12-2001

Delivering a Shopping Website Using Apache Web Server on UNIX

Norman Dean Harrod

Follow this and additional works at: https://scholar.dsu.edu/theses
MSIS INTEGRATIVE EXPERIENCE
INFORMATION SYSTEMS PROJECT

Project Approval Form (PAF)

Student Name: Norman Dean Harrod

SSN: ____________________________

Expected Graduation Date: December 16, 2001

Master’s Project Title: Delivering a Shopping Website Using Apache Web Server on UNIX

Approvals/Signatures:

Student: ________________________ Date: 12/18/01
Faculty supervisor: ______________ Date: 12/18/01
Committee member: ______________ Date: 12/18/01
Committee member: ______________ Date: __________
Dean: __________________________ Date: 12/18/01

Copies to:
Original to Graduate Office; copies to: Advisor, Dean’s Office, and Student

Revised 12/18/2001
Delivering a Shopping Website Using Apache Web Server on UNIX

BY: Norman D. Harrod

A project submitted in partial fulfillment of the requirements for the Master of Science in Information Systems

Dakota State University
December 2001
Abstract

This project explores three areas related to deploying a small-scale website that can be implemented on a home-built UNIX server. The first component is the installation and configuration of a UNIX Solaris 8 operating system on a Sun Microsystems SPARCstation 5. The second area is the installation and configuration of Apache Web Server on UNIX with the PHP scripting language module. The third piece is the construction of a shopping website to demonstrate PHP programming and usability. The result is, "Delivering a Shopping Website Using Apache Web Server on UNIX."
# Table of Contents

Introduction ........................................................................................................... 1

Chapter 1 Selecting the Hardware and Software ...................................................... 3  
   SPARCstation 5 .................................................................................................. 3  
   Solairs/UNIX ................................................................................................. 4  
   Apache Webserver ....................................................................................... 6  
   PHP ............................................................................................................... 7  
   Other Technologies Used ............................................................................ 8

Chapter 2 Installation and Configuration ............................................................... 10  
   Network Information and Partitioning ......................................................... 10  
   Connecting to the Internet ........................................................................... 13  
   DNS and the Personal Web Server ............................................................... 15  
   Apache Web Server and PHP ...................................................................... 17

Chapter 3 Serving a Usable Shopping Website ....................................................... 19  
   Jakob Nielsen's Alertbox and ADA Guidelines .......................................... 19  
   The Website ................................................................................................ 22  
   The Code .................................................................................................... 25  
   Usability Testing ......................................................................................... 30

Conclusion .......................................................................................................... 33

References .......................................................................................................... 36

Appendix A: Gantt Chart ...................................................................................... 38
Appendix B: Kempston's Connecting to Any ISP ............................................. 40
Appendix C: General Installation for Apache with PHP Module ................. 56
Appendix D: Checklist of Checkpoints for ADA Guidelines-Priority 1 ....... 58
Appendix E: Example of Improvement After Usability Testing ............... 60
Appendix F: Website Source Code ................................................................. 64
Introduction

The purpose of the project is to gain individual experience on an unfamiliar UNIX system, take advantage of low cost solutions, and design a website-shopping cart that conforms to the ideas of usability (see Appendix A). The majority of course work in the Master of Information Systems degree at Dakota State University centered on the common Windows 95, 98, or 2000 operating systems. Some experience on a UNIX system was provided from the point of a user, but little from the administrative perspective.

Low cost solutions are the bottom line with this project as well as with many other business projects. Open source software provides powerful solutions to supporting a website and satisfying low cost requirements. Two popular free technologies used in this project are Apache Web Server and PHP. Sun Microsystems has recently offered Solaris 8 operating system for no cost. Hardware costs have also dramatically dropped over the years. Thus, offering equipment and software to test drive UNIX for a MSIS project.

The website shopping cart portion of the project came from my experiences of auctioning collectible pressure lanterns and stoves on Ebay. The project presented a chance to design a shopping website to take this venture a little further. Auction bidders from around the world often requested to view and shop for products before the items appear on Ebay. Perhaps, the buyers were looking to catch good deals before the high bidding begins.

Usability should be the goal of any e-commerce site. If the website discriminates based on design or technology, then sales and customers will go
elsewhere. Jakob Nielsen and the W3C have covered a number of issues related to design and usability. What is good design? How do you test for usability? This report will summarize some of the major properties of a well-designed site and describe the techniques to test and evaluate site usability.

The paper is broken down into three areas. Chapter 1 describes selecting the hardware and software and gives a brief background of those components. Chapter 2 describes the installation and configuration of UNIX, Apache Web Server, and PHP. This chapter focuses on major points that must be considered to successfully install Solaris 8 operating system. The chapter also describes connecting to the internet and the difficulties of providing connectivity to a standalone home server system. The installation of Apache Web Server and PHP shows what must be done to compile and install the software. Chapter 3 describes the tasks to serving a useable shopping website and looks into the code, design considerations, and results of the usability testing. Finally, a conclusion of the project will cover the successes, problems of the project, and future enhancements.
Chapter 1

Selecting the Hardware and Software

Hardware and software for the project had to be low cost or free. Surplus hardware can be found on auction sites for reasonable cost. A large number of Sun Microsystems UNIX workstations are available. Open source software, such as Apache and PHP, provide powerful options to build the project on. The following section briefly covers the software and hardware selected.

SPARCstation 5:

The SPARCstation 5 is a common Sun Microsystems computer manufactured during the mid 1990's based on the sun4m architecture. This workstation came in four cpu varieties; 70Mhz, 85Mhz, 110Mhz, and 170Mhz. The SPARC 4 and 5 are single processor based while the 10 and 20 support multiple processors. All were based on the same chassis, nicknamed the "pizza box" among users. (Cirulnick, 2001) As the processor number and price increases, so does the price!

Ebay’s auction website offered a prime location to pick up hardware. The SPARC 5 system offers a low cost solution for home development of the project and can support a small-scale virtual website. Compared to travel costs to campus labs, this system easily paid for itself. The SPARCstation 5 was never equipped with a modem to dial in or out. Internet dial-up connectivity could only be accomplished with the addition of an external serial modem. The US
Robotics Courier V. Everything (33.6/28.8 Kbps) supports UNIX connectivity through the RS-232 serial port. Upgrading to cable modem service was not an option for the project due to cost considerations. The specifications of the Sun Microsystems SPARCstation 5 setup for this project are listed below:

- 2 x 2 GB Hard Drives 5400 RPM
- 256 MB Memory
- 110MHz TurboSparc CPU
- Internal Diskette Drive
- 3 SBus cards (AFX)
- SCSI
- Parallel
- AUI Ethernet
- Twisted-pair Ethernet
- Two serial ports
- Integrated sound
- External SCSI CD-rom Caddy Load

Solaris/UNIX:

Simply stated, UNIX is a multi-processing, multi-user, and multi-tasking operating system with a high degree of portability. These types of features are typically required among program developers, businesses, and educational institutions using a number of connected workstations requiring control and access to programs, files, or resources. Although choosing the right operating system is quite arguable, Solaris 8 by Sun Microsystems was selected for this project based on its price (free), reliability, compatibility with SPARCstation, and ability to run Linux applications.

UNIX is difficult to define, as the term is general to a number of releases from various vendors since 1969. One definition that reflects this is "UNIX is the name of a computer operating system and its family of related utility programs." (Sobell,
The UNIX operating system was developed by AT&T and has progressed since then. Two popular varieties of UNIX are System V (AT&T) and BSD (Berkeley Software Distribution). However, there are a number of characteristics common to all:

1. A kernel that has the responsibility of allocating system resources evenly among multiple users or jobs. The kernel is written in the C programming language that is portable.
2. Hierarchical or tree-like file system beginning with a root directory and branching out into other determined directories.
3. System hardware devices such as ports or external devices are represented as special files.
4. Process based, in which all services and shells are represented by one process identifying number called the process ID or PID.
5. Share a set of command-line utilities that perform some function (i.e. display, sort, copy, print).
6. User processes can be started from a shell, which execute applications.
7. Multiple processes can be executed concurrently by a single user and sent to the background.
8. Multiple users can execute commands concurrently by logging into pseudoterminals. (Watters & Veeraraghavan, 2000, and Sobell, 1995)

An in-depth report on each UNIX or UNIX-like release is far beyond the scope of this project.

Sun Microsystems is currently offering the Solaris 8 operating system under the Free Solaris Binary License Program (http://www.sun.com/solaris/binaries/). This is certainly an attractive offer to learn and explore the offerings of a UNIX operating system. Major highlights are:

- Power, stability, predictability, backwards compatible
- Stable kernel design for increased load balancing across multiple processors
- Ability to handle heavy traffic, large data sets, and solve complex problems
- Secure environment with support for other security protocols and technology
- Support for 37 languages and 123 locales (regions)
- Simplified installation and integration capabilities
- Supports latest network protocols and complies to all major industry standards. (Sun Microsystems, 2001)

Solaris is available for SPARC (32- and 64-bit) and Intel Architecture (32-bit) platforms and supports up to 8 processors. However, not all PC hardware is supported because of the large number of manufactured devices (Watters & Veeraraghavan, 2000). Another interesting note is Solaris has extensive Linux compatibility through the "lxrun" utility. This allows Linux applications to run on Solaris platforms (Sun Microsystems, 2001). Sun Microsystems offers this for download (huge) or cost of media ($75). Another option is auction sites such as Ebay. The savings will depend on the competing interested bidders. The choice of Solaris 8 for this project was due to its UNIX-like system structure, graphical interface, cost, and "supposed" ease of installation.

Apache Web Server:

Apache is a powerful open source web server software package available for free. Early beginnings sprung from the collaboration of a number of webmasters requiring a common distribution of server software in 1995. They began to exchange changes among each other in the form of "patches", hence Apache. Next, the group formed the Apache Group and continued refinements. The first release of Apache server 0.6.2 was an instant hit and the project gained additional members, volunteers, and user popularity up to present day. The mindset behind the server development was to ensure that no company could
dominate the entire market through exclusive ownership. (Wainwright, 1999, & Engleschall, 2001) The goals of the Apache project:

- Web server that is of commercial grade
- Flexibility through the concept of adding or removing modules
- Able to readily incorporate other technology standards
- Professional tool
- Small server "footprint"
- Continual development from the contributions of others (Wainwright, 1999, & Engleschall, 2001)

Apache Web Server is a major market leader. Netcraft gathers information on web server usage on the internet and posts the Web Server Survey results via the internet. For the last 6 years, Apache has ranked at the top (rankings - Apache, Microsoft, iPlanet, NCSA, Other). The figures showed approximately 60% or 8 million active servers in August 2001. (Netcraft, 2001) No doubt the module functionality seems to be the most noticeable feature of Apache and certainly a reason for the success of Apache Web Server. Modules allow Apache to use other technologies. Even though Apache server is included with many software packages, the latest version is always recommended for download directly from the website. The choice of Apache for this project was based on prior experience with Apache on Windows 95 and 98, cost, and its reputation as proven server.

PHP:

PHP is an open source server-side scripting language for producing dynamic web pages. PHP began in 1994 as a way to track visitors online and embed SQL queries (Atkinson, 2001). Based on the concept of sharing, other
interested parties asked for copies of the code that powered an online resume. The author, Rasmus Lerdorf released Personal Home Page Tools version 1.0. Thus the beginning of another successful open source project evolved through core refinements. Through the contributions of other authors such as Zeev Suraski and Andi Gutmans, PHP (Hypertext Preprocessor – recursive naming) and Zend (e-business company) developed in parallel. Zend represents a commercialized version of PHP for e-commerce applications and solutions. Zend provides support to the PHP open-source community. (Zend Technologies Limited, no date available) Some of highlights of PHP are:

- Short development time
- Ease of use
- Cross platform or platform independent (Microsoft, UNIX, Mac, etc)
- Database support (Informix, Microsoft SQL server, MySQL, Oracle, etc) (Zend Technologies Limited, no date available)

The latest version of PHP is 4.0. PHP runs on server software such as Apache Web Server, Web Ten Web Server, and Microsoft’s Internet Information Server. The operating system can be Linux, Solaris, Windows NT, Windows 95/98/2000, IBM’s OS/2, Apple ‘s OS X and Macintosh OS. (Atkinson, 2001) The choice of PHP to build the dynamic website for this project was based on prior experience with Apache on Windows 95 and 98, cost, and familiar coding.

Other Technologies Used:

   Here, I will briefly point out the other technologies used in the project. HTML, JavaScript, and Cascading Style Sheets are some of the other languages used in this project. HyperText Markup Language (HTML) is a common language used
to create web pages. This project also uses a dash of JavaScript. JavaScript is a popular client-side language used to take advantage of the browser running on the client. There is no guarantee that the user's browser will be able to support this technology. Therefore, testing websites on older browsers should be completed to identify scripts that will not work on older browsers or optional solutions should be included into the website to present the material. Cascading Style Sheets (CSS) help to separate presentation from content in one central file. The idea is to give freedom back to the user, who by use of CSS is not restricted to set design elements that do not work. CSS also allows the page to be viewed on other devices. Pages must continue to work when the style sheets are disabled or not supported. (Nielsen, 1997) The original idea behind HTML was basic page formatting such as paragraphs, text, and backgrounds. HTML is limited in its ability to provide style design. CSS is the answer to providing any kind of design element needed. Although older browsers such as Internet Explorer 3.0 and Navigator 4.0 have bugs and absolutely no support in earlier versions, the need for this separation is logical. (Holzschlag, 1999) Unlike JavaScript, the function of CSS is not required to work in order to view the information.
Chapter 2
Installation and Configuration of UNIX

Installing, configuring Solaris 8, and connecting to the internet on a SPARCstation 5 proved to be challenging. The SPARCstation 5 required a fresh install of a UNIX operating system because the system was shipped with none. This chapter covers details of what must be done to setup a standalone SPARCstation 5 for connectivity to the internet through a dial-up ISP situation. Installation of Apache Web Server and PHP on Solaris 8 operating system takes a little research. It is necessary to download the software from the internet and follow installation instruction that can readily be found on the internet, books, or in the documentation provided with the downloads. If Apache or PHP does not install successfully, it is a good bet that some additional utilities will be needed. This chapter also covers major installation procedures related to this project.

Network Information and Partitioning:

Installation of the Solaris 8 (SPARC Platform Edition) 4/01 software was straightforward but some difficulties did arise. Solaris 8 requires network and hard drive partitioning information. The network information was simply non-networked or standalone. If networked, be prepared to supply parameters such as hostname, IP address, domain name, and subnet mask. Partitioning new hard drives depends on a number of issues related to expected use. Of course, actual use can change because of system demands. Partitioning enhances the
use of the drive and provides a degree of security. The security refers to other
daemon processes (log files) or users filling up the drive. Thus, the system may
become unusable or locked up. Particular attention was given to the following
partitions as recommended by Solaris (Watters & Veeraraghavan, 2000):

- `/` “This is the root partition where essential files such as the kernel are
  stored. It should be large enough to hold the kernel and some additional
  files, with enough free space to handle a few large log files and core files
  from errant processes.”
- `/opt` “This partition is used to store optional applications to be installed on
  the local system. If you need to install several optional applications, this
  partition should be quite large.”
- `/usr` “This partition stores the utilities that make up the heart of the Solaris
  system, including the “bin” and “sbin” subdirectories. If you intend to
  install some of your optional applications in `/usr` instead of in `/opt`, then this
  partition will need to be quite large.”
- `/var` “This partition is used to store volatile files, such as log files, mail
  files, and print jobs. If you are running a mail server or a print
  server...large...”
- `swap` “This allows you to run more programs than will fit into memory at
  once. You always want this.” (Watters & Veeraraghavan, 2000)
- `/export/home` (just for personal user files)

This led to some experimentation and a number of long re-installations.

However, not all problems were exclusively related to partitioning (Note: average
installation takes approximately 3 hrs with a 1x external Sun CD-ROM drive). If
there was only one hard disk drive, the recommendation was four partitions: root,
/usr, swap, and one partition for everything else. The problem was that the
project machine had two separate 2 GB hard drives. The general
recommendation for two drives:

- Disk 1: root, `/usr`, swap, `/export/home`
- Disk 2: `/opt`, swap

The first attempted installation left all of the second drive to `/usr`. Unexpectedly,
this resulted in extremely poor CPU and disk drive performance. Apparently, a
swap partition on both hard drives is necessary for good overall performance. Disk 1 has partitions swap, /, /opt, /var. Disk 2 has partitions swap, and /usr. Certainly the exact MB allotted to each partition could be adjusted upon future system requirements. The partitions were based on Solaris recommendations. There was no reason to have a /home directory for additional users and no partition was created. Simply, this was a web server and no other users need to use the SPARCstation 5. There was no cookbook answer because partitioning is dependent on the needs of the system – an estimated guess. (Rutgers University Computing Services, 2000)

Common Desktop Environment, Xterm/Terminal, and Admintool utilities were used through the rest of the project for file manipulation, connecting to the internet, and startup and shutdown. The Common Desktop Environment (CDE) is the graphical user interface to Solaris. Windows users will immediately find this interface as something friendly to use. Upon setup, CDE was selected as the default for login. Login and system adjustments were done under root, which allows access to adjusting many files, programs, and accounts. That is the extent of any similarity to Windows unless one backs up to DOS. Any major adjustments to the system will usually have to be done through a terminal program like Xterm, Terminal, or other utilities. For example, editing system files for dialup connection. Admintool allows for some system changes and viewing processes through a graphical interface. Admintool will also identify serial ports and setting connection parameters for modem dialing. If new to UNIX, prepare for some rocky times as it is easy to make mistakes, cause system problems
requiring expert remedy, or typing tons of command lines. It is helpful to have reference books and patience.

Connecting to the Internet:

Connecting to the internet was a daunting task. Sun SPARCstations were not equipped with a modem for dial-up. As prices have dropped over the years, these machines have made their way to home users who want to explore the power of UNIX. An internet source, [http://www.kempston.net/solaris/connectanyisp.html](http://www.kempston.net/solaris/connectanyisp.html) (see Appendix B) gives step-by-step instructions on how to connect a SPARCstation to an internet service provider (ISP). Essential information for connection is the domain of the ISP, primary and secondary Domain Name Servers IP addresses, and authentication method. Sounds easy? Not really, most ISP providers are servicing users who have PCs with dialup programs and installed modems that do all the work. Simply supply the telephone number, username, and password to gain internet connectivity to the internet on a PC (i.e. NetZero, Prodigy).

Locating the SPARCstation information requires a direct letter to the ISP provider or deep digging into the ISP's website. Depending on the version of Solaris, there are some installed PPP (Point-to-Point Protocol) packages that must be located to verify that they are installed. Solaris 8 had all of the necessary packages. When connecting the external modem to a serial port, it is important to verify which port the modem is on. Otherwise, changes made by Admintool and test commands to the modem will not work. After the script files are made
active for dial-up and hang-up, additional editing might need to be performed. A lot of extra time was committed to changing modem dip switches and editing files because configuring was originally done for the wrong serial port. Once communication with the modem was established, finding out the authentication procedure and actually connecting to the ISP can be frustrating.

Communication between the hardware and ISP must be synchronized. Authentication methods to choose from with the ISP are Login and Password, PAP, or CHAP negotiation. If this information can not be readily found from the ISP, trial and error will eventually establish the authenticated internet connection. Selecting the correct speed for communication between the serial port hardware and the modem is essential. Upon dial-up, communication can be viewed via another terminal window. If there are problems, garbage and garble will be displayed. The SPARCstation 5 serial port supports a serial Data Terminal Equipment (DTE) speed of 9600 and all files related to the dialup script had to be changed to 9600. Eureka! An established connection has been accomplished with the ISP provider and the modem indicators set idle with an active connection. The next issue was finding a browser to surf the internet to get the required files for the project.

Netscape 4.75 default Solaris 8 installation package would not communicate through a dial-up ISP connection. Suggested fixes for earlier versions 4.x did not resolve the problem. To speculate, perhaps some additional security methods or settings needed to be adjusted or cleared – nothing could be found specific to 4.75. Since earlier versions had no problems, Netscape 3.01 was
installed with the browser option only. With Netscape 4.75 still installed and
3.01 on the same system, no internet surfing could be completed. With another
fresh install of the operating system without Netscape 4.75, Netscape 3.01 could
access internet sites. Netscape 3.01 was adequate to download files required
for installing and supporting Apache and PHP. The download speed for pages
and files was extremely slow. For example, Ebay’s home page took 3 minutes to
download. Netscape 3.01 did not support many of the internet technologies
common to dynamic and application website content. Opera 5.0 for Solaris Beta
1 provided all of the necessary support to visit current websites, download files,
and test upcoming website design on the SPARCstation 5 (i.e. cascading style
sheets).

DNS and the Personal Web Server:

Reasons for running a personal web server are many. The core of this
project relies on setting up and running a server that could respond to page
requests over the internet. Making adjustments for a low cost project requires
one important factor, a static IP address. If other users on the internet know the
IP address that a web server is connected to, file serving can be done through
the internet.

ISP’s provide dynamic IP addresses. The reason that the number changes is
that there are many users requiring internet connections and so few IP numbers.
ISP’s only allocate enough numbers to cover the number of clients that they have
connected at any one time. Most users are not connected to the internet all the
time. Therefore, IP addresses allocated get shared around to use resources efficiently. The problem is how to keep others coming to the current IP address. The answer is the Domain Name Server (DNS).

The DNS has registered names such as "Ebay.com" pointing to IP address numbers. There are free and non-free sites providing services to register a Domain Name and point to the static or dynamic IP address used to access the internet. Ethics and caution should be exercised at this time. ISP providers may not permit secondary activities such as running a business requiring high bandwidth through a regular user internet account or competing services. A number of Windows and UNIX programs are offered to update the DNS to continually point to the currently used dynamic or static IP address. Because the credibility of these free DNS services could not be readily verified, a static IP address was requested from DSU's Computing Services. Unfortunately, Computing Services could not accommodate this request because they are setup for dynamic IP addressing. By typing in the set IP address into the browser URL address bar, the personal web server can be accessed over the internet. The alternative was uploading all website project files to an account with DSU for demo-only or testing purposes.
Apache Web Sever and PHP:

Installing Apache Web Server and PHP requires some extra supporting utility files. The Solaris 8 installation package supplies an Apache Web Server version. Although this may seem to be convenient, it is best to download the latest version from the Apache website because usually the Apache software is not supplied on the current system or included in an upgrade or initial operating system installation. Also, the latest version would feature all recent enhancements and fixes. Apache 1.3.20 was downloaded in this manner. PHP 4.0 was downloaded directly from the PHP website.

The downloaded Apache and PHP files come in a compressed format and must be uncompressed (similar to the zipped files in Windows). The directories contain the source code, utilities, and documentation. On the first Apache compilation, it was found that the required compilers were not available. PHP noted a list of required software utilities for compiling and related tools (gcc, make, flex, bison, m4, autoconf, automake, perl, gzip, tar) (PHP, 2001). The needed tools were acquired from http://www.sunfreeware.com/. All utilities were to be installed in /usr/local director and this made installation a snap! Apache compiled with no problems. The common directory picked to install Apache is /usr/local/httpd/ (Watters & Veeraraghavan, 2000). The best directions for installing Apache with the PHP module are described in Appendix C.

On a side note, at some point it may become necessary to shut down the system. It is important to note that shutting down the UNIX system must be done correctly. UNIX can not just be powered off while running the CDE interface or
during the login prompt. Shutting down the power switch at any point during operation is risky business. A number of errors will be introduced into the system tables that manage disk space, which will need correcting during the next power-up. It is possible to damage the hard drives and render the machine inoperable (Watters & Veeraraghavan, 2000). A combination of two commands successfully brought the system down. In the CDE terminal screen, issue the command "/usr/sbin/shutdown." The command brings the system down and warns other users who might still be on the system. Then, a prompt will be waiting for the administrator to login. Login and issue the "poweroff" command to completely shutdown the system correctly. This will help ensure that all of the installation work will not be lost to an unresponsive or damaged file system on the hard drive.
Chapter 3

Serving a Useable Shopping Website

This portion of the project deals with designing a shopping website that adheres to some of Jakob Nielsen’s AlertBox and ADA Guidelines. Nielsen covers many areas of usability, but only the major ones will be applied to the shopping portion of the website. One can think of this as a “before and after” shot of a section of the website – the shopping cart. While Nielsen addresses people with disabilities, the World Wide Web Consortium (W3C) also gives recommended checklists to promote and check for usability compliance. The website acts as a testing ground for applying the checklist and exploring additional tools that can be used to meet usability issues. The shopping website idea comes from experiences of auctioning a small quantity of hard-to-find pressurized lanterns and stoves on Ebay. The selection of different items rarely exceeds twenty as the market is specialized. This chapter describes usability design issues, ADA guidelines, and testing methods used on this project.

Jakob Nielsen’s AlertBox and ADA Guidelines:

Web usability looks at design issues and the way users interact with web technologies. Dr. Jakob Nielsen has authored many columns related to web design and disabilities. Column articles that appeared relevant to this project were given priority.
Jakob Nielsen has some columns that could be addressed as the "Top Tens."

These are important suggestions regarding common mistakes and are well-worth
adhering to. At the time of this writing, Nielsen is currently presenting his list of
"top ten web-design mistakes of 2001" at the User Experience 2001/2002
conferences and no article is available.

The following is a summation of common mistakes:

1. Splitting up of the webpage by using frames. New browsers are
   offering better support for bookmarking, but some do not bookmark the
   actual information in the frame.

2. Relying heavily on cutting edge technology. Beta releases can crash
   user systems. Use technology wisely and appropriately to the
   content. Let the technology develop. If a user runs into a JavaScript
   error, they usually leave the site. All too familiar! The exception here
   is the companies who are promoting the new products or services.

3. Scrolling text, marquees, and blinking or animations are distracting.
   Advertisers are trying to utilize this more and users are ignoring it.

4. Orphan pages resulting from no navigation because a user accessed
   by some other way besides the home page. The user becomes lost!
   Orphan pages are still there but users have learned how to get to the
   home page.

5. Long scrolling pages. All critical information and navigation should be
   viewable. Users have also learned that some links are lower. More
   tolerance to longer pages but use only if absolutely needed.

6. Lack of navigation support. Users need extra structure to go by, as
   they are not familiar with the site. Suggestions are site maps or a site
   index. Lack of navigation support is no longer an issue, but a big
   mistake if navigation is not there.

7. Non-standard link colors. Leave them as they are; not seen – blue or
   visited – purple or red. Users get confused with anything that is
   underlined (use of underlining not recommended).

8. Out-dated information needs to be removed. Maintenance is a part of
   the web site. This problem has only increased as the internet has
   grown.

9. Long download times. Ten to fifteen seconds is the traditional user
   tolerance time limit. Sites are still overusing graphics. Bandwidth is a
   continual problem in comparison to user growth. (Nielsen, 1996, &
   Nielsen, 1999 May 6)
The following summation features the latest good design features a site should implement:

1. Name and logo on every page with the logo linking back to the homepage. Logo on the homepage should not have a link as no link should come back to the same page the link is on.
2. Search engine for a site with more than 100 pages.
3. Simple headlines and page titles that make sense out-of-context on search engine results.
4. Increase scanning on pages to help readers quickly find what they are looking for. For example a outline type structure with main topics standing out.
5. Topics or products should be broken down into multiple linked pages to help users find what they want to see.
6. Use product photos with thumbnails and if needed, use larger ones on later pages for more detail.
7. Relevance-enhanced image reduction. This means instead of making a large picture smaller and losing detail, focus on a particular part that might convey the best clues about the item.
8. Link titles give previews of what is on the up-coming page if it can not be deducted from the link before clicking.
9. Accessibility for users with disabilities.
10. Do the same as the big boys. “Jakob’s Law of the Web User Experience: users spend most of their time on other sites, so that’s where they form their expectations for how the web works” (Nielsen, 1999 October 3)

Nielsen also has a number of articles in support of disabled user accessibility.

Other points are brought into the spectrum but he states, “We should consider these users as users: As people who have jobs to perform and goals to accomplish when they use websites and intranets” (Nielsen, 2001).

Websites can further increase their usability by meeting the ADA guidelines presented by World Wide Web Consortium (W3C). Disabilities include visual, auditory, motor, and cognitive. The W3C is a central organization composed of many groups interested in promoting standards across the World Wide Web for the benefit of all involved. One of the domain activities is the Web Accessibility Initiative (WAI). The WAI is focused on accessibility to all people. Included
areas are technology, guidelines, tools, education and outreach, and research and development. The recommended guidelines are found in Web Content Accessibility Guidelines 1.0 (http://www.w3.org/Consortium/). Accessibility checkpoints are rated as priority 1 (must), 2 (should), and 3 (may). Priority 1 is the minimal requirement and the goal of the virtual shopping website portion of this project. The Checklist of Checkpoints for Web Content Accessibility Guidelines 1.0 (W3C, 1999) will be used. See Appendix D for a detailed listing.

Some useful programs are available to validate the guidelines presented by the W3C. Bobby, Vischeck, and IBM Home Page Reader 3.01 will be used to evaluate this section of the project. Bobby helps identify and suggest possible repairs/causes to accessibility problems (http://www.cast.org/bobby/). Vischeck simulates colorblind vision (http://vischeck.com/index.php3). IBM Home Page Reader 3.01 acts as a reader for vision-impaired users (http://www-3.ibm.com/able/hpr.html).

The Website:

The shopping website portion of the project was designed to showcase programming technologies and usability features. The virtual site caters to collectors of hard-to-find pressure lanterns and stoves. The market is small and items only come in small quantities. The code is written in PHP and documented through comments. The pages were tested to see how they performed against various accessibility tests. A page from the created website displaying pre- and post-test results can be found in Appendix E.
Figure 1, Overview of website page structures.

The site layout was constructed to look like other top sites such as Amazon.com and Ebay.com. The standard logo, Lantern and Stove Connection, is displayed in the upper left corner. The letters are dark brown positioned on a white background. The background of each page appears “new tan” in color. Navigation is provided just below the logo across the top. The bar appears white to help the links stand out. The pages are Home, Products, View Cart, Empty Cart, Check Out, Customer Service and Links, with legal information located at the bottom of the page. The links are the standard colors of blue before being visited and purple afterwards. On mouse-over, the clickable link turns green due to the cascading style sheets. The page structures can be viewed in Figure 1.

The concentration of the usability testing was placed on the shopping cart pages. A shopping cart is very similar to a shopping cart in a grocery or department store. Items are viewed (just like items on a shelf), added (picked off the shelf and dropped in the cart), and removed (placed back on the shelf).
When the shopping cart contains the desired items, the customer proceeds to checkout where necessary payment and shipping information is taken.

The Products page lists all currently available items. The user can add the item and the quantity desired in addition to viewing the description. A detailed product description and picture of the item comes up in a pop-up window when the description link is clicked (see Appendix E, p 63). Once the “Add Checked Items to Shopping Cart” button is clicked, the customer is taken to the View Cart page (Appendix E, p 62). This page shows what is currently in the cart, quantity ordered, price per item, total per item, subtotal, and allows for removal or editing of quantities. By returning to the Products page at this point, the customer can view all products not currently in shopping cart or removed earlier and make desired changes. From here, the customer will most likely empty the cart or proceed to checkout. The Empty Cart page simply states that the cart has been emptied and empties the cart. The Checkout page shows the shopping cart with subtotal and a total with tax less shipping. At this point the customer can submit the order for verification by clicking on the “Submit Order” button. If the send is successful the customer will receive an automated e-mail response acknowledging the order. The order must be e-mailed to customer service to check for availability of product quantity and shipping costs. The Customer Service page allows the customer to ask a question, in addition to supplying other general customer service information. To submit a question, the customer simply types the question in the Comments box and clicks the “Submit Question” button. An automated e-mail response will be sent to the customer.
The form "reset" buttons were removed to increase usability. During testing on some users, the "reset" button was clicked in hopes of making the changes selected. This just cleared all of the choices and a confused look was a result. Upon research at Jakob Nielsen’s Alertbox (2000), he suggested reset and cancel buttons should not be used. The reset button can be clicked accidentally. Users should simply be able to edit the check or text boxes with little effort without the aid of a complete reset. By removing this button, the natural flow of the next step is a simple one click. Keep the steps simple and narrow as possible. (Nielsen, 2000)

The Code:

Site information relies on the manipulation of six session variables (see figure 2).

```php
<?
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
```

**Figure 2, Registering session variables using PHP.**

User sessions are stored as files on the server. A session identifier is sent to the user as a cookie. It is possible for the user to reject the use of cookies. PHP allows this information to also be passed via the URL. (Atkinson, 2001) Five of the session variables store arrays.
Arrays are lists of information that are indexed. An analogy is building blocks lined up in a horizontal row and each block has a piece of information. The blocks start from position 0 to 4 (the index numbers). Yes, the first block starts with zero. This is one array. Now, five of these arrays will be lined up in parallel and are the same length or same number of indexes. Looking at the diagram in Figure 3, the first ARRAY0 can be used to determine whether or not the lower arrays will be looped through in the same index position. Programming control structures can be used to loop through the number of the index and the number contained in the array name. By naming each array by “ARRAY” and then number, it is easy to loop through each array containing product information by the index number. The number in the array name can then be associated with a count variable. Each index number represents one item for sale on the website (0-4 or five items can be found on this site). Cutting vertically down through each of the arrays at the same index number, product information is given for one specific item (shown in the highlighted position in Figure 3). The shopping cart elements that are edited by the user and stored by the shopping cart structure are the quantity and the state of ARRAY0.

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRAY0</td>
<td>True, False Add, Remove</td>
<td>True, False Add, Remove</td>
<td>True, False Add, Remove</td>
<td>True, False Add, Remove</td>
<td>True, False Add, Remove</td>
</tr>
<tr>
<td>ARRAY1</td>
<td>item</td>
<td>item</td>
<td>item</td>
<td>item</td>
<td>item</td>
</tr>
<tr>
<td>ARRAY2</td>
<td>quantity</td>
<td>quantity</td>
<td>quantity</td>
<td>quantity</td>
<td>quantity</td>
</tr>
<tr>
<td>ARRAY3</td>
<td>price</td>
<td>price</td>
<td>price</td>
<td>price</td>
<td>price</td>
</tr>
<tr>
<td>ARRAY4</td>
<td>description</td>
<td>description</td>
<td>description</td>
<td>description</td>
<td>description</td>
</tr>
</tbody>
</table>

Figure 3, Arrays in parallel to loop through product information.

The session variable Setcheck sets the state of the shopping cart as load, filled, or empty. When Setcheck equals “load”, the arrays are filled with the
information and the shopping cart is filled. Setcheck is then set to equal "filled" and means the cart has been filled with some items. If no items are selected or all the items in the cart are removed, then Setcheck is set to "empty" which means the cart is empty. By using checkboxes, textboxes, and forms, a variety of things can be done to those values in ARRAY0 and ARRAY2. The following section of code in Figure 4 is an edited example from the View Cart page.

```php
if($SETCHECK == "filled"){
    print("<form method="post" action="viewcartedit.php" width="100%">"
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == TRUE){
            print("<tr><td><b>Remove</b></td><input type="checkbox" name="Prod">
        print("$i")
        print("\n value="remove">$ARRAY1[$i]<td><td">
        print("<b>Qty </b>"
        print("$ARRAY2[$i]<td><td">
        print("<b>Edit Qty </b>"
        print("<input type="text" size="5" name="changeqty">
        print("$i")
        print("\n value=""</td><td">
        print("PRICE \$ $ARRAY3[$i] Each</td><td">
        print("<b>Price </b>"
        $pricetotal = $ARRAY2[$i] * $ARRAY3[$i];
        printf("$\%5.2f", $pricetotal);
        print("</td></tr>");
    }
}
``` 

Figure 4, Looping control structure for showing items in the shopping cart.

Depending on the state of the shopping cart, the Setcheck variable is tested by an "if" control structure. If the Setcheck variable equals "filled", the HTML table and form is formed by looping through the number of items in a "for" control
structure. ARRAY0 determines whether or not the rest of the array will be printed based on index number (item 0, item 1, etc.). In this particular case, ARRAY0 contains TRUE if the customer has checked the checkbox. If not checked, ARRAY0 contains no information and is FALSE. To view the shopping cart, a “for” control structure loops through each item contained on the website by using a count variable going from zero to four (only five items on the website). Within the “for” control structure, ARRAY0 is tested at each index position to see if it is TRUE. If TRUE, the same index number in each parallel array will be printed in the HTML table, prices of individual items figured accordingly, option to remove the item by a checkbox, and edit the quantity of items through a textbox. If FALSE, the item is ignored and not shown to the customer. This same type of approach is used throughout the rest of the website for adding items or removing items.

To help centralize repeatable code, the PHP server side “include” function was used. These chunks of PHP code held in a separate file are inserted into the page and executed depending on where the function call is made. An example is adjusting the navigation links to not display the same link as the current page or display them all if need be. This was done through a “switch” case control statement in Figure 5. If “index” (we are on the homepage) is passed to the function named loadNavigation, the string “index” will be contained in the “$page” variable and the “switch” control structure will see which case it matches. So “index” matches “index” and the homepage link on the homepage will not be shown. No other testing is required and the code stops execution
because of "break." Therefore, this same function can be called on each page by passing the string name needed to produce the right navigation bar. The server side "include" function is also used for the footer of each page because the same information is repeated on each and every page. Also, all of the functions used are contained in one file and added to each page by using the "include" function.

```php
function loadNavigation($page){
    switch($page){
    case "index":
        print(""
            <center>
                <b>
                    <a href="prod.php">Products</a>
                    <a href="viewcart.php">View Cart</a>
                    <a href="emptycart.php">Empty Cart</a>
                    <a href="checkout.php">Check Out</a>
                    <a href="service.php">Customer Service</a>
                    <a href="links.php">Links</a>
                </b>
            </center>
        ");
        break;
    case "prod":
        print(""
            <center>
                <b>
                    <a href="index.php">Home</a>
                    <a href="viewcart.php">View Cart</a>
                    <a href="emptycart.php">Empty Cart</a>
                    <a href="checkout.php">Check Out</a>
                    <a href="service.php">Customer Service</a>
                    <a href="links.php">Links</a>
                </b>
            </center>
        ");
        break;
    }
}
```

Figure 5, Function using a switch control structure to display navigation.
Presentation features of the site are found in the Cascading Style Sheets which is a single file. The style of the site can be altered from a single page compared to changing the code on each and every page. All source code can be found in Appendix F.

Usability Testing:

Bobby checks a website page for compliance with ADA requirements. Bobby v 3.2 passed most pages as Priority 1 compliant (see Appendix D). The online testing program noted that tables not used for layout should be tested for voice speech synthesizer structure. Pages that rely on dynamic structure, such as the View Cart, could not produce results. The dynamic View Cart has the same problems with the product price tables. The Bobby program noted the use of Cascading Style Sheets and that all pages should be viewable without them. The CSS file was deleted and all pages were indeed readable when viewed by the user. The JavaScript button placed to close the window on the individual product pictured produced a browser compatibility error flag (later browser testing showed no error). Bobby download times were on average below 6 seconds at 28,800 baud modem. The shopping cart portion of the website was redone to include the proper table layout. Appendix E (p 61-62) shows what visual changes were made to accommodate the recommendations by Bobby for screen readers.
Vischeck passed all pictures successfully on typically encountered conditions of color blindness. Pictures were tested based on three forms of vision problems; “deuteranope (a form of red/green color deficit), protanope (another form of red/green color deficit) and tritanope (a blue/yellow color deficit—very rare).” (Vischeck, no date) Due to the color of text (dark brown) and logo picture (dark brown), one could easily speculate that this site would have little or no problems with color vision problems because red and green were avoided. Vischeck does not accommodate cascading style sheets (Vischeck, no date). Therefore, rendering a site in certain colors may cause problems if the user does not turn off the cascading style sheets. Vischeck is currently working on CSS support so the color of the CSS design elements can be tested.

IBM Home Page Reader 3.01 is a page reader for the visually impaired. The test revealed that improvement is needed. As also noted by Bobby, voice readers need additional markup to present the listener with understandable remarks. The initial Products page and the View Cart (Appendix E, p 61-62) was redone to make improvements on the reader test and comply with Bobby’s findings.

Browser tests were completed on Netscape 3.01 and 4.x, Internet Explorer 5.x, and Opera. Browsers on other computer systems were used to test these pages as the SPARCstation 5 only had Netscape 3.01 and Opera. It is incorrect to assume that everyone upgrades to the latest browser. The design of the website intentionally used server-side scripting to avoid problems with browser versions or plug-ins. Visiting with classmates, instructors, family, and friends
certainly showed how cautious users are when JavaScript errors occur or plug-in software is required to view site content. Either the user totally avoids the site or just will not download the extra software. This is a design issue that every web author must address. Deciding to use client-side scripting programs will require coding adjustments to ensure presentation or applications will still work across many browser formats (yes, it can be done but be prepared to write extra code to take advantage of competing browsers). Absolutely no problems were encountered on any of the above browsers when testing the website shopping cart.

This is a very small site and the hardware is adequate for low numbers of users. If usage exceeds the processing capabilities of the hardware, it is logical to have the client-side browser do some of the processing work. This entails some more coding to be done by the web author to compensate for different browser releases. The hardware could also be upgraded. Performance testing should be done upon public release.
Conclusion

UNIX is comprised of many utilities and this report only touches on a few of the features of the Solaris 8 operating system. Beginners will find the system unfriendly without the help of detailed reference materials. This is especially true if used from the point of the administrator and not that of a user with an account. With the availability of Solaris 8 at no cost, this is an excellent opportunity to take a new operating system for a test drive.

Connecting to a dial-up ISP was difficult in this case. The Sun SPARCstations were designed for networked environments and not the typical home user. Finding support to make the required changes is time-consuming. If connecting is a problem, double check the serial port connection, script changes, and DTE speed. In terms of speed, an alternative ISP connection will be needed.

Apache and PHP open source software offers inexpensive solutions to driving a web server. Installation on a UNIX system provides some unique options. The module concept allows customization of the server by only using what is needed. The addition of the PHP server-side programming language allows for the creation of a dynamic website.

Usability should be the goal of any website. Increasing the usability can only result in benefits. By using a variety of tests, the site can continue to offer better usability for all, including those with disabilities. Although there are some testing tools available on the internet, they are not 100% effective. Bobby could not test
dynamic content like viewing the picked shopping cart contents. This most likely is due to how the session variables are being handled between the Bobby program window and the user's browser (Bobby says the file can not be found). IBM Home Page Reader 3.01 is a good tool to hear how the site sounds for usability. Since the testing was done from the perspective of a user with no visual impairment, it is difficult to access how successful the changes made actually are. Vischeck needs to enhance its software to accommodate current technologies to test more than just pictures.

The goals of the project were completed. All of the tasks were completed according to the approved Gantt chart (see Appendix A). The research provided many ways to approach the project. Solaris 8 operating system was installed along with Apache Web Server and PHP. The shopping cart provided the opportunity to practice programming skills and design a site that utilized the technology and hardware on a project level. The website also provided the testing ground for various usability requirements and testing tools. Other interested parties may find useful information from the work done on this project.

Additional work can be done in many areas. A web server has many administration, performance, and security tasks surrounding a publicly accessed site. The optional desired features of a database and mail server would enhance the site. Unfamiliarity with the Sun Microsystems software and hardware pushed these optional add-ons outside of the original set of deliverables. The code to send e-mail and auto-respond to customer service was completed, but requires a properly configured mail server to perform the
A database could be a central place for the web pages to gather information on real-time inventory, transaction information, chat rooms, mailing lists, etc. The hard coding of the inventory into arrays certainly causes a lot of extra work for the programmer. Adjustments to inventory have to be made to the arrays and count variables each time an item is added or removed from the inventory. The arrays could be filled from a database and all adjustments can be made to the database. Finally, the site can continually improve on usability and general error checking. Utilizing actual users and users with disabilities would add a lot to testing usability besides software.

The biggest problem on this project was scope creep on research and inexperience with UNIX. Many interesting issues surround delivering a website on any server. History, technologies, enhancements, discoveries, problems, and many other issues can quickly exceed the timeline required to complete a project. Familiarity with Windows 95, 98, or 2000 platform would have greatly decreased the development time for additional features of this project.

The successes of this project were setting up a home based UNIX server, running Apache Webserver, and delivering a usable shopping website. Future expectations are to have this server running full time on the internet for collectors of pressurized lanterns and stoves. The additional future experience will continue past the end of this MSIS project. Hopefully, this project will help others learn about the topics covered and encourage future ideas.
References


http://useit.com/alertbox/990502.html

http://useit.com/alertbox/990530.html


http://useit.com/alertbox/20011111.html


http://info.rutgers.edu/Teachdir/solarisbody.html#partitioning


<table>
<thead>
<tr>
<th>WBS</th>
<th>Task Name</th>
<th>Aug</th>
<th>Sept</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Plan the Project</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td>Develop MSIS project proposal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>Selection of project advisor and two graduates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Develop WBS and Gantt Chart</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>Submit proposal for review and approval</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Key Research</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>Solaris/Unix</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td>Jakob Nielsen's AlertBox</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>ADA compliance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4</td>
<td>Apache Webserver</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.5</td>
<td>Outline findings &amp; specification plan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.6</td>
<td>Notify committee of milestone completion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>3</td>
<td>Installation &amp; Configuration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td>Solaris</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2</td>
<td>Apache Webserver</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3</td>
<td>PHP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td>Database*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5</td>
<td>Mail server*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6</td>
<td>Conduct testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.7</td>
<td>Identify problems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.8</td>
<td>Solution research</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.9</td>
<td>Modify</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.10</td>
<td>Possible modification of project upon progress review</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.11</td>
<td>Notify committee of milestone completion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>4</td>
<td>Design Virtual Shopping Website</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1</td>
<td>Develop site layout</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>Design site/programming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Conduct testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>Modify</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.5</td>
<td>Notify committee of milestone completion</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5</td>
<td>Close Out Project</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>Prepare Paper</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.2</td>
<td>Prepare Presentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.3</td>
<td>Submit Paper</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.4</td>
<td>Present Presentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Preliminary Summary Schedule: Milestone X

NOTE: * denotes an additional task depending on time and development
Appendix B
Solaris Resources at Kempston
Connecting to Any ISP with Solaris 7 and Solaris 8
www.kempston.net/solaris/ www.kempston.org/solaris/

This page contains detailed step-by-step instructions to configure PPP on Solaris 7 and Solaris 8 to connect to any ISP. These instructions also work on Solaris 2.6.

Note for German readers:
Für alle deutschsprachigen Leser ist diese Anleitung freundlicherweise von Andreas Backhaus übersetzt worden. Die deutsche Übersetzung ist sowohl im PDF als auch im Postscript Format verfügbar. (For all German-speaking readers, this guide has been kindly translated by Andreas Backhaus. The German translation is available either in PDF or in Postscript Format)

If you have an account with Demon Internet, Netcom UK, U-Net, Direct Connection or Freeserve, which are all Internet Service Providers in the UK, or with Netgate, an ISP in California, please select the appropriate link below. These provide step-by-step instructions with all the details which are specific to these ISPs.

Configuring PPP to connect to Demon Internet
Configuring PPP to connect to Netcom UK
Configuring PPP to connect to Netgate
Configuring PPP to connect to U-Net
Configuring PPP to connect to Direct Connection
Configuring PPP to connect to Freeserve

Some people who have used these pages successfully have kindly provided specific details for some other ISPs. Have a look to see whether your ISP is listed. If not, please email me at mike@kempston.net with the details listed at the foot of this page and I'll add your ISP to the list.

This page contains rather more explanation than is given in the instructions for the specific ISPs listed above so that you can modify the instructions for your particular ISP. The ISP-specific information which you will need in order to implement these instructions is:

- your dialup username
- your dialup password
- your static IP address (if you have one - most dial-up users don't)
- details of the login or authentication script
- the domain name of your ISP
- the IP addresses of your ISP's name servers (usually two)
- the telephone number your modem dials to connect to your ISP
whether you wish to authenticate using a username and password or by means of PAP or CHAP

Some ISPs require a dial up system to authenticate by responding to conventional Login and Password prompts, some require the use of PAP or CHAP where authentication is part of the initial PPP negotiation, and some allow either method. I recommend using the username and password technique first, if your ISP supports it, because it's easier to see what's happening. When this is working, you can change to PAP or CHAP if you wish and this establishes the connection a little faster.

I've used an imaginary ISP called Zulu Internet in these instructions. If there is an ISP of this name, these instructions don't necessarily apply to it!

Important Notes

The instructions assume that your modem is external and attached to the first serial port but the instructions identify the changes needed to use a modem connected to the second serial port. Note that you must have a real modem; monstrosities known as WinModems are of no use whatsoever with unix. Unfortunately, I have no experience of using internal modems with Solaris and there may be additional configuration issues with these. I'm not able to advise about issues with internal modems but there's some useful advice at Celeste Stokely's Web pages and especially in her Tutorial on Solaris 2.x Modems & Terminals which is highly recommended.

These instructions do work and many people have configured a dial up connection using them. If your system fails to connect to your ISP after following them, the most likely explanation is that you've edited one or more of the files incorrectly. Please note that the format of some of the changes to configuration files is critical; some require a tab character between fields and some require a space. The safest way of making the changes is to save this Web page in a file, load it into a Web browser on Solaris and copy and paste lines from the Web page into the files which are edited.

If you're reading this page on a Windows system, note that Solaris is quite happy to read files saved under Windows. If you have a system which can boot into either Windows or Solaris and your Windows c: drive is the master disc on the primary IDE controller, the following commands mount your c: drive under a directory called /dos in the Solaris file system:

```
# mkdir /dos
# mount -F pcfs /dev/dsk/c0d0p0:c /dos
```

and you can perform this mount automatically at boot time by adding a line to /etc/vfstab which reads:

```
/dev/dsk/c0d0p0:c /dos pcfs -s
```

Note that "c" on the end of the device name means the first DOS partition on the drive. The following command mounts the second DOS partition on the same drive:

```
# mount -F pcfs /dev/dsk/c0d0p0:d /dos
```

Now for the instructions. No unix knowledge is assumed, other than the ability to edit files using a text editor.
A step-by-step Guide to configuring PPP for any ISP

Please note that you must be logged in as root while configuring PPP.

1. Check that the UUCP packages have been installed:

   # pkginfo | grep UUCP

   Check that the following are installed:
   
   system SUNWbnur Networking UUCP Utilities, (Root)
   system SUNWbnnu Networking UUCP Utilities, (Usr)

   If this command just returns with a prompt, the packages aren't installed. Use pkgadd to install them as follows:

   **For Intel (x86) Solaris 8:**
   Insert the CD marked "Solaris 8 Intel Platform Edition Software CD 2 of 2" and type:
   
   # pkgadd -d /cdrom/sol_0_ia_2/Solaris_8/Product SUNWbnnu
   # pkgadd -d /cdrom/sol_0_ia_2/Solaris_8/Product SUNWbnur

   **For SPARC Solaris 8:**
   Insert the CD marked "Solaris 8 SPARC Platform Edition Software CD 2 of 2" and type:
   
   # pkgadd -d /cdrom/sol_0_sparc_2/Solaris_8/Product SUNWbnuu
   # pkgadd -d /cdrom/sol_0_sparc_2/Solaris_8/Product SUNWbnur

   **For Intel (x86) Solaris 7:**
   Insert the Intel Solaris 7 CD and type:
   
   # pkgadd -d /cdrom/sol_7_x86/s2/Solaris_2.7/Product SUNWbnnu
   # pkgadd -d /cdrom/sol_7_x86/s2/Solaris_2.7/Product SUNWbnuu

   **For SPARC Solaris 7:**
   Insert the SPARC Solaris 7 CD and type:
   
   # pkgadd -d /cdrom/sol_7_sparc/s0/Solaris_2.7/Product SUNWbnnu
   # pkgadd -d /cdrom/sol_7_sparc/s0/Solaris_2.7/Product SUNWbnuu

2. Check that the PPP packages have been installed:

   # pkginfo | grep ppp

   Check that (at least) the following are installed:

   **On Solaris 8 and Solaris 7 (11/99):**

   system SUNWapppr PPP/IP Asynchronous PPP daemon
   configuration files
   system SUNWapppu PPP/IP Asynchronous PPP daemon and PPP
   login service
   system SUNWpppk PPP/IP and IPdialup Device Drivers

   **On earlier versions of Solaris 7:**

   system SUNWpppk Solstice PPP Device Drivers
   system SUNWapppu PPP/IP Asynchronous PPP daemon and PPP
   login service
   system SUNWapppr PPP/IP Asynchronous PPP daemon
   configuration files

   If you have 64-bit Solaris 7 or 8 installed, you should also see the package:

   system SUNWpppxx PPP/IP and IPdialup Device Drivers (64-bit)

   If not, use pkgadd to install them in the order shown below.

   **For Intel (x86) Solaris 8:**
Insert the CD marked "Solaris 8 Intel Platform Edition Software CD 2 of 2" and type:

```sh
# pkgadd -d /cdrom/sol_8_i686/Solaris_8/Prodct SUNWpppk
# pkgadd -d /cdrom/sol_8_i686/Solaris_8/Prodct SUNWapppu
# pkgadd -d /cdrom/sol_8_i686/Solaris_8/Prodct SUNWapppr
```
And for 64-bit Solaris 8:

```sh
# pkgadd -d /cdrom/sol_8_i686/Solaris_8/Prodct SUNWpppkx
```

**For SPARC Solaris 8:**

Insert the CD marked "Solaris 8 SPARC Platform Edition Software CD 2 of 2" and type:

```sh
# pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Prodct SUNWpppk
# pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Prodct SUNWapppu
# pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Prodct SUNWapppr
```
And for 64-bit Solaris 8:

```sh
# pkgadd -d /cdrom/sol_8_sparc_2/Solaris_8/Prodct SUNWpppkx
```

**For Intel (x86) Solaris 7:**

Insert the Intel Solaris 7 CD and type:

```sh
# pkgadd -d /cdrom/sol_7_i686/Solaris_2.7/Prodct SUNWpppk
# pkgadd -d /cdrom/sol_7_i686/Solaris_2.7/Prodct SUNWapppu
# pkgadd -d /cdrom/sol_7_i686/Solaris_2.7/Prodct SUNWapppr
```
And for 64-bit Solaris 7:

```sh
# pkgadd -d /cdrom/sol_7_i686/Solaris_2.7/Prodct SUNWpppkx
```

**For SPARC Solaris 7:**

Insert the SPARC Solaris 7 CD and type:

```sh
# pkgadd -d /cdrom/sol_7_sparc/Solaris_2.7/Prodct SUNWpppk
# pkgadd -d /cdrom/sol_7_sparc/Solaris_2.7/Prodct SUNWapppu
# pkgadd -d /cdrom/sol_7_sparc/Solaris_2.7/Prodct SUNWapppr
```
And for 64-bit Solaris 7:

```sh
# pkgadd -d /cdrom/sol_7_sparc/Solaris_2.7/Prodct SUNWpppkx
```

If you happen to have the (commercial) server edition of Solaris 7 or Solaris 8, do not install the optional Solstice PPP product: it stops the bundled aspppd from working.

3. Configure your modem connection:

Start admintool under CDE:

```sh
# admintool&
```
Select the Browse menu and then Serial Ports from the drop-down list. Click on the line starting a if your modem is attached to the first serial port or on the line starting b if your modem is connected to the second serial port. Select the Edit menu and then Modify from the drop-down list.
Change the following fields:

```
Template:    Modem - Dial out Only
Baud Rate:   38400
```
Click on OK and close admintool.

4. Test communication with the modem:

If your modem is attached to the first serial port:

```sh
# tip /dev/cua/a
```
If your modem is attached to the second serial port:

```sh
# tip /dev/cua/b
```
tip should respond:
    connected
which indicates that tip has connected to the serial port. To check that the system
can communicate with the modem, type:
    ATZ
and the modem should respond:
    OK
Close tip by typing:
    ~
If this doesn't work, and especially if you have an internal modem, have a look at
Celeste Stokely's Tutorial on Solaris 2.x Modems & Terminals which contains a
wealth of useful information about modems.

5. Edit /etc/uucp/Devices:

Edit /etc/uucp/Devices to comment out the last two lines:
    ACU cu/a/b - Any hayes
    Direct cu/a/b - Any direct
by prefixing them with a # symbol and add:
    ACU cu/a - Any hayes
If your modem is attached to the second serial port, this line should read:
    ACU cu/b - Any hayes

6. Edit /etc/uucp/Dialers:

Edit /etc/uucp/Dialers to comment out the existing line starting 'hayes' by
prefixing it with a # symbol and add:
    hayes =,-,   "" P_ZERO "" \d "" AT&F1\r OK ATDT\T\t\r\c CONNECT
In this line:
    hayes =,-, identifies the modem type
    is a translation table reference
and the rest of the line is a chat script for the first part of the dialup process. It
consists of a series of prompts expected from the modem followed by the
responses which should be made by your system, with each prompt and
response string separated with a space character:
    "" P_ZERO Wait for nothing and set zero parity
    "" \d Wait for nothing and delay for 2 seconds
    "" AT&F1\r Wait for nothing and send "AT&F1" to initialise
the modem
    OK ATDT\T\t\r\c Wait for an "OK" response and dial the telephone
    number listed in the entry in /etc/uucp/Systems
    CONNECT Wait for a "CONNECT" response from the modem
The initialisation string 'AT&F1' is correct for 3Com and US Robotics modems
and sets the modem to hardware flow control. If your modem needs a different
initialisation string to set hardware flow control, replace 'AT&F1' with the required
string.

7. Edit /etc/uucp/Systems:

The changes needed to /etc/uucp/Systems are different depending on whether
you wish to authenticate by responding to Login and Password prompts or
whether you wish to use PAP or CHAP.
If you're using the conventional Login and Password method, add to the end of
/etc/uucp/Systems a line such as:

isp-ppp Any ACU 115200 telephonenumber ologin: username word:
password "" STTY=crtscroll

The elements in this line are described below:

isp-ppp details for a name specified in the /etc/asppp.cf file which is described later. As an example, if the name "zulu-ppp"

Any Make the connection at any time
ACU The device type
115200 The speed used in communication between the PC and modem

telephonenumber The telephone number used to dial up your ISP.

The rest of this line is a chat script defining the login or authentication process. It consists of a series of prompts expected from the ISP followed by the responses which should be made by your system, with each prompt and response string separated with a space character. The prompt strings in this script should specify the end of the expected response from the ISP. The example chat script listed above is interpreted as follows:

When a connection has been made, wait for a prompt from the ISP ending "oigin:" (the end of a prompt which may read "Zulu Internet login:")) and reply with username. Then wait for another prompt which ends "word:" (the end of a prompt which reads "Password:")) and reply with password. Replace username and password with your own dialup username and password. When the password has been sent, wait for nothing (the two quote characters) and set hardware flow control (STTY=crtscroll). Hardware flow control is essential when using high-speed modems. When your system has received and replied to the last prompt on this line, it will switch to PPP to complete the connection process.

The script "oigin: username word: password" is sufficient for most ISPs but some may prompt with "Username" instead of "login:" and some may require further responses. One possible login sequence is:

Zulu Internet login: username
Password: password
Protocol: ppp
Zulu Internet status: all systems go
HELLO

And the corresponding chat script for this sequence is "oigin: username word: password ocol: ppp HELLO". Waiting for the final "HELLO" stops your system switching to PPP too early.

If you don't know the exact login sequence used by your ISP, either ask them or dial in using an ordinary terminal program and observe the prompts.
If you wish, or need, to authenticate using PAP or CHAP, the changes to this file are simpler. Just add a line to /etc/uucp/Systems reading:

isp-ppp Any ACU 115200 telephonenumber \n STTY=crtstc

The "\n" in this line waits for the end of the CONNECT string from the modem so that the connect speed is recorded in the asppp.log file.

If you want to be able to connect to more than one ISP, you can add additional lines to this file for the other ISPs.

Note: There must be a line feed (LF) character at the end of the line added to this file. If using a graphical editor, such as the one supplied with CDE, ensure that the cursor is positioned at the start of the line following the inserted line before saving the file. If this LF character is omitted, the aspppd log file will report:

Call Failed: SYSTEM NOT IN Systems FILE

8. A word about modem speeds

The 115200 in the /etc/uucp/Systems file sets the speed at which the serial port hardware communicates with the modem to 115,200 bits per second. It's important to understand that there are two speeds to be considered in modem communication. There's the speed at which the host computer talks to a connected modem and this is known as the Data Terminal Equipment or DTE speed. There's also the speed at which the modem transmits and receives data across the telephone line and this is known as the Data Communication Equipment or DCE speed. A 56K or V90 modem is capable of a DCE speed of up to 56K, although this is unlikely to be achieved in practice. Data sent down a telephone connection is often compressed with the receiving modem uncompressing the data before sending it to the host computer. So a modem to modem speed (DCE) of 56K could result in a modem to computer speed (DTE) well in excess of 56K. For this reason, the DTE speed should be set at least 50% higher than the maximum DCE speed and a DTE speed of 115200 is recommended for use with 56K modems.

However, not all Sun hardware is capable of talking to a serial port at this speed. If you have a Sun Ultra 5 or better, or an Intel P90 or better, your system should be able to sustain a DTE speed of 115200. If you have an older Sun system, you may well not be able to drive a V90 modem at its maximum speed and will need to experiment to find the highest DTE speed that your hardware supports. If 115200 doesn't work, try specifying 38400, 19200 or 9600 in the line in /etc/uucp/Systems.

9. Create /etc/resolv.conf.isp:

domain domain-name
nameserver first-name-server-IP-address
nameserver second-name-server-IP-address

Name this file "/etc/resolv.conf" with a suffix of ".isp" where isp is the name of your ISP. This file is activated by means of a dialup script, as shown later, which allows simple configuration of Solaris to connect to more than one ISP if necessary. Replace domain-name with the domain name of your ISP (such as zulu.net), first-name-server-IP-address with the IP address of your ISP's primary
name server and second-name-server-IP-address with the IP address of your ISP's secondary name server.
The "domain" line in the file allows you to specify short names for systems within your ISP's domain: if domain name is "zulu.net" then typing "telnet gate" while connected to Zulu Internet is equivalent to typing "telnet gate.zulu.net. The name server IP addresses identify the name servers which should be used for translating fully qualified domain names into the corresponding IP addresses.

Note: Some editors (including CDE's text editor) create files with permissions of 600 which means that only the owner can read the file. If /etc/resolv.conf.isp has these permissions, only root will be able to use the dial-up connection. To ensure that the dial-up connectivity is accessible to all users, change the permissions on this file:

    # chmod 644 /etc/resolv.conf.isp

10. Edit /etc/asppp.cf:

Make two copies of /etc/asppp.cf:

    # cp /etc/asppp.cf /etc/asppp.cf.original
    # cp /etc/asppp.cf /etc/asppp.cf.isp

where ".isp" is the name of your ISP. You can edit the original version of this file if you wish. But be aware that the aspppd daemon starts automatically at boot time if /etc/asppp.cf exists and dials on demand if this file contains an "ifconfig" line. This may be inconvenient, unexpected, and costly in telephone charges. It's much safer to make these changes in a copy of the file and activate the copy in a script as shown later: this gives you control over the dialup process.

If your ISP allocates dynamic IP addresses (most do), edit /etc/asppp.cf.isp and add to the end of this file:

    ifconfig ipdpt0 plumb 1.1.1.1 2.2.2.2 up
    path
    inactivity_timeout 300 # Terminate the connection after
    5 minutes of inactivity
    interface ipdpt0 # Use the point-to-point
    negotiate_level 9 # Full logging
    negotiate_address on # Obtain a dynamic IP address
    from the ISP
    default_route # Use the dialup link as the
default route for IP packets
    peer_system_name isp-ppp # Must match the name
    used in /etc/uucp/Systems

If you have a static IP address, edit /etc/asppp.cf.isp and add to the end of this file:

    ifconfig ipdpt0 plumb your-IP-address 2.2.2.2 up
    path
    inactivity_timeout 300 # Terminate the connection after
    5 minutes of inactivity
    interface ipdpt0 # Use the point-to-point
    debug_level 9 # Full logging
    default_route # Use the dialup link as the
    default route for IP packets
peer_system_name isp-ppp  # Must match the name used in /etc/uucp/Systems replacing *your-IP-address* with the static IP address allocated to you by your ISP. Note that the name specified on the peer_system_name line must match the name used in /etc/uucp/Systems. A common mistake is to use a hyphen in one file (*isp-ppp*) and an underline character in the other (*isp_ppp*). Make sure that they match!

The ifconfig line configures network interface parameters:

```
    ipdptp0  The point-to-point interface
    plumb    Open the device and set up streams needed for
TCP/IP
    1.1.1.1   A dummy IP address for your system when using a
dynamic IP address
    2.2.2.2   A dummy IP address for the ISP's gateway system
```

The IP addresses 1.1.1.1 and 2.2.2.2 are replaced automatically during the PPP negotiation phase with the dynamic IP address assigned by your ISP to the connection and with the real address of the ISP’s gateway. Adjust the inactivity timeout if desired.

If you're authenticating using PAP (don't do this if you're using the conventional Login and Password method), add three additional lines to /etc/asppp.cf.isp immediately after the 'peer_system_name' line, reading:

```
    will_do_authentication pap
    pap_id username
    pap_password password
```

replacing *username* with your username and *password* with your password.

If you're authenticating using CHAP (don't do this if you're using either the conventional Login and Password method or PAP), add three additional lines to /etc/asppp.cf.isp immediately after the 'peer_system_name' line, reading:

```
    will_do_authentication chap
    chap_name username
    chap_secret password
```

replacing *username* with your username and *password* with your password. Note that some ISPs, such as Worldnet, require the fully qualified domain name in the "chap_name" field, such as 9999999999@worldnet.att.net but most ISPs simply require *username*.

Change permissions on the asppp.cf files:

```
    # chmod 600 /etc/asppp.cf.original /etc/asppp.cf.isp
```

11. Check /etc/nsswitch.dns and /etc/nsswitch.files:

Check whether your system has files named /etc/nsswitch.dns and /etc/nsswitch.files:

```
    # ls /etc/nsswitch.dns /etc/nsswitch.files
    /etc/nsswitch.dns /etc/nsswitch.files
```

These files exist in later versions of Solaris 7 and in Solaris 8, but not in early versions of Solaris 7. If both files exist, skip the rest of this step.

The nsswitch.files file is used while your system isn't connected to your ISP and tells your system to look in /etc/inet/hosts when it needs to look up an IP address. The nsswitch.dns file is used while your system is connected to your ISP and tells your system to look in /etc/inet/hosts first but to use your ISP's DNS name servers if the address isn't listed in the local hosts file.
If one or both files don't exist, create them as follows:

```
/etc/nsswitch.dns:
Create this file by copying /etc/nsswitch.conf:
    # cp /etc/nsswitch.conf /etc/nsswitch.dns
Edit /etc/nsswitch.dns and check that the line starting "hosts:" reads as follows, changing it if necessary:
    hosts:      files dns
Make sure that the file has the correct permissions:
    # chmod 644 /etc/nsswitch.dns
```

```
/etc/nsswitch.files:
Create this file by copying /etc/nsswitch.conf:
    # cp /etc/nsswitch.conf /etc/nsswitch.files
Edit /etc/nsswitch.files and check that the line starting "hosts:" reads as follows, changing it if necessary:
    hosts:      files
Make sure that the file has the correct permissions:
    # chmod 644 /etc/nsswitch.files
```

12. Create the aspppd log file:

```
    # touch /var/adm/log/aspppd.log
```
This file is written to by the aspppd daemon and is invaluable for diagnosing errors in the configuration process.

13. Inhibit sending of RIP packets:

in.routed may send routing information packets (RIP) on the dialup link. This is undesirable and is prevented by creating a file named /etc/gateways containing the single line:
```
    norip ipdptp0
```

14. Stop the system acting as a router:

Solaris may decide that your system is a router, running routed and router discovery. This is undesirable on a dialup connection and is prevented by creating an empty file named /etc/notrouter:
```
    # touch /etc/notrouter
```

15. Disable host caching:

The name service cache daemon (nscd) causes problems with name lookups on intermittently-connected systems. Disable host caching by editing the /etc/nscd.conf file and uncommenting the line:
```
    # enable-cache hosts no
```
so that it reads:
```
    enable-cache hosts no
```
Then stop and restart the nscd daemon:
```
    # /etc/init.d/nscd stop
    # /etc/init.d/nscd start
```

16. Log incoming TCP/IP connections:

Change the last line in /etc/rc2.d/S72inetsvc from:
```
    /usr/sbin/inetd -s &
```
to:
```
    /usr/sbin/inetd -s
```
/usr/sbin/inetd -s -t &
This instructs inetd to log incoming TCP/IP connections to /var/adm/messages. Note that if you use the Open Windows text editor to edit the file, the editor saves the original file in /etc/rc2.d as S72inetsvc%. If this file is left in place, Solaris will execute both the original file and the edited file during boot up and this will result in error messages being reported to the console. After editing the file, delete the saved original file by doing:
  # rm /etc/rc2.d/S72inetsvc%
It's necessary to do this only when the file is edited using the Open Windows text editor. The CDE text editor doesn't save the original file.

17. Create a dialup script named /dialup:
   cp -p /etc/resolv.conf.isp /etc/resolv.conf
   cp -p /etc/asppp.cf.isp /etc/asppp.cf
   cp -p /etc/nsswitch.dns /etc/nsswitch.conf
   /etc/init.d/asppp start
   /usr/sbin/ping 2.2.2.2
The script activates your ISP-specific versions of /etc/resolv.conf, /etc/asppp.cf and /etc/nsswitch.conf and starts the aspppd daemon which dials on demand and routes PPP packets across the dialup link. Finally, it pings the dummy gateway address which causes aspppd to start the dialup process.

18. Create a hangup script named /hangup:
   /etc/init.d/asppp stop
   rm /etc/resolv.conf
   cp -p /etc/asppp.cf.original /etc/asppp.cf
   cp -p /etc/nsswitch.files /etc/nsswitch.conf
   /usr/sbin/route -f
This script stops aspppd which terminates the dialup connection, deletes /etc/resolv.conf and re-instates the original versions of /etc/asppp.cf and /etc/nsswitch.conf. Finally, it flushes the gateway entry from the routing table.

19. Make these scripts executable:
   # chmod 700 /dialup /hangup

20. Reboot:
    # init 6
This isn't essential (Solaris very rarely requires a reboot) but it has the effect of testing the changes you've made to scripts which are run at boot time.

Connecting to Your ISP
After rebooting, your system should be fully configured for dialling into your ISP.

To test the dial-up link:
Open two terminal windows on your screen. In the first window, type:
  # tail -f /var/adm/log/asppp.log
This will display the contents of the PPP log file as it is written to by the asppp daemon and enables the progress of the connection to be monitored.
In the second window, run the dial-up script by typing:
# /dialup

Watch the log window and you'll see the interaction with the modem, the telephone number being dialled and the CONNECT response from the modem when the two modems at each end of the link have negotiated a connection. If you're authenticating using a Login and Password script, the log window will show the login followed by some PPP frames as the PPP negotiation is performed. If you've configured your system to perform PAP or CHAP authentication, the log file will show PPP frames as soon as a connection has been established.

The PPP frames displayed in the log window may not make much sense unless you're familiar with the detail of the PPP protocol. There are some examples of typical PPP negotiations, with an explanation of what it all means, in my page on Analysing the asppp log file on Solaris. Have a look at this if you're interested and especially if the connection fails for any reason.

When PPP negotiation has finished, the log file displays:

```
start_ip: IP up on interface ipdptp0, timeout set for 300 seconds
```

which indicates that the link is fully up and ready to carry traffic. After this point, the log window shows each PPP frame carried on the link. The log file can grow large very quickly and you may wish to reduce the logging level when you're happy that the connection process is working properly.

Note that the 'ping' command in the dial-up script will fail because the pinged address doesn't exist. Don't worry about this; its purpose is simply to prod aspppd into action and start the dial-up process.

After connection is established, test the link using ping and nslookup, or just fire up Netscape for Solaris and have a look at www.sunfreeware.com

Be aware that it's not advisable to connect to the Internet routinely while logged in as root and, in particular, running any Web browser as root may compromise the security of your system. Once you've checked that the PPP configuration is working, I'd recommend that you always start the dialup process while logged in as an ordinary non-privileged user. There's a small problem here, in so far as the /dialup and /hangup scripts must be run with root privileges. The solution is an excellent program called sudo (Super User Do) that enables specified programs and scripts to be run as root from an ordinary user. Please see my Web page on installing sudo for details.

To stop the dial-up link, type:

```
# /hangup
```

To monitor TCP/IP packets while connected:

```
# snoop -d ipdptp0
```

Further configuration
Logging levels

When you're sure that the dialup connection is working properly, you may like to reduce the logging information written to /var/adm/log/asppp.log by changing the line in /etc/asppp.cf.isp which reads:

```
debug_level 9 # Full logging
to:

debug_level 5 # Log all uucp chat script info
```

Level 5 logs the connection process but not PPP messages or raw IP packets.

Netscape

Some recent versions of Netscape for Solaris don't work properly if the name server cache daemon (nscd) is configured not to cache host name lookups as described on this page. Symptoms are that Netscape works fine when given the IP address of a Web server but times out when given the name of a Web server. This problem doesn't occur on all systems but, if it happens on your system, the solution is to undo the changes to /etc/nscl.conf. Edit this file to replace:

```
    enable-cache  hosts  no
```

with:

```
    #    enable-cache  hosts  no
```

Then stop and restart the nscl daemon:

```
    # /etc/init.d/nscl stop
    # /etc/init.d/nscl start
```

A graphical Internet Dialer

Dave Everly has written an excellent Internet Dialer script for Solaris that enables you to select an ISP and connect to the Net with just a few clicks of the mouse. This script assumes that you have already configured your Solaris system to connect to the Net as described here and provides a graphical user interface as an alternative to using the /dialup and /hangup scripts directly. The script can be downloaded from my Web page on A GUI Internet Dialer for Solaris 7 which also contains a screenshot and full details on installing and configuring the script.

Configuring mail

Now that you've successfully configured Solaris to make a dial up connection to your ISP, you may like to follow my instructions for Configuring mail so that you can send and receive email using Solaris.

Security

Consider installing TCP wrappers to give you at least some security from unwelcome visitors when your Solaris system is connected to the Net.

Connecting to multiple ISPs:

One advantage of creating ISP-specific files for use with a dialup script is that it's simple to configure Solaris to dial into one of several ISPs under user control. My /dialup script reads as follows:

```
#!/sbin/sh
```
# PPP Dialup script for Demon, Netcom, U-Net and Freeserve
# Copyright (c) mike@kempston.net 1999

case "$1" in
  'demon')
    /etc/init.d/sendmail stop
cp /etc/resolv.conf.demon /etc/resolv.conf
cp /etc/asppp.cf.demon /etc/asppp.cf
cp /etc/nsswitch.dns /etc/nsswitch.conf
/etc/init.d/asppp start
/usr/sbin/ping 158.152.1.222
;;
  'netcom')
    cp /etc/resolv.conf.netcom /etc/resolv.conf
cp /etc/asppp.cf.netcom /etc/asppp.cf
cp /etc/nsswitch.dns /etc/nsswitch.conf
/etc/init.d/asppp start
/usr/sbin/ping 2.2.2.2
;;
  'unet')
    cp /etc/resolv.conf.unet /etc/resolv.conf
cp /etc/asppp.cf.unet /etc/asppp.cf
cp /etc/nsswitch.dns /etc/nsswitch.conf
/etc/init.d/asppp start
/usr/sbin/ping 2.2.2.2
;;
  'freeserve')
    cp /etc/resolv.conf.freeserve /etc/resolv.conf
cp /etc/asppp.cf.freeserve /etc/asppp.cf
cp /etc/nsswitch.dns /etc/nsswitch.conf
/etc/init.d/asppp start
/usr/sbin/ping 2.2.2.2
;;
*)
  echo "Usage: $0 { demon | netcom | unet | freeserve }"
  exit 1
;;
esac
exit 0

So I can choose which ISP I dial into by typing one of:
  # /dialup demon
  # /dialup netcom
  # /dialup unet
  # /dialup freeserve

Has this guide been useful?
I hope these instructions prove useful. If you have any comments or suggestions for improvement, or have found any technical errors, please email me at mike@kempston.net

This page is copyright © mike@kempston.net 2000. Reproduction is forbidden without the author's permission. You are welcome to link to these pages if you wish but please point to the pages at www.kempston.net: Solaris Resources at Kempston

Would you like to help others to connect to your ISP?

If you successfully connect to your ISP using these instructions, please consider emailing me at mike@kempston.net with the following information:

A copy of the line in your /etc/uucp/Systems file which contains the chat script (without your password!)

A copy of your /etc/resolv.conf file

whether your ISP supports Login and Password authentication, PAP, or both of these

and I'll be happy to add to these pages the details which are specific to your ISP.
Appendix C
Details of installing PHP with Apache on Unix

You can select arguments to add to the configure on line 8 below.

Installation Instructions (Apache Module Version)

1. gunzip apache_1.3.x.tar.gz
2. tar xvf apache_1.3.x.tar
3. gunzip php-x.x.x.tar.gz
4. tar xvf php-x.x.x.tar
5. cd apache_1.3.x
6. ./configure --prefix=/www
7. cd ../php-x.x.x
8. ./configure --with-mysql --with-apache=../apache_1.3.x --enable-track-vars
9. make
10. make install
11. cd ../apache_1.3.x
12. for PHP 3: ./configure --activate-module=src/modules/php3/libphp3.a
   for PHP 4: ./configure --activate-module=src/modules/php4/libphp4.a
13. make
14. make install

Instead of this step you may prefer to simply copy the httpd binary overtop of your existing binary. Make sure you shut down your server first though.

15. cd ../php-x.x.x
   for PHP 4: cp php.ini-dist /usr/local/lib/php.ini

You can edit your .ini file to set PHP options. If you prefer this file in another location, use --with-config-file-path=/path in step 8.

17. Edit your httpd.conf or srm.conf file and add:

   For PHP 3:  AddType application/x-httpd-php3 .php3
   For PHP 4:  AddType application/x-httpd-php .php

   You can choose any extension you wish here. .php is simply the one we suggest. You can even include .html.

18. Use your normal procedure for starting the Apache server. (You must stop and restart the server, not just cause the server to reload by use a HUP or USR1 signal.)

<table>
<thead>
<tr>
<th>Priority 1 checkpoints</th>
<th>Yes</th>
<th>No</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In General (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 Provide a text equivalent for every non-text element (e.g., via &quot;alt&quot;, &quot;longdesc&quot;, or in element content). This includes images, graphical representations of text (including symbols), image map regions, animations (e.g., animated GIFs), applets and programmatic objects, ascii art, frames, scripts, images used as list bullets, spacers, graphical buttons, sounds (played with or without user interaction), stand-alone audio files, audio tracks of video, and video.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1 Ensure that all information conveyed with color is also available without color, for example from context or markup.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1 Clearly identify changes in the natural language of a document's text and any text equivalents (e.g., captions).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.1 Organize documents so they may be read without style sheets. For example, when an HTML document is rendered without associated style sheets, it must still be possible to read the document.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.2 Ensure that equivalents for dynamic content are updated when the dynamic content changes.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.1 Until user agents allow users to control flickering, avoid causing the screen to flicker.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.1 Use the clearest and simplest language appropriate for a site's content.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>And if you use images and image maps (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.2 Provide redundant text links for each active region of a server-side image map.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1 Provide client-side image maps instead of server-side image maps except where the regions cannot be defined with an available geometric shape.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>And if you use tables (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1 For data tables, identify row and column headers.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.2 For data tables that have two or more logical levels of row or column headers, use markup to associate data cells and header cells.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>And if you use frames (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12.1 Title each frame to facilitate frame identification and navigation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>And if you use applets and scripts (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.3 Ensure that pages are usable when scripts, applets, or other programmatic objects are turned off or not supported. If this is not possible, provide equivalent information on an alternative accessible page.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>And if you use multimedia (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3 Until user agents can automatically read aloud the text equivalent of a visual track, provide an auditory description of the important information of the visual track of a multimedia presentation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4 For any time-based multimedia presentation (e.g., a movie or animation), synchronize equivalent alternatives (e.g., captions or auditory descriptions of the visual track) with the presentation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>And if all else fails (Priority 1)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.4 If, after best efforts, you cannot create an accessible page, provide a link to an alternative page that uses W3C technologies, is accessible, has equivalent information (or functionality), and is updated as often as the inaccessible (original) page.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix E
BEFORE TESTING:

View Cart webpage shows all items currently in the shopping cart. The design incorporates Cascading Style Sheets and usability features recommended by Jakob Nielsen. Bobby, IBM Home Page Reader 3.01, and Checklist of Checkpoints Priority 1 revealed that there might be problems with the HTML tables used to present the items in the shopping cart. Bobby was unable to test dynamic content. This error could be from the interface between the user browser window with the mini-view window to navigate the website being tested or where the session variables are actually stored during testing (i.e. user, Bobby, or Bobby does not allow them?).
AFTER TESTING:

View Cart with changes made to the HTML tables of the shopping cart. The appearance is much more appealing. The code was based on recommendations by the W3C Web Content Accessibility Guidelines 1.0. It is difficult to access whether or not this page is 100% usable because the screen reader test was done through the perspective of a user with no visual impairment and accustomed to seeing and working with the pages. Usability tools provide an excellent option to improve usability. To be properly approved, websites must be Priority 3 compliant. The goal was Priority 1. Another important note, the "reset" button was removed because one user thought the button would make the changes. Instead, the selected changes were erased and changes had to be re-entered.
DESCRIPTION WINDOW:

The Products page gives description links for each item for sale. When 
clicked, a new browser pop-up window appears on the monitor screen. This 
page produced a possible browser compatibility error under Bobby because of 
the JavaScript button, which closes the browser window. After testing with a 
number of browsers (even Netscape 3.01), there were no problems with the 
"Close Description Window" button. Vischeck tested the lantern picture for some 
colorblind types. No problems were found. Vischeck is currently working to 
support testing full webpages for design elements utilizing color through 
technologies such as Cascading Style Sheets.
Appendix F
<?

/**************
* Name: Norman Harrod
* Course: INFS 790:
***************/

/* vap.php */

<title>Lantern Stove Connection</title>
<link rel="stylesheet" type="text/css" href="sitestyle.css">
</head>
<body>
<table class="ad">
<tr><td><p><h1>VAPALUX</h1><form><input type="button" name="quit" value="Close Description Window" onClick="self.close();"></form></p><img src="vap.jpg" alt="VAPALUX"></td></tr>
<tr><p><b>Description:</b>
Lantern is in new condition. Construction is brass, steel, stove
enameled hood, and cast aluminum. Comes with
two-piece white shade, funnel, spirit can, spare parts, and instruction
tag. Burns kerosene only.</p></tr>
<tr><p><b>Manufacturer:</b> Bairstow
Brothers</p></tr>
<tr><p><b>Country:</b> United Kingdom</p></tr>
<tr><p><b>Model:</b> M320</p></tr>
<tr><p><b>Year:</b> 2001</p></tr>
<tr><p><b>Output:</b> 300 C.P.</p></tr>
<tr><p><b>Dimensions:</b> NA</p></tr>
</table>
</body>
</html>
<? /* *
Name: Norman Harrod
Course: INES 790: 
*/ viewcart.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!--[-- Links html page to cascading style sheet --]>
<link rel="stylesheet" type="text/css" href="sitestyle.css">
</head>
<body>
<? include("functions.php") ?>

<table width="100%">
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<tr>
<td class="navigation">
<? loadNavigation("viewcart") ?>
</td>
</tr>
</table>
<h2>View Cart</h2>
<?
# set check variable
if($SETCHECK == "load"){
# initialize session variables
# array0 watches the check box TRUE or FALSE and add or remove if checked
$ARRAY0 = array("$Prod0", "$Prod1", "$Prod2", "$Prod3", "$Prod4");
# array1 holds the name of the item
$ARRAY1 = array("Austramax", "Coleman", "Petromax", "Tilley", "Vapalux");
# array2 holds the quantity
$ARRAY2 = array("$qty0", "$qty1", "$qty2", "$qty3", "$qty4");
# array 3 holds the price per item
$ARRAY3 = array(67.89, 45.23, 64.78, 79.58, 58.97);
# array 4 holds the description page

# check to see if any of the items have been selected
if($Prod0 == FALSE AND $Prod1 == FALSE AND $Prod2 == FALSE AND $Prod3 == FALSE AND $Prod4 == FALSE){
# nothing selected so cart is empty
$SETCHECK = "empty";
} else{
    # shopping cart has items to process
    $SETCHECK = "filled";
}

if($SETCHECK == "filled"){
    # form elements based on selected items that are checked
    into a form for quantity edit
    print("<form method="post" action="viewcartedit.php" onsubmit="return onSubmit()">")
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == TRUE){
            print("<tr><td><b>Remove</b><br><input type="checkbox" name="Frod" value="remove">$ARRAY1[$i]"/></td><td>"
            print("<b>Qty</b> <div class="quantity">"
            print("$ARRAY2[$i]"./td><td>");
            print("<input type="text" name="changeqty" value="">");
            print("<b>Price</b> <div class="price">"
            print("$ARRAY3[$i] Each"./td><td>"));
            print("<tr>"./td><td>"));
            print("$PRICE \$ $ARRAY3[$i] Each"./td><td>"));
            print("<tr>"./td><td>"));
            # take quantity times price to give total per
            # item
            $pricetotal = $ARRAY2[$i] * $ARRAY3[$i];
            printf("\$%.2f", $pricetotal);
            print("<tr>"./td><td>"));
        }
    }
    # calculate subtotal of the order
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == TRUE){
            $subtotal = $subtotal + ($ARRAY2[$i] *
            $ARRAY3[$i]);
    }
}

print("<tr><td>Total</td><td><b>Total</b><br><input type="submit" value="Make Shopping Cart Changes"></form>"./td><td><b>Total</b><br><input type="submit" value="Make Shopping Cart Changes"></form>"./td><td>
if($SETCHECK != "filled"){
    print("Shopping Cart is Empty");
}
}</td>
<tr>
<td class="footer">
</td>
</tr>
</table>
</body>
</html>
<? /* ********************************************
* Name: Norman Harrod
* Course: INFS 790:
*********************************************/
/* viewcartada.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css"/>
<body>
<? include("functionsada.php") ?>
<table width="100%">
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<tr class="navigation">
<? loadNavigation("viewcart") ?>
</td>
</tr>
<tr class="information">
<h2>View Cart</h2>
<br>
# set check variable
if($SETCHECK == "load"){
    # initialize session variables
    # array0 watches the check box TRUE or FALSE and add
    or remove if checked
    $ARRAY0 = array("$Prod0", "$Prod1", "$Prod2",
    "$Prod3", "$Prod4");
    # array1 holds the name of the item
    $ARRAY1 = array("Austramax", "Coleman", "Petromax",
    "Tilley", "Vapalux");
    # array2 holds the quantity
    $ARRAY2 = array("$qty0", "$qty1", "$qty2", "$qty3",
    "$qty4");
    # array 3 holds the price per item
    $ARRAY3 = array(67.89, 45.23, 64.78, 79.58, 58.97);
    # array 4 holds the description page
    $ARRAY4 = array("aus.php", "col.php", "pet.php",
    "til.php", "vap.php");
    # check to see if any of the items have been
    selected
    if($Prod0 == FALSE AND $Prod1 == FALSE AND $Prod2 ==
    FALSE AND $Prod3 == FALSE AND $Prod4 == FALSE){
        # nothing selected so cart is empty
        $SETCHECK = "empty";
    }
```php
else{
    # shopping cart has items to process
    $SETCHECK = "filled";
}

if($SETCHECK == "filled"){
    # form elements based on selected items that are checked
    action="viewcarteditada.php"
    <table summary="This table lists the items in the shopping cart which can be removed by checking a check box or change the quantity ordered in a text box. All changes must be submitted by clicking the button make shopping cart changes."
    width="100%">
    <caption>Items currently in shopping cart</caption>
    <tr>
        <th scope="col">Check to Remove</th>
        <th scope="col">Product</th>
        <th scope="col">Quantity Ordered</th>
        <th scope="col">Edit Quantity</th>
        <th scope="col">Price Each</th>
        <th scope="col">Total</th>
    </tr>
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == TRUE){
            name="Prod";
            print("<tr><td><center><input type="checkbox" name="$i">
            print("$i"ollider value="remove"></center></td><td><center>$ARRAY1[$i]</center></td><td>");

            print("<center>$ARRAY2[$i]</center></td><td><center></center></td><td><center">
        print("<input type="text" size="5"
            name="changeqty"></center></td><td><center"></center>");

        print("$ARRAY3[$i]");
        # take quantity times price to give total per item
        $pricetotal = $ARRAY2[$i] * $ARRAY3[$i];
        printf("\$%.2f", $pricetotal);
        print("</center></td><tr>");
    }

    # calculate subtotal of the order
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == TRUE){
            $subtotal = $subtotal + ($ARRAY2[$i] * $ARRAY3[$i]);
    }

    print("</table><p align="right"><b>SubTotal</b><p>
        print("<br>
    </b>$$subtotal</b></p>");
    print("<input type="submit" value="Make Shopping Cart Changes"></form">;
}
```
if($SETCHECK != "filled"){
    print("Shopping Cart is Empty");
}
</td>
<tr>
    <td class="footer">
        <? include("footer.php") ?>
    </td>
</tr>
</table>
</body>
</html>
<?*
//****************************************************************************
* Name: Norman Harrod
* Course: INFS 790:
****************************************************************************/
/* viewcartedit.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<? include("functions.php") ?>
<table width="100%"
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<td class="navigation">
<? loadNavigation(""") ?>
</td>
</tr>
<td class="information">
<?
    # check to see if the quantity needs to be changed
    if($changeqty0 != ""){
        $ARRAY2[0] = $changeqty0;
    }
    if($changeqty1 != ""){
        $ARRAY2[1] = $changeqty1;
    }
    if($changeqty2 != ""){
        $ARRAY2[2] = $changeqty2;
    }
    if($changeqty3 != ""){
        $ARRAY2[3] = $changeqty3;
    }
    if($changeqty4 != ""){
        $ARRAY2[4] = $changeqty4;
    }

    # check to see if the item needs to be removed and then set to false
    if($Prod0 == "remove"){
        $ARRAY0[0] = FALSE;
    }
    if($Prod1 == "remove"){
        $ARRAY0[1] = FALSE;
    }
    if($Prod2 == "remove"){


71
$ARRAY0[2] = FALSE;
}
if($Prod3 == "remove"){
 $ARRAY0[3] = FALSE;
}
if($Prod4 == "remove"){
 $ARRAY0[4] = FALSE;
}

# check to see if the items needs to be added and then set to true
if($Prod0 == "add"){
 $ARRAY0[0] = TRUE;
}
if($Prod1 == "add"){
 $ARRAY0[1] = TRUE;
}
if($Prod2 == "add"){
 $ARRAY0[2] = TRUE;
}
if($Prod3 == "add"){
 $ARRAY0[3] = TRUE;
}
if($Prod4 == "add"){
 $ARRAY0[4] = TRUE;
}

# check to see if the cart has been totally emptied
 $SETCHECK = "empty";
} else{
 $SETCHECK = "filled";
}

if($SETCHECK == "filled"){
 # form elements based on selected items that are checked
 into a form for quantity edit
 print("Changes have been made to the shopping cart<br>");
}
if($SETCHECK != "filled"){
 print("Shopping Cart is Empty");
}

</td>
<tr>
<td class="footer">
<i>?$include("footer.php") ?</i>
</td>
</tr>
</table>
</body>
</html>
<? /*
  *****************************************
  * Name: Norman Harrod
  * Course: INFS 790:
  *****************************************/
/* aus.php */

<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<body>
<table class="ad">
<tr><td><p><h1>AUSTRAMAX</h1><form><input type="button" name="quit" value="Close Description Window" onClick="self.close();"></form></p><img src="aus.jpg" alt="AUSTRAMAX"></td></tr>
</table>
<p>Description:</p>
Lantern is in new condition. Construction is stainless steel with chrome plating, steel, brass, and aluminum. Comes with spirit bottle for preheating and cardboard box which doubles as a carrying case. Uses kerosene only.</p>
<tr><td><p><b>Manufacturer:</b> Austramax</p></td></tr>
<tr><td><p><b>Country:</b> Australia</p></td></tr>
<tr><td><p><b>Model:</b> 3/300</p></td></tr>
<tr><td><p><b>Year:</b> 2001</p></td></tr>
<tr><td><p><b>Output:</b> 300 C.P.</p></td></tr>
<tr><td><p><b>Dimensions:</b> NA</p></td></tr>
</table>
</body>
</html>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<body>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<? include("functionsada.php") ?>
<table width="100%"
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
</table>
<tr>
<td class="navigation">
<? loadNavigation(""") ?>
</td>
</tr>
<tr>
<td class="information">
<?
# check to see if the quantity needs to be changed
if($changeqty0 != ""){
    $ARRAY0[0] = $changeqty0;
}
if($changeqty1 != ""){
    $ARRAY0[1] = $changeqty1;
}
if($changeqty2 != ""){
    $ARRAY0[2] = $changeqty2;
}
if($changeqty3 != ""){
    $ARRAY0[3] = $changeqty3;
}
if($changeqty4 != ""){
    $ARRAY0[4] = $changeqty4;
}
# check to see if the item needs to be removed and then set to false
if($Prod0 == "remove"){
    $ARRAY0[0] = FALSE;
}
if($Prod1 == "remove"){
    $ARRAY0[1] = FALSE;
}
if($Prod2 == "remove"){


74
```php
$array0[2] = FALSE;
if($Prod3 == "remove"){
    $array0[3] = FALSE;
}
if($Prod4 == "remove"){
    $array0[4] = FALSE;
}

# check to see if the items needs to be added and then set to true
if($Prod0 == "add"){
    $array0[0] = TRUE;
}
if($Prod1 == "add"){
    $array0[1] = TRUE;
}
if($Prod2 == "add"){
    $array0[2] = TRUE;
}
if($Prod3 == "add"){
    $array0[3] = TRUE;
}
if($Prod4 == "add"){
    $array0[4] = TRUE;
}

# check to see if the cart has been totally emptied
    $setcheck = "empty";
} else{
    $setcheck = "filled";
}

if($setcheck == "filled"){
    # form elements based on selected items that are checked into a form for quantity edit
    print("Changes have been made to the shopping cart<br>");
} else{
    print("Shopping Cart is Empty");
}
</td>
<tr>
    <td class="footer">
        <? include("footer.php") ?>
    </td>
</tr>
</table>
</body>
</html>
```
<?

/*********************************************
* Name: Norman Harrod
* Course: INFS 790:
*********************************************/

// checkout.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>

<html>
<head>
<title>Surplus Lanterns</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
</head>
<body>
<? include("functions.php") ?>
<table width="100%">
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a></td>
</tr>
<tr>
<td class="navigation">
<? loadNavigation("checkout") ?>
</td>
</tr>
<tr>
<td class="information">
<h2>Check Out</h2>
<? if($SETCHECK == "filled"){
    # form the product elements based on selected items
    print("<table width="100\%">\n")
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == TRUE){
            print("<tr><td>$ARRAY1[$i]</td><td></td></tr>");
            print("<b>Quantity </b>\n")
            print("$ARRAY2[$i]<td><td></td></tr>");
            print("<b>Price Per Item </b>\n")
            print("$ARRAY3[$i]<td><td></td></tr>");
            print("<b>Price </b>\n")
            print("$ARRAY4[$i]<td></td></tr>");
            # take quantity times price to give total
            $pricetotal = $ARRAY2[$i] * $ARRAY3[$i];
            printf("\n$f\%.2f", $pricetotal);
            print("</td><tr></td>");
        }
    }
}
print("</table>");
print("Flat Sales Tax of 6%<br>");
# calculate total of the order with sales tax
for($i = 0; $i <= 4; ++ $i){
    if($ARRAY1[$i] == TRUE){
        $subtotal = $subtotal +
	}
($ARRAY2[$i] * $ARRAY3[$i]);
}

$taxamount = $subtotal * .06;
$total = $subtotal + $taxamount;
print("<b>Total LESS shipping and handling:  \$</b>". $total);
print("<br>");
print("<p>Due to the variety of shipping methods and regulations, we are unable to give on the spot grand totals. ").
print("Please fill out the form and your shopping cart will automatically be sent to our order department. ").
print("Our customer service will respond with the grand total, services, and rates that are available.";
);

Please fill out the following information to expedite your order inquiry:
<form method="post" action="order.php">
<table width="100%">
<tr>
<td>First Name:</td><td><input type="text" name="fname" size="45" maxlength="25"></td>
</tr>
<tr>
<td>Last Name:</td><td><input type="text" name="lname" size="45" maxlength="25"></td>
</tr>
<tr>
<td>Country:</td><td><input type="text" name="country" size="45" maxlength="25"></td>
</tr>
<tr>
<td>Valid E-mail:</td><td><input type="text" name="useremail" size="45" maxlength="40"></td>
</tr>
<tr>
<td>Comments:</td><td><textarea name="comments" cols="45" rows="4"></textarea></td>
</tr>
</table>
<br>
<input type="submit" value="Submit Order">
</form>

?} if($SETCHECK ! = "filled"{ 
print("Shopping Cart is Empty<br>");
}

<br>
<tr>
<td class="footer">

? include("footer.php") />
</tr>
</table>
</body>
</html>
<?
/*******************
* Name: Norman Harrod
* Course: INES 790:
***************************/
/* col.php */
?>
<html>
<head>
<title>Lantern Stove Connection</title>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
</head>
<body>
<table class="ad">
<tr><td><p><h1>COLEMAN</h1></p><form><input type="button" name="quit" value="Close Description Window" onClick="self.close();"></form></td><td>
<p><img src="col.jpg" alt="COLEMAN"></p>
</td></tr>
<tr><td><b>Description:</b>
Lantern is in new condition. Construction is steel and features spare parts cylinder in fuel found - specifically designed for the US Military. Comes with certificate of authenticity, spare parts, and chained on funnel.
</td></tr>
<tr><td><b>Manufacturer:</b> Coleman</td><td><b>Country:</b> United States</td></tr>
<tr><td><b>Model:</b> 252A</td><td><b>Year:</b> 1952</td></tr>
<tr><td><b>Output:</b> 350 C.P.</td><td><b>Dimensions:</b> NA</td></tr>
</table>
</body>
</html>
<?
/*****************************************************************************/
* Name: Norman Harrod
* Course: INFS 790:
*****************************************************************************/
/* emptycart.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<? include("functions.php") ?>
<table width="100%"
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<tr>
<td class="navigation">
<? loadNavigation("emptycart") ?>
</td>
</tr>
<tr>
<td class="information">
<h2>Empty Cart</h2>
<?
$SETCHECK = "empty";
print("Shopping Cart Has Been Emptied");
?>
</td>
</tr>
<tr>
<td class="footer">
<? include("footer.php") ?>
</td>
</tr>
</table>
</body>
</html>
<?
/*******************************
* Name: Norman Harrod
* Course: INFS 790:
*******************************/
/* emptycartada.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<? include("functionsada.php") ?>
<table width="100%"
<tr>
   <td class="header">
      <a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
   </td>
</tr>
<tr>
   <td class="navigation">
      <? loadNavigation("emptycart") ?>
   </td>
</tr>
<tr>
   <td class="information">
      <h2>Empty Cart</h2>
      <?
      $SETCHECK = "empty";
      print("Shopping Cart Has Been emptied");
      ?>
   </td>
</tr>
<tr>
   <td class="footer">
      <? include("footer.php") ?>
   </td>
</tr>
</table>
</body>
</html>
<?*
/****************************
* Name: Norman Harrood
* Course: INFS 790:
/*****************************/
/* faq.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
</html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
</body>
<? include("functions.php") ?>
<table width="100%"
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo" /></a>
</td>
</tr>
<tr>
<td class="navigation">
<? loadNavigation("") ?>
</td>
</tr>
<tr>
<td class="information">
<h2>Frequently Asked Questions</h2>
<a name="questions"></a>
<ul>
<li><a href="#order">How do I place an order?</a></li>
<li><a href="#delivery">How will my order be delivered?</a></li>
<li><a href="#status">How can I check the status of my order?</a></li>
<li><a href="#contact">How do I contact a customer service representative?</a></li>
<li><a href="#returns">How do I return a product?</a></li>
</ul>
<a name="order">1. How do I place an order?</a>
<p>From any page, click on the "Products" link to view available products.<br>
Place a checkmark in the box to the left of the products you would like to order and indicate</p>
the quantity you would like to purchase. Then choose
the "Add to Shopping Cart" button.
If you have further questions, please email us at x@x.x
or call (xxx) xxx-xxxx.</p>

<a href="#questions">Go back to the questions</a>
<hr/>
<a name="delivery">2. How will my order be
delivered?</a>
<br/>
<p>Orders are shipped after payment is approved. Deliveries are made via Federal Express, United Parcel
Service, or United States Postal Service. There are restrictions placed on service according to the carrier. Insurance will be an additional fee.</p>

<a href="#questions">Go back to the questions</a>
<hr/>
<a name="status">3. How can I check the status of my order?</a>
<br/>
<p>When we receive your order and after we have obtained your payment information, your order is assigned a tracking number.</p>

<a href="#questions">Go back to the questions</a>
<hr/>
<a name="contact">4. How do I contact Lantern Stove Connection?</a>
<br/>
<p>Call (xxx) xxx-xxxx or Fax (xxx) xxx-xxxx or Email x@x.x or Mail: Lantern Stove Connection</p>

<a href="#questions">Go back to the questions</a>
<hr/>
<a name="returns">5. How do I return a product?</a>
<br/>
<p>Send returns within 30 days to: Lantern Stove Connection</p>

<a href="#questions">Go back to the questions</a>
</td>
</tr>
<tr>
<td class="footer">
<? include("footer.php") ?>
</td>
</tr>
</table>
</body>
</html>
<?
/*
 ********************************************
 * Name: Norman Harrod
 * Course: INFS 790:
 *************************************************/
/* footer.php */

# standard repeated footer code
print("<hr/>
<a href="legal.php">Copyright</a>\&\#032;\&\#169;\&\#032;2001\&\#032;LanternStove
Connection.com<br>
<a href="legal.php">Legal</a>\&\#032;\&\#124;\&\#032;
<a href="legal.php">Online Privacy</a>\&\#032;\&\#124;\&\#032;
<a href="legal.php">Site Terms of Use</a>\&\#032;\&\#124;\&\#032;
<a href="legal.php">Warranty & Return Policy</a>"
);
?>
<? /* Name: Norman Harrod 
Course: INFS 790: */ 

// loads the navigation bar according to the page
function loadNavigation($page){
    switch($page)
    {
        case "index":
            print("<center>
            <b>
            <a href="prod.php">Products</a>
            <a href="viewcart.php">View Cart</a>
            <a href="emptycart.php">Empty Cart</a>
            <a href="checkout.php">Check Out</a>
            <a href="service.php">Customer Service</a>
            </b>
            <a href="links.php">Links</a>
            </center>
            ");
            break;
        case "prod":
            print("<center>
            <b>
            <a href="index.php">Home</a>
            <a href="viewcart.php">View Cart</a>
            <a href="emptycart.php">Empty Cart</a>
            <a href="checkout.php">Check Out</a>
            <a href="service.php">Customer Service</a>
            </b>
            <a href="links.php">Links</a>
            </center>
            ");
            break;
        case "viewcart":
            print("<center>
            <b>
            <a href="index.php">Home</a>
            <a href="prod.php">Products</a>
            <a href="emptycart.php">Empty Cart</a>
            <a href="checkout.php">Check Out</a>
            <a href="service.php">Customer Service</a>
            </b>
            <a href="links.php">Links</a>
            </center>
            ");
            break;
        case "emptycart":
            print("<center>
            <b>
            <a href="index.php">Home</a>
            <a href="prod.php">Products</a>
            <a href="viewcart.php">View Cart</a>
            <a href="checkout.php">Check Out</a>
            </b>
            <a href="links.php">Links</a>
            </center>
<a href="service.php">Customer</a>
</b>
</center>
"
);
break;
break;
case "checkout":
print("<center>
<b>
<a href="index.php">Home</a>
<a href="prod.php">Products</a>
<a href="viewcart.php">View Cart</a>
<a href="emptycart.php">Empty Cart</a>
<a href="checkout.php">Check Out</a>
<a href="links.php">Links</a>
</b></center">
</b>
</center>
"
);
break;
case "service":
print("<center>
<b>
<a href="index.php">Home</a>
<a href="prod.php">Products</a>
<a href="viewcart.php">View Cart</a>
<a href="emptycart.php">Empty Cart</a>
<a href="checkout.php">Check Out</a>
<a href="service.php">Customer</a>
</b></center>
"
);
break;
default:
print("<center>
<b>
<a href="index.php">Home</a>
<a href="prod.php">Products</a>
<a href="viewcart.php">View Cart</a>
<a href="emptycart.php">Empty Cart</a>
<a href="checkout.php">Check Out</a>
<a href="service.php">Customer</a>
</b></center>
"
);


```html

</center>

});

}

# function for autoreply message
function replyUser($useremail){

  # recipient
  $recipient .= "$useremail";

  # subject
  $subject = "Lantern Stove Connection AutoResponder";

  # message
  $message = <<<MESSAGE_BODY
We have received your e-mail and will handle your request as promptly as possible.
You should receive a personal reply within 40 hrs from one of our friendly service representatives.

Sincerely, 
Lantern Stove Connection
MESSAGE_BODY;

  # header piece(s)
  $headers .= "From: nharrod@prodigy.net";
  $headers .= "Reply-To: nharrod@prodigy.net";

  # mail it
  $result = mail($recipient, $subject, $message, $headers);

  # successful send ?
  if($result == TRUE){
    print("Expect a response from our customer service shortly");
  } else{
    print("Error, contact the webmaster");
  }
}

# function loads the arrays, calls printForm, and returns the setcheck to load
function printForm(){
  # initialize session variable
  # array1 holds the name of the item
  $ARRAY1 = array("Austramax", "Coleman", "Petromax", "Tilley", "Vapalux");
  # array 3 holds the price per item
  $ARRAY3 = array(67.89, 45.23, 64.78, 79.58, 58.97);
  # array 4 holds the description page

  print("<form method="post" action="viewcart.php">".
    "<table width="100%">")
    for($j = 0; $j <= 4; ++ $j){
      print("<tr><td><b>Add</b><input type="checkbox" name="Prod$j" value="add"/>\<i>$ARRAY1[$j]\</i></td>
        <td><a href="$ARRAY4[$j]" target="_blank">Description</a></td><td>$ARRAY3[$j] Each</td>
        <td><input type="text" size="5" name="qty$j" value=""/></td></tr>");
  }

```

86
print("</table><br><input type="submit" value="Add Checked Items to Shopping Cart"/></form>");

$SETCHECK = "load";
return($SETCHECK);

?>
<?
/* Name: Norman Harrold
 * Course: INFS 790:
 */

// loads the navigation bar according to the page
function loadNavigation($page){
    switch($page)
    {
        case "index":
            print("<center>
                <b>
                    <a href="prodada.php">Products</a>
                    <a href="viewcartada.php">View Cart</a>
                    <a href="emptycartada.php">Empty Cart</a>
                    <a href="checkout.php">Check Out</a>
                    <a href="service.php">Customer Service</a>
                </b>
            </center>
            ");
            break;
        case "prod":
            print("<center>
                <b>
                    <a href="index.php">Home</a>
                    <a href="viewcartada.php">View Cart</a>
                    <a href="emptycartada.php">Empty Cart</a>
                    <a href="checkout.php">Check Out</a>
                    <a href="service.php">Customer Service</a>
                </b>
            </center>
            ");
            break;
        case "viewcart":
            print("<center>
                <b>
                    <a href="index.php">Home</a>
                    <a href="prodada.php">Products</a>
                    <a href="emptycartada.php">Empty Cart</a>
                    <a href="checkout.php">Check Out</a>
                    <a href="service.php">Customer Service</a>
                </b>
            </center>
            ");
            break;
        case "emptycart":
            print("<center>
                <b>
                    <a href="index.php">Home</a>
                    <a href="prodada.php">Products</a>
                </b>
            </center>
            ");
            break;
    }
}
?>
function replyUser($useremail) {
    $recipient = "$useremail";
    $subject = "Lantern Stove Connection AutoResponder";
    $message = "We have received your e-mail and will handle your request as promptly as possible. You should receive a personal reply within 48 hrs from one of our friendly service representatives.

Sincerely,
Lantern Stove Connection

MESSAGE BODY;

    $headers = "From: ntharrod@prodigy.net";
    $headers .= "Reply-To: ntharrod@prodigy.net";
    $result = mail($recipient, $subject, $message, $headers);
    if($result == TRUE) {
        echo "Expect a response from our customer service shortly";
    } else {
        echo "Error, contact the webmaster";
    }
}

function printForm() {
    $array1 = array("Austramax", "Coleman", "Petromax", "Tilley", "Vapalux");
    $array2 = array(67.89, 45.23, 64.78, 79.58, 58.97);

    print("<form method="post" action="/viewcartada.php" width="100%">
        <table summary="This table lists the items currently

available for purchase and can be added by check to add. The item must be checked to add to the cart and a quantity should be entered into the text box."

```php
for($j = 0; $j <= 4; ++ $j){
    print("<tr><td><center><input type="checkbox" name="Prod$j" value="add"></center></td>
    <td><center><i>$ARRAY1[$j]</i></center></td>
    <td><center>a href="$ARRAY4[$j]" target="_blank">Click HERE for Description</a></center></td>
    <td><center>$ARRAY2[$j]</center></td>
    <td><center><input type="text" size="5" name="qty$j" value=""></center></td>
</tr>");
}
print("</table><br><input type="submit" value="Add Checked Items to Shopping Cart" /></form>");

$SETCHECK = "load";
return($SETCHECK);

?>
```
<?
/*
** Name: Norman Harrod
** Course: INFS 790:
*****************************************************************************/
/* index.php */

# start a session
session_start();
?>
<html>
<head>
  <title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
  <link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<? include("functions.php") ?>
<table width="100%">
<tr>
  <td class="header">
    <img src="logo.jpg" alt="Lantern Stove Connection Logo">
  </td>
</tr>
<tr>
  <td class="navigation">
    <? loadNavigation("index") ?>
  </td>
</tr>
<tr>
  <td class="information">
    <p>Welcome! <i>Lantern Stove Connection</i> is your one-stop online source for quality pressurized lamps & cookers, from everyday users to hard-to-find specialties and collectibles. We carry quality and tested lamps backed by great service. These are all original, hand-picked, tough-as-nails, and sometimes rare to come by items.</p>
    We want you to be completely satisfied with your purchase from <i>Lantern Stove Connection</i>. Limited stock ranges across the globe, such as Germany, United States, Australia, Sweden, Canada, Singapore, and many other locations. Our warranties vary depending on the condition, availability, and age of the light or burner.</p>
    Our items are popular with interior designers, hunters, campers, collectors, construction sites, and survivalists to name a few. Please check our stock frequently for changing inventory.</p>
  </td>
</tr>
<tr>
  <td class="footer">
    <? include("footer.php") ?>
  </td>
</tr>
</table>
<?
# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
</body>
</html>
<?
//***************************
* Name: Norman Harrod
* Course: INPS 790:
***************************
/* legal.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
</head>
<? include("functions.php") ?>
<table width="1008"
<tr>
  <td class="header">
    <a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo" /></a>
  </td>
</tr>
<tr>
  <td class="navigation">
    <? loadNavigation(""") ?>
  </td>
</tr>
<tr>
  <td class="information">
    <h2>Legal</h2>
    <!-- Following information came from www.dell.com DELL Computers support webpages and altered for the purposes of this project -->
    <p><b>Privacy &amp;#038; Data Security:</b> Your right to privacy and data security is a primary concern. That's why, when you visit Lantern Stove Connection, we help you maintain control over your personal data on the Internet. In a few areas on our Web site and online customer support forms, we ask you to provide information that will enable us to enhance your site visit, to assist you with support issues or to follow up with you after your visit. It is completely optional for you to participate. For example, we request information from you when you:
    <ul>
      <li>Place an order</li>
      <li>Provide feedback in an online form</li>
    </ul>
</p>
</td>
</tr>
</table>
</body>
</html>
<li>Request e-mail notification of your order</li>
<li>Request online technical support</li>
<li>Request assistance from our customer support</li>
</ul>

In each of the instances above, we may ask for your name, e-mail address, phone number, address, type of business, customer preference information, customer number, as well as other similar personal information that is needed. The information you provide will be kept confidential and used to support your customer relationship.

Lantern Stove Connection will not disclose your personal information to any outside organization for its use in marketing without your consent. Information regarding you (such as name, address and phone number) or your order and the products you purchase will not be given or sold to any outside organization for its use in marketing or solicitation without your consent. Your information may be shared with agents for the purpose of performing services for Lantern Stove Connection. Lantern Stove Connection wants to help you keep your personal information accurate. You can request the individual information that Lantern Stove Connection has collected about you via the Internet at x@x.com. You can have factual inaccuracies in this information corrected by contacting x@x.com

Copyright: The copyright in all material provided on this site ("Site") is held by the original creator of the material. Except as stated herein, none of the material may be copied, reproduced, distributed, republished, downloaded, displayed, posted or transmitted in any form or by any means, including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Lantern Stove Connection or the copyright owner. Permission is granted to display, copy, distribute and download the materials on this Site for personal, non-commercial use only, provided you do not modify the materials and that you retain all copyright and other proprietary notices contained in the materials. You also may not, without permission, "mirror" any material contained on this Site on any other server. This permission terminates automatically if you breach any of these terms or conditions. Upon termination, you must immediately destroy any downloaded and printed materials. Any unauthorized use of any material contained on this Site may violate copyright laws, trademark laws, the laws of privacy and publicity, and communications regulations and statutes.

Site Terms of Use: The following are terms of a legal agreement between you and Lantern Stove Connection. By accessing, browsing
and/or using this site ("Site"), you acknowledge that you have read, understood, and agree, to be bound by these terms and to comply with all applicable laws and regulations, including U.S. export and re-export control laws and regulations. If you do not agree to these terms, do not use this Site. The material provided on this Site is protected by law, including, but not limited to, United States Copyright Law and international treaties. This Site is controlled and operated by Lantern Stove Connection within the United States. Lantern Stove Connection makes no representation that materials in the Site are appropriate or available for use in other locations, and access to them from territories where their contents are illegal is prohibited. Those who choose to access this Site from other locations do so on their own initiative and are responsible for compliance with applicable local laws. Any claim relating to, and the use of, this Site and the materials contained herein is governed by the laws of the state of South Dakota.</p><p><b>Limitation of Liability:</b> UNDER NO CIRCUMSTANCES, INCLUDING, BUT NOT LIMITED TO, NEGLIGENCE, SHALL Lantern Stove Connection BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING, BUT NOT LIMITED TO, LOSS OF DATA OR PROFIT, ARISING OUT OF THE USE, OR THE INABILITY TO USE, THE MATERIALS ON THIS SITE, EVEN IF Lantern Stove Connection HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IF YOUR USE OF MATERIALS FROM THIS SITE RESULTS IN THE NEED FOR SERVICING, REPAIR OR CORRECTION OF EQUIPMENT OR DATA, YOU ASSUME ANY COSTS THEREOF. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.</p><p><b>30-Day Limited Warranty &amp; Return Policy:</b> Lantern Stove Connection acquires its products from parts surplus, auctions, and estates. Products from Lantern Stove Connection will be free from major defects in materials and workmanship. The warranty term is 30-days beginning on the date of invoice, as further described below. Damage due to shipping is NOT covered under this warranty. Otherwise, this warranty does not cover damage due to external causes, including accident, abuse, misuse, problems with seals, servicing not authorized by Lantern Stove Connection, usage not in accordance with product instructions, failure to perform required preventive maintenance, and problems caused by use of parts and components not supplied as original parts. Accessories or parts added to the product after the item is shipped from Lantern Stove Connection. Lantern Stove Connection will repair or replace products returned to the facility at our discretion. To request warranty service, you must...
contact Lantern Stove Connection within the warranty period. If warranty service is required, Lantern Stove Connection will issue a Return Material Authorization Number. You must ship the products back to Lantern Stove Connection in their original or equivalent packaging, prepay shipping charges, and insure the shipment or accept the risk of loss or damage during shipment. Lantern Stove Connection will ship the repaired or replacement products to you freight prepaid if you use an address in the United States. Shipments to other locations will be prepaid.

If Lantern Stove Connection repairs or replaces a product, its warranty term is not extended. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS THAT VARY FROM STATE TO STATE (OR JURISDICTION TO JURISDICTION). Lantern Stove Connection RESPONSIBILITY FOR MALFUNCTIONS AND DEFECTS IN HARDWARE IS LIMITED TO REPAIR AND REPLACEMENT AS SET FORTH IN THIS WARRANTY STATEMENT. ALL EXPRESS AND IMPLIED WARRANTIES FOR THE PRODUCT, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF AND CONDITIONS OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE WARRANTY PERIOD SET FORTH ABOVE AND NO WARRANTIES, WHETHER EXPRESS OR IMPLIED, WILL APPLY AFTER SUCH PERIOD. SOME STATES (OR JURISDICTIONS) DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU. Lantern Stove Connection DOES NOT ACCEPT LIABILITY BEYOND THE REMEDIES SET FORTH IN THIS WARRANTY STATEMENT OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING WITHOUT LIMITATION ANY LIABILITY FOR PRODUCTS NOT BEING AVAILABLE. SOME STATES (OR JURISDICTIONS) DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE EXCLUSION OR LIMITATION MAY NOT APPLY TO YOU.
<?

/****************************
* Name: Norman Harrod
* Course: INF 790:
****************************/

/* links.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?
</html>
</head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">

<body>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">

<body>
<? include("functions.php") ?>
<table width="100%"
<tr>
</td class="header">
</a><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<tr>
</td class="navigation">
<? loadNavigation("links") ?>
</td>
</tr>
<tr>
</td class="information">
<h2>Links</h2>
<p><a href="http://www.spiritburner.com" title="Online resource for the collector and enthusiast of pressure stoves.">Classic Camp Stoves</a></p>
<p><a href="http://www.asahi-net.or.jp/~we2a-sod/index.htm" title="Japanese collector with galleries of stoves for the viewing.">Super Shioshio's Room</a></p>
<p><a href="http://ourworld.compuserve.com/homepages/awm/" title="Most comprehensive pressure lantern resource on the web for discovering the history of a lantern.">Pressure Lamps Unlimited</a></p>
</td>
</tr>
<tr>
</td class="footer">
<? include("footer.php") ?>
</td>
</tr>
</table>
</body>
</html>
<?

/**************************************************************************
 * Name: Norman Harrod
 * Course: INFS 790:
**************************************************************************/

/* order.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");

?>

<html>
<head>
<title>Lantern Stove Connection</title>
</head>

<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">

<body>
"? include("functions.php") ?"
"?

# prepare inquiry text for body of message
for($i = 1; $i <= 5; ++ $i){
  if($ARRAY1[$i] == TRUE){
    $text .= "Product $i ";
    $text .= "$ARRAY1[$i] ";
    $text .= "Quantity ";
    $text .= "$ARRAY2[$i] ";
    $text .= "Price Per Item ";
    $text .= "\$";
    $text .= "$ARRAY3[$i]";
    $text .= "\n";
  }
}

# check for e-mail
if($useremail == ""){
?

<tr>
  <td class="header">
   <a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
  </td>
</tr>

<tr>
  <td class="navigation">
   "? loadNavigation("") ?"
  </td>
</tr>

<tr>
  <td class="information">
   A valid e-mail address must be included to complete this inquiry successfully, Please return to the form.
  </td>
</tr>

<tr>
  <td class="footer">
   "? include("footer.php") ?"
  </td>
</tr>

</body>
</html>
<?
}

else{
    # recipients
    $recipient = "ntharrod@prodigy.net";

    # subject
    $subject = "Order Inquiry";

    # message
    $message = "<<MESSAGE_BODY
FIRST NAME: $fname
LAST NAME: $lname
COUNTRY: $country
E-MAIL: $useremail

ORDER INQUIRY:
$text

CUSTOMER COMMENTS:
$comments

MESSAGE_BODY;

    # header piece(s)
    $headers = "X-Priority: 1\n"; //Urgent message!
    $headers .= "Reply-To: ntharrod@prodigy.net"; //Return path for errors

    # mail it
    $result = mail($recipient, $subject, $message, $headers);

    # successful send ?
    if($result == TRUE){
        <table width="100%"
        <tr>
            <td class="header">
                <a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
            </td>
        </tr>
        <tr>
            <td class="navigation">
                <![ loadNavigation("" ) ]>
            </td>
        </tr>
        <tr>
            <td class="information">
                Form Has Been Sent Successfully!
            </td>
        </tr>
        <tr>
            <td class="footer">
                <![ include("footer.php") ]>
            </td>
        </tr>
    </table>
} else {
    <table width="100%"
    <tr>
        <td class="header">
            <a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
        </td>
    </tr>
    <tr>
        <td class="navigation">
            <? loadNavigation("" ) ?>
        </td>
    </tr>
    <tr>
        <td class="information">
            Error, Please contact the WebMaster or try Again
        </td>
    </tr>
    <tr>
        <td class="footer">
            <? include("footer.php") ?>
        </td>
    </tr>
</table>
<?
}
}?>

</body>
</html>
Lantern Stove Connection

Lantern is in new condition. Construction is brass, steel, and subdued in grey military finish. Comes with olive drab steel transport case, spare parts, and funnel. Multi-fuel burning.

Manufacturer: Graetz K.G.
Country: Germany
Model: 829B
Year: 1962
Output: 400 C.P.
Dimensions: NA
<?
/*
 * Name: Norman Harrod
 * Course: INF5 790:
 */
/* prod.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css" />
<body>
<? include("functions.php") ?>
<table width="100%">
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<tr>
<td class="navigation">
<? loadNavigation("prod") ?>
</td>
</tr>
</table>
<tr>
<td class="information">
<h2>Products</h2>
<?
# if setcheck is not filled and not empty, print the form
if($SETCHECK != "filled" AND $SETCHECK != "empty"){
    # call printForm and set setcheck
    $SETCHECK = printForm();
}
# if setcheck is filled, its ready for checkout or need to empty and start over
if($SETCHECK == "filled"){
    print("Your shopping cart is already filled with some items, but here are the items not currently in your shopping cart.");
    # form elements based on selected items that are checked into a form for quantity edit
    print("<form method="post" action="viewcartsedit.php">\n<table width="100%">\n");
    for($i = 0; $i <= 4; ++ $i){
        if($ARRAY0[$i] == FALSE){
            print("<tr><td><b>Add</b></td><input type="checkbox" name="Prod" value="$i"/>
            print("<\tr>\t\t\" value="add\"">");
            print("<\t\t\ti>$ARRAY1[$i]\t</td><td"></td>");
            print("<a href="$ARRAY4[$i]"
            target="_blank">Description</a></td></tr>");
        }
    }
    print("PRICE \$ $ARRAY3[$i] Each\</td><td"></td>");
    print("<b>Quantity</b></td>\n")
?>
</table></form>\n\n</body>
name="changeqty" class="changeqty">
    print("<input type="text" size="5"
    name="changeqty">
    
    print("$i";
    print("\" value="\">\</td></tr>");
  }
}

print("</table><br><input type="submit" value="Add Checked Items to Shopping Cart"></form">");

# if setcheck is empty, print the form
if($SETCHECK == "empty"){
    # call printForm and set setcheck
    $SETCHECK = printForm();
}

</td>
<tr>
<td class="footer">

<? include("footer.php") ?>

</td>
</tr>
</table>
</body>
</html>
<?
/
*********************************
* Name: Norman Harrod
* Course: INFS 790:
***************************/
/* prodada.php */

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<? include("functionsada.php") ?>
<table width="100%">
<tr>
    <td class="header">
        <a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
    </td>
</tr>
<tr>
    <td class="navigation">
        <? loadNavigation("prod") ?>
    </td>
</tr>
<tr>
    <td class="information">
        <h2>Products</h2>
        <? # if setcheck is not filled and not empty print the form
        if($SETCHECK != "filled" AND $SETCHECK != "empty"){
            # call printForm and set setcheck
            $SETCHECK = printForm();
        } # if setcheck is filled, its ready for checkout or need
to empty and start over
        if($SETCHECK == "filled"){
            print("<p>Your shopping cart is already filled with some
items, but here are the items not currently in your shopping cart.
If no items are listed, then everything is in
your shopping cart.</p>");
        } # form elements based on selected items that are checked
        into a form for quantity edit
        print("<form method="post"
action="viewcarteditada.php">
<table width="100%"
    <table summary="This table lists the items
currently available for purchase and can
be added by check to add. The item must be
checked to add to the cart
and a quantity
should be entered into the text box."
    width="100%">
    <caption>Current items for
    purchase.</caption>

<tr>
<th scope="col">Check to Add</th>
<th scope="col">Product</th>
<th scope="col">Open Description Window</th>
<th scope="col">Price Each</th>
<th scope="col">Quantity</th>

</tr>

for($j = 0; $j <= 4; ++$j){
    if($ARRAY0[$j] == FALSE){
        print("<tr><td><center><input type="checkbox" name="Prod$j" value="add"/></center></td>
               <td><center><a href="$ARRAY4[$j]" target="_blank">Click HERE for Description</a></center></td>
               <td><center><a href="$ARRAY3[$j]" target="_blank"></center></td>
        </tr>");
    }
}

print("</table><br><input type="submit" value="Add Checked Items to Shopping Cart"></form">");

if(!$SETCHECK == "empty"){
    # call printForm and set setcheck
    $SETCHECK = printForm();
}

</tr>

<td class="Footer">
    </td>
</tr>
</table>
</body>
</html>
<?
/
.HasValue
/* Name: Norman Harrod
 * Course: INF5 790:
.HasValue
.HasValue
.HasValue
.HasValue
.HasValue
/>

# register session variables
session_register("ARRAY0");
session_register("ARRAY1");
session_register("ARRAY2");
session_register("ARRAY3");
session_register("ARRAY4");
session_register("SETCHECK");
?>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<!-- Links html page to cascading style sheet -->
<link rel="stylesheet" type="text/css" href="sitestyle.css">
<body>
<? include("functions.php") ?>
<table width="100%"
<tr>
<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<td class="navigation">
<? loadNavigation("service") ?>
</td>
</tr>
<td class="information">
<h2>Customer Service Comes FIRST!</h2>
<b>Please fill out the following information to expedite your order inquiry or question</b>
<form method="post" action="order.php">
<table width="100%">
<tr>
<td>First Name:</td><td><input type="text" name="fname" size="45" maxlength="25"></td>
</tr>
<tr>
<td>Last Name:</td><td><input type="text" name="lname" size="45" maxlength="25"></td>
</tr>
<tr>
<td>Country:</td><td><input type="text" name="country" size="45" maxlength="25"></td>
</tr>
<tr>
<td>Valid E-mail:</td><td><input type="text" name="useremail" size="45" maxlength="40"></td>
</tr>
</tbody></table></form>
</td>
</tr>
</tbody></table></body></html>
<td>Comments:</td><td><textarea name="comments" cols="45" rows="4"></textarea></td></tr>
</table>
<br>
<input type="submit" value="Submit Question"

Please click one of the lines below to find out more about us.
<ul>
<li><a href="faq.php">Frequently Asked Questions</a></li>
<li><a href="testimonials.php">What do other customers think about Lantern Stove Connection</a></li>
</ul>
</td>
</tr>
<td class="footer">
<? include("footer.php") ?>
</td>
</tr>
</table>
</body>
</html>
Name: Norman Harrod
Course: INFS 790
Cascading Style Sheets file sitestyle.css

/* Table background New Tan */

table {background: #EBC79E;
}

/* Class for navigation with white background and Forest Green text */
td.navigation {text-align: center;
border-style: solid;
border-width: 1px;
border-color: #215E21;
background: #FFFFFF;
}

/* Horizontal rule Forest Green */
hr {color: #215E21;
}

/* Color of links changes to Forest Green when mouse is over the link */
A:hover {color: #215E21;}
<html>
<head>
<title>Lantern Stove Connection</title>
</head>
<body>
<? include("functions.php") ?>
<table width="100%"
<tr>

<td class="header">
<a href="index.php"><img src="logo.jpg" alt="Lantern Stove Connection Logo"></a>
</td>
</tr>
<tr>
<td class="navigation">
<? loadNavigation(""") ?>
</td>
</tr>
<tr>
<td class="information">
<h2>Customer Testimonials</h2>
I was looking for a site that offered a variety of lanterns at reasonable prices.
Lantern Stove Connection fits the bill!
- Mike Johnson, South Dakota, USA
I've dabbled in some auction sites, but none measure up to the service I received.
- Ross Spirits, London, UK
The customer service and packing is awesome! I have been eagerly looking forward to new items, as they're in excellent condition!
- Stan Man, California.
We are here to help you! If you have questions please send us an email through checkout or call 1-800-xxx-xxxx.
</td>
</tr>
<tr>
<td class="footer">
<? include("footer.php") ?>
</td>
</tr>
</body>
</html>
</tr>
</table>
</body>
</html>
Lantern Stove Connection

TILLEY

Manufacturer: Tilley
Country: United Kingdom
Model: TL136
Year: 1958
Output: 300 C.P.
Dimensions: NA