Apr-2001 10:23	Delete this directory
IT	19-Apr-2001 10:24	Delete this directory
Services	19-Apr-2001 10:24	Delete this directory
The Key	23-Apr-2001 10:41	Delete this directory
Training	19-Apr-2001 10:23	Delete this directory

Create a new directory:
Name:
Create directory

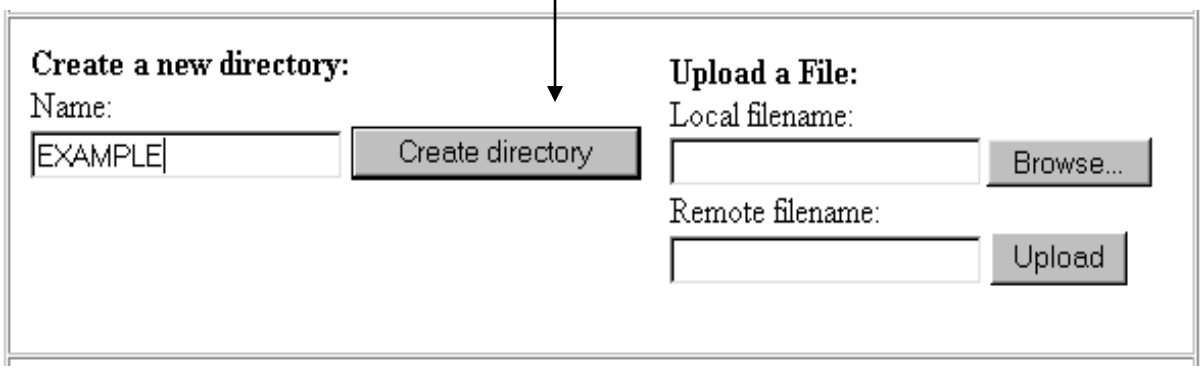
Upload a File:
Local filename:
Browse...
Remote filename:
Upload

Last updated:

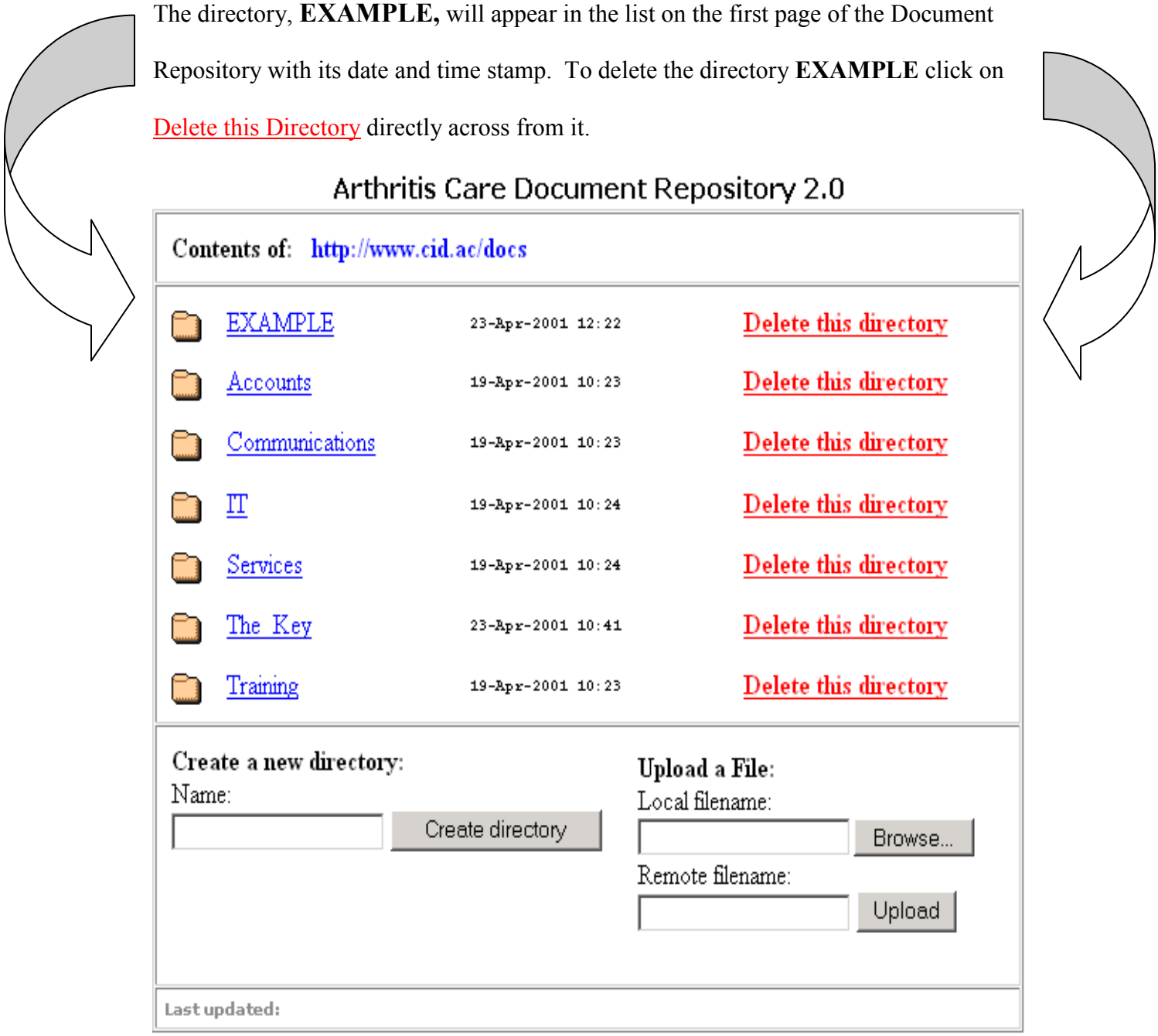
To create a new directory click on the text box under **Create a new directory.** Type in the name of the directory, used here titled as **EXAMPLE,** as shown below.



To add the directory **EXAMPLE** to the document repository click on the **Create Directory** button 



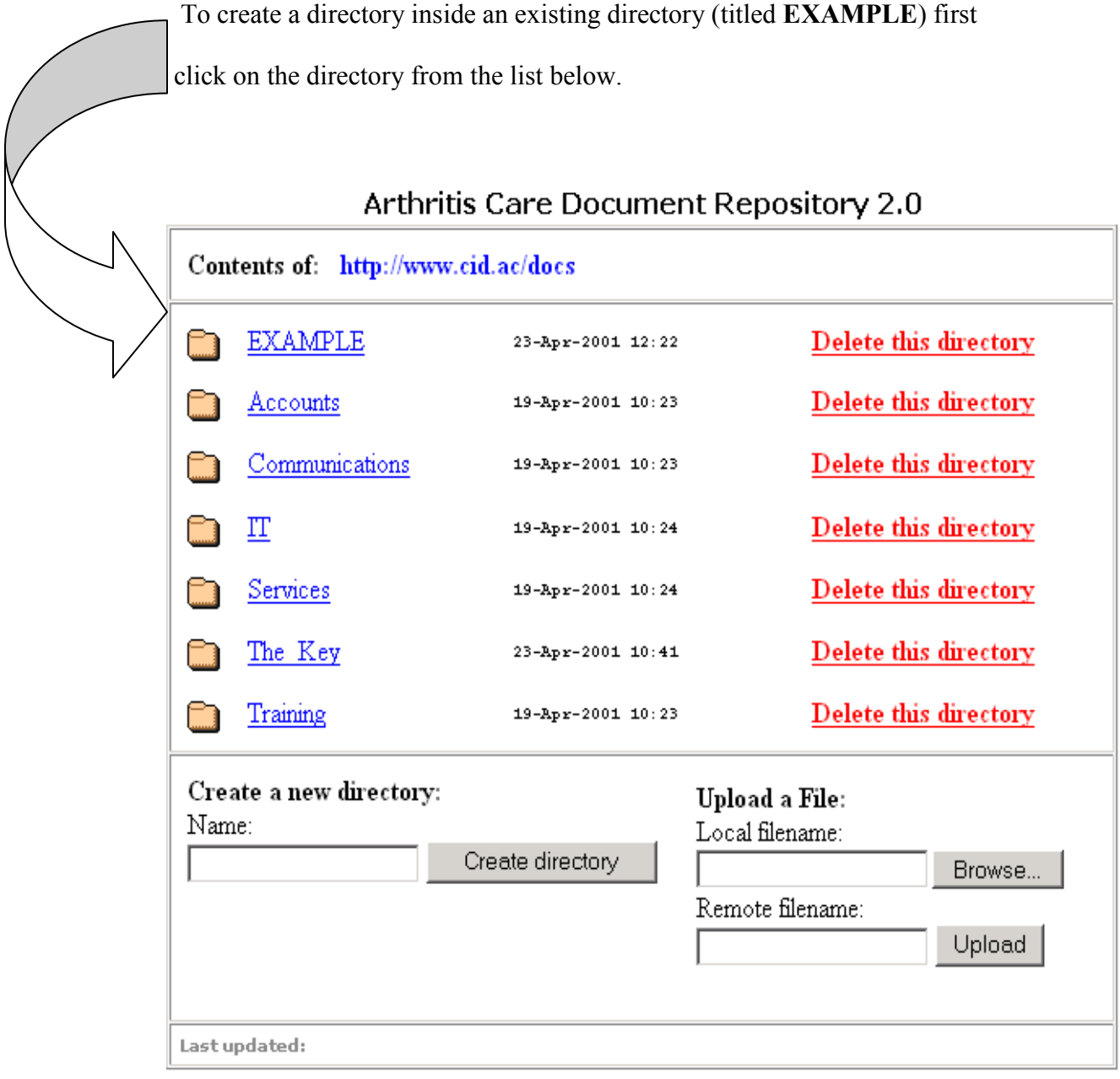
The directory, **EXAMPLE,** will appear in the list on the first page of the Document Repository with its date and time stamp. To delete the directory **EXAMPLE** click on Delete this Directory directly across from it.



6.2.1 ADDITIONAL DIRECTORIES

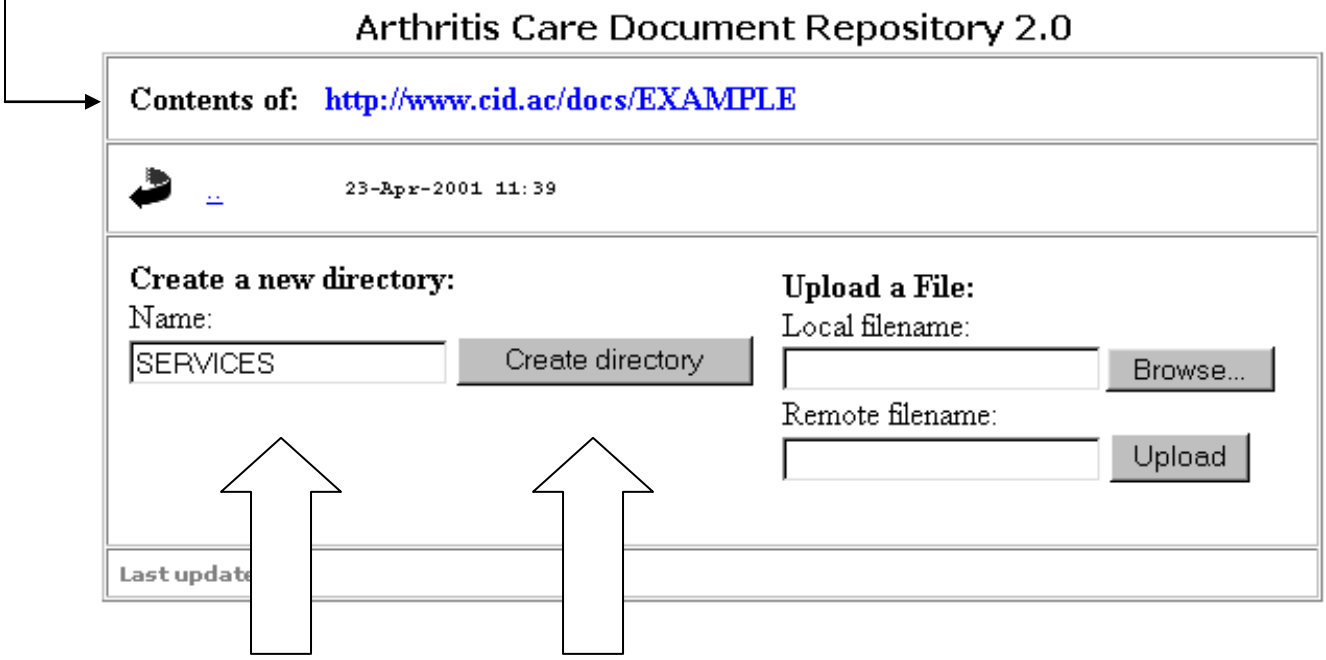
The **Document Repository** also allows the user to create a directory inside an existing directory and so on. This feature allows for better organisation of the directories and files that are stored in the **Document Repository**. The following steps below show how to create a directory inside another directory.

To create a directory inside an existing directory (titled **EXAMPLE**) first click on the directory from the list below.

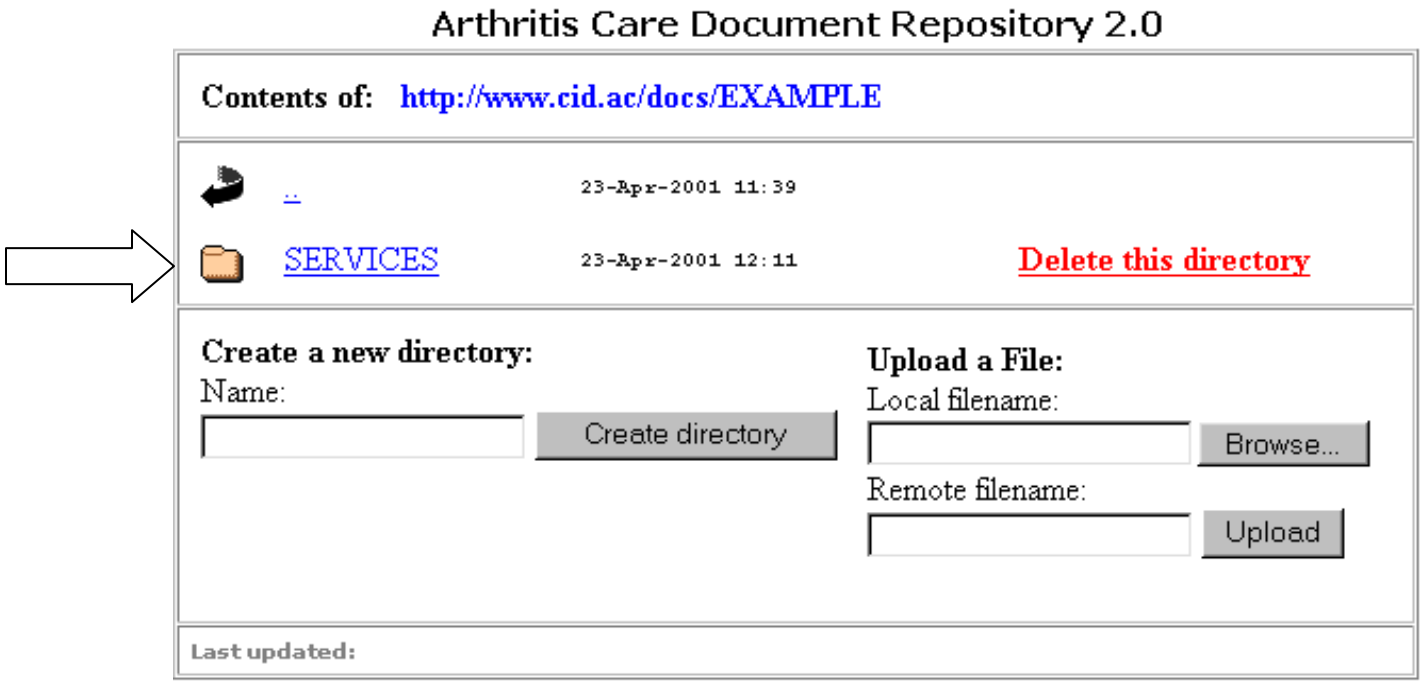


The screen shown below displays the **Contents of** the **EXAMPLE** directory.

To put a new directory into the **EXAMPLE** directory enter the name of the new directory, used here titled as **SERVICES**, in the text box under **Create a new directory**. Then click on the **Create directory** button.



Clicking on the **Create Directory** button creates the new directory, **SERVICES,** inside the **EXAMPLE** directory as shown below:



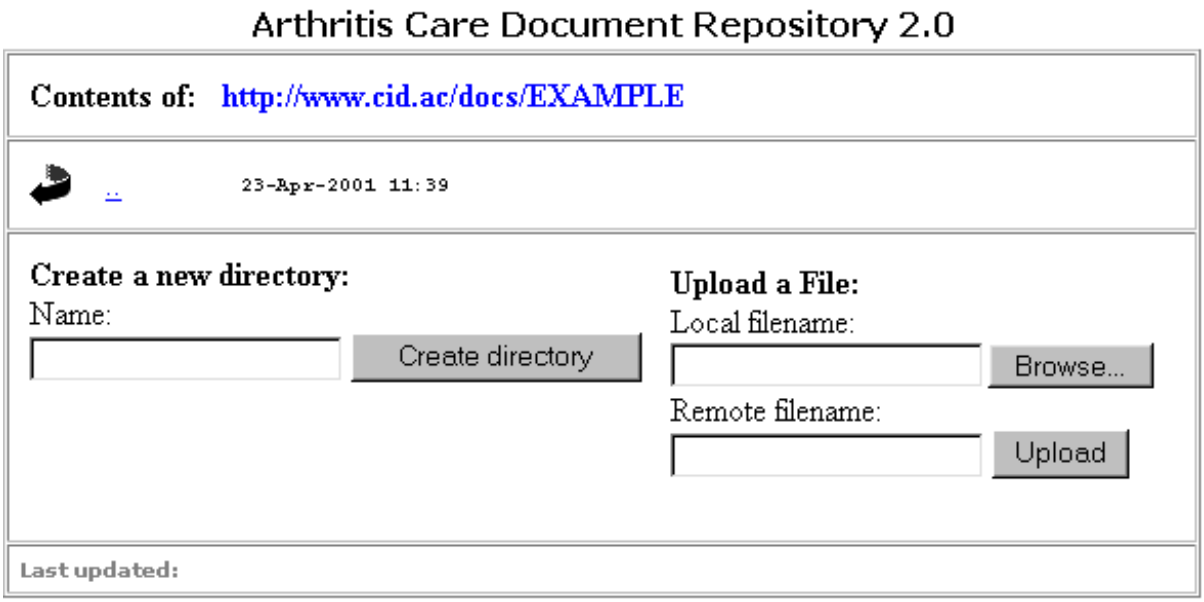
The above screen shows that the **SERVICES** directory is contained in the directory **EXAMPLE**. From this screen the user can:

- Create another new directory inside the **EXAMPLE** directory by repeating the process described above in section **6.2.1.**
- Create a new directory inside the **SERVICES** directory by clicking on SERVICES from the above screen and repeating the process described in section **6.2.1.**
- **Upload a file** to the **SERVICES** directory by clicking on SERVICES and following the steps in section **6.3** for uploading files to directories.
- Delete the **SERVICES** directory by clicking on Delete this Directory, directly across from the directory name.

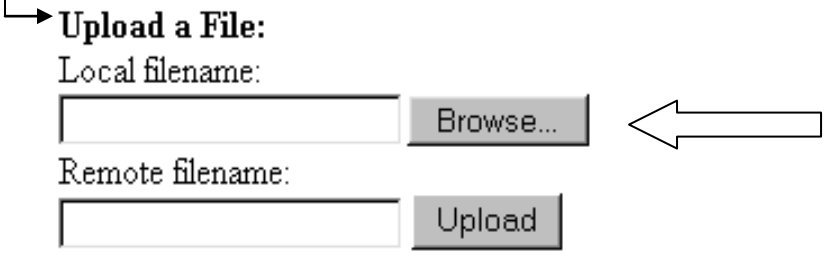
6.3 UPLOAD A FILE

To upload a file to the directory you just created, **EXAMPLE**, click on it from the list below.

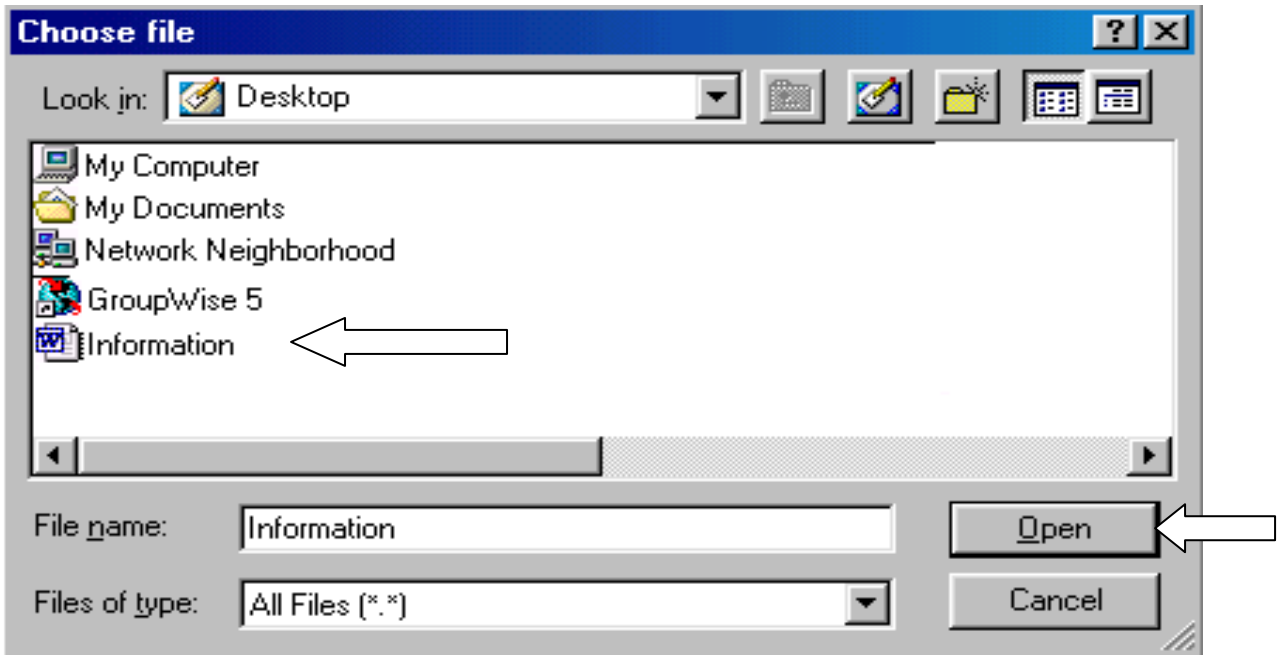
Clicking on the directory, **EXAMPLE,** will bring you to a new screen that displays what is contained in the **EXAMPLE** directory, which is currently empty, as shown below:



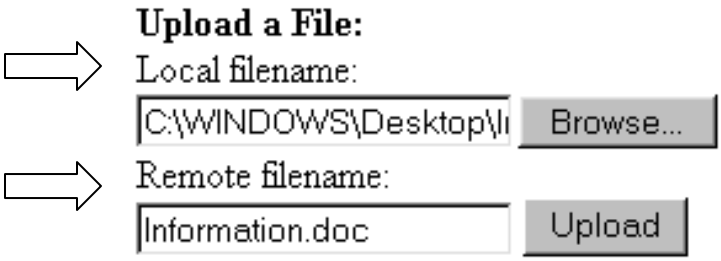
To upload a file from your computer to the Document Repository go to **Upload a File** and click on the **Browse** button as shown below.



Shown below is a window that will appear on your screen allowing you to search for files on the computer or on a disk. To upload the text document, , titled **Information** from the computer click on it with the mouse and then click on the **Open** button.



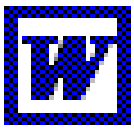
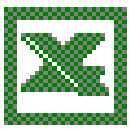
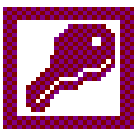
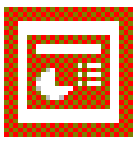
Clicking on the Open button will automatically enter the **Local filename** and **Remote filename** of the text document **Information.doc** in the two spaces shown below.



NOTE-

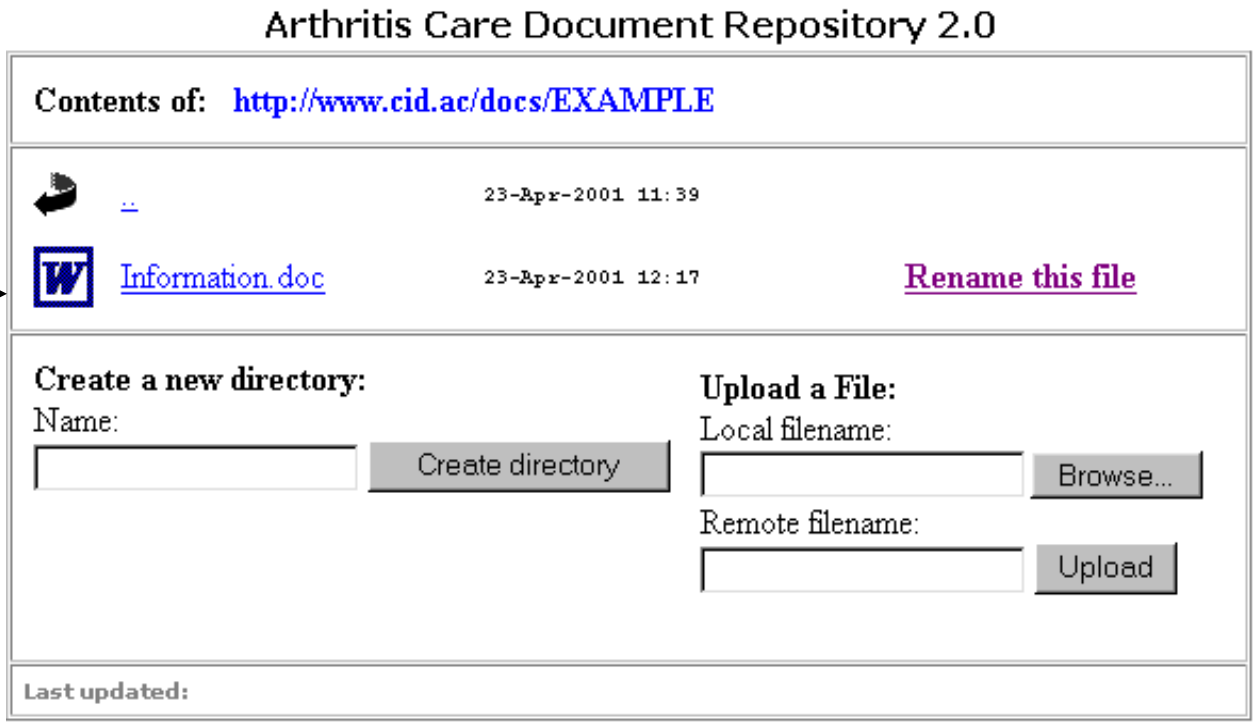
Before uploading a file, the Remote file name can be changed to another name by clicking on the space under Remote file name and typing in a new name. It is important that **YOU MUST GIVE THE FILE NAME AN EXTENTION FOR IT TO WORK PROPERLY**. For example: if it is a word document the proper extension is **.doc**, giving the **Remote filename: Information.doc**

Listed below are the extensions for each particular type of document:

Document	Extension
 Microsoft Word	**.doc**
 Microsoft Excel	**.xls**
 Microsoft Access	**.mdb**
 Microsoft PowerPoint	**.ppt**

To upload this file, **Information.doc**, to the Document Repository click on the **Upload** button shown below.

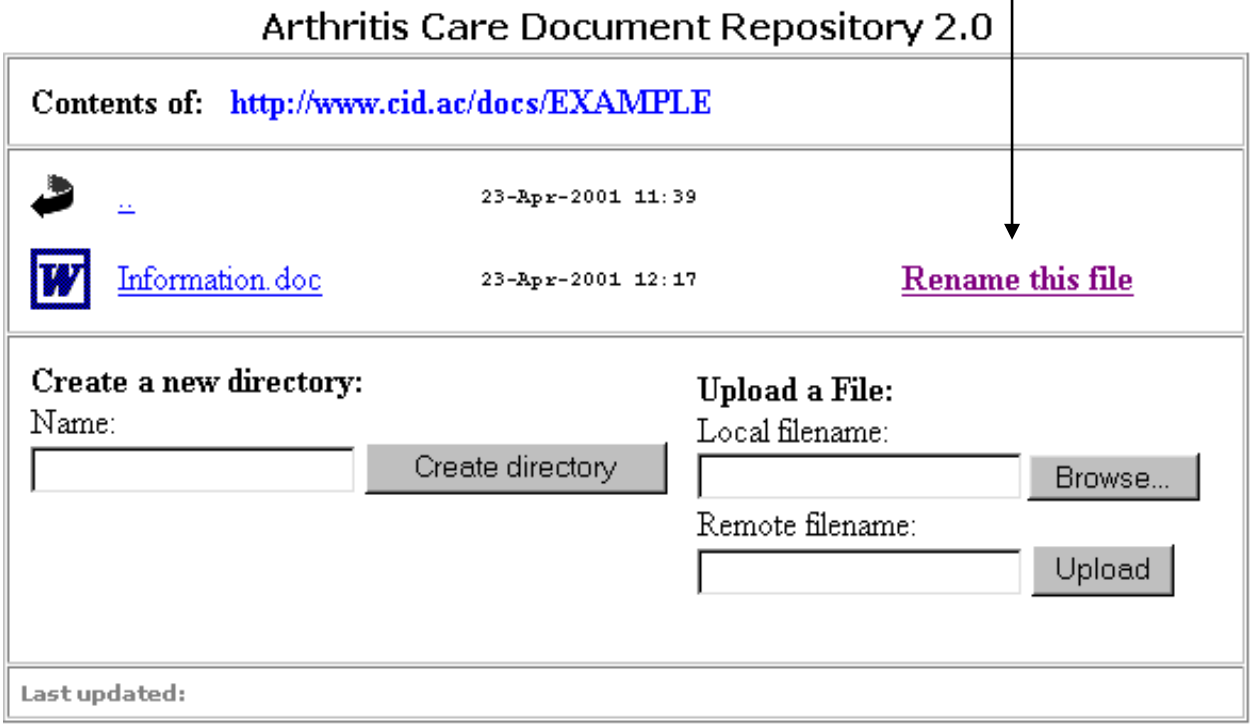
The file will appear in the directory titled **EXAMPLE** shown in the screen below.



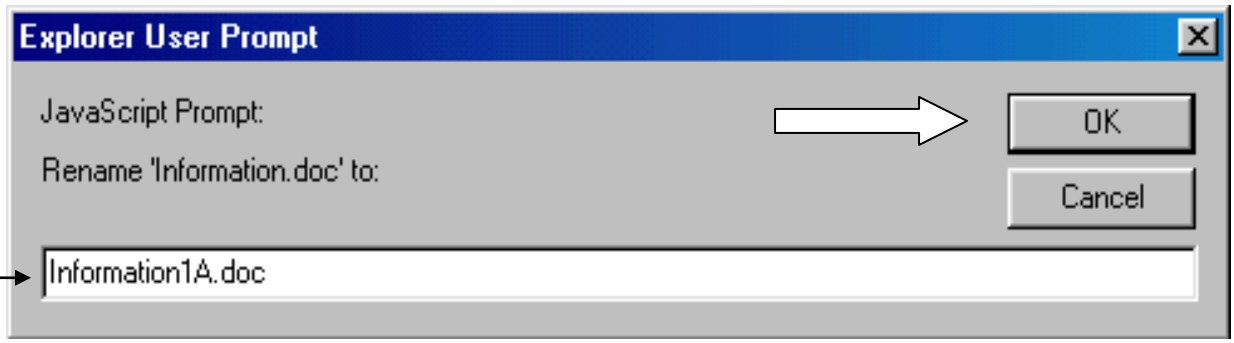
6.3.1 RE-NAME AN UPLOADED FILE

An existing uploaded file can be easily renamed. The file named **Information.doc** has been uploaded to the directory **EXAMPLE.** To change the name of this file from **Information.doc** to **Information1A.doc** is very easy.

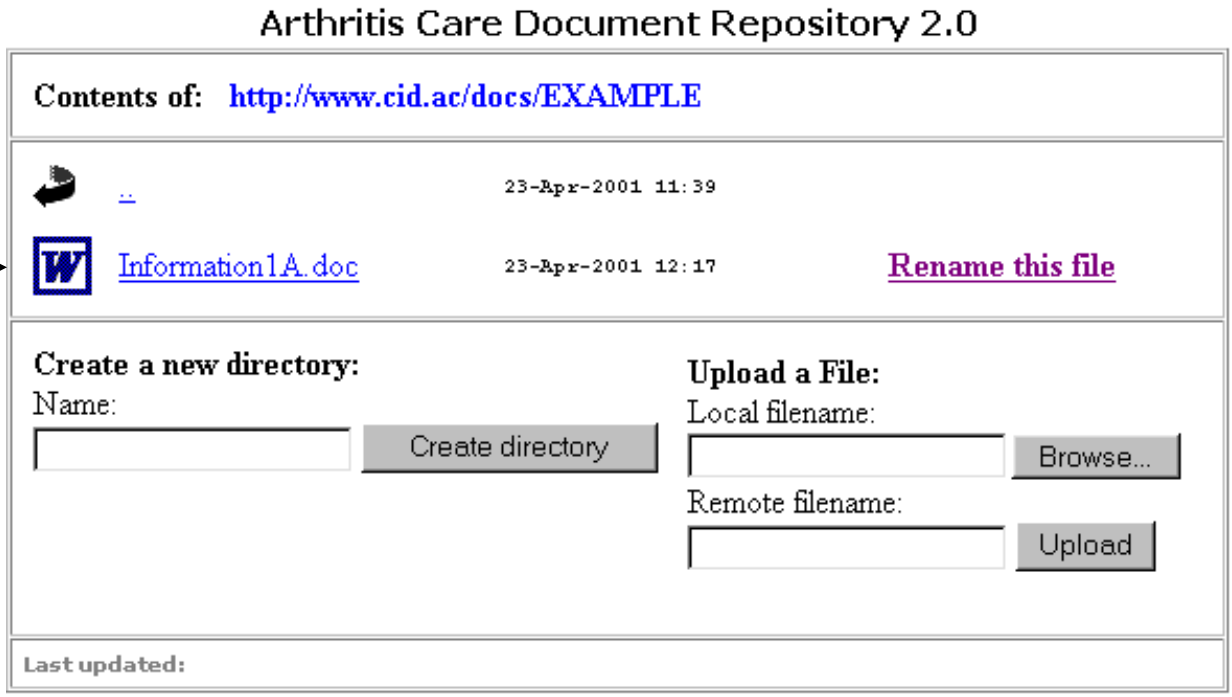
To change the file name first click on **Rename this file**



Clicking on **Rename this File** will immediately open a new box on the screen where you can enter the new name **Information1A.doc** as shown below**.** Click on **OK** for the change to occur.



The name change from **Information.doc** to **Information1A.doc** will appear as shown below.



7.0 DIRECTORY

The **Directory** is a listing of Arthritis Care staff contact information. Information such as name, address, phone number, e-mail and position in Arthritis Care can be accessed easily and efficiently with this directory. This information can also be edited at any time to provide the most accurate and up-to-date information about the staff of Arthritis Care. This page can be accessed by going to the Arthritis Care Internal Connection main page and clicking on the Directory link or its icon  . The following sections describe how to use the directory and conduct searches.

The first page of the Directory that appears after clicking on the Directory link or its icon is shown below.

7.1 QUICK SEARCH

The **Quick Search** button allows the user to search for an Arthritis Care staff member by name. Below is the first page of the directory with the Quick Search option. The full name and exact spelling does not have to be known to conduct a search. The search can be narrowed or expanded depending on how complete the name is.

QUICK SEARCH FEATURE:

The Quick Search option allows the user to search the directory without having to know the full name or exact spelling of the person. This means that a search can be done by first names only, surnames only, misspelled names and full names. However, this search is not as narrow; it provides the user with a list of names to choose from that are close to what they are looking for.

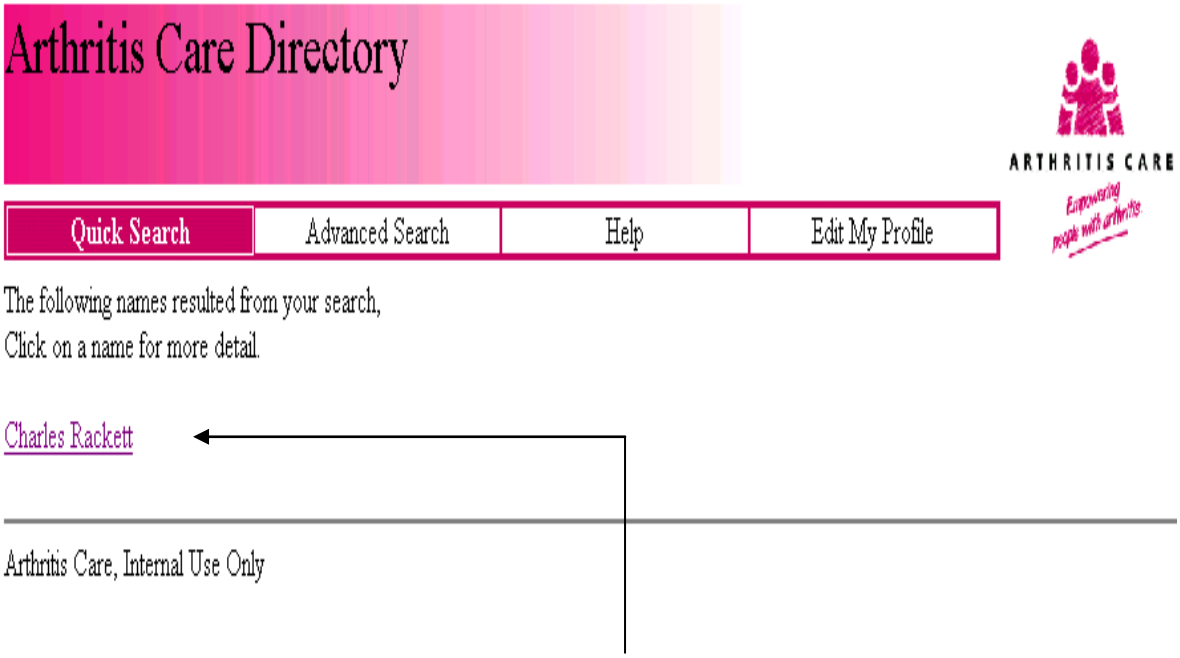
Using the Quick Search option, typing in the letters **Rac** will produce a list of names where the letters appear in that order in either the first or sir name. Typing in the name **Tracy** produces a shorter list with all the Arthritis Care staff named **Tracy**. Typing in the first and sir name of a staff member, such as **Charles Rackett**, in the directory will list only that person in the search results. This is shown below:

QUICK SEARCH RESULTS:

Quick Search Results for typing in the letters: **Rac.**	Quick Search Results for typing in the name: **Tracy**.	Quick Search Results for typing in the name: **Charles Rackett**
Rachel Johnson	Tracy Dudley	Charles Rackett
John Puract	Tracy Billings	
Bill Toomracs		
Beth Racell		
Rachel Wickett		
Charles Rackett		
Rob Belrach		
Tracy Dudley		
David Bachrach		
William Racene		
Tracy Billings		

To do a **Quick Search**, type the name of the staff member in the text box shown below and click on the **Search** button

The search results are shown below:



To view information about the staff member **Click** on their name from the search results page shown above. After clicking on the name the contact and organisational information of that person will appear as shown below:

Charles Rackett

Contact Information

Email Address	RackettC@arthritiscare.org.uk	Address 1	00 Main Street
Work Phone	123 4567 8910	Address 2	
Mobile	no mobile	Address 3	
Fax		Address 4	
		Address 5	
		Postal Code	ABC 123

Organisational Information

Job Title	
Hours	part time
Team	Services
Location or Division	UK
Notes	based at home

On the bottom of the contact and organisational information screen there is a Tool bar with three options: **Home**, **Directory** and **Documents**.

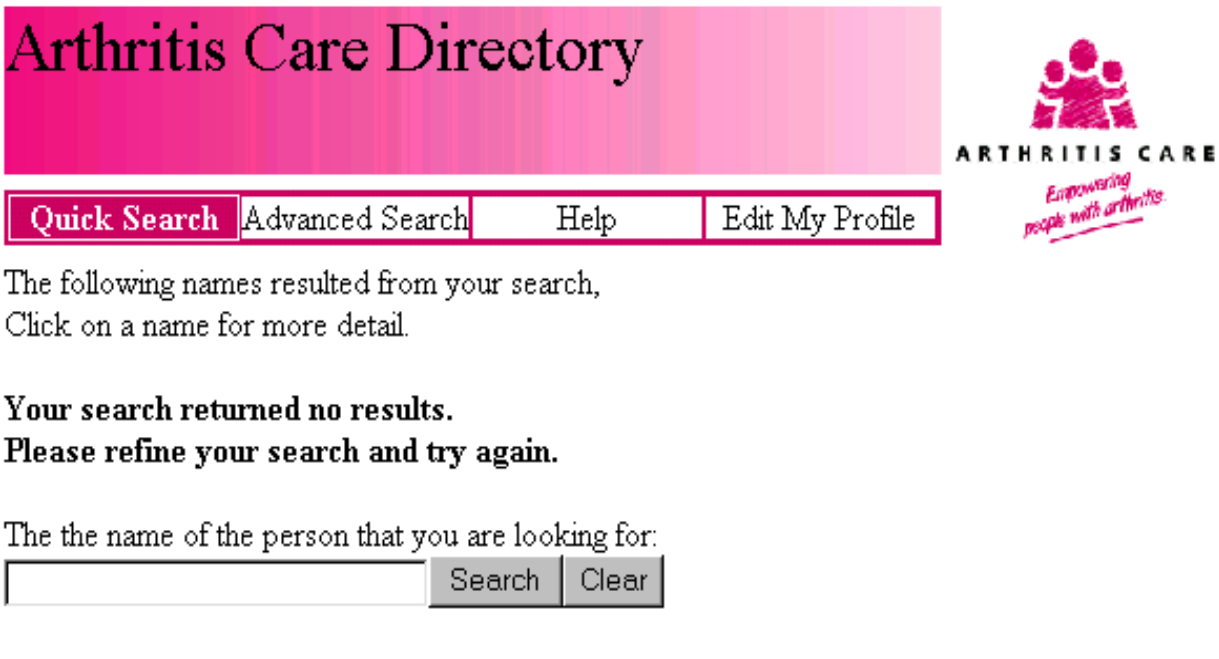
Clicking on the **Home** button will bring you to the first, main page of the Arthritis Care Internal Connection.

Clicking on the **Directory** button will bring you back main directory page to start a new search.

Clicking on the **Documents** button will bring you to the Document Repository main page.

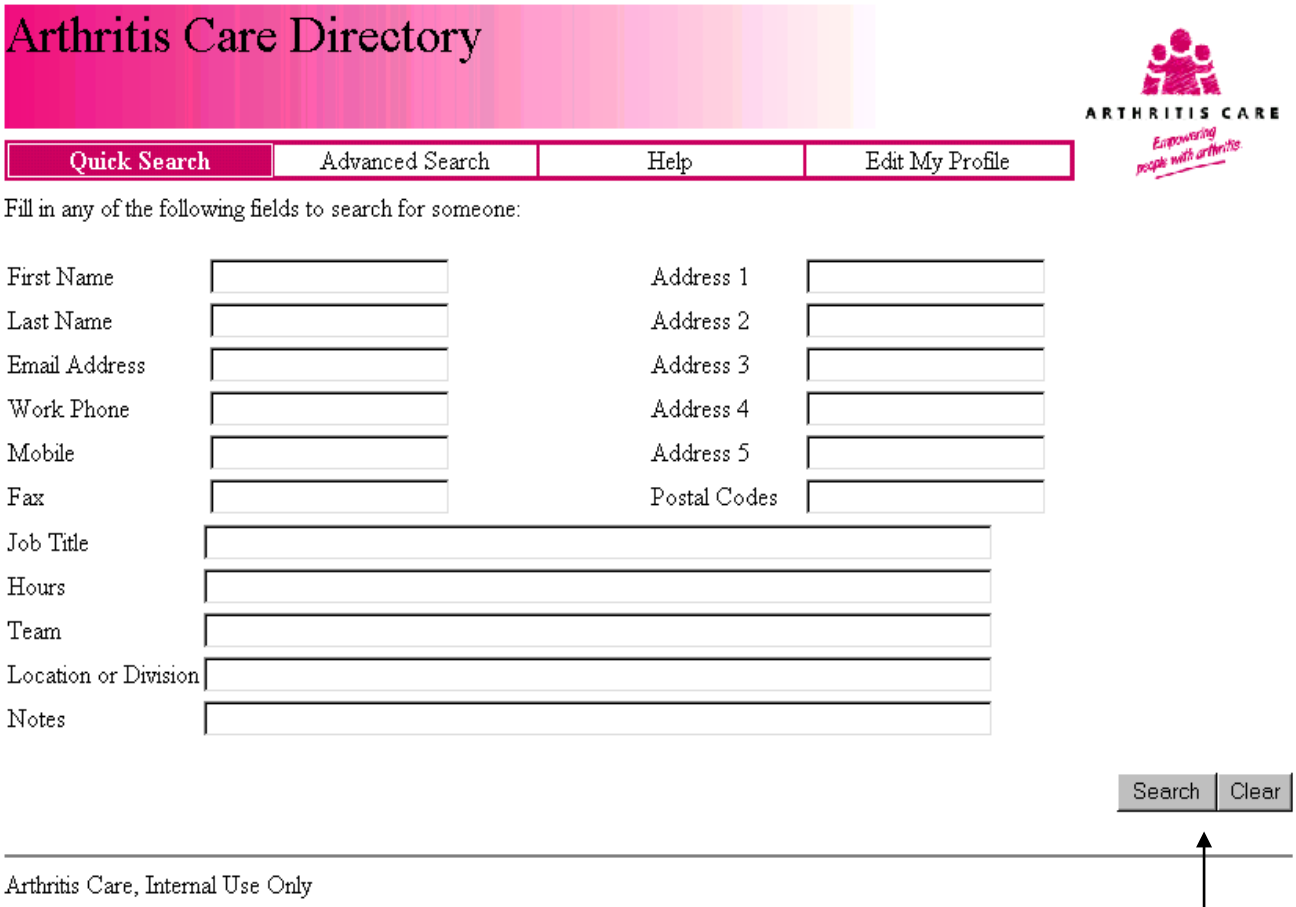
-Unsuccessful Quick Search

Shown below is the screen that appears when the name typed into the **Quick Search** does not match any of the names in the database. If the **Quick Search** returns no results it is best to refine the name and try it again.



7.2 ADVANCED SEARCH

The **Advanced Search** allows the user to search the staff database for a particular person or group of people based on any form of contact and organisational information they chose to include on their profile. A search can be done using one, or more than one, of the available contact and organisational information. For example, a search can be done for staff members at the Arthritis Care **central office** who all have the same first **name**. The screen shown below is where the user can perform a search for a staff member using the **Advanced Search** feature.



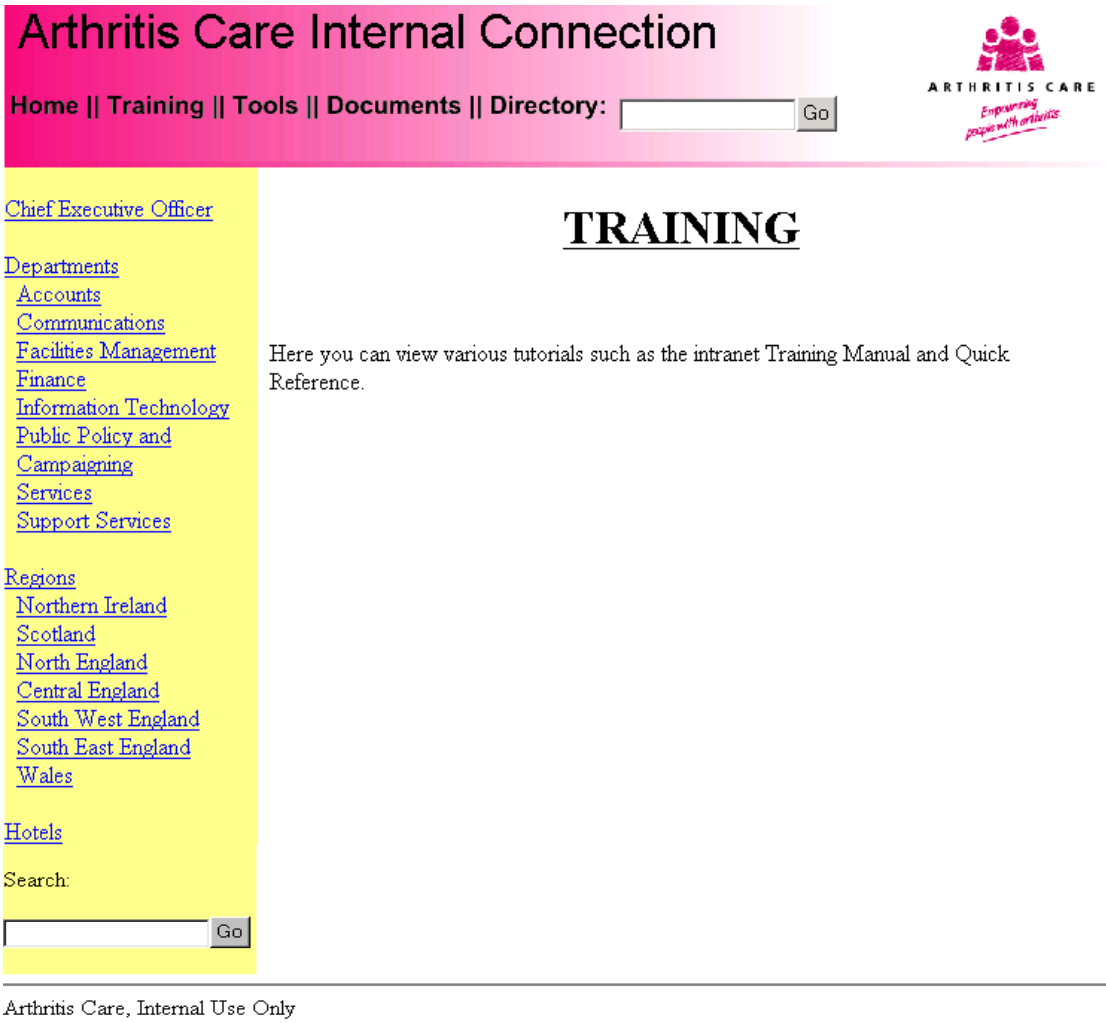
To search for staff members at the Arthritis Care central office who all have the same first name, click on the space next to **First Name** and the space next to **Location or Division**. Enter the information into the spaces and click on the **Search** button at the bottom right.

Note:

The amount of search options is not limited by two as shown in the example above. Any variety or combination of the above Advanced Search options can be used to best narrow the search.

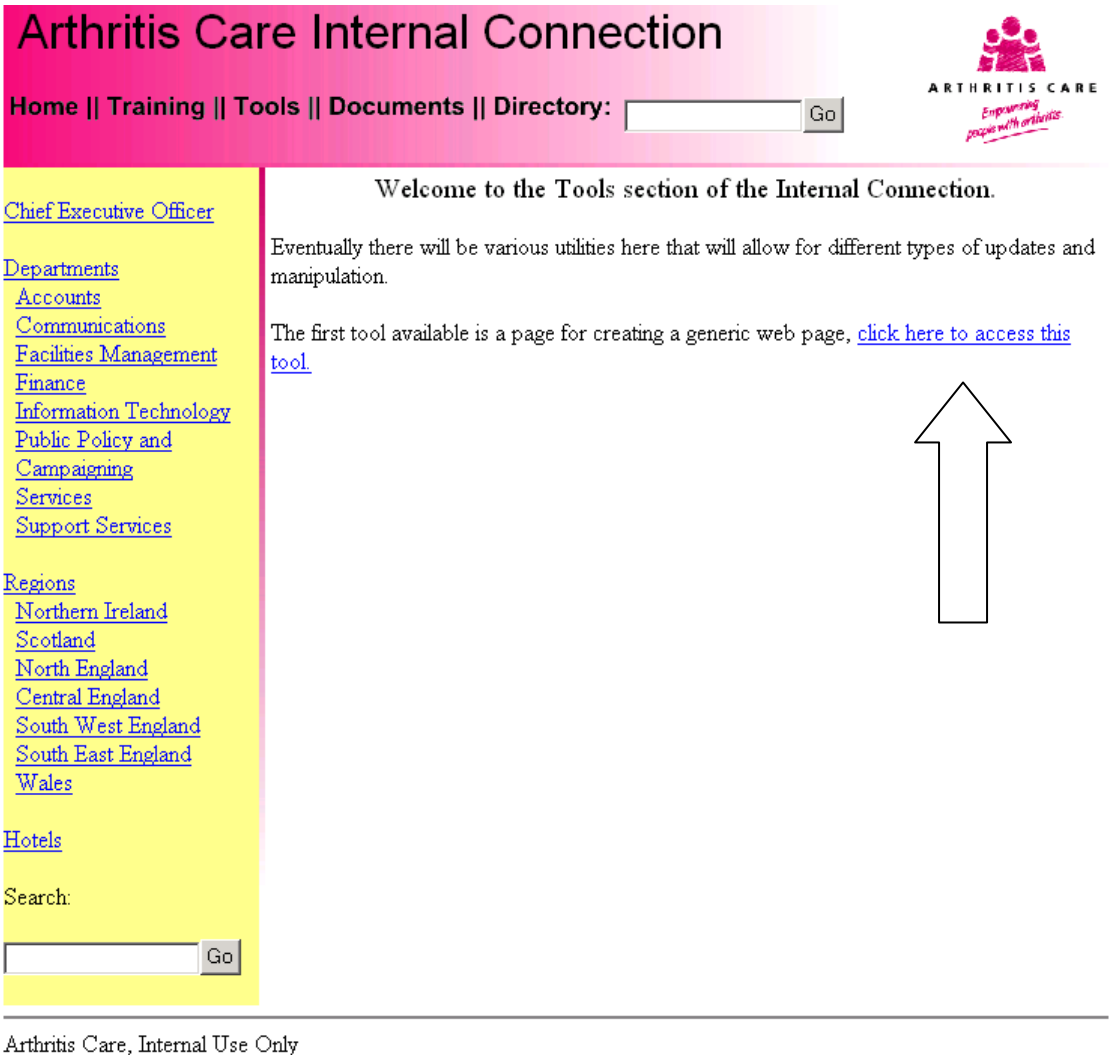
8.0 TRAINING

Training is a feature of the intranet that contains Arthritis Care training tutorials and the Training Manual/Quick Reference sheet for the intranet. These materials can be viewed and printed at the users personal computer. To access the **Training** feature, click on either the Training link or its icon  found on the main page of the intranet. The first page of **Training** is shown below:

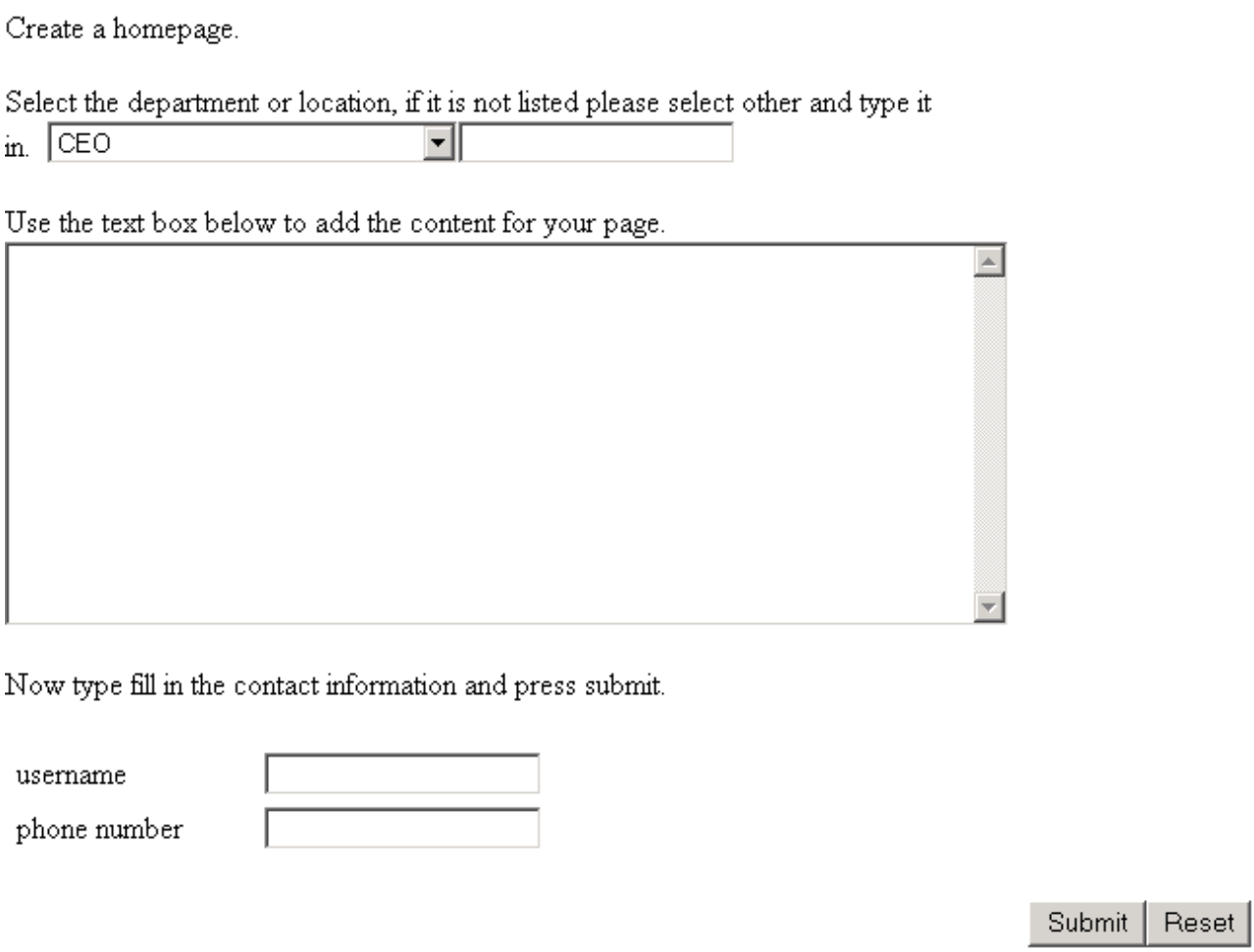


9.0 TOOLS

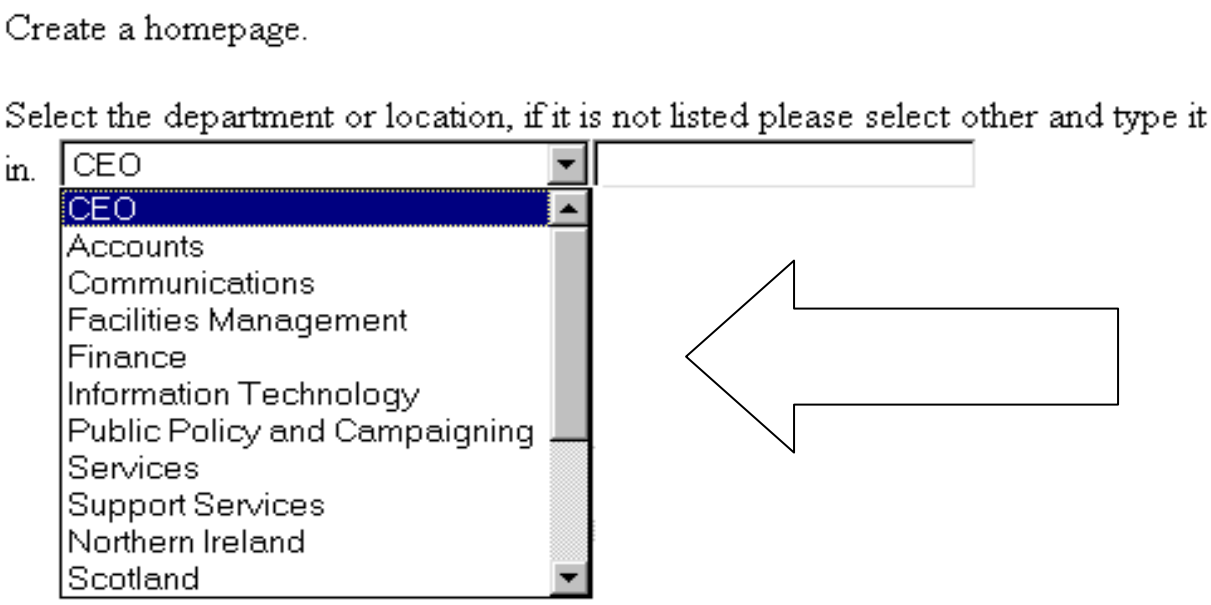
The **Tools** feature of the intranet allows users to create generic web pages for departments in the central office and regional Arthritis Care offices. To access the **Tools** feature, click on the Tools link or its icon  found on the main page of the intranet. The link to create a generic web page is found on the first page of the **Tools** feature as shown below:



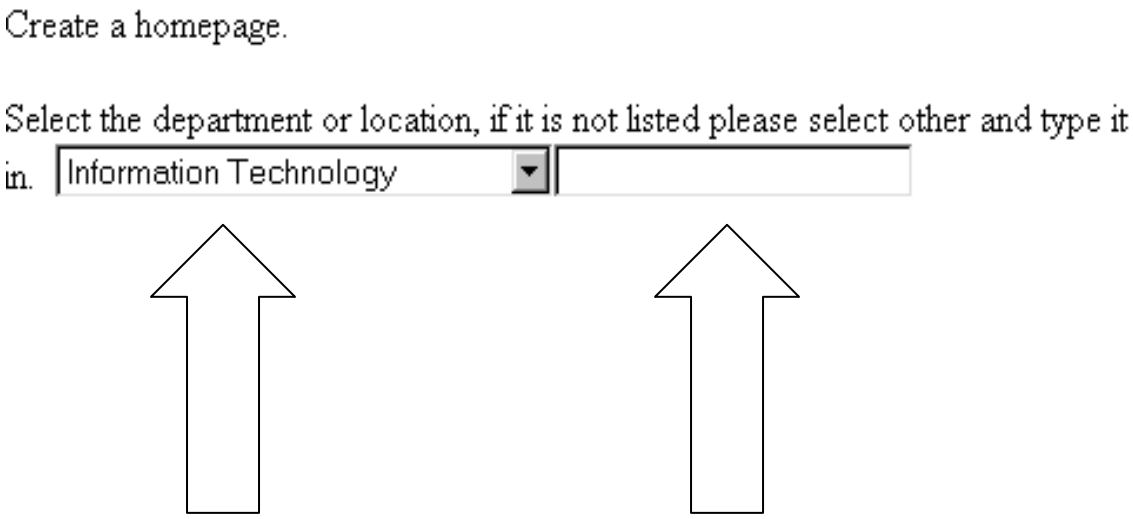
After clicking on this link from the main page as shown above, the form to create a generic web page will appear as shown below:



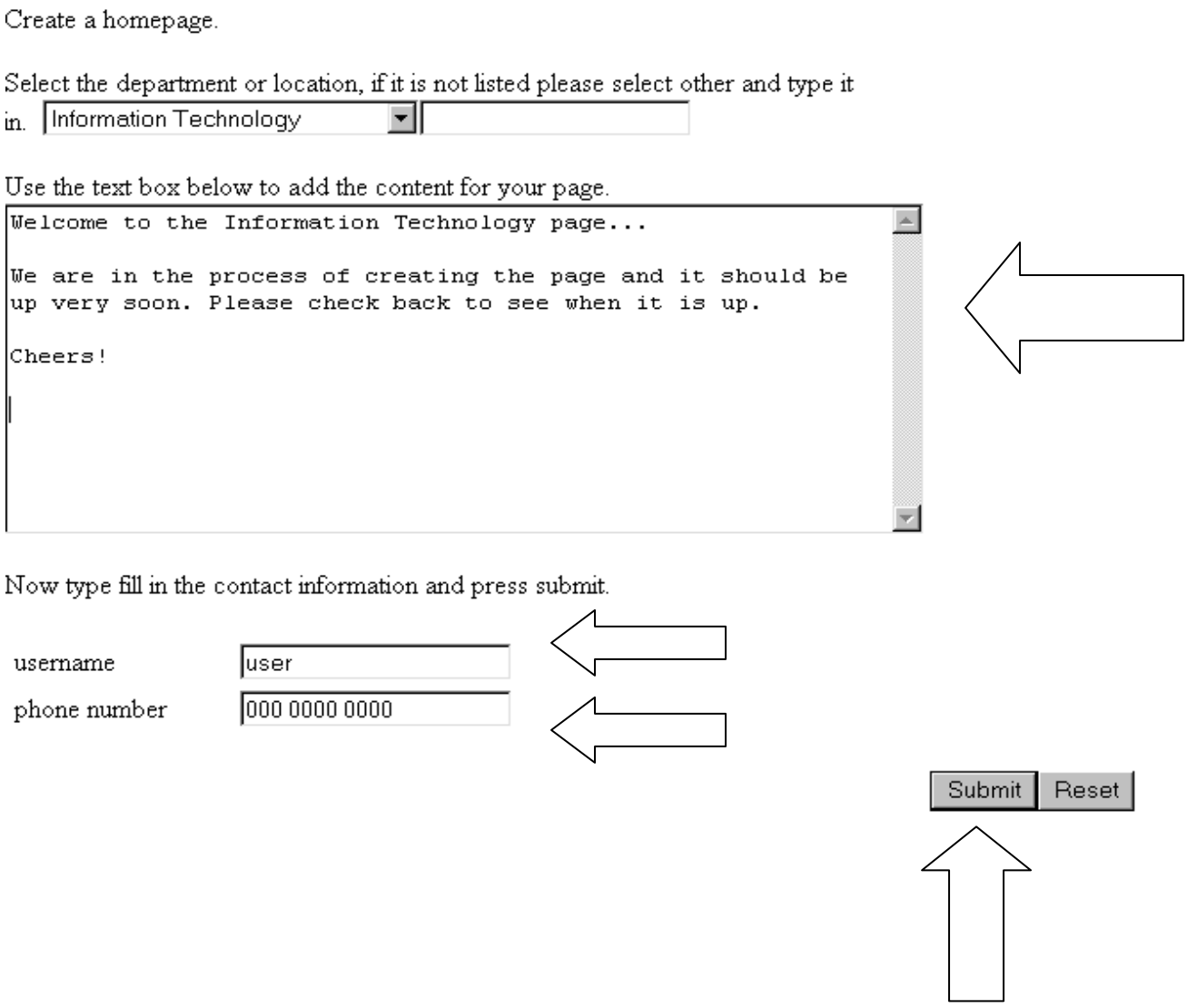
To create a web page for a department or regional office first choose the department or location from the list that appears in the drop box by clicking on the down arrow  as shown below:



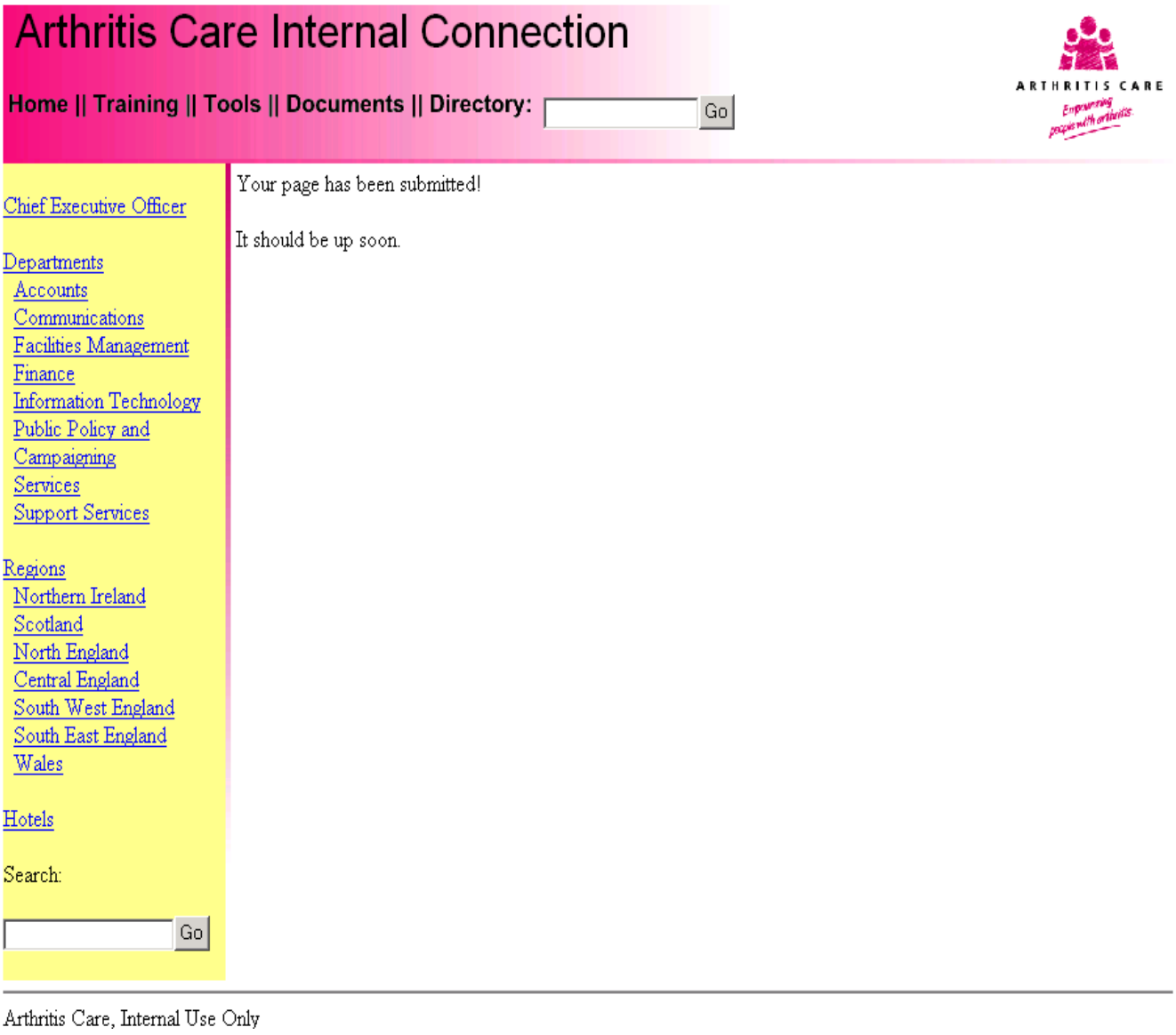
To create a web page for the **Information Technology** department select it from the list by clicking on it as shown above. Clicking on **Information Technology** from the list will automatically enter it into the text box as shown below. If the department or location is not represented in the list, enter it into the text box to the right.



The content for the web page that you wish to create is entered in the large space shown below. The username and phone number of the person who created the page must also be entered in the appropriate spaces below. To submit the web page you created click on the submit button shown at the bottom right of the content form.



After clicking on the submit button, a new page will appear confirming that the web page you created was submitted successfully as shown below:



GLOSSARY

Advanced Search: This feature allows the user to search the staff database for a particular person or group of people based on any form of contact and organisational information they chose to include in their profile.

Directory: A listing that includes staff and member contact information as well as job descriptions.

Document Repository: Location where various forms of information can be stored and organised.

Homepage: The option on the Internet Explorer to set a particular web site to automatically open once the web browser has been opened.

Icon: A picture or graphic on a computer screen that represents an available function. Icons work the same way that links do. By clicking on the icon, the user can get to a new destination, whether it is a new page or document.

Internet: Transfers information over the international telephone system and connects thousands of personal computers that are on local, metropolitan, and wide area networks. Also referred to as the "World Wide Web".

Internet Explorer: A Web Browser that you can use to search the Internet.

Intranet: A smaller more self-contained version of the Internet that allows all the communication features of the Internet but is not accessible to the outside world.

Link: A word or phrase on a web page that is usually in a different color, bold print, and/or underlined. It serves as a connection from one page to another destination.

Local Filename: Contains the complete file name from where the document was originally obtained. This can be from a disk, computer's hard drive, disk or network.

Networking: Information that is transferred from one computer to another by means of a connection.

Quick Search: This feature allows the user to search the directory for an Arthritis Care staff member by name without knowing the full name or exact spelling of the name.

Remote Filename: The name of the document chosen at your discretion.

Search Engine: Device used to search frequently used databases that are open to the public on the Internet for searching.

Tools: Here you will find tools to modify your account

Tool Bar: Basic buttons like: search, forward and backward arrows and the stop button are found on the Internet Explorer.

Training: Find various training manuals and publications here

World Wide Web: Transfers information over the international telephone system and connects thousands of personal computers that are on local, metropolitan, and wide area networks. Also referred to as the Internet.

Appendix H – Maintenance Guide for Intranet

The Internal Connection is currently hosted by clara.net. All pages are stored on their Unix servers. They are responsible for the domain name, cid.ac, and linking it to the directory on the Unix server.

The site is a combination of HTML, Perl and JavaScript. All static pages are written in html. Pages that are dependant on user input are a combination of Perl and JavaScript. Perl is used on the server to generate data, and JavaScript is used to verify input. The Perl pages have the extension of cgi, and are executed by the server.

Now knowing what the pages are made up of, it must be understood how they are stored. A graphical layout of the entire site is available in Figure 1, at the end of this appendix. The layout has been designed to be as easy to use and understand as possible. By referring to each location and department as an instance, a directory is then very easy to get to the instance's location by the URL. All boxes under http://www.cid.ac in Figure 1 start with a (/), the reason for this is that this is the way the actual instance would be accessed. For example, to get to the Wales Regional Office webpage, the address would be http://www.cid.ac/home/regions/wales, and this address will bring you directly there, therefore the name of the page does not need to be known. This also makes it easier for instances to create their pages, as they can have multiple pages in their directory and have there own index page. The reason for naming the main page under each section index.htm or index.html is that the web server will automatically display the contents of that page when the directory is entered. The diagram is also colour coded based on the restriction levels of the directory. The colour blue represents a global area that anyone can access without a password. The yellow section represents the main password protected intranet. These directories include the /home and /cgi-bin, both use the same password file and

are synchronized. The red section of the website is restricted to administrators, this section uses a password file that is different from the yellow section.

By keeping the intranet under the /home directory, the entire structure below /home is protected by the same password. This also allows the first page seen when you enter www.cid.ac to be viewable by all and then you have the ability to log in. The directories that have been populated thus far with actual content is /home, /docs, /tools, /training, and /help. All other directories have had a temporary index file uploaded to them so that something will appear when the page is selected.

The /cgi-bin directory is where all Perl/cgi scripts are stored. The scripts will be referred to as cgi scripts from henceforth because that is the format that they must be stored in to be executed on the server, even though they are all written in Perl. Under the /cgi-bin directory, there is also a directory called /secure. This directory is for administrative scripts, they must be kept below the /cgi-bin directory in order to be recognised by the server and allowed to be executed.

The last main area is the /secure section, underneath the /secure directory are two subdirectories, logs, and staff_directory. The staff_directory is the directory in which the contact databases are stored and accessed from. These files can be updated and then stored back in to the same location. The logs directory has been set up for scripts that produce logs so that they can be seen and accessed. The /secure directory also has an index page that links all of the administrative tools to the rest of the website.

Now that the directory structure has been explained as well as the logic behind it, the next thing to know is how to get the files onto the server. The server, as mentioned earlier, is a Unix server run by clara.net. All the files must be stored on that machine. There are two ways of getting files on to the server; the first is through

an administrative script that is the same as the document repository except that it has more power and access. The second is through ftp. Both ways will be explained, as there are benefits to both.

The administrative version of the document repository is a powerful tool for manipulating the entire site. The tool has access to the entire site, its root level is www.cid.ac, and this means that it can see all files and directories below there. This also means that you can use this version to upload files into the different instance directories and to delete other files from the site. This is an easy way to navigate the site and manipulate from the web.

The second was of getting documents onto the server is through ftp. This is an easy way of moving files around as well. It would be recommend to use a gui-based ftp program as it makes it easier to manipulate and upload files than from a dos prompt, but it can be done either way. To ftp to the site, you would use the address ftp://www.cid.ac or use the www.cid.ac in a gui-based ftp program. When coming in from the ftp level, the first directory that you will see will be the \ directory. This is because you are using the Unix structure, you then have to switch into the docs directory and from there the Internal Connection structure can be seen. An explanation of the Unix structure will be cover as well.

Once the files have been uploaded to the appropriate directory, the next thing to do is to make sure that they have the correct permissions to be viewed. The best way to do this is to log in to the Unix servers of clara.net. To do this, you must telnet to the server, which can be accomplished by running the command telnet www.cid.ac, when prompted enter cid as the username and then the appropriate password. Once you are logged into the system, you will be in the root directory. There are four directories under the root directory, they are:

Docs – This is where the directory structure starts for the website

Logs – This is where access logs are kept for the web site

Bin – This directory host the binaries need to manipulate the Unix account

ftp – This is the area for allowing anonymous ftp access

Once logged in, you have the ability to manipulate files and directories. Here is a list of commands that are commonly used in Unix.

ls – This is the DOS dir equivalent, by using ls with the –la flag you will see all the file details.

cd – The same as in DOS, this allows you to change your current path.

chmod – This command allows you to change the properties of a file or directory, this is needed for making scripts executable or files visible. For more information type help chmod from the Unix prompt.

mkdir - This command is similar to the DOS command for creating a directory.

rmdir - This command is used to remove directories.

To make sure that the files have the correct permissions, you would change into the directory that you want to check and then execute the command ls –la. The resulting list will list the permission beside the name of the file. All html files should have -rw-r--r--, cgi files should have -rwx-r-xr-x and directories should have drwxr-xr-x or drwxrwxrwx. To make these changes to the file properties, you would use the chmod command, for example, to change the properties on a cgi file you would use the command chmod 755 filename.cgi or chmod a+rx filename.cgi. To change to

permissions on a n ht ml file you w ould us e t he command c hmod a+r f ile.html a nd chmod u+w file.html.

Once the files are in place and have the correct permissions, the files will be ready t o be vi ewed. T he nex t thing t o di scuss i s t he a ctual c ontent of t he files, of which there is two types. The two are the static html and the cgi files. The static html files, have the same basic layout. Figure 2 shows this layout, A is the top navigation bar that remains static on all of the pages. Part B is a navigation utility that is present on m ost pa ges a nd a llows a ccess t o di fferent i nstances on t he pa ges. P art D i s t he footer t hat i s on e very page, i t i ncludes t he s tandard di sclaimer, a nd c ontact information about the page. The last section, part C, is where the main content goes on every page. In the actual html, it is commented where each section actually lies.

<table>
<tr><td colspan="2">A</td></tr>
<tr><td>B</td><td>C</td></tr>
<tr><td colspan="2">D</td></tr>
</table>

Figure 2 - Page Layout

The cgi scripts are set up ve ry di fferently from the html pages. M any of t he scripts actually contain the code for the cgi scripts right in them. T he general layout for t he s cripts i s t o ha ve a configuration s ection a t t he t op t o c onfigure t he ur l references a nd ot her pa ths. S ome of t he s cripts do not ha ve t his section but a re commented, to where such changes would be needed. The best way to understand the scripts is just to look through the code and the comments.

Another item that should be kept consistent is the links. All the links throughout the site are referenced starting from /home, this prevents any problems that might arise from linking within a directory to another directory. This is also done so that all images are stored in one location. All images should be referenced by /Images/filename. If a directory is being referenced then the index file should not be included, for example if the Wales Regional Office webpage was to be linked from another page it should be done in the format /home/regions/wales/ and that will bring you to http://www.cid.ac/home/regions/wales/ and the index page for the Wales directory will be automatically displayed.

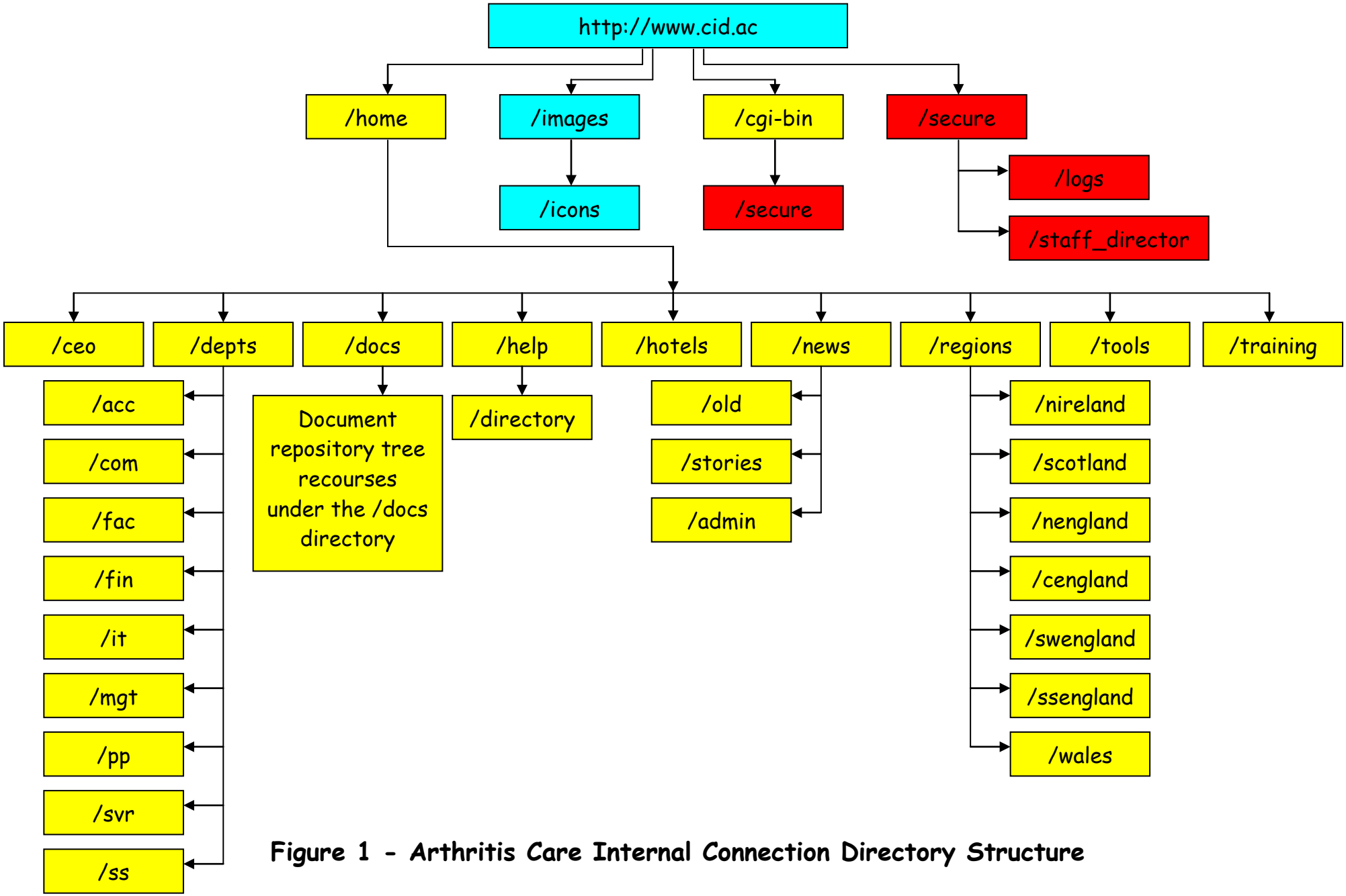


Figure 1 - Arthritis Care Internal Connection Directory Structure

Appendix I – Code for Intranet

```
---------------------- Code For Basic Web Page -----------------------------
<html>

<head>
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Arthritis Care Internal Connection</title>
</head>

<body>

<!—header →
<table border="0" width="100%" cellspacing="0" cellpadding="0">
  <tr>
    <td width="100%">
      <table border="0" width="100%" cellspacing="0" cellpadding="0">
        <tr>
          <td width="100%">
            <table border="0" width="100%" cellspacing="0" cellpadding="0">
              <tr>
                <td width="95%" background="/Images/tp_bg.jpg" height="100%">
                  <font size="6" face="Arial"> Arthritis
                  Care Internal Connection</font></td>
              </tr>
              <tr>
                <td width="95%" background="/Images/tp_bg.jpg" valign="bottom">
                 

                 </td>
              </tr>
              <tr>
                <td width ="95%" background="/Images/tp_bg.jpg">
                <form method="POST" action="/cgi-bin/dbsearch.cgi">
                  <IMG SRC="/Images/top_nav.gif" USEMAP="#top_nav.gif"
                  width=409 height=26 BORDER=0 align="bottom"><MAP
                  NAME="top_nav.gif">
                    <AREA shape=RECT coords="5, 0, 64, 25" HREF="/home/">
                    <AREA shape=RECT coords="68, 0, 138, 25"
                    href="/home/training/">
                    <AREA shape=RECT coords="150, 0, 197, 25"
                    href="/home/tools/">
                    <AREA shape=RECT coords="212, 0, 312, 25" HREF="/cgi-
                    bin/docs.cgi">
                    <AREA shape=RECT coords="323, 0, 408, 25"
                    HREF="/home/directory/">
                  </MAP>
                    <input type="text" name="name" size="16"><input
                    type="submit" value="Go" name="B1">
                  </form>
                </td>
            </table>
          </td>
          <td width="9%" valign="top" align="right"><img border="0"
          src="/Images/logo_top.gif" width="126" height="58"><br>
            <img border="0" src="/Images/logo_bot.gif" width="126"
             height="34"></td>
        </tr>
      </table>
    </td>
  </tr>
```

```
 <tr>
  <td width="100%"><img border="0" src="/Images/nb_top.jpg" width="100%"
  height="3"></td>
</tr>
<!-- End header, begin lefft navigation bar -->
<tr>
  <td width="100%">
    <table border="0" width="100%" cellspacing="0" cellpadding="0"
    height="416">
      <tr>
        <td width="170" bgcolor="#FFFF99" valign="top" height="345">
          <table border="0" width="170" cellspacing="0" cellpadding="0"
          height="283">
            <tr>
              <td width="170" valign="top" bgcolor="#FFFF8C"
              height="283"><br>
                <a href="/home/ceo/">
                Chief Executive Officer</a><br>
                <br>
                <a href="/home/depts/">
                Departments</a><br>
                  <a href="/home/depts/acc/">Accounts</a><br>
                  <a href="/home/depts/com/">Communications</a><br>
                  <a href="/home/depts/fac/">Facilities
                 Management</a><br>
                  <a href="/home/depts/fin/">Finance</a><br>
                  <a href="/home/depts/it/">Information
                 Technology</a><br>
                  <a href="/home/depts/pp/">Public Policy and</a><br>
                  <a href="/home/depts/pp/">Campaigning</a><br>
                  <a href="/home/depts/svr/">Services</a><br>
                  <a href="/home/depts/ss/">Support Services</a><br>
                <br>
                <a href="/home/regions/">Regions</a><br>
                  <a href="/home/regions/nireland/">Northern
                 Ireland</a><br>
                  <a href="/home/regions/scotland/">Scotland</a><br>
                  <a href="/home/regions/nengland/">North
                England</a><br>
                  <a href="/home/regions/cengland/">Central
                 England</a><br>
                  <a href="/home/regions/swengland/">South West
                 England</a><br>
                  <a href="/home/regions/seengland/">South East
                 England</a><br>
                  <a href="/home/regions/wales/">Wales</a><br>
                <br>
                <a href="/home/hotels/">Hotels</a><br>
                  
              </td>
            </tr>
          </table>
        </td>
        <td width="5" background="/Images/nb_side.jpg" rowspan="2"
        height="416"> </td>
        <!—End Left Navigation bar, Start main content section →
        <td width="100%" valign="top" rowspan="2" height="416">
          <table border="0" width="100%" cellspacing="1">
            <tr>
              <td width="100%">
                <table border="0" width="100%" cellspacing="2">
                  <tr>
                    <td width="100%">
```

```
            <table border="0" width="100%" cellspacing="0"
             cellpadding="0">
              <tr>
                <td width="50%">Organisation News</td>
                <td width="50%" align="right">Previous News</td>
              </tr>
            </table>
          </td>
        </tr>
        <tr>
          <td width="100%" bgcolor="#99CCFF">News1</td>
        </tr>
        <tr>
          <td width="100%" bgcolor="#99CCFF">News2</td>
        </tr>
        <tr>
          <td width="100%" bgcolor="#99CCFF">News3</td>
        </tr>
        <tr>
          <td width="100%">
            <p align="right">Submit a news story</p>
          </td>
        </tr>
      </table>
    </td>
  </tr>
  <tr>
    <td width="100%">
      <hr noshade color="#808080">
      <form method="POST" action="">
        <p align="left"><select size="1" name="D1">
          <option>Quick questions and links to there
           answers</option>
        </select><input type="submit" value="Submit"
        name="B1"><input type="reset" value="Reset" name="B2"></p>
      </form>
      <hr noshade color="#808080">
    </td>
  </tr>
  <tr>
    <td width="100%">
      <table border="0" width="100%" cellspacing="1" height="96">
        <tr>
          <td width="15%" valign="middle" height="67"><a
           href="/cgi-bin/docs.cgi"><img border="0"
           src="/Images/j0299535.gif" width="95" height="70"
           align="middle"></a> </td>
          <td width="35%" valign="middle" height="67"><a
           href="/cgi-bin/docs.cgi">Documents<br>
            </a>The document repository is available
           here </td>
          <td width="50%" height="67">
            <table border="0" width="100%" cellspacing="1">
              <tr>
                <td width="15%"><img border="0"
               src="/Images/j0158037.gif" width="95"
               height="70"></td>
                <td width="65%"><a
                  href="/home/training/">Training<br>
                  </a>Find various training manuals and
                  publications
                  here</td>
              </tr>
```

```
                    </table>
                  </td>
                </tr>
                <tr>
                  <td width="15%" valign="middle" height="21"><a
                    href="/home/directory/"><img border="0"
                    src="/Images/j0312176.gif" width="95" height="70"
                    align="middle"></a> </td>
                  <td width="35%" valign="middle" height="21"><a
                    href="/home/directory/">Directory<br>
                    </a>Find people and locations here</td>
                  <td width="50%" height="21">
                    <table border="0" width="100%" cellspacing="1">
                      <tr>
                        <td width="1%">
                          <p align="center"><img border="0"
                          src="/Images/j0234105.gif" width="95"
                          height="70"></td>
                        <td width="65%"><a href="/home/tools/">Tools<br>
                          </a>Here you will find tools to modify your
                          account</td>
                      </tr>
                    </table>
                  </td>
                </tr>
              </table>
            </td>
          </tr>
        </table>
      </td>
    </tr>
    <tr>
      <!—End Content, add search button to left navigation bar →
      <td width="170" bgcolor="#FFFF8C" height="71">
      Search:
              <form method="POST" action="">
                   <p><input type="text" name="T1" size="18"><input
                   type="submit" value="Go" name="B1"></p>
              </form>
      </td>
    </tr>
  </table>
</td>
</tr>
<tr>
  <td width="100%">
    <hr noshade color="#808080">
    Arthritis Care, Internal Use Only<br>
    <br>
    Maintained by: <a href="mailto:webmaster@cid.at">webmaster</a><br>
    Last modified:  <!--webbot bot="Timestamp" S-Type="REGENERATED"
    S-Format="%d %B %Y %H:%M" -->
  </td>
</tr>
</table>

</body>

</html>
```

```
---------------------- Code for document repository ----------------------------
#!/usr/bin/perl

# ==============================================================


# Required libraries
# ------------------------------------------------------
#   use strict;     # File uploads don't work with use strict in place, although script compiles with use strict.
    use vars qw(%config %icons $in);
    use CGI  qw(:cgi);
    $in = new CGI;

# Configuartion
# ------------------------------------------------------
    %config = (
            root_dir            => "/ccd0/vhosts/cid/docs/home/docs",
            logfile             => "/ccd0/vhosts/cid/docs/secure/logs/doc.log",
            password_dir        => "/secure",
            #root_url           => "http://127.0.0.1/docs",
            #script_url         => "http://127.0.0.1/cgi-bin/docs.cgi",
            #icondir_url        => 'http://127.0.0.1/icons',
            root_url            => "http://www.cid.ac/docs",
            script_url          => "http://www.cid.ac/cgi-bin/docs.cgi",
            icondir_url         => "http://www.cid.ac/Images/icons",
            use_flock           => 1,
            allowed_space       => 50000,
            max_upload          => 10000,
            show_size           => 0,
            show_date           => 1,
            show_perm           => 0,
            show_icon           => 1,
            show_pass           => 0,
            version             => '2.0'
    );

    %icons = (
            'gif jpg jpeg bmp'              => 'image2.gif',
            'txt'                   => 'quill.gif',
            'cgi pl'                => 'script.gif',
            'zip gz tar'                    => 'uuencoded.gif',
            'htm html shtm shtml'   => 'world1.gif',
            'wav au mid mod'        => 'sound1.gif',
            'doc'                           => 'Word.bmp',
              'mdb'                 => 'access.bmp',
              'xls csv'                     => 'Excell.bmp',
              'ppt'                         => 'Powerpoint.bmb',
              folder                        => 'folder.gif',
            parent                          => 'back.gif',
            unknown                         => 'unknown.gif'
    );
# ------------------------------------------------------

# ------------------------------------------------------
# Run the program and trap fatal errors.
    eval { &main; };
```

```
   if ($@) { &cgierr ("Fatal Error: $@"); }
# ----------------------------------------------------------

sub main {
# ============================================================
# 1. Get the form input, and print the HTTP headers.
#
   $|++;                    # Flush Output
   print $in->header('text/html');

   my ($working_dir)       = $in->param('wd');              # Our current working directory.
   my ($filename)          = $in->param('fn');              # Filename to edit, delete, etc.
   my ($name)              = $in->param('name');          # Org. filename to rename.
   my ($newname)           = $in->param('newname');     # New filename in rename.
   my ($directory)         = $in->param('dir');             # Directory to make/delete/change
   my ($newperm)           = $in->param('newperm');     # New permissions to set.
   my ($action)            = $in->param('action');          # Action to take.
   my ($user)              = $in->param('user');            # Username to add to password list.
   my ($pass)              = $in->param('pass');            # Password to add to password list.

# 2. Validate the form input. This makes sure any passed in information is valid. After this
#    the information is assumed safe.
   my ($error);
   ($working_dir, $error) = &is_valid_dir  ($working_dir); $error and &user_error ("Invalid Directory:
'$working_dir'. Reason: $error", "$config{'root_dir'}/$working_dir");
   ($filename,    $error) = &is_valid_file ($filename);    $error and &user_error ("Invalid Filename:
'$filename'. Reason: $error", "$config{'root_dir'}/$working_dir");
   ($name,        $error) = &is_valid_file ($name);        $error and &user_error ("Invalid Filename: '$name'.
Reason: $error", "$config{'root_dir'}/$working_dir");
   ($newname,     $error) = &is_valid_file ($newname);     $error and &user_error ("Invalid Filename:
'$newname'. Reason: $error", "$config{'root_dir'}/$working_dir");
   ($newperm,     $error) = &is_valid_perm ($newperm);     $error and &user_error ("Invalid Permissions:
'$newperm'. Reason: $error", "$config{'root_dir'}/$working_dir");
   ($user,        $error) = &is_valid_user ($user);        $error and &user_error ("Invalid Username: '$user'.
Reason: $!", "$config{'root_dir'}/$working_dir");
   ($pass,        $error) = &is_valid_user ($pass);        $error and &user_error ("Invalid Password: '$pass'.
Reason: $!", "$config{'root_dir'}/$working_dir");

# New directory name is special. It has to pass both a filename, and directory test.
   ($directory, $error)   = &is_valid_dir  ($directory);   $error and &user_error ("Invalid Directory:
'$directory'. Reason: $error", "$config{'root_dir'}/$working_dir");
   ($directory, $error)   = &is_valid_file ($directory);   $error and &user_error ("Invalid Directory:
'$directory'. Reason: $error", "$config{'root_dir'}/$working_dir");

# 3. Set the current working directory, and current working url.
   my ($dir, $url);
   if ($working_dir) {
      $dir       = "$config{'root_dir'}/$working_dir";
      $url       = "$config{'root_url'}/$working_dir";
   }
   else {
      $dir       = $config{'root_dir'};
      $url       = $config{'root_url'};
   }

# 4. Print HTML intro.
```

```
# Javascript form validation.
   my $javascript = qq~
<script language="Javascript">
<!-- Hide from old browsers
   function validateFileEntry(validString, field) {
      var isCharValid = true;
      var i, invalidChar;
      for (i=0; i<validString.length; i++) {
         if (validString.charAt(0) == '.') {
            isCharValid = false;
            validString = validString.substr(1, validString.length-1);
            i = validString.length;
         }
         if (validateCharacter(validString.charAt(i)) == false) {
            isCharValid = false;
            invalidChar = validString.charAt(i);
            validString = validString.substr(0, i) + validString.substr(i+1, validString.length-1);
            i = validString.length;
         }
      }
      if (i < 1) { return false; }
      if (isCharValid == false) {
         if (invalidChar) alert("Invalid filename. Can't contain '" + invalidChar + "'. Filename adjusted.");
         else alert('Invalid filename. Filename adjusted.');
         if (field) {
            field.value = validString;
            field.focus();
            field.select();
         }
         return false;
      }
      return true;
   }

   function validateCharacter(character) {
      if ((character >= 'a' && character <= 'z') || ( character >='A' && character <='Z') || ( character >= '0'
&& character <= '9') || ( character =='-') || ( character == '.') || ( character == '_')) return true;
      else return false;
   }

   function isNum(passedVal) {
      if (!passedVal) { return false  }
      for (i=0; i<passedVal.length; i++) {
         if (passedVal.charAt(i) < "0") { return false }
         if (passedVal.charAt(i) > "7") { return false }
      }
      return true
   }

   function renameFile ( name ) {
      var newname = window.prompt("Rename '" + name + "' to: ","")
      if (newname != null) {
         if (validateFileEntry(newname)) {
            window.location.href = "$config{'script_url'}?action=rename&name=" + name + "&newname="
+ newname +"&wd=$working_dir"
```

```
        }
      }
   }

   function deleteFile ( name ) {
      if (window.confirm("Are you sure you want to delete '" + name + "'")) {
         window.location.href = "$config{'script_url'}?action=delete&fn=" + name + "&wd=$working_dir"
      }
   }

   function deleteDir ( name ) {
      if (window.confirm("Are you sure you want to delete the directory '" + name + "'")) {
         window.location.href = "$config{'script_url'}?action=removedir&dir=" + name +
"&wd=$working_dir"
      }
   }

   function changePermissions ( name ) {
      var newperm = window.prompt("Change file permissions for '" + name + "' to: ","")
      if (newperm == null) {  return; }
      if (!isNum(newperm) || (newperm == "") || (length.newperm > 2)) {
         alert ("Three digits only please! Enter the permissions in octal. EG 766.")
      }
      else {
         window.location.href = "$config{'script_url'}?action=permissions&name=" + name +
"&newperm=" + newperm +"&wd=$working_dir"
      }
   }

   function serverFileName() {
      var fileName = window.document.Upload.data.value.toLowerCase();
      window.document.Upload.fn.value = fileName.substring(fileName.lastIndexOf("\\\\") +
1,fileName.length);
   }

// -->
</script>
   ~;

# Text to be displayed if the user does not have Javascript turned on.
   my $nojavascript = qq~
      <noscript>
      <table border=0 bgcolor="#FFFFFF" cellpadding=5 cellspacing=3 width=100% valign=top>
      <tr>
      <td><font color="red"><B>Stop:   </B></font><FONT COLOR="black">

      Your browser must have <font color="red"><b>JavaScript turned on</b></font> -- The Document
Repositary uses JavaScript.
      Please open your browser preferences, and <b>enable JavaScript</b>. You can then click on the
<b>Reload</b> button and use the Document Repositary.
      </FONT></td></tr></table>
      </noscript>
   ~;

# Print the HTML Header.
   print qq~
```

```
<html>
<head>
<title>Arthritis Care Document Repository $config{'version'}</title>
$javascript
</head>
<body bgcolor="#ffffff">
<table border="0" width="100%" cellspacing="0" cellpadding="0">
  <tr>
    <td width="100%">
      <table border="0" width="100%" cellspacing="0" cellpadding="0">
        <tr>
          <td width="100%">
            <table border="0" width="100%" cellspacing="0" cellpadding="0">
              <tr>
                <td width="95%" background="/Images/tp_bg.jpg" height="100%">
                  <font size="6" face="Arial"> Arthritis
                  Care Internal Connection</font></td>
              </tr>
              <tr>
                <td width="95%" background="/Images/tp_bg.jpg" valign="bottom">
                 

                                        </td>
              </tr>
              <tr>
                <td width ="95%" background="/Images/tp_bg.jpg">
                <form method="POST" action="/cgi-bin/dbsearch.cgi">
                <IMG SRC="/Images/top_nav.gif" USEMAP="#top_nav.gif" width=409 height=26 BORDER=0
align="bottom"><MAP NAME="top_nav.gif">
                                                <AREA shape=RECT coords="5, 0, 64, 25"
HREF="/home/">
                                                        <AREA shape=RECT coords="68, 0, 138, 25"
href="/home/training/">
                                                        <AREA shape=RECT coords="150, 0, 197, 25"
href="/home/tools/">
                                                        <AREA shape=RECT coords="212, 0, 312, 25"
HREF="/cgi-bin/docs.cgi">
                                                        <AREA shape=RECT coords="323, 0, 408, 25"
HREF="/home/directory/">
                                                </MAP>
                  <input type="text" name="name" size="16"><input type="submit" value="Go" name="B1">
                </form>
                        </td>
            </table>
          </td>
          <td width="9%" valign="top" align="right"><img border="0" src="/Images/logo_top.gif"
width="126" height="58"><br>
            <img border="0" src="/Images/logo_bot.gif" width="126" height="34"></td>
        </tr>
      </table>
    </td>
  </tr>
  <tr>
    <td width="100%"><img border="0" src="/Images/nb_top.jpg" width="100%" height="3"></td>
  </tr>
  <tr>
```

```
   <td width="100%">
    <table border="0" width="100%" cellspacing="0" cellpadding="0" height="416">
     <tr>
      <td width="170" bgcolor="#FFFF99" valign="top" height="345">
       <table border="0" width="170" cellspacing="0" cellpadding="0" height="283">
        <tr>
         <td width="170" valign="top" bgcolor="#FFFF8C" height="283"><br>
          <a href="/home/ceo/">
          Chief Executive Officer</a><br>
          <br>
          <a href="/home/depts/">
          Departments</a><br>
            <a href="/home/depts/acc/">Accounts</a><br>
            <a href="/home/depts/com/">Communications</a><br>
            <a href="/home/depts/fac/">Facilities Management</a><br>
            <a href="/home/depts/fin/">Finance</a><br>
            <a href="/home/depts/it/">Information Technology</a><br>
            <a href="/home/depts/pp/">Public Policy and</a><br>
            <a href="/home/depts/pp/">Campaigning</a><br>
            <a href="/home/depts/svr/">Services</a><br>
            <a href="/home/depts/ss/">Support Services</a><br>
          <br>
          <a href="/home/regions/">Regions</a><br>
            <a href="/home/regions/nireland/">Northern Ireland</a><br>
            <a href="/home/regions/scotland/">Scotland</a><br>
            <a href="/home/regions/nengland/">North England</a><br>
            <a href="/home/regions/cengland/">Central England</a><br>
            <a href="/home/regions/swengland/">South West England</a><br>
            <a href="/home/regions/seengland/">South East England</a><br>
            <a href="/home/regions/wales/">Wales</a><br>
          <br>
          <a href="/home/hotels/">Hotels</a><br>
            
         </td>
        </tr>
       </table>
      </td>
<!--
      <td width="170" bgcolor="#FFFF8C" height="71">
      Search:
           <form method="POST" action="">
             <p><input type="text" name="T1" size="18"><input type="submit" value="Go"
name="B1"></p>
           </form>
      </td>
-->

      <td width="5" background="/Images/nb_side.jpg" rowspan="2"> </td>
      <td width="100%" valign="top" rowspan="2">
<!--
    <table border=1 bgcolor="#FFFFFF" cellpadding=2 cellspacing=1 width="630" align=center
valign=top>
     <tr> <td bgcolor="white" align=left><a href="javascript:history.go(-1)"><font face="Verdana, Arail"
size=2><b>Back</b></font></a></td>
        <td bgcolor="navy"  align=center width=90%><font color="white" face="Verdana, Arail"
size=3><b>File Manager $config{'version'}</b></font></td>
```

```
        <td bgcolor="white" align=right><a href="$config{'script_url'}"><font face="Verdana, Arail"
size=2><b>Root</b></font></a></td>
      </tr>
    </table>
-->
          <br>
    <table border=0 bgcolor="#FFFFFF" cellpadding=2 cellspacing=1 width="450" align=center
valign=top>
      <tr> <!-- <td bgcolor="white" align=left><a href="javascript:history.go(-1)"><font face="Verdana,
Arail" size=2><b>Back</b></font></a></td> -->
          <td bgcolor="white"  align=center width=90%><font color="black" face="Verdana, Arail"
size=3><b>Arthritis Care Document Repository $config{'version'}</b></font></td>
          <!-- <td bgcolor="white" align=right><a href="$config{'script_url'}"><font face="Verdana, Arail"
size=2><b>Root</b></font></a></td> -->
      </tr>
    </table>
<table border="1" bgcolor="#FFFFFF" cellpadding="2" cellspacing="1" width="450" align="center"
valign="top">
      <tr><td>
~;

# 5. Figure out what to do.
   my ($result, @disk_space);
   CASE: {
      ($action eq 'write')        and do {
                                    @disk_space = &checkspace($config{'root_dir'});
                                    if ($disk_space[0] < 50) { &delete_only_error; }
                                    else {
                                       $result = &write ($dir, $filename, $in->param('data'), $url);
                                       &list_files ($result, $working_dir, $url, @disk_space);
                                    }
                                    &log_action ($result, $dir) if ($config{'logfile'});
                                    last CASE;
                                  };
      ($action eq 'delete')       and do {
                                    $result = &delete ($dir, $filename);
                                    @disk_space = &checkspace ($config{'root_dir'});
                                    &list_files ($result, $working_dir, $url, @disk_space);
                                    &log_action ($result, $dir) if ($config{'logfile'});
                                    last CASE;
                                  };
      ($action eq 'makedir')      and do {
                                    @disk_space = &checkspace($config{'root_dir'});
                                    if ($disk_space[0] < 50) { &delete_only_error; }
                                    else {
                                       $result = &makedir    ($dir, $directory);
                                       &list_files ($result, $working_dir, $url, @disk_space);
                                       &log_action ($result, $dir) if ($config{'logfile'});
                                    }
                                    last CASE;
                                  };
      ($action eq 'removedir')    and do {
                                    @disk_space = &checkspace($config{'root_dir'});
                                    $result = &removedir  ($dir, $directory);
                                    &list_files ($result, $working_dir, $url, @disk_space);
                                    &log_action ($result, $dir) if ($config{'logfile'});
```

```
                                last CASE;
                            };
    ($action eq 'rename')     and do {
                                @disk_space = &checkspace($config{'root_dir'});
                                $result = &rename_file ($dir, $name, $newname);
                                &list_files   ($result, $working_dir, $url, @disk_space);
                                &log_action ($result, $dir) if ($config{'logfile'});
                                last CASE;
                            };
    ($action eq 'edit')       and do {
                                @disk_space = &checkspace($config{'root_dir'});
                                if ($disk_space[0] < 50) { &delete_only_error; }
                                else { &edit ($dir, $filename, $working_dir, $url); }
                                last CASE;
                            };
    ($action eq 'upload')     and do {
                                @disk_space = &checkspace($config{'root_dir'});
                                if ($disk_space[0] < 50) { &delete_only_error; }
                                else {
                                   my $file_space;
                                   ($file_space, $result) = &upload ($dir, $in->param('data'), $filename,
$disk_space[0]);
                                   $disk_space[0] -= $file_space; $disk_space[2] += $file_space;
                                   &list_files ($result, $working_dir, $url, @disk_space);
                                   &log_action ($result, $dir) if ($config{'logfile'});
                                }
                                last CASE;
                            };
    ($action eq 'permissions')  and do {
                                if ($config{'show_perm'}) {
                                   @disk_space = &checkspace($config{'root_dir'});
                                   $result = &change_perm ($dir, $name, $newperm);
                                   &list_files ($result, $working_dir, $url, @disk_space);
                                   &log_action ($result, $dir) if ($config{'logfile'});
                                   last CASE;
                                }
                            };
    ($action eq 'protect_form') and do {
                                if ($config{'show_pass'}) {
                                   &protect_form ($working_dir, $directory, '');
                                   last CASE;
                                }
                            };
    ($action eq 'add_user')    and do {
                                if ($config{'show_pass'}) {
                                   $result = &add_user ($user, $pass, $working_dir, $directory);
                                   &protect_form ($working_dir, $directory, $result);
                                   &log_action ($result, $dir) if ($config{'logfile'});
                                   last CASE;
                                }
                            };
    ($action eq 'remove_user')  and do {
                                if ($config{'show_pass'}) {
                                   $result = &remove_user ($user, $working_dir, $directory);
                                   &protect_form ($working_dir, $directory, $result);
                                   &log_action ($result, $dir) if ($config{'logfile'});
```

```
                                    last CASE;
                                }
                            };
# Default Case
        do {
            @disk_space = &checkspace($config{'root_dir'});
            print $nojavascript;
            &list_files ('File and Directory Listing.', $working_dir, $url, @disk_space);
        };
    };

# 6. Wrap up and print the last of the HTML.
    print qq~
                </td></tr>
            <tr><td><table border=0 width=100%>
                    <tr>
                                        <!--#config timefmt="%d-%b-%Y at %T"-->
                                        <td align=left><b><font color="#888888" size=1 face="Verdana,
Arial">Last updated: <!--#echo var="LAST_MODIFIED"--></font></b></td>
                    </tr>
                </table>
            </td></tr>


        </table>

          </td>
        </tr>
      </table>
    </td>
  </tr>
</table>

<table border=0 width=100%>
  <tr>
    <td width="100%">
      <hr noshade>
      Arthritis Care, Internal Use Only<br>
      <br>
      Maintained by: <a href="mailto:webmaster\@cid.ac">webmaster</a><br>
      Last modified:  <!--webbot bot="Timestamp" S-Type="REGENERATED"
      S-Format="%d %b %Y %H:%M" -->
    </td>
  </tr>
</table>

        </body>
</html>
    ~;
}
# ==========================================================================

sub list_files {
# ---------------------------------------------------
# Displays a list of files for a given directory.
#
```

```
   my ($message, $working_dir, $url, @disk_space) = @_;
   my ($directory)   = "$config{'root_dir'}/$working_dir";
   my ($diskUsage)   = "'Disk Usage:\\n\\nAllowed disk space:  $disk_space[1] kb\\nDisk space
used:      $disk_space[2] kb\\n\\nDisk space
free:      $disk_space[0] kb'";

# Print out table header with free disk space.
   print qq~
      <P>
      <table border=0 bgcolor="#FFFFFF" cellpadding=5 cellspacing=3 width=100% valign=top>
         <tr>
                  <td><B>Contents of:   <FONT COLOR="blue">$url</font></B></td>
           <!-- <td><B>Contents of:   <a href="$url"><FONT
COLOR="blue">$url</font></A></B></td> -->
           <!-- <td align="right"><B><a href="javascript:alert($diskUsage)"><font color="blue">Disk
usage</font></a></B></td> -->
         </tr>
         <!-- <tr>
           <td><b>Action: <font color=red>$message</font><br></td>
           <td align="right"><b><B>Free space: $disk_space[0] kb </B></td>
         </tr> -->
      </table>
   </td></tr>
   <tr><td>
      <P>
      <table border=0 bgcolor="#FFFFFF" cellpadding=5 cellspacing=3 width=100% valign=top>
   ~;

# Get the list of files using readdir.
   opendir (DIR, $directory) or &cgierr ("Can't open dir: '$directory'.\nReason: $!");
   my @ls = readdir(DIR);
   closedir (DIR);

# Then go through the results of ls and work out the files..
   my (%directory, %text, %graphic);
   my ($temp_dir, $newdir, @nest, $fullfile, $filesize, $filedate, $fileperm, $fileicon, $file);

   FILE: foreach $file (@ls) {
# Skip the "." entry and ".." if we are at root level.
      next FILE if  ($file eq '.');
      next FILE if (($file eq '..') and ($directory eq "$config{'root_dir'}/"));

# Get the full filename, file size, file modification date and file permissions.
      $fullfile = "$directory/$file";
      ($filesize, $filedate, $fileperm) = (stat($fullfile))[7,9,2];
      $fileperm = &print_permissions ($fileperm)      if ($config{'show_perm'});
      $filesize = &print_filesize    ($filesize)      if ($config{'show_size'});
      $filedate = &get_date($filedate)                if ($config{'show_date'});

      if (-d $fullfile ) {
# Let's work out the relative path if it is a directory.
         if ($file eq '..') {
            @nest = split (/\//, $working_dir);
            (pop (@nest)) ?
               ($newdir = "$config{'script_url'}?wd=" . join ("/", @nest)) :
               ($newdir = "$config{'script_url'}");
```

```
        }
        else {
           $working_dir ? ($temp_dir = "$working_dir%2F$file") : ($temp_dir = "$file");
           $newdir   = "$config{'script_url'}?wd=$temp_dir";
        }
        $newdir = $in->escapeHTML($newdir);
# .. directory
        if ($file eq '..') {
           $fileicon = "$config{'icondir_url'}/$icons{'parent'}"  if ($config{'show_icon'});
           $directory{$file}  = qq~ <tr>\n~;
           $directory{$file} .= qq~     <td width="25"><b><a href="$newdir"><img src="$fileicon"
align=middle border=0></a></td> \n~ if ($config{'show_icon'});
#             $directory{$file} .= qq~     <td><a href="$url/$file"><font
color=blue>$file</font></a></b></td> \n~;
           $directory{$file} .= qq~     <td><a href="$newdir"><font color=blue>$file</font></a></b></td>
\n~;
                     $directory{$file} .= qq~     <td><b><tt><a
href="javascript:changePermissions('$file')"><font color="gray" size=1>$fileperm</font></a></b></td>
\n~ if ($config{'show_perm'});
           $directory{$file} .= qq~     <td><b><tt><font size=1>$filedate</font></tt></b></td> \n~ if
($config{'show_date'});
           $directory{$file} .= qq~     <td></td>~;
#             $directory{$file} .= qq~     <td><b><a href="$newdir"><font
color=black>chdir</font></a></B></td>;
           $directory{$file} .= qq~     <td><br></td></tr>~;
        }
# Regular directory.
        else {
           $fileicon = "$config{'icondir_url'}/$icons{'folder'}"  if ($config{'show_icon'});;
           $directory{$file}  = qq~ <tr>\n~;
           $directory{$file} .= qq~     <td width="25"><b><a href="$newdir"><img src="$fileicon"
align=middle border=0></a></td> \n~ if ($config{'show_icon'});
#             $directory{$file} .= qq~     <td><a href="$url/$file"><font
color=blue>$file</font></a></b></td> \n~;
                     $directory{$file} .= qq~     <td><a href="$newdir"><font
color=blue>$file</font></a></b></td> \n~;
           $directory{$file} .= qq~     <td><b><tt><a href="javascript:changePermissions('$file')"><font
color="gray" size=1>$fileperm</font></a></b></td> \n~ if ($config{'show_perm'});
           $directory{$file} .= qq~     <td><b><tt><font size=1>$filedate</font></tt></b></td> \n~ if
($config{'show_date'});
           $directory{$file} .= qq~     <td></td>~;
          # $directory{$file} .= qq~     <td><b><a href="$newdir"><font
color=black>chdir</font></a></b></td>\n~;
           $directory{$file} .= qq~     <td width="175"><b><a href="javascript:deleteDir('$file')"><font
color=red>Delete this directory</font></A></b></td>\n~;
           $directory{$file} .= qq~     <td><b><a
href="$config{'script_url'}?action=protect_form&wd=$working_dir&dir=$file"><font
color=brown>pass</font></A></b></td>\n~ if ($config{'show_pass'});
           $directory{$file} .= qq~ </tr>\n~;
        }
      }
# Text Files.
      elsif (-T $fullfile) {
         $fileicon = &get_icon($fullfile) if ($config{'show_icon'});
         $text{$file}  = qq~  <tr>\n~;
```

```
        $text{$file} .= qq~     <td width="25"><b><a href="$url/$file"><img src="$fileicon" align=middle
border=0></a></td> \n~ if ($config{'show_icon'});
        $text{$file} .= qq~     <td><a href="$url/$file"><font color=blue>$file</font></a></b></td> \n~;
        $text{$file} .= qq~     <td><b><tt><a href="javascript:changePermissions('$file')"><font
color="gray" size=1>$fileperm</font></a></b></td> \n~ if ($config{'show_perm'});
        $text{$file} .= qq~     <td><b><tt><font size=1>$filedate</font></tt></b></td> \n~ if
($config{'show_date'});
        $text{$file} .= qq~     <td><b><tt><font size=1>$filesize</font></tt></b></td> \n~ if
($config{'show_size'});
        ($disk_space[0] > 150) ?
           ($text{$file} .= qq~
                        <!-- <td><b><a
href="$config{'script_url'}?action=edit&fn=$file&wd=$working_dir"><font
color=green>edit</font></a></b></td> --> <td><b><font color=white></font></b></td>
           ~) :
           ($text{$file} .= qq~
                        <td><br></td>
           ~);
        $text{$file} .= qq~
                        <!-- <td width="175"><b><a href="javascript:deleteFile('$file')"><font
color=red>Delete this file</font></a></b></td> -->
                        <td width="150"><b><a href="javascript:renameFile('$file')"><font
color=purple>Rename this file</font></a></b></td></tr>
        ~;
      }
# Binary Files.
      else {
        $fileicon = &get_icon($fullfile) if ($config{'show_icon'});
        $graphic{$file}  = qq~  <tr>\n~;
        $graphic{$file} .= qq~      <td width="25"><b><a href="$url/$file"><img src="$fileicon"
align=middle border=0></a></td> \n~ if ($config{'show_icon'});
        $graphic{$file} .= qq~      <td><a href="$url/$file"><font color=blue>$file</font></a></b></td>
\n~;
        $graphic{$file} .= qq~      <td><b><tt><a href="javascript:changePermissions('$file')"><font
color="gray" size=1>$fileperm</font></a></b></td> \n~ if ($config{'show_perm'});
        $graphic{$file} .= qq~      <td><b><tt><font size=1>$filedate</font></tt></b></td> \n~ if
($config{'show_date'});
        $graphic{$file} .= qq~      <td><b><tt><font size=1>$filesize</font></tt></b></td> \n~ if
($config{'show_size'});
        $graphic{$file} .= qq~      <td><br></td>
                          <!--  <td width="175"><b><a href="javascript:deleteFile('$file')"><font
color=red>Delete this file</font></a></b></td> -->
                            <td width="150"><b><a href="javascript:renameFile('$file')"><font
color=purple>Rename this file</font></a></b></td></tr>
        ~;
      }
   }
   foreach (sort keys %directory) {
      print $directory{$_};
   }
   foreach (sort keys %text) {
      print $text{$_};
   }
   foreach (sort keys %graphic) {
      print $graphic{$_};
   }
```

```
# Print the footer.
   if ($disk_space[0] < 50) {
      print qq~
         </table>
         <p><blockquote>
         <b>You are running out of disk space. Please delete some files before
         creating new ones.</b></blockquote></p>~;
   }
   else {
      print qq~
         </table>
      </td></tr>
      <tr><td>
         <table border=0 cellpadding=5 cellspacing=3 width="575" valign=top>
            <!--
                <tr><td align="left" valign="top" width=50%>
               <form method=post action="$config{'script_url'}" name="createfile">
                  <input type=hidden name="action" value="edit">
                  <input type=hidden name="wd"     value="$working_dir">
                  <font color="black"><B>Create a new document:</B><br>
                     Filename:<br> <input type=text name="fn" onBlur="validateFileEntry(this.value, this)"
><br>
                  <input type=submit value="Create file"></font>
               </form>
                     -->
            </td><td align="left" rowspan=2 valign="top" width="300">
               <form method=post action="$config{'script_url'}">
                  <input type=hidden name="action" value="makedir">
                  <input type=hidden name="wd"     value="$working_dir">
                  <font color="black"><B>Create a new directory:</B><br>
                     Name:<br> <input type=text name="dir" onBlur="validateFileEntry(this.value, this)" >
                  <input type=submit value="Create directory"></font>
               </form>
            </td></tr><tr><td valign="top" align="left">
               <form method=post action="$config{'script_url'}" NAME="Upload"
ENCTYPE="multipart/form-data">
                  <input type=hidden name="wd"     value="$working_dir">
                  <input type=hidden name="action" value="upload">
                  <font color="black"><B>Upload a File:</B><br>
                     Local filename:
                     <INPUT NAME="data" TYPE="file" onBlur="serverFileName()"><br>
                     Remote filename:<br> <INPUT NAME="fn" onFocus="select()"
onBlur="validateFileEntry(this.value, this)">
                  <input type="submit" value="Upload"></font>
               </form>
            </td></tr>
         </table>
      ~;
   }
} # End List Files Procedure.

sub delete {
# ----------------------------------------------------
# Begin Delete File Procedure:
#
```

```
   my ($directory, $filename) = @_;
   my ($fullfile);

# Check to make sure a file name was entered.
   (!$filename) and return "Delete File: No filename was entered!";

# Get the full path to the file.
   ($directory =~ m,/$,) ? ($fullfile = "$directory$filename") : ($fullfile = "$directory/$filename");

# Delete it if it exists.
   if (&exists($fullfile)) {
      unlink ($fullfile) ?
         return "Delete File: '$filename' was removed." :
         return "Delete File: '$filename' could not be deleted. Check file permissions.";
   }
   else {
      return "Delete File: '$filename' could not be deleted. File not found.";
   }
}

sub edit {
# --------------------------------------------------
# Begin Edit Text File Procedure:
#
   my ($directory, $filename, $working_dir, $url) = @_;
   my ($lines, $fullfile, $full_url);

# Check to make sure a file name was entered.
   (!$filename) and return "Edit File: No filename was entered!";

# Build full file name and full url.
   ($directory =~ m,/$,) ? ($fullfile = "$directory$filename") : ($fullfile = "$directory/$filename");
   $full_url   = "$url/$filename";

# Either load the contents from a file..
   if (&exists($fullfile)) {
      open (DATA, "<$fullfile") or &cgierr ("Can't open '$fullfile'\nReason: $!");
      $lines = join ("", <DATA>);
      $lines =~ s/<\/TEXTAREA/<\/TEXT-AREA/ig;
      close DATA;
      print qq!<p>Modify <a href="$full_url"><B>$filename</B></A> as needed:</p>!;
   }
   else {
# Or use the following as a template.
      $lines = qq~
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>

<BODY BGCOLOR="#FFFFFF" TEXT="#000000" LINK="#FF0000" VLINK="#800000"
ALINK="#FF00FF">

</BODY>
</HTML>
```

```
        ~;
        print "<p>This is a new file. Input your HTML below:</p>";
    }

# Print out the editing and saving options.
    print qq~
        <p><blockquote>
            After editing, select "Save Document" to save <B>$filename</B> and return to
            the main menu.
        </blockquote></p>

        <form method=post action="$config{'script_url'}">
        <textarea name="data" rows=40 cols=60 wrap=virtual>$lines</textarea>

        <p>Alternate Filename:
            <input type=text name="fn" value="$filename"><br>
                (entering an alternate filename will leave <B>$filename</B>
                untouched and will place the text above into a file with the
                alternate name. Note that if a file already exists with the alternate
                filename, you will overwrite it completely.)<P>
            <input type=hidden name="action" value="write">
            <input type=hidden name="wd"     value="$working_dir">
            <input type=submit               value="Save Document">
        </form>
        </p>
    ~;
}

sub write {
# --------------------------------------------------
# Begin Write Text File Procedure:
#
    my ($directory, $filename, $data, $url) = @_;
    my ($fullfile, $new);

# Make sure a filename was passed in.
    (!$filename) and return "Edit File: No filename was entered!";

# Get the full path.
    ($directory =~ m,/$,) ? ($fullfile = "$directory$filename") : ($fullfile = "$directory/$filename");

# Check to see if this is a new or existing file.
    $new = 1;
    (&exists($fullfile)) and ($new = 0);

# Fix textarea tags.
    $data =~ s,</TEXT-AREA,</TEXTAREA,ig;

# Write the file to the system.
    open(FILE,">$fullfile") or &cgierr ("Can't open: '$fullfile'.\nReason: $!");
        print FILE $data;
    close(FILE);

    if (&exists($fullfile)) {
        ($new eq 'yes') ?
            return ("Edit File: '$filename' has been created.") :
```

```
        return ("Edit File: '$filename' has been edited.");
    }
    else {
        return  ("Edit File: Cannot save '$filename'. Check permissions.");
    }
}

sub upload {
# ----------------------------------------------------
# Begin Upload File Procedure:

    my ($directory, $data, $filename, $free_space) = @_;
    my ($bytesread, $buffer, $fullfile, $file_size);

# Make sure we have a filename to upload.
    (!$filename) and return (0, "Upload: No filename was entered!");

# Get the full file name.
    ($directory =~ m,/$,) ?
        ($fullfile = "$directory$filename") :
        ($fullfile = "$directory/$filename");
    $file_size = 0;

# Open the output file and save the upload. We abort if the file is
# to big, or not enough free disk space.
    open    (OUTFILE, ">$fullfile") or &cgierr ("Can't open: '$fullfile'.\nReason: $!");
    binmode (OUTFILE);  # For those O/S that care.
    while ($bytesread=read($data,$buffer,1024)) {
        ($fullfile =~ /cgi|pl$/) and ($buffer =~ s/\r//g);
        print OUTFILE $buffer;
        $file_size += 1024;
        if (($file_size / 1000) > $free_space) {
            close OUTFILE;
            unlink ($fullfile) or &cgierr ("Can't unlink: $fullfile. Reason: $!");
            return (0, "Upload: Not enough free space to upload that file. Space left: $free_space kb.");
        }
        if (($file_size / 1000) > $config{'max_upload'}) {
            close OUTFILE;
            unlink ($fullfile) or &cgierr ("Can't unlink: $fullfile. Reason: $!");
            return (0, "Upload: Aborted as your file is larger then the maximum uploadable file size of
$config{'max_upload'} kb!");
        }
    }
    close OUTFILE;
    &exists($fullfile) ?
        return (int($file_size / 1000), "Upload: '$filename' uploaded.") :
        return (int($file_size / 1000), "Upload: Cannot upload '$filename'. Check permissions.");
}

sub makedir {
# ----------------------------------------------------
# Begin Make Directory Procedure:
#
    my ($root, $new) = @_;
    my ($fulldir);
```

```
# Make sure we have a directory name.
   (!$new) and return "Make Directory: You forgot to enter in a directory name!";

# Get the full path.
   ($root =~ m,/$,) ? ($fulldir = "$root$new") : ($fulldir = "$root/$new");

# Create the directory unless it already exists.
   if (&exists($fulldir)) {
      return "Make Directory: '$new' already exists.";
   }
   else {
      mkdir ($fulldir, 0755) ?
         return "Make Directory: '$new' directory created." :
         return "Make Directory: Unable to create the directory. Check permissions.";
   }
}

sub removedir {
# ----------------------------------------------------
# Removes a directory.
#
   my ($root, $new) = @_;
   my ($fulldir);

# Make sure we have a directory name to delete.
   (!$new) and return "Remove Directory: No directory name was entered!";

# Get the full directory.
   ($root =~ m,/$,) ? ($fulldir = "$root$new") : ($fulldir = "$root/$new");

# Then remove if possible.
   if (!&exists($fulldir)) {
      return "Remove Directory: '$new' does not exists.";
   }
   else {
      rmdir($fulldir) ?
         return "Remove Directory: '$new' has been removed." :
         return "Remove Directory: '$new' was <B>not</B> removed. Check that the directory is empty.";
   }
}

sub rename_file {
# ----------------------------------------------------
# Renames a file using perls rename() function.
#
   my ($directory, $oldfile, $newfile) = @_;

# Make sure we have both an old name and a new name.
   (!$oldfile or !$newfile) and return "Rename: Both a source and destination file must be entered!";

# Get the full path of each file.
   my ($full_oldfile, $full_newfile);
   ($directory =~ m,/$,) ?
      ($full_oldfile = "$directory$oldfile"  and $full_newfile = "$directory$newfile") :
      ($full_oldfile = "$directory/$oldfile" and $full_newfile = "$directory/$newfile");
```

```
# Make sure the oldfile exists, and the new file doesn't.
   (&exists($full_oldfile)) or  return "Rename: Old file '$oldfile' does not exist.";
   (&exists($full_newfile)) and return "Rename: New file '$newfile' already exists.";

# Rename.
   rename ($full_oldfile, $full_newfile) or &cgierr("Unable to rename '$full_oldfile' to '$full_newfile'.
Reason: $!");
   return "Rename: '$oldfile' has been renamed '$newfile'.";
}

sub change_perm {
# ------------------------------------------------------------
# Changes the permission attributes of a file
#
   my ($directory, $file, $newperm) = @_;
   my ($full_filename, $octal_perm);

# Make sure we have both a filename and a permission.
   (!$file)    and return "Change Permissions: No file entered!";
   (!$newperm) and return "Change Permissions: No new permissions entered!";

# Check to make sure the file exists.
   $full_filename = "$directory/$file";
   (&exists($full_filename)) or return "Change Permissions: '$file' does not exist.";

# Permissions have to be in octal.
   $octal_perm = oct($newperm);
   chmod ($octal_perm, $full_filename) or &cgierr("Unable to change permissions for '$file' to '$newperm'.
Reason: $!");
   return "Change Permissions: '$file' permissions have been changed.";
}

sub print_permissions {
# ------------------------------------------------------------
# Takes permissions in octal and prints out in ls -al format.
#
   my $octal  = shift;
   my $string = sprintf "%lo", ($octal & 07777);
   my $result = '';
   foreach (split(//, $string)) {
      if    ($_ == 7) { $result .= "rwx "; }
      elsif ($_ == 6) { $result .= "rw- "; }
      elsif ($_ == 5) { $result .= "r-x "; }
      elsif ($_ == 4) { $result .= "r-- "; }
      elsif ($_ == 3) { $result .= "-wx "; }
      elsif ($_ == 2) { $result .= "-w- "; }
      elsif ($_ == 1) { $result .= "--x "; }
      elsif ($_ == 0) { $result .= "--- "; }
      else            { $result .= "unkown '$_'!"; }
   }
   return $result;
}

sub protect_form {
# ------------------------------------------------------------
# Presents the users with form to protect directory.
```

```
#
   my ($working_dir, $directory, $result) = @_;

# Set the working directory and get the password file.
   my ($pass_file);
   $working_dir ? ($pass_file = "$working_dir/$directory.pass") : ($pass_file = "$directory.pass");
   $pass_file =~ s,/,_,g; $pass_file = "$config{'password_dir'}/$pass_file";

# Get the user list, and print out the forms.
   my (@users)     = &load_users ($pass_file);
   my ($user_list);
   my ($local_dir) = "$working_dir/$directory"; $local_dir =~ s,^/,,;
   print qq~<p>Password protection for <i><b><a
href="$config{'root_url'}/$local_dir">$directory</a></b></i> : </p>~;
   print qq~<p>Result: <font color=red>$result</font></p>~ if ($result);
   print qq~
               <form action="$config{'script_url'}" method="post">
                  <input type=hidden name="action" value="add_user">
                  <input type=hidden name="wd" value="$working_dir">
                  <input type=hidden name="dir" value="$directory">
                  Add a new user, name: <input name="user" size=10> pass: <input name="pass" size=10>
<input type=submit value="Add">
               </form>
   ~;
   if ($#users > -1) {
      foreach (@users) {
         $user_list .= qq~<option value="$_">$_\n~;
      }
      print qq~
               <form action="$config{'script_url'}" method="post">
                  <input type=hidden name="action" value="remove_user">
                  <input type=hidden name="wd" value="$working_dir">
                  <input type=hidden name="dir" value="$directory">
                  Delete an authorized user: <select name='user'>$user_list</select> <input type=submit
value="Delete">
               </form>
      ~;
   }
}

sub add_user {
# ----------------------------------------------------------
# Protects directory with htacces files.
#
   my ($user, $pass, $working_dir, $directory) = @_;

# Set the working directory and get the password file.
   my ($pass_file);
   $working_dir and ($directory = "$working_dir/$directory");
   $pass_file = "$directory.pass";
   $pass_file =~ s,/,_,g; $pass_file = "$config{'password_dir'}/$pass_file";

# Make sure we have a username and password.
   if (length($user) < 3) { return "Add User: Username '$user' too short."; }
   if (length($pass) < 3) { return "Add User: Password '$pass' too short."; }
```

```
# Encrypt the password.
   my @salt_chars = ('A' .. 'Z', 0 .. 9, 'a' .. 'z', '.', '/');
   my $salt = join '', @salt_chars[rand 64, rand 64];
   my $encrypted = crypt($pass, $salt);

# Add/modify the user.
   my ($output, $found);
   if (&exists($pass_file)) {
      open (PASS, "<$pass_file") or &cgierr("Unable to open password file '$pass_file'. Reason: $!");
      while (<PASS>) {
         next unless (/^([^:]+)/);
         if ($user eq $1) {
            $output .= "$user:$encrypted\n";
            $found = 1;
         }
         else {
            $output .= $_;
         }
      }
      close PASS;
      if (!$found) { $output .= "$user:$encrypted\n"; }
   }
   else {
      $output = "$user:$encrypted\n";
   }
   open (PASS, ">$pass_file") or &cgierr("Unable to open password file '$pass_file'. Reason: $!");
   print PASS $output;
   close PASS;

# Create the .htaccess file if neccessary.
   &create_htaccess ($directory, $pass_file);

   return "Add User: '$user' added to password file.";
}

sub remove_user {
# ------------------------------------------------------------
# Removes a user from the .htaccess file and the password file.
#
   my ($user, $working_dir, $directory) = @_;
   my ($output);

# Set the working directory and get the password file.
   my ($pass_file);
   $working_dir and ($directory = "$working_dir/$directory");
   $pass_file = "$directory.pass";
   $pass_file =~ s,/,_,g; $pass_file = "$config{'password_dir'}/$pass_file";

# Make sure we have a username and password.
   if (length($user) < 3) { return "Remove User: '$user' too short or not specified."; }

# Update the password file.
   open (PASS, "<$pass_file") or &cgierr("Unable to open password file '$pass_file'. Reason: $!");
   while (<PASS>) {
      next if (/^\Q$user\E:/gio);
      $output .= $_;
```

```
    }
    close PASS;

# If we have users left, rewrite the password file. Otherwise, remove the password file
# and the .htaccess file.
    if ($output) {
        open (PASS, ">$pass_file") or &cgierr("Unable to open password file '$pass_file'. Reason: $!");
            print PASS $output;
        close PASS;
    }
    else {
        unlink ("$config{'root_dir'}/$directory/.htaccess") or &cgierr("Can't remove htaccess file
'$config{'root_dir'}/$directory/.htaccess'. Reason: $!");
        unlink ("$pass_file")                    or &cgierr("Can't remove password file '$pass_file'. Reason:
$!");
    }
    return "Remove User: '$user' removed successfully.";
}

sub create_htaccess {
# ----------------------------------------------------------
# Writes an .htaccess file in the specified directory.
#
    my ($directory, $pass_file) = @_;
    my $fulldir = "$config{'root_dir'}/$directory";

    if (!&exists("$fulldir/.htaccess")) {
        open (PASS, ">$fulldir/.htaccess") or &cgierr ("Unable to open htaccess file: '$directory/.htaccess'.
Reason: $!");
        print PASS qq~
AuthUserFile $pass_file
AuthGroupFile /dev/null
AuthName 'Protected Area'
AuthType Basic

<limit GET PUT POST>
require valid-user
</limit>
~;
        close PASS;
    }
}

sub load_users {
# ----------------------------------------------------------
# Loads the list of valid users from the password file.
#
    my $pass_file = shift;
    my (@users, $user, $pass);

    if (&exists("$pass_file")) {
        open (PASS, "<$pass_file") or &cgierr("Unable to open password file '$pass_file'. Reason: $!");
        while (<PASS>) {
            ($user, $pass) = split (/:/);
            push (@users, $user);
        }
```

```
        close PASS;
    }
    return @users;
}

sub print_filesize {
# ---------------------------------------------------------
# Prints out the file size.

    my $size = shift;
    my $formatted_size = int($size / 1000) . " kb";
    $formatted_size == 0 ?
        return "$size bytes" :
        return $formatted_size;
}

sub checkspace {
# ---------------------------------------------------------
# Check for allowed disk space to determine whether we can allow
# editing or uploads.
#
    use File::Find;

    my ($directory)     = shift;
    my ($size, $used_space, $free_space) = 0;

    &find ( sub { $size += -s }, $directory );
    $used_space = int ($size / 1024);
    $free_space = ($config{'allowed_space'} - $used_space);

    return ($free_space, $config{'allowed_space'}, $used_space);
}

sub exists {
# ---------------------------------------------------------
# Checks to see if a file exists.
#
    return -e shift;
}

sub get_icon {
# ---------------------------------------------------------
# Get the associated icon based on a files extension
#
    my ($file) = lc(shift);
    my ($ext)  = $file =~ /\.([^.]+)$/;
    if (!$ext) { return "$config{'icondir_url'}/$icons{'unknown'}"; }
    foreach (keys %icons) {
        next if (/folder/);
        next if (/unknown/);
        next if (/parent/);
        ($_ =~ /$ext/i) and return "$config{'icondir_url'}/$icons{$_}";
    }
    return "$config{'icondir_url'}/$icons{'unknown'}";
}
```

```
sub get_date {
# ---------------------------------------------------------
   my $time = shift;
   $time or ($time = time);
   my @months = qw!Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec!;

   my ($min, $hr, $day, $mon, $yr) = (localtime($time))[1,2,3,4,5];
   $yr = $yr + 1900;
   ($min < 10) and ($min = "0$min");
   ($hr  < 10) and ($hr  = "0$hr");
   ($day < 10) and ($day = "0$day");

   return "$day-$months[$mon]-$yr $hr:$min";
}

sub is_valid_file
# ---------------------------------------------------------
# Checks to see if a file is valid (proper form).
#
   my ($file, $okfile) = "";
   $file = shift;

   ($file =~ m,^([A-Za-z0-9\-_. ]*)$,) ?
      ($okfile = $1) :
      (return ($file, "Illegal Characters in Filename. Please use letters, numbers, -, _ and . only."));

   ($file =~ m,\.\.,)   and return ($file, "No double .. allowed in file names.");
   ($file =~ m,^\.,)    and return ($file, "no leading '.' in file names.");
   (length($file) > 30) and return ($file, "File name is too long. Please keep it to under 20 characters.");

   return ($okfile, "");
}

sub is_valid_dir {
# ---------------------------------------------------------
# Checks to see if a file is valid (proper form).
#
   my ($dir, $okdir, $last_dir) = "";
   $dir = shift;

   my (@size) = split (/\//, $dir);
   $last_dir  = pop (@size);

   ($dir =~ m,^([A-Za-z0-9\-_/]*)$,) ?
      ($okdir = $1) :
      (return ($dir, "Illegal Characters in Directory. Please use letters, numbers, - and _ only."));

   ($dir =~ m,^/,)         and return ($dir, "No leading / in directory names.");
   ($dir =~ m,/$,)         and return ($dir, "No trailing / in directory names.");
   ($#size > 8)            and return ($dir, "Directory level too deep.");
   (length($last_dir) > 15) and return ($dir, "Directory Name too Long. Please keep it to less then 15
characters.");

   return ($okdir, "");
}
```

```
sub is_valid_user {
# ----------------------------------------------------
# Makes sure a username is ok.
#
   my ($user) = shift;
   (!$user) and return ($user, "");
   ($user =~ /^([A-Za-z0-9-_]+)$/) ?
      return ($1, "") :
      return ($user, "Can only contain letters, numbers and -, _");
}

sub is_valid_perm {
# ----------------------------------------------------
# Makes sure entered permissions are ok.
#
   my ($perm) = shift;
   (!$perm)                    and return ($perm, "");
   ($perm =~ /^([0-7][0-7][0-7])$/) or return ($perm, "Permissions must be three digits only, 0 to 7.");
   return ($1, "");
}

sub log_action {
# ----------------------------------------------------
# Logs an action to the log file. Format is:
#   time ip remotehost referer working_dir action
#
   my ($action, $wd) = @_;
   open (LOG, ">>$config{'logfile'}") or &cgierr ("Unable to open logfile: $config{'logfile'}. Reason: $!",
1);
   if ($config{'use_flock'}) {
      flock (LOG, 2) or &cgierr ("Unable to get exlcusive lock: $config{'logfile'}. Reason: $!", 1);
   }
   print LOG join ("\t",
      scalar(localtime()),
      $ENV{'REMOTE_ADDR'},
      $ENV{'REMOTE_HOST'},
      $ENV{'HTTP_REFERER'},
      $wd,
      $action,
      "\n"
   );
   close LOG;
}

sub delete_only_error {
# ----------------------------------------------------
# Prints out an error message if the user tries to add anything when he is running
# out of disk space.
#
   print qq~
      <BLOCKQUOTE>
      <FONT FACE="arial" SIZE=4>
      This action was aborted, because your disk space allotment is
      full or near full (less than thirty kilobytes).<P>
      Please delete some files or directories before proceeding. Alternately,
      you may ask the webmaster to allocate more disk space to this
```

```
      account.</BLOCKQUOTE><br><br><br>
   ~;
}

sub user_error {
# ----------------------------------------------------------
# Displays a message about illegal filenames and whatsuch.
#
   my ($error, $wd) = @_;

   print qq|
<html>
<head>
   <title>File Manager $config{'version'}</title>
</head>

<body bgcolor="#DDDDDD">
   <center>
      <table border=1 bgcolor="#FFFFFF" cellpadding=2 cellspacing=1 width="630" align=center
valign=top>
         <!-- <tr> <td bgcolor="white" align=left><a href="javascript:history.go(-1)"><font face="Verdana,
Arail" size=2><b>Back</b></font></a></td>
            <td bgcolor="navy"  align=center width=90%><font color="white" face="Verdana, Arail"
size=3><b>File Manager $config{'version'}</b></font></td>
            <td bgcolor="white" align=right><a href="$config{'script_url'}"><font face="Verdana, Arail"
size=2><b>Root</b></font></a></td>
         </tr> -->
            <tr> <!-- <td bgcolor="white" align=left><a href="javascript:history.go(-1)"><font
face="Verdana, Arail" size=2><b>Back</b></font></a></td> -->
            <td bgcolor="white"  align=center width=90%><font color="black" face="Verdana, Arail"
size=3><b>Arthritis Care Document Repository $config{'version'}</b></font></td>
            <!-- <td bgcolor="white" align=right><a href="$config{'script_url'}"><font face="Verdana,
Arail" size=2><b>Root</b></font></a></td> -->
            </tr>
         <tr><td colspan=3>
            <p><b>Error!</b> The following error occured: </p>
            <p><blockquote><font color=red><b>$error</b></font></blockquote></p>
            <p>Please press <a href="javascript:history.go(-1)">back</a> on your browser to fix the
problem.</p>
         </td></tr>
         <tr><td colspan=3>
            <table border=0 width=100%>
               <tr><td align=left><b><font color="#888888" size=1 face="Verdana, Arial">Lat Updated:
</font></b></a></td>
               </tr>
            </table>
         </td></tr>
      </table>
   </center>
</body>
</html>
   |;
   &log_action ("Form Input Error: $error", $wd) if ($config{'logfile'});
   exit;
}
```

```
sub cgierr {
# --------------------------------------------------------
# Displays any errors and prints out FORM and ENVIRONMENT
# information. Useful for debugging.
#
   my ($key, $env);
   my ($error, $nolog) = @_;
   print "</td></tr></table>";
   print "</td></tr></table></center></center>";

   print "<PRE>\n\nCGI ERROR\n===============================================\n";
   $error    and print "Error Message       : $error\n";
   $0        and print "Script Location     : $0\n";
   $]        and print "Perl Version        : $]\n";

   print "\nConfiguration\n-------------------------------------------\n";
   foreach $key (sort keys %config) {
      my $space = " " x (20 - length($key));
      print "$key$space: $config{$key}\n";
   }

   print "\nForm Variables\n-----------------------------------------\n";
   foreach $key (sort $in->param) {
      my $space = " " x (20 - length($key));
      print "$key$space: " . $in->param($key) . "\n";
   }
   print "\nEnvironment Variables\n--------------------------------------------\n";
   foreach $env (sort keys %ENV) {
      my $space = " " x (20 - length($env));
      print "$env$space: $ENV{$env}\n";
   }
   print "\n</PRE>";
   &log_action ("CGI ERROR: $error") if (!$nolog and $config{'logfile'});
   exit;
}
```

```
---------------------- Code for directory search ----------------------------
#!/usr/bin/perl -wT

#use strict;
use CGI;
use DBI;
use DBD::ODBC;


#Configuration
#-----------------------------------------------------------
my $q = new CGI;

my $name = $q->param( "name" );

my $fname ="";
my $lname ="";
my $counter = 0;
my $results;

my $ddb;

($fname, $lname) = split " ", $name;

print qq~
<html>

<head>
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Arthritis Care Internal Web Page</title>
</head>

<body>

<table border="0" width="100%" cellspacing="0" cellpadding="0" height="245">

  <tr>
    <td width="100%" height="102" >
      <table border="0" width="100%" cellspacing="0" cellpadding="0" height="104">
        <tr>
          <td width="76%" height="104">
            <table border="0" width="100%" cellspacing="0" cellpadding="0" height="115">
              <tr>
                <td width="100%" background="/Images/tp_bg.jpg"><font size="6">Arthritis Care
Directory</font>
                </td>
              </tr>
              <tr>
                <td width ="95%" background="/Images/tp_bg.jpg">
                <form method="POST" action="/cgi-bin/dbsearch.cgi"><br>
                <MAP NAME="top_nav.gif">
                                                <AREA shape=RECT coords="5, 0, 64, 25"
HREF="/home/">
                                                        <AREA shape=RECT coords="68, 0, 138, 25"
href="/home/training/">
```

```
                                                    <AREA shape=RECT coords="150, 0, 197, 25"
href="/home/tools/">
                                                    <AREA shape=RECT coords="212, 0, 312, 25"
HREF="/cgi-bin/docs.cgi">
                                                    <AREA shape=RECT coords="323, 0, 408, 25"
HREF="/home/directory/">
                                                </MAP><IMG SRC="/Images/top_nav.gif"
USEMAP="#top_nav.gif" width=409 height=26 BORDER=0 align="bottom"><input type="text"
name="name" size="16"><input type="submit" value="Go" name="B1">
              </form>
                        </td>
          </tr>
          <tr>
            <td width="100%" height="39">
              <table border="2" width="100%" bordercolor="#CE0063" cellspacing="0" cellpadding="0">
                <tr>
                  <td width="25%" align="center" bgcolor="#CE0063" bordercolor="#FFFFFF"
bordercolorlight="#FFFFFF" bordercolordark="#FFFFFF"><b><font color="#FFFFFF">Quick
Search</font></b></td>
                  <td width="25%" align="center"><a href="/home/directory/directoryAS.htm">Advanced
Search</a></td>
                  <td width="25%" align="center"><a href="/home/help/directory/">Help</a></td>
                  <td width="25%" align="center">Edit My Profile</td>
                </tr>
              </table>
            </td>
          </tr>
        </table>
      </td>
      <td width="12%" align="right" height="104"><img border="0" src="/Images/logo_top.gif"
width="139" height="63"><br>
        <img border="0" src="/Images/logo_bot.gif" width="133" height="37"></td>
    </tr>
  </table>
 </td>
</tr>
<tr>
 <td width="100%" height="77">
The following names resulted from your search,<br>
Click on a name for more detail. <p>
~;

#connect to a database

$db = DBI->connect('DBI:ODBC:driver=Microsoft Access Driver
(*.mdb);dbq=/ccd0/vhosts/cid/docs/secure/staff_directory/contacts.mdb',",")
 or die print qq~ Cannot connect: $DBI::errstr~;

if ($lname ne '') {
  $ddb = $db->prepare("SELECT * FROM contact WHERE fname LIKE '%$fname%' AND lname LIKE
'%$lname%'");}
else {

$ddb = $db->prepare("SELECT * FROM contact WHERE fname='$fname'
            OR fname like '%$fname%'
            OR lname like '%$fname%'
```

```
            OR lname='$fname'");}

$ddb->execute;

while(@data = $ddb->fetchrow_array()) {
  print qq~ <a href="http://www.cid.ac/cgi-bin/dbperson.cgi?fname=$data[1]\&lname=$data[2]">$data[1]
$data[2]<p> ~;
  $counter += 1;
}

if ($counter == 0) {
  print qq~
    <p>
    <b><font-size=2>Your search returned no results.<br>
    Please refine your search and try again.</font></b>
    <form method="POST" action="/cgi-bin/dbsearch.cgi">
      <p>The the name of the person that you are looking for: <br>
       <input type="text" name="name" size="27"><input type="submit" value="Search"
name="B1"><input type="reset" value="Clear" name="B2"></p>
    </form>
  ~;
}

$ddb->finish;



print qq~
  </td>
  </tr>
  <tr>
    <td width="100%" height="66">
      <hr noshade>
      Arthritis Care, Internal Use Only<br>
      <br>
      Maintained by: <a href="mailto:webmaster\@cid.ac">ACG</a><br>
      Last modified:20 Apr 2001 10:25 
    </td>
  </tr>
</table>

</body>

</html>
~;
```

```
---------------------- Code for directory person results ----------------------

#!/usr/bin/perl

##########################################
# Arthritis Care                         #
# Directory: Individual Person Page      #
#                                        #
# worsham@wpi.edu                        #
##########################################


use CGI;
use DBI;
use DBD::ODBC;


# configuration for links
# make it easy to change the hosts!
#####################################################
my $home_url = "http://www.cid.ac/home/index.htm";
my $dir_url  = "http://www.cid.ac/home/directory/directoryQS.htm";
my $docs_url = "http://www.cid.ac/cgi-bin/docs.cgi";

my $q = new CGI;

my $fname = $q->param( "fname" );
my $lname = $q->param( "lname" );

my $ddb;

print qq~
<html>

<head>
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Arthritis Care Internal Web Page</title>
</head>

<body>

<table border="0" width="100%" cellspacing="0" cellpadding="0" height="245">

  <tr>
    <td width="100%" height="102" >
      <table border="0" width="100%" cellspacing="0" cellpadding="0" height="104">
        <tr>
          <td width="76%" height="104">
            <table border="0" width="100%" cellspacing="0" cellpadding="0" height="115">
              <tr>
                <td width="100%" background="/Images/tp_bg.jpg"><font size="6">Arthritis Care
Directory</font>
                </td>
              </tr>
              <tr>
```

```
            <td width ="95%" background="/Images/tp_bg.jpg">
            <form method="POST" action="/cgi-bin/dbsearch.cgi"><br>
            <MAP NAME="top_nav.gif">
                                                <AREA shape=RECT coords="5, 0, 64, 25"
HREF="/home/">
                                                        <AREA shape=RECT coords="68, 0, 138, 25"
href="/home/training/">
                                                        <AREA shape=RECT coords="150, 0, 197, 25"
href="/home/tools/">
                                                        <AREA shape=RECT coords="212, 0, 312, 25"
HREF="/cgi-bin/docs.cgi">
                                                        <AREA shape=RECT coords="323, 0, 408, 25"
HREF="/home/directory/">
                                                </MAP><IMG SRC="/Images/top_nav.gif"
USEMAP="#top_nav.gif" width=409 height=26 BORDER=0 align="bottom"><input type="text"
name="name" size="16"><input type="submit" value="Go" name="B1">
              </form>
                        </td>
           </tr>
           <tr>
             <td width="100%" height="39">
               <table border="2" width="100%" bordercolor="#CE0063" cellspacing="0"
cellpadding="0">
                 <tr>
                   <td width="25%" align="center" bgcolor="#CE0063" bordercolor="#FFFFFF"
bordercolorlight="#FFFFFF" bordercolordark="#FFFFFF"><b><font color="#FFFFFF">Quick
Search</font></b></td>
                   <td width="25%" align="center"><a
href="/home/directory/directoryAS.htm">Advanced Search</a></td>
                   <td width="25%" align="center"><a href="/home/help/directory/">Help</a></td>
                   <td width="25%" align="center">Edit My Profile</td>
                 </tr>
               </table>
             </td>
           </tr>
         </table>
        </td>
        <td width="12%" align="right" height="104"><img border="0" src="/Images/logo_top.gif"
width="139" height="63"><br>
          <img border="0" src="/Images/logo_bot.gif" width="133" height="37"></td>
       </tr>
     </table>
   </td>
  </tr>
  <tr>
    <td width="100%" height="77">
~;

#connect

$db = DBI->connect('DBI:ODBC:driver=Microsoft Access Driver
(*.mdb);dbq=/ccd0/vhosts/cid/docs/secure/staff_directory/contacts.mdb','','')
  or die "Cannot connect: ". $DBI::errstr;

#$db->Sql("SELECT * FROM contact WHERE fname='$fname'");
```

```
$ddb = $db->prepare("SELECT * FROM contact WHERE fname='$fname%'");

$ddb->execute;

while(@data = $ddb->fetchrow_array()) {
if (($data[1] =~ $fname) && ($data[2] =~ $lname)) {

print qq~
                  <td width="29%" valign="top">Email Address</td>
                  <td width="71%"><a href="mailto:$data[13]">$data[13]</a></td>
                </tr>
                <tr>
                  <td width="29%">Work Phone</td>
                  <td width="71%">$data[10]</td>
                </tr>
                <tr>
                  <td width="29%">Mobile</td>
                  <td width="71%">$data[12]</td>
                </tr>
                <tr>
                  <td width="29%">Fax</td>
                  <td width="71%">$data[11]</td>
                </tr>
              </table>
            </td>
            <td width="50%" valign="top">
              <table border="0" width="100%" cellspacing="1">
                <tr>
                  <td width="29%">Address 1</td>
                  <td width="71%">$data[4]</td>
                </tr>
                <tr>
                  <td width="29%">Address 2</td>
                  <td width="71%">$data[5]</td>
                </tr>
                <tr>
                  <td width="29%">Address 3</td>
                  <td width="71%">$data[6]</td>
                </tr>
                <tr>
                  <td width="29%">Address 4</td>
                  <td width="71%">$data[7]</td>
                </tr>
                <tr>
                  <td width="29%">Address 5</td>
                  <td width="71%">$data[8]</td>
                </tr>
                <tr>
                  <td width="29%">Postal Code</td>
                  <td width="71%">$data[9]</td>
                </tr>
              </table>
            </td>
          </tr>
          <tr>
            <td width="100%" colspan="2"><b><font size="3">Organisational
```

```
            Information</font></b></td>
          </tr>
          <tr>
            <td colSpan=2 width="100%">
              <table border="0" width="100%" cellspacing="1">
                <tr>
                  <td width="21%">Job Title</td>
                  <td width="79%">$data[3]</td>
                </tr>
                <tr>
                  <td width="21%">Hours</td>
                  <td width="79%">$data[16]</td>
                </tr>
                <tr>
                  <td width="21%">Team</td>
                  <td width="79%">$data[15]</td>
                </tr>
                <tr>
                  <td width="21%">Location or Division</td>
                  <td width="79%">$data[14]</td>
                </tr>
                <tr>
                  <td width="21%">Notes</td>
                  <td width="79%">$data[17]</td>
                </tr>
              </table>
            </td>
          </tr>
        </table>
      </td>
    </tr>
  </table>
 </td>
</tr>
<tr>
 <td width="100%" height="34">
  <table border="2" width="100%" bordercolor="#CE0063" cellspacing="0" cellpadding="0"
bordercolorlight="#CE0063" bordercolordark="#CE0063" bgcolor="#FFFFFF">
    <tr>
     <td width="33%" align="center"><a href="$home_url">Home</a></td>
     <td width="33%" align="center"><a href="$dir_url">Directory</a></td>
     <td width="34%" align="center"><a href="$docs_url">Documents</a></td>
    </tr>
  </table>
 </td>
</tr>
<tr>
 <td width="100%" height="97">
  <hr noshade>
  Arthritis Care, Internal Use Only<br>
  <br>
  Maintained by: <a href="mailto:webmaster\@cid.ac">webmaster</a><br>
  Last modified: 
 </td>
</tr>
</table>
```

```
</body>

</html>
~;
}
}
$ddb->finish;
```

```
---------------------- Code for directory advanced search ----------------------
#!/usr/bin/perl -wT

use CGI;
use DBI;
use DBD::ODBC;

#COnfiguration
#-----------------------------------------------------------
my $local_root_url  = "http://www.cid.ac/cgi-bin";
my $remote_root_url = "http://www.cid.ac/cgi-bin";

my $q = new CGI;

my $fname  = $q->param( "c_fname" );
my $lname  = $q->param( "c_lname" );
my $email  = $q->param( "c_email" );
my $wphone = $q->param( "c_wphone" );
my $mobile = $q->param( "c_mobile" );
my $fax    = $q->param( "c_fax" );
my $jtitle = $q->param( "c_jtitle" );
my $add1   = $q->param( "c_add1" );
my $add2   = $q->param( "c_add2" );
my $add3   = $q->param( "c_add3" );
my $add4   = $q->param( "c_add4" );
my $add5   = $q->param( "c_add5" );
my $pcode  = $q->param( "c_pcode" );
my $lod    = $q->param( "c_lod" );
my $team   = $q->param( "c_team" );
my $hours  = $q->param( "c_hours" );
my $notes  = $q->param( "c_notes" );

my $query = '';
my $counter = 0;
my $results;

my $ddb;

print qq~
<html>

<head>
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<title>Arthritis Care Internal Web Page</title>
</head>

<body>

<table border="0" width="100%" cellspacing="0" cellpadding="0" height="245">

  <tr>
    <td width="100%" height="102" >
      <table border="0" width="100%" cellspacing="0" cellpadding="0" height="104">
        <tr>
          <td width="76%" height="104">
```

```
        <table border="0" width="100%" cellspacing="0" cellpadding="0" height="115">
          <tr>
            <td width="100%" background="/Images/tp_bg.jpg"><font size="6">Arthritis Care
Directory</font>
            </td>
          </tr>
          <tr>
            <td width ="95%" background="/Images/tp_bg.jpg">
            <form method="POST" action="/cgi-bin/dbsearch.cgi"><br>
            <MAP NAME="top_nav.gif">
                                              <AREA shape=RECT coords="5, 0, 64, 25"
HREF="/home/">
                                                  <AREA shape=RECT coords="68, 0, 138, 25"
href="/home/training/">
                                                  <AREA shape=RECT coords="150, 0, 197, 25"
href="/home/tools/">
                                                  <AREA shape=RECT coords="212, 0, 312, 25"
HREF="/cgi-bin/docs.cgi">
                                                  <AREA shape=RECT coords="323, 0, 408, 25"
HREF="/home/directory/">
                                             </MAP><IMG SRC="/Images/top_nav.gif"
USEMAP="#top_nav.gif" width=409 height=26 BORDER=0 align="bottom"><input type="text"
name="name" size="16"><input type="submit" value="Go" name="B1">
            </form>
                        </td>
          </tr>
          <tr>
            <td width="100%" height="39">
              <table border="2" width="100%" bordercolor="#CE0063" cellspacing="0"
cellpadding="0">
                <tr>
                  <td width="25%" align="center" bgcolor="#CE0063" bordercolor="#FFFFFF"
bordercolorlight="#FFFFFF" bordercolordark="#FFFFFF"><b><font color="#FFFFFF">Quick
Search</font></b></td>
                  <td width="25%" align="center"><a
href="/home/directory/directoryAS.htm">Advanced Search</a></td>
                  <td width="25%" align="center"><a href="/home/help/directory/">Help</a></td>
                  <td width="25%" align="center">Edit My Profile</td>
                </tr>
              </table>
            </td>
          </tr>
        </table>
      </td>
      <td width="12%" align="right" height="104"><img border="0" src="/Images/logo_top.gif"
width="139" height="63"><br>
        <img border="0" src="/Images/logo_bot.gif" width="133" height="37"></td>
    </tr>
  </table>
 </td>
</tr>
<tr>
 <td width="100%" height="77">
The following names resulted from your search,<br>
Click on a name for more detail. <p>
~;
```

```
$db = DBI->connect('DBI:ODBC:driver=Microsoft Access Driver
(*.mdb);dbq=/ccd0/vhosts/cid/docs/secure/staff_directory/contacts.mdb','','')
  or die "Cannot connect: ". $DBI::errstr;

$query = "SELECT * FROM contact WHERE ";
if ($lname ne ''){$query .= "lname like '%$lname%' AND";}
if ($fname ne ''){$query .= "fname like '%$fname%' AND";}
if ($email ne ''){$query .= "email like '%$email%' AND";}
if ($wphone ne ''){$query .= "wphone like '%$wphone%' AND";}
if ($mobile ne ''){$query .= "mobile like '%$mobile%' AND";}
if ($fax ne ''){$query .= "fax like '%$fax%' AND";}
if ($add1 ne ''){$query .= "add1 like '%$add1%' AND";}
if ($add2 ne ''){$query .= "add2 like '%$add2%' AND";}
if ($add3 ne ''){$query .= "add3 like '%$add3%' AND";}
if ($add4 ne ''){$query .= "add4 like '%$add4%' AND";}
if ($add5 ne ''){$query .= "add5 like '%$add5%' AND";}
if ($pcode ne ''){$query .= "pcode like '%$pcode%' AND";}
if ($jtitle ne ''){$query .= "jtitle like '%$jtitle%' AND";}
if ($notes ne ''){$query .= "notes like '%$notes%' AND";}
if ($lod ne ''){$query .= "lod like '%$lod%' AND";}
if ($team ne ''){$query .= "team like '%$team%' AND";}
if ($hours ne ''){$query .= "hours like '%$hours%' AND";}

# Remove the trailing AND
chop($query);
chop($query);
chop($query);

$ddb = $db->prepare($query);
$ddb->execute;

while(@data = $ddb->fetchrow_array()){
  print qq~ <a href="http://www.cid.ac/cgi-
bin/dbperson.cgi?fname=$data[1]\&lname=$data[2]">$data[1] $data[2]<p> ~;
  $counter += 1;
}

if ($counter == 0) {
  print qq~
    <p>
    <b><font-size=2>Your search returned no results.<br>
    Please refine your search and try again.</font></b>
    <form method="POST" action="/cgi-bin/dbsearch.cgi">
      <p>The the name of the person that you are looking for: <br>
       <input type="text" name="name" size="27"><input type="submit" value="Search"
name="B1"><input type="reset" value="Clear" name="B2"></p>
    </form>
  ~;
}

$ddb->finish;



print qq~
  </td>
```

```
  </tr>
  <tr>
    <td width="100%" height="66">
      <hr noshade>
      Arthritis Care, Internal Use Only<br>
      <br>
      Maintained by: <a href="mailto:webmaster\@cid.ac">webmaster</a><br>
      Last modified: 
    </td>
  </tr>
</table>

</body>

</html>
~;
```

```
---------------------- Code for password manipulation ----------------------
#!/usr/bin/perl
#
#
############################################################
#                   Access Denied version 1.2
#
#          Created by: Solution Scripts
#                   Email: solutions@solutionscripts.com
#                   Web: http://solutionscripts.com
#
############################################################
#
#
# COPYRIGHT NOTICE:
#
# Copyright 2000 Solution Scripts  All Rights Reserved.
#
# This program is being distributed as freeware.  It may be used and
# modified free of charge, so long as this copyright notice, the header
# above and all the footers in the program that give me credit remain
# intact. Please also send me an email, and let me know
# where you are using this script.
#
# By using this program you agree to indemnify Solution Scripts from any liability.
#
# Selling the code for this program without prior written consent is
# expressly forbidden.  Obtain permission before redistributing this
# program over the Internet or in any other medium.  In all cases
# copyright and header must remain intact.
#
############################################################
#
#
# PLEASE READ THE README INCLUDED IN THIS ZIP OR
# VISIT http://faq.solutionscripts.com/docs/accessdenied
# BEFORE ATTEMPTING TO INSTALL ACCESS-DENIED

$protect{'Home'} = "/ccd0/vhosts/cid/docs/.htpasswd";
$protect{'Secure'} = "/ccd0/vhosts/cid/docs/.htsecure";
#$protect{'Test 2'} = "/home/httpd/cgi-bin/demos/access_denied/.htpasswd1";


# Full path to password file ..... including file name
$password_location = "pass.txt";

# THATS ALL FOLKS.....................
############################################################


$thisurl = $ENV{'SCRIPT_NAME'};
$use_password = 0;

$VERSION = "1.2";

@char_set = ('a'..'z','0'..'9');
```

```
## CREATE SELECT LIST ##
foreach $key (sort_hashs(\%protect)) {
	$select .= "<option value=\"$key\">$key";
}

### FORM DATA ###
if ($ENV{'QUERY_STRING'}) {
	@pairs=split(/&/,$ENV{'QUERY_STRING'});
	$ftype = "GET";
}
if ($ENV{'CONTENT_LENGTH'}) {
	read(STDIN, my $buffer, $ENV{'CONTENT_LENGTH'});
	push(@pairs,split(/&/, $buffer));
	$ftype = "POST";
}

foreach $pair (@pairs) {
	($name, $value) = split(/=/, $pair);
	$value =~ tr/+/ /;
	$value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
	if ($INPUT{$name}) { $INPUT{$name} = $INPUT{$name}.",".$value; }
	else { $INPUT{$name} = $value; }
}

@passset = ('a'..'z');
for ($i = 0; $i < 2; $i++) {
	$randum_num = int(rand($#passset + 1));
	$salt .= @passset[$randum_num];
}


print "Content-type: text/html \n\n";

&Top;

open (PASSWORD, "$password_location");
$password = <PASSWORD>;
close (PASSWORD);

chop ($password);

### SET NEW PASSWORD
&newpass if $INPUT{'newpass'};

### SET NEW PASSWORD ###
unless ($password || $use_password) {
	print qq~
	<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC" width=500>
	<TR><TD><TABLE cellpadding=5 cellspacing=1 border=0 width="100%">
	<TR BGCOLOR="#FFFFFF"><TD align=center>
	Before you can do anything else, you'll need to set your administrative password.
	This will allow you to use the administrative functions.
	<HR noshade size=1 width=85%>
	Please enter your desired password below. (Enter it twice.)
	<FORM METHOD=POST ACTION="$thisurl">
```

```
		<INPUT TYPE=HIDDEN NAME="newpass" value="hit return">
		<INPUT TYPE=SUBMIT NAME=newpass VALUE="Set Admin Password:">
		<INPUT TYPE=PASSWORD NAME=passad SIZE=10>
		<INPUT TYPE=PASSWORD NAME=passad2 SIZE=10>
		</FORM>
		</TD></TR></TABLE>
		</TD></TR></TABLE>
		<BR><BR>
		~;

		&Bottom(0);
		exit;
}

### PROMPT FOR PASSWORD ###
unless ($INPUT{'password'} || $use_password) {
		print qq~
		<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC" width=500>
		<TR><TD><TABLE cellpadding=5 cellspacing=1 border=0 width="100%">
		<TR BGCOLOR="#FFFFFF"><TD align=center>
		Welcome, please enter your password
		<HR noshade size=1 width=85%>
		<FORM METHOD=POST ACTION="$thisurl">
		<INPUT TYPE=HIDDEN NAME="admin" value="hit return">
		<INPUT TYPE=PASSWORD NAME=password SIZE=10>  
		<INPUT TYPE=SUBMIT NAME=admin VALUE=" Proceed ">
		</FORM>
		</TD></TR></TABLE>
		</TD></TR></TABLE>
		<BR><BR>
		~;

		&Bottom(0);
		exit;
}

&checkpassword unless $use_password;

if ($INPUT{'add'}) { &add; }
elsif ($INPUT{'remove'}) { &remove; }
elsif ($INPUT{'delete_select'}) { &delete_select; }
elsif ($INPUT{'delete_final'}) { &delete_final; }
else { &admin; }

sub admin {

### PRINT MAIN ADMIN SCREEN ###
print qq~

<form action="$thisurl" method=post>
<INPUT TYPE="HIDDEN" NAME="password" value="$INPUT{'password'}">

<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#000000">
<TR><TD><TABLE cellpadding=4 cellspacing=1 border=0 width="100%">
```

```
<TR bgcolor="#CCCCCC"><TD colspan=3 align=center><B>Add or remove
logins</B></TD></TR>

<TR bgcolor="#FFFFFF">
<TD>Login:</TD>
<TD><input type=text name="login" size=25></TD>
</TR>

<TR bgcolor="#FFFFFF">
<TD>Password:</TD>
<TD><input type=text name="pass" size=25></TD>
</TR>

<TR bgcolor="#FFFFFF">
<TD>Retype:</TD>
<TD><input type=text name="pass2" size=25></TD>
</TR>

<TR BGCOLOR="#FFFFFF">
<TD>Select Dir:</TD>
<TD><SELECT NAME="dir">$select</SELECT></TD>
</TR>

<TR BGCOLOR="#FFFFFF">
<TD>Use rules:</TD>
<TD><input type="Checkbox" name="rules" value="on" checked> Yes</TD>
</TR>

<TR BGCOLOR="#FFFFFF">
<TD colspan=2 align=center><INPUT TYPE="SUBMIT" NAME="add" VALUE=" Add Login ">
  
<INPUT TYPE="SUBMIT" NAME="remove" VALUE=" Delete Login "></TD>
</TR>

</TABLE></TD></TR></TABLE>


<BR><BR>



<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#000000">
<TR><TD><TABLE cellpadding=7 cellspacing=1 border=0 width="100%">

<TR BGCOLOR="#CCCCCC">
<TD align=center colspan=2><B>Manage Password Files</B></TD>
</TR>

<TR BGCOLOR="#FFFFFF">
<TD>View passwords in the dir:</TD>
<TD>Begining with the letter: </TD>
</TR>

<TR BGCOLOR="#FFFFFF" align=center>
<TD><SELECT NAME="manage_dir">$select</SELECT></TD>
<TD valign=top rowspan=2><SELECT name=letters multiple size=6>
```

```
<OPTION value="ALL Letters" Selected>All Letters
~;

foreach $ch(@char_set) {
	print "<OPTION VALUE=\"$ch\">$ch\n";
}

print qq~
</SELECT></TD>
</TR>

<TR BGCOLOR="#FFFFFF" align=center>
<TD><INPUT TYPE=SUBMIT name="delete_select" value=" Submit "></TD>
</TR>

</TABLE>
</TD></TR></TABLE>
</FORM>
~;

&Bottom(0);
exit;
}

##### SET NEW ADMIN PASSWORD ####
sub newpass {

unless ($INPUT{'passad'} eq $INPUT{'passad2'}) {
	print qq~
	<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
	<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
	<TR BGCOLOR="#FFFFFF"><TD align=center>
	Your administrative password was
	not set, as the two entries were different!
	</TD></TR></TABLE>
	</TD></TR></TABLE>
	~;
	&Bottom(0);
	exit;
}


if ($INPUT{'passad'}) {
	$newpassword = crypt($INPUT{'passad'}, $salt);
}
else {
	print qq~
	<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
	<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
	<TR BGCOLOR="#FFFFFF"><TD align=center>You must enter a password!
	</TD></TR></TABLE>
	</TD></TR></TABLE>
	~;
	&Bottom(0);
	exit;
}
```

```
unless ( -e "$password_location" ) {
        open (PASSWORD, ">$password_location")|| &error("Error setting Password");;
        print PASSWORD "$newpassword";
        close (PASSWORD);
}
else {
        print qq~
        <TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
        <TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
        <TR BGCOLOR="#FFFFFF"><TD align=center>
        Error setting your admin password<BR><BR>A $password_location file was found in this
dir.<BR><BR>
        Due to security reasons, it will not be overwritten.
        </TD></TR></TABLE>
        </TD></TR></TABLE>
        ~;
        &Bottom(1);
        exit;
}

print qq~
<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
<TR BGCOLOR="#FFFFFF"><TD align=center>
Your administrative password has been set.
</TD></TR></TABLE>
</TD></TR></TABLE>
<BR><BR>
~;

&admin;
exit;
}

#### ADD A NEW PASSWORD ####
sub add {

$err = '';
unless ($INPUT{'login'})   {
        $err .= "You have not entered a login, please go back an try again<BR>\n";
}
unless ($INPUT{'pass'} eq $INPUT{'pass2'}) {
        $err .= "The two passwords you entered must be the same<br>\n";
        }
if ($INPUT{'rules'}) {
        $pwderr = &check_pwd($INPUT{'pass'},$INPUT{'login'},'');
        $err .= "Password Error: ". $pwderr if ($pwderr && ($pwderr != 1));
}

if ($err) {
        print qq~
        <TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
        <TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
        <TR BGCOLOR="#FFFFFF"><TD align=center>
        <B>Error setting login/password</b><BR><BR>
```

```
		$err
		<BR><BR>
		To set the password as-is, be sure to uncheck the rules box.
		</TD></TR></TABLE>
		</TD></TR></TABLE>
		~;
		&Bottom(1);
		exit;
}

$password = crypt($INPUT{'pass'}, $salt);

open(LIST,"$protect{$INPUT{'dir'}}");
@addresses=<LIST>;
close(LIST);

foreach $line(@addresses) {
		@login = split(/\:/,$line);
		if ($login[0] eq $INPUT{'login'}) {
			print qq~
			<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
			<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
			<TR BGCOLOR="#FFFFFF"><TD align=center>The login you entered,
<b>$INPUT{'login'}</b> already exists in the $INPUT{'dir'} directory,
			therefore was not set.
			</td></TR></TABLE>
			</td></TR></TABLE>
			~;
			&Bottom(1);
			exit;
		}
}

open(LIST,">>$protect{$INPUT{'dir'}}") || &error("Unable to print to $protect{$INPUT{'dir'}}");
print LIST "$INPUT{'login'}:$password\n";
close(LIST);

print qq~
<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
<TR BGCOLOR="#FFFFFF"><TD align=center>The login <b>$INPUT{'login'}</b> was
successfully added to the $INPUT{'dir'} directory
</td></TR></TABLE>
</td></TR></TABLE><BR><BR>
~;

&admin;
exit;
}

#### REMOVE A PASSWORD ####
sub remove{

open(LIST,"$protect{$INPUT{'dir'}}");
@addresses=<LIST>;
close(LIST);
```

```
open(LIST,">$protect{$INPUT{'dir'}}") || &error("Unable to write to $protect{$INPUT{'dir'}}");
$found=0;
foreach $line (@addresses) {
	@login = split(/\:/,$line);
	if ($login[0] eq $INPUT{'login'}) {
		$found=1;
	}
	else {
		print LIST $line;
	}
}
close(LIST);

if ($found) {
	print qq~
	<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
	<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
	<TR BGCOLOR="#FFFFFF"><TD align=center>
	The login <b>$INPUT{'login'}</b> has been removed from the $INPUT{'dir'} directory
	</TD></TR></TABLE>
	</TD></TR></TABLE>
	<BR><BR>
	~;
	&admin;
	exit;
}
else {
	print qq~
	<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
	<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
	<TR BGCOLOR="#FFFFFF"><TD align=center>
	The login <b>$INPUT{'loginn'}</b> has was not found in the $INPUT{'dir'} directory,
	therefore not removed
	</TD></TR></TABLE>
	</TD></TR></TABLE>
	<BR><BR>
	~;
	&admin;
	exit;
}

}


#### DELETE MULTIPLE ####
sub delete_select {

open(LIST,"$protect{$INPUT{'manage_dir'}}") || &error("Can not open
$protect{$INPUT{'manage_dir'}} for reading");
@addresses=<LIST>;
close(LIST);
@addresses = sort {$a cmp $b} @addresses;

print <<EOF;
<FORM METHOD=POST ACTION="$thisurl">
```

```
<input type="Hidden" name="password" value="$INPUT{'password'}">
<input type="Hidden" name="manage_dir" value="$INPUT{'manage_dir'}">

<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#000000">
<TR><TD><TABLE cellpadding=5 cellspacing=1 border=0 width="100%">
<TR bgcolor="#CCCCCC"><TD align=center>
Showing all logins from the <B>$INPUT{'manage_dir'}</B> directory<BR>
that start with the letter: <B>$INPUT{'letters'}</B>
<BR><BR>
Select logins to delete
</TD></TR></TABLE>
</TD></TR></TABLE>

<br><br>

<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#000000">
<TR><TD><TABLE cellpadding=3 cellspacing=1 border=0 width="100%">
<TR align=left bgcolor="#FFFFFF">


EOF

$num_email=0;
$color{'1'} = "#DDDDDD";
$color{'2'} = "#CCCCCC";
$colors=1;
$total =0;

foreach $line(@addresses) {
        chomp($line);
        @login = split(/\:/,$line);

        my @accarray = split(//,$login[0]);
        ## LETTER SELECT ##
        unless ($INPUT{'letters'} =~ /ALL/) {
                unless ($INPUT{'letters'} =~ /$accarray[0]/) {
                        next;
                }
        }

        $total++;

        if ($num_email == 3) {
                print "</TR><TR align=left bgcolor=\"#FFFFFF\">";
                $num_email=0;
        }
        print "<TD bgcolor=$color{$colors}>";
        print "<INPUT TYPE=\"CHECKBOX\" NAME=\"delete\" VALUE=\"$line\"> --
$login[0]</TD>";
        $num_email++;
        if ($colors == 2) {
                $colors = 0;
        }
        $colors++;
}
```

```
$span = 3 - $num_email;

if ($span) {
        print qq~
        <TD colspan=$span> </TD>
        ~;
}

print qq~
</TR>
<TR bgcolor="#DDDDDD" align=center>
<TD colspan=3><B>$total</B> logins found</TD>
</TR>
<TR bgcolor="#CCCCCC" align=center>
<TD colspan=3><INPUT TYPE="SUBMIT" NAME="delete_final" VALUE="Delete selected
logins"></TD>
</TR></TABLE>
</TD></TR></TABLE>
</form>
~;
&Bottom(1);
exit;
}

#### DELETE MULITIPLE LOGIN ####
sub delete_final {

open(LIST,"$protect{$INPUT{'manage_dir'}}")|| &error("Unable to open
$protect{$INPUT{'manage_dir'}}");
@addresses=<LIST>;
close(LIST);

print qq~
<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
<TR BGCOLOR="#FFFFFF"><TD align=center>
The following logins were deleted from the $INPUT{'manage_dir'} directory<BR><BR>
~;

@deleting = split(/\,/,$INPUT{'delete'});
foreach $line(@deleting) {
        @addresses = grep{ !(/^$line/i) } @addresses;
        @login = split(/\:/,$line);
        print "$login[0] <BR>";
}

print qq~
</TD></TR></TABLE>
</TD></TR></TABLE>
<BR><BR>
~;

open(LIST,">$protect{$INPUT{'manage_dir'}}") || &error("Unable to write to
$protect{$INPUT{'manage_dir'}}");
print LIST @addresses;
close(LIST);
```

```
&admin;
exit;
}

#### CHECK PASSWORD ####
sub checkpassword {

open (PASSWORD, "$password_location") || &error("Unable to open $password_location");
$password = <PASSWORD>;
close (PASSWORD);

if ($INPUT{'password'}) {
        $newpassword = crypt($INPUT{'password'}, $password);
        unless ($newpassword eq $password) {
                print qq~
                <TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
                <TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
                <TR BGCOLOR="#FFFFFF"><TD align=center>Wrong Password
                </TD></TR></TABLE></TD></TR></TABLE>
                ~;

                &Bottom(0);
                exit;
        }
}
else {
        print qq~
        <TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
        <TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
        <TR BGCOLOR="#FFFFFF"><TD align=center>You must enter your admin password
        </TD></TR></TABLE>
        </TD></TR></TABLE>
        ~;
        &Bottom(0);
        exit;
}

}

#### PRINT TOP ####
sub Top {

print qq~
<HTML><HEAD><TITLE>Access Denied -- Solution Scripts</TITLE>
<style type="text/css">
  body { font-size: 12px; font-family: verdana,helvetica,arial; }
  td { font-size: 12px; font-family: verdana,helvetica,arial; }
</STYLE>
</HEAD>
<BODY LEFTMARGIN="0" RIGHTMARGIN="0" TOPMARGIN="0" BOTTOMMARGIN="0"
BGCOLOR="white" TEXT="black" link=blue vlink=blue alink=blue>

<TABLE WIDTH=100% CELLPADDING=4 CELLSPACING=0 BORDER=0 BGCOLOR="NAVY">
<TR><TD width=100%>
<FONT color="white" size=+2>Access Denied</FONT>
```

```
   
<FONT color="white" size=+1>from</FONT>
<A HREF="http://solutionscripts.com"><FONT color="white" size=+1>Solution
Scripts</FONT></A>
</TD>
<TD NOWRAP align=right>
<FONT color="white">
Version $VERSION<BR>
<A HREF="http://faq.solutionscripts.com/docs/accessdenied"><FONT
color="white">Manual</FONT></A>
</TD>
</TR></TABLE>
<BR><BR>
<DIV ALIGN=CENTER>

~;

}

#### BOTTOM ####
sub Bottom {


print qq~
</DIV>
<BR>
<BR>
<TABLE bgcolor=white border=0 cellpadding=2 width=100%>
<TR>
<TD NOWRAP align=right><font size=-2>Copyright 2000 <A
HREF="http://solutionscripts.com">Solution Scripts</A>   </FONT></TD>
</TR></TABLE>
</BODY></HTML>
~;

}

#### ERROR ####
sub error{

$errors = $_[0] ;

print qq~
<TABLE cellpadding=0 cellspacing=0 border=0 bgcolor="#CCCCCC">
<TR><TD><TABLE cellpadding=8 cellspacing=1 border=0 width="100%">
<TR BGCOLOR="#FFFFFF"><TD align=center>
<B>An error has occured</B><BR><BR>
The error is: <B>$errors</B><BR>
<B>$!</B>
<BR><BR>

<A HREF="http://faq.solutionscripts.com/docs/accessdenied" target="_BLANK">Access Denied
Documentation</A>
<BR>
<A HREF="http://forum.solutionscripts.com" target="_BLANK">Solution Scripts help forums</A>
```

```
</TD></TR></TABLE>
</TD></TR></TABLE>
~;
&Bottom;
exit;

}

sub sort_hashs {
        my $x = shift;
        my %array = %$x;
        sort { $array{$b} cmp $array{$a}; } keys %array;
}

######### SUB CHECK PASSWORD #########
sub check_pwd {

# portions of code taken from anlpasswd
# ftp://coast.cs.purdue.edu/pub/tools/unix/anlpasswd
# Systems Support Group
# Mathematics & Computer Science Division
# Argonne National Laboratory
# Argonne, Illinois

(my $pass,my $username,my $nickname) = @_;

$nickname = $username unless $nickname;

my $pwderror;

#       if ($pass && length($pass) > 8) {
#          $pass = substr($pass,0,8);
#   };

   # Embedded null can spoof crypt routine.

   if ($pass =~ /\0/) {
        $pwderror = "Please don't use the null character in your password.\n";
        return ($pwderror);
   }

   if ($pass =~ /:/) {
        $pwderror = "Please don't use the colon character in your password.  Some vendors
have\n";
        $pwderror .= "a problem in changing passwds to something new when the old has a
colon.\n";
        return ($pwderror);
        }

   if (length($pass) < 6) {
        $pwderror = "Please use at least 6 characters.\n";
        return ($pwderror);
   }

        if ($pass =~ m!^[A-Z]*\d+$!) {
                $pwderror = "Upper case numbering\n";
```

```
            return ($pwderror);                                 #Upper case with numbers
}

if ($pass =~ m!^\d+\s$!) {
        $pwderror = "Watch that whitespace\n";
            return ($pwderror);                                 #Numbers with spaces
}

if ($pass =~ m!^\s\d+$!) {
        $pwderror = "Watch that whitespace\n";
            return ($pwderror);                                 #Spaces with numbers
}

if ($pass =~ m!^[A-Z]*\s$!) {
        $pwderror = "Watch that whitespace\n";
            return ($pwderror);                                 #Uppercase with spaces
}

if ($pass =~ m!^[a-z]*\s$!) {
        $pwderror = "Watch that whitespace\n";
            return ($pwderror);                                 #lower case with space
}

if ($pass =~ m!^\s[A-Z]*$!) {
        $pwderror = "Watch that whitespace\n";
            return ($pwderror);                                 #spaces with Upper case
}

if ($pass =~ m!^\s[a-z]*$!) {
        $pwderror = "Watch that whitespace\n";
            return ($pwderror);                                 #spaces with lower case
}

if ($pass =~ m!^[a-z]*$!) {
        $pwderror = "Mix those cases\n";
            return ($pwderror);                                 #all lower case
}

if ($pass =~ m!^\d+$!) {
        $pwderror = "It's passWORD not NUMBER\n";
            return ($pwderror);                         #all numbers
}

if ($pass =~ m!^[a-z]*\d+$!) {
        $pwderror = "Something a little more complex please\n";
            return ($pwderror);             #lower case with number
}

if ($pass =~ m!^[A-Z]*$!) {
        $pwderror = "no need to SHOUT your choice\n";
            return ($pwderror);                         #all uppercase
}

if ($pass =~ m!^\d+[a-z]*$!) {
        $pwderror = "Something a little more complex please\n";
            return ($pwderror);             #number and lower case
```

```
        }

        if ($pass =~ m!^\d+[A-Z]*$!) {
                $pwderror = "Something a little more complex please\n";
                   return ($pwderror);              #number and upper case
        }

        if ($pass =~ m!^[A-Z]*[a-z]$!) {
                $pwderror = "A little backwards today?\n";
                   return ($pwderror);                      #all upper followed by lower
        }

        if ($pass =~ m!^[A-Z][a-z]*$!) {
                $pwderror = "Standard capitolization\n";
                   return ($pwderror);                                  #Standard first letter cap.
        }

        if ($pass =~ m!^[a-z][A-Z]*$!) {
                $pwderror = "Feeling a little shifty today\n";
                   return ($pwderror);                      #Backwards of above
        }

        if ($pass =~ m!^[a-z]*[A-Z]$!) {
                $pwderror = "Mix it up a little more\n";
                   return ($pwderror);                                  #all lower with last cap.
        }


 if ($pass =~ m!^[-\d/]*$!) {
        if ($pass =~ m!^\d{3}-\d{2}-\d{4}$! ||
           $pass =~ m!^\d\d\d\d\d\d\d\d\d$!) {
           $pwderror = "Please don't use a Social Security Number!\n";
           return ($pwderror);
        }
        if ($pass =~ m!^\d*/\d*/\d*$! ||
           $pass =~ m!^\d*-\d*-\d*$!) {
           $pwderror = "Please don't use dates.\n";
           return ($pwderror);
        }
        if ($pass =~ m!^\d\d\d-?\d\d\d\d$!) {
           $pwderror = "Please don't use a phone number.\n";
           return ($pwderror);
        }
        if ($pass =~ m!^\d{6,7}$!) {
           $pwderror = "Please don't use a short number.\n";
           return ($pwderror);
        }
 }

        my $mo;
if ($mo = ($pass =~ /^[ \d]*([a-zA-Z]{3,5})[ \d]*$/) &&
        ($mo =~ /^(jan|feb|mar(ch)?|apr(il)?|may|june?|july?|aug|sept?|oct|nov|dec)$/i) ) {
        $pwderror = "Please don't use dates.\n";
        return ($pwderror);
 }
```

```
    if ($pass =~ /$username/i) {
            $pwderror = "Please don't use your login id.\n";
            return ($pwderror);
    }

    if ($pass =~ /$nickname/i) {
            $pwderror = "Please don't use part of your name.\n";
            return ($pwderror);
    }

    # A sequence of keyboard keys?
    (my $foo = $pass) =~ y/A-Z/a-z/;
    $foo =~ y/qwertyuiop[]asdfghjkl;'zxcvbnm,.\//a-la-ka-j/;
    $foo =~ y/!@#\$%^&*()_+|~/abcdefghijklmn/;
    $foo =~ y/-1234567890=\\`/kabcdefghijlmn/;
    my @ary = unpack('C*',$foo);
    my $ok = 0;
    for (my $i = 0; $i < $#ary; ++$i) {
            my $diff = $ary[$i+1] - $ary[$i];
            $ok = 1 if $diff > 1 || $diff < -1;
    }
    if (!$ok) {
            $pwderror = "Please don't use consecutive keys.\n";
            return ($pwderror);
    }


    # Repeated patterns: ababab, abcabc, abcdabcd
    if ( $pass =~ /^(..)\1\1/
      || $pass =~ /^(...)\1/
      || $pass =~ /^(....)\1/ ) {
            $pwderror = "Please don't use repeated sequences of $1.\n";
            return ($pwderror);
    }

    # Reversed patterns: abccba abcddcba

    if ( $pass =~ /^(.)(.)(.)\3\2\1/
      || $pass =~ /^(.)(.)(.)(.)\4\3\2\1/ ) {
            $pwderror = "Please don't use palindromic sequences of $1$2$3$4.\n";
            return ($pwderror);
            }

    my $reverse = reverse $username;
    if ($pass =~ /$reverse/i) {
            $pwderror = "Please don't use your login id spelled backwards.\n";
            return ($pwderror);
    }

return(1); ## GOOD PASSWORD ## whew....!!
}
1;
```

```
---------------------- Code for creating a page ----------------------
#!/usr/bin/perl

use CGI;

my $q = new CGI;

my $name   = $q->param("name");
my $pnum   = $q->param("pnumber");
my $inst   = $q->param("instance");
my $inst_t = $q->param("instance_text");
my $page   = $q->param("page");

$SENDMAIL = '/usr/sbin/sendmail';

@AUTHURLS = ('www.cid.ac','cid.ac');

$TO       = 'worsham@wpi.edu';

$SUBJECT  = 'Web page creation request';

$REDIRECT = 'http://www.cid.ac/home/tools/confirm.htm';


###############################################################################

# Check to make sure this script was called by an authorized URL(s)

        &check_url;

# Format Local Date/Time for Email Message Date

        &get_date;

# Reformat Form Contents

        &reformat_form_data;

# Send the form data to the recipient via e-mail

        &send_email;

# Redirect user to confirmation page

        print "Location: $REDIRECT\n\n";

        exit();

###################################################################

sub check_url
{
        if ($ENV{'HTTP_REFERER'})
        {
                foreach $AUTHURL (@AUTHURLS)
                {
```

```
                        if ($ENV{'HTTP_REFERER'} =~ /$AUTHURL/i)
                        {
                                $check_url = '1';
                                last;
                }
                }
        }
        else
        {
                $check_url = '1';
        }

        if ($check_url != 1)
        {
                print "Content-type: text/html\n\n";
                print "<html>\n <head>\n  <title>Unauthorized URL Referrer - Access
Denied</title>\n </head>\n";
                print " <body>\n  <center>\n   <h1>Unauthorized URL Referrer - Access
Denied</h1>\n  </center>\n";
                print "The form that is trying to use this script resides at: \n";
                print "$ENV{'HTTP_REFERER'}, which is not allowed to access this cgi
script.<p>\n";
                print "Sorry!\n";
                print "</body></html>\n";
                exit;
        }

}

sub get_date
{
        @days = ('Sunday','Monday','Tuesday','Wednesday','Thursday','Friday','Saturday');
        @months = ('January','February','March','April','May','June','July',
                'August','September','October','November','December');

        ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
        if ($hour < 10)
                { $hour = "0$hour"; }
        if ($min < 10)
                { $min = "0$min"; }
        if ($sec < 10)
                { $sec = "0$sec"; }
        if ($year >= 100)
                { $year = $year - 100; }
        if ($year < 10)
                { $year = "0$year"; }
        if ($year < 90)
                { $cent = "20"; }
        else
                { $cent = "19"; }

        $date = "$days[$wday], $months[$mon] $mday, $cent$year at $hour\:$min\:$sec";

        $mon = $mon + 1;
        if ($mon < 10)
                { $mon = "0$mon"; }
```

```
        if ($mday < 10)
                { $mday = "0$mday"; }

        $dateShort = "$cent$year\-$mon\-$mday";
        $timeShort = "$hour\:$min\:$sec";

}

sub reformat_form_data
{



}

sub send_email
{
# Build the 'from' address of the form: "name <email address>"

        $from_name=($CONFIG{'Name'} . " <" . $CONFIG{'email'} . "> ");

        open(MAIL,"|$SENDMAIL -t") || die "Can't open $mailprog!\n";

# Output the mail header

        print MAIL "To: $TO\r\n";
        print MAIL "From: $name\@arthritiscare.org.uk\r\n";
        print MAIL "Reply-To: $name\@arthritiscare.org.uk\r\n";
        print MAIL "Subject: Web page upload request\r\n\n";

# Output the mail message header with the Local Date/Time

        print MAIL "---------------------------------------------------------------\n\n";
        print MAIL "   The following information was submitted by: \n";
        print MAIL "   $name at $pnum on $date\n\n";
        print MAIL "   They would like the $inst or $inst_t page upload\n\n";
        print MAIL "--------------------cut and paste from here down----------------\n\n";

# Output the mail body
        print MAIL <<end_of_mail;

<html>

<head>
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<meta name="GENERATOR" content="Microsoft FrontPage 4.0">
<meta name="ProgId" content="FrontPage.Editor.Document">
<title>Arthritis Care Internal Connection</title>
</head>

<body>

<table border="0" width="100%" cellspacing="0" cellpadding="0">
  <tr>
    <td width="100%">
```

```
    <table border="0" width="100%" cellspacing="0" cellpadding="0">
      <tr>
        <td width="100%">
          <table border="0" width="100%" cellspacing="0" cellpadding="0">
            <tr>
              <td width="95%" background="/Images/tp_bg.jpg" height="100%">
                <font size="6" face="Arial"> Arthritis
                Care Internal Connection</font></td>
            </tr>
            <tr>
              <td width="95%" background="/Images/tp_bg.jpg" valign="bottom">
               

                                        </td>
            </tr>
            <tr>
              <td width ="95%" background="/Images/tp_bg.jpg">
              <form method="POST" action="/cgi-bin/dbsearch.cgi">
              <IMG SRC="/Images/top_nav.gif" USEMAP="#top_nav.gif" width=409 height=26
BORDER=0 align="bottom"><MAP NAME="top_nav.gif">
                                                <AREA shape=RECT coords="5, 0, 64, 25"
HREF="/home/">
                                                        <AREA shape=RECT coords="68, 0, 138, 25"
href="/home/training/">
                                                        <AREA shape=RECT coords="150, 0, 197, 25"
href="/home/tools/">
                                                        <AREA shape=RECT coords="212, 0, 312, 25"
HREF="/cgi-bin/docs.cgi">
                                                        <AREA shape=RECT coords="323, 0, 408, 25"
HREF="/directory/directoryQS.htm">
                                                </MAP>
                <input type="text" name="name" size="16"><input type="submit" value="Go"
name="B1">
              </form>
                            </td>
          </table>
        </td>z
        <td width="9%" valign="top" align="right"><img border="0" src="/Images/logo_top.gif"
width="126" height="58"><br>
          <img border="0" src="/Images/logo_bot.gif" width="126" height="34"></td>
      </tr>
    </table>
  </td>
 </tr>
 <tr>
  <td width="100%"><img border="0" src="/Images/nb_top.jpg" width="100%" height="3"></td>
 </tr>
 <tr>
  <td width="100%">
    <table border="0" width="100%" cellspacing="0" cellpadding="0" height="416">
      <tr>
        <td width="170" bgcolor="#FFFF99" valign="top" height="345">
          <table border="0" width="170" cellspacing="0" cellpadding="0" height="283">
            <tr>
              <td width="170" valign="top" bgcolor="#FFFF8C" height="283"><br>
                <a href="/home/ceo/">
```

```
            Chief Executive Officer</a><br>
            <br>
            <a href="/home/depts/">
            Departments</a><br>
              <a href="/home/depts/acc/">Accounts</a><br>
              <a href="/home/depts/com/">Communications</a><br>
              <a href="/home/depts/fac/">Facilities Management</a><br>
              <a href="/home/depts/fin/">Finance</a><br>
              <a href="/home/depts/it/">Information Technology</a><br>
              <a href="/home/depts/pp/">Public Policy and</a><br>
              <a href="/home/depts/pp/">Campaigning</a><br>
              <a href="/home/depts/svr/">Services</a><br>
              <a href="/home/depts/ss/">Support Services</a><br>
            <br>
            <a href="/home/regions/">Regions</a><br>
              <a href="/home/regions/nireland/">Northern Ireland</a><br>
              <a href="/home/regions/scotland/">Scotland</a><br>
              <a href="/home/regions/nengland/">North England</a><br>
              <a href="/home/regions/cengland/">Central England</a><br>
              <a href="/home/regions/swengland/">South West England</a><br>
              <a href="/home/regions/seengland/">South East England</a><br>
              <a href="/home/regions/wales/">Wales</a><br>
            <br>
            <a href="/home/hotels/">Hotels</a><br>
              
           </td>
          </tr>
         </table>
        </td>
        <td width="5" background="/Images/nb_side.jpg" rowspan="2" height="416"> </td>
        <td width="100%" valign="top" rowspan="2" height="416">
         <table border="0" width="100%" cellpadding="2">
          <tr>
           <td width="100%">
         end_of_mail
         print MAIL $page;
         print MAIL <<end_of_mail;
</tr>
         </table>
        </td>
       </tr>
       <tr>
        <td width="170" bgcolor="#FFFF8C" height="71">
        Search:
             <form method="POST" action="--WEBBOT-SELF--">
              <!--webbot bot="SaveResults"
              U-File="C:\htdocs\_private\form_results.txt"
              S-Format="TEXT/CSV" S-Label-Fields="TRUE" -->
              <p><input type="text" name="T1" size="18"><input type="submit" value="Go"
name="B1"></p>
             </form>
        </td>
       </tr>
      </table>
    </td>
  </tr>
```

```
  <tr>
    <td width="100%">
      <hr noshade color="#808080">
      Arthritis Care, Internal Use Only<br>
      <br>
      Maintained by: <a href="mailto:webmaste@cid.ac">webmaster</a><br>
      Last modified:  <!--webbot bot="Timestamp" S-Type="REGENERATED"
      S-Format="%d %B %Y %H:%M" -->
    </td>
  </tr>
</table>

</body>

</html>
end_of_mail;


# Output the mail footer

        print MAIL "<REMOTE HOST>    $ENV{'REMOTE_HOST'}\n";
        print MAIL "<REMOTE ADDRESS> $ENV{'REMOTE_ADDR'}\n";
        print MAIL "<USER AGENT>     $ENV{'HTTP_USER_AGENT'}\r\n";

# Close the pipe and send the mail

        close(MAIL);
}
```