1998

A web-based universal encyclopedia/dictionary

Francis Hyeongwoo Lee

Follow this and additional works at: http://scholarworks.lib.csusb.edu/etd-project

Part of the Digital Communications and Networking Commons

Recommended Citation

http://scholarworks.lib.csusb.edu/etd-project/1812

This Project is brought to you for free and open access by the John M. Pfau Library at CSUSB ScholarWorks. It has been accepted for inclusion in Theses Digitization Project by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.
A WEB-BASED UNIVERSAL ENCYCLOPEDIA/DICTIONARY

A Project
Presented to the
Faculty of
California State University,
San Bernardino

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Francis Hyeongwoo Lee
June 1998
A WEB-BASED UNIVERSAL ENCYCLOPEDIA/DICTIONARY

A Project
Presented to the
Faculty of
California State University,
San Bernardino

by
Francis Hyeongwoo Lee
June 1998

Approved by:

Dr. George M. Georgiou, Chair, Computer Science

Dr. Josephine Mendoza

Dr. Kerstin Voigt

6/11/98
Web-based Universal Encyclopedia/Dictionary (WUED) is a software application that provides a database independent user tool that places online encyclopedia or dictionary type data. WUED creates a searchable encyclopedia/dictionary of user provided data through a user-friendly Graphical User Interface via an Internet browser. It allows a user to display terms and related resources online so that information of interest can be easily browsed. It provides a one-stop source of information about stored data, and includes cross-referencing and hyperlinks to related resources elsewhere.
# TABLE OF CONTENTS

ABSTRACT ................................................................. iii  
LIST OF TABLES .......................................................... vi  
LIST OF FIGURES ........................................................ viii  
CHAPTER 1 SOFTWARE REQUIREMENT SPECIFICATION ............... 1  
  1.1 Introduction ....................................................... 1  
  1.2 Product Overview and Summary ................................... 3  
  1.3 Development, Operating, and Maintenance Environments .......... 5  
    1.3.1 Development Environment .................................. 5  
    1.3.2 Operating Environment .................................... 6  
    1.3.3 Maintenance Environment .................................. 6  
  1.4 Graphical User Interface (GUI) .................................... 7  
    1.4.1 General Layout .............................................. 7  
    1.4.2 Home ......................................................... 11  
    1.4.3 History .................................................... 12  
    1.4.4 Random ..................................................... 14  
    1.4.5 Contents .................................................. 15  
    1.4.6 Edit ......................................................... 17  
    1.4.7 Statistics ................................................. 19  
    1.4.8 Help ....................................................... 20  
    1.4.9 Submit ..................................................... 21  
    1.4.10 Contact ................................................... 22  

iv
LIST OF TABLES

Table 1. User inputs and corresponding actions in Menu .............. 10
Table 2. User inputs and corresponding actions in Edit .............. 18
Table 3. Database Table: wued ................................... 30
Table 4. Database View Table: v_wued ............................... 31
Table 5. Database Table: guest .................................... 31
Table 6. Database Table: stat ...................................... 32
Table 7. Database Table: submit .................................... 32
Table 8. Product Test ................................................ 60
Table 9. Performance Test: Data Size 8.5 MB ......................... 63
Table 10. Performance Test: Data Size 12.8 MB ....................... 64
Table 11. Performance Test: Data Size 17.0 MB ....................... 64
Table 12. Performance Test: Data Size 21.3 MB ....................... 65
Table 13. Performance Test: Data Size 25.5 MB ....................... 65
Table 14. Performance Test: Data Size 29.8 MB ....................... 66
Table 15. Performance Comparison for Full-text Search .............. 67
Table 16. Performance Comparison for Exact-match Search .......... 67
LIST OF FIGURES

Figure 1. Menu ........................................ 7
Figure 2. Full-text Search .............................. 8
Figure 3. Exact-match Search ............................ 9
Figure 4. Home .......................................... 11
Figure 5. History ........................................ 12
Figure 6. Random ........................................ 14
Figure 7. Contents ....................................... 15
Figure 8. Display of Entries ............................ 16
Figure 9. Edit Options ................................... 17
Figure 10. Statistics .................................... 19
Figure 11. Help ......................................... 20
Figure 12. Submit ....................................... 21
Figure 13. Contact ...................................... 22
Figure 14. About ....................................... 23
Figure 15. Overall Algorithm ........................... 25
Figure 16. State Diagram ............................... 26
Figure 17. WUED Algorithm ............................. 27
Figure 18. Directory Structure .......................... 28
Figure 19. Conceptual Model Diagram .................. 29
Figure 20. Operation with Database for Search and Statistics .................. 38
Figure 21. Operation with Database for Search and Statistics .......................... 40

Figure 22. Performance Comparison Graph ............. 68
CHAPTER 1 SOFTWARE REQUIREMENT SPECIFICATION

1.1 Introduction

The Internet is rapidly becoming mainstream media conduit for communication between individuals, companies, and global dwellers. As part of the Internet, the Web has been growing extremely fast in recent years, and its applications served as the major tools of exchanging the information. Many of the Web applications have used a simple ASCII text file to store data. However, more efficient and convenient ways of storing data are demanded since applications are becoming more flexible and complicated, and requiring storing larger amounts of data. The Web-based database was carefully considered to solve the problem and can be used to develop other Web applications.

One of the Web applications that easily provides useful information to browsers is the online dictionary. There are already several hundreds of online dictionaries on the Web allowing browsers to access the information of interest. For example, the Free On-line Dictionary of
Computing (FOLDOC), http://wombat.doc.ic.ac.uk/ [1] is an online searchable dictionary designed by Denis Howe in 1985. The dictionary is stored as a single text file and contains over 11,000 definitions totaling more than four megabytes. It allows user to enter queries and displays the related information.

WUED was designed to provide a user tool that creates something with similar functionality to the existing online dictionaries and uses a database instead of using ad hoc scripts and the flat files. WUED uses generic database functions in order to be independent of the database and can be easily deployed when data is available.
1.2 Product Overview and Summary

The software product is designed to provide a database independent user tool that creates Web-base online encyclopedia/dictionary. WUED creates a searchable encyclopedia/dictionary of user provided data through the user-friendly GUI of Internet browsers. The online encyclopedia/dictionary allows a user to display his/her terms and definitions online so that browsers can quickly and easily browse information of interest. It aims to provide a one-stop source of information about data stored which can include cross-references and hyperlinks to related resources elsewhere. The CGI (Common Gateway Interface) script is written in Perl (Practical Extraction and Report Language). Extensive use of Perl's regular expression matching facilities is made to provide fast search and retrieval of an entry as well as full-text search. WUED offers easy installation, flexibility in selecting a database and simplicity of use.

WUED provides an INSTALL script that performs all the necessary installation procedures. The interactive procedure of the INSTALL script defines the name of the encyclopedia/dictionary, the type of database management
system that is used and other configuring information. During installation of WUED a user is selecting any one of the database systems that the Perl5 Database Interface (DBI) and Database Driver (DBD) support. DBI defines and implements a common interface to enable interaction between applications and various database engines. DBD contains implementations of the DBI functions written using the private interface functions of the corresponding database engines [2].

WUED provides the browser information that contains hyperlinks to the Web, to audio and image files that are stored locally. Such files may contain music, pronunciations of words, pictures, maps, etc.

WUED’s EDIT functionality provides an easy way to edit the information stored in encyclopedia/dictionary. It allows a user to import, insert, delete and modify data in the database. This utility is only available to the administrator of the encyclopedia/dictionary via a password, and general browsers are not allowed to access it.
1.3 Development, Operating, and Maintenance Environments

1.3.1 Development Environment

WUED uses the following hardware and software on the server. Clients can use standard Web browsers such as Netscape Navigator, Internet Explorer to access.

1. A PC (Personal Computer) with Redhat Linux 5.0 specifically
   a. 100 Mhz Pentium
   b. 1.2 GB hard disk
   c. 32 MB RAM
   d. Linux Kernel v2.0.31

2. Perl v5.004_04

3. The Perl5 Database Interface (DBI) v0.63

4. Database Driver for PostgreSQL (DBD::Pg) v0.90

5. POSTGRES95

6. Apache Server 1.2
1.3.2 Operating Environment

The product shall operate within the environment as specified above in section 1.3.1

1.3.3 Maintenance Environment

The product shall be maintained within the environment as specified above in section 1.3.1
1.4 Graphical User Interface (GUI)

WUED's GUI offers an easy way for browsing the encyclopedia/dictionary. This section describes each of the GUI that WUED provides.

1.4.1 General Layout

A menu is displayed at the top of each window to enable a browser to navigate the encyclopedia/dictionary by a simple click of a tool button (Figure 1). The menu consists of six tool buttons, a text box and a drop-down list. Tool buttons allow a browser to move on to the other WUED utilities of interest. The utilities that each tool button allows a browser to access are Home, History, Random, Contents, Edit, Help and Search. Table 1 summaries the effects of the various user inputs.

![Diagram of menu layout]

Figure 1. Menu
A browser can conduct a full-text search or exact-match search by typing a search query and hitting the search button or Enter.

The search will run on the database and display related information. The full-text search displays the number of entries found and a list of entries along with brief summaries of each entry. Full-text Search searches through all keywords and terms that contain the keywords specified in the search are displayed as hyperlinks (Figure 2).

Figure 2. Full-text Search

The exact-match search displays the definition of an entry, if present in the database. If the definition contains other resources such as related URLs, images or
sounds, these resources are converted to hyperlinks for easy access by the browser (Figure 3).

![Web-based Universal Encyclopedia/Dictionary - Netscape](image)

**Information**

**Routing information protocol**

1. (RIP) A distance vector, as opposed to link state, routing protocol. RIP is an Internet standard Interior Gateway Protocol defined in STD 34, RFC 1058 and updated by RFC 1388.

Figure 3. Exact-match Search

In addition to the tool buttons, each page of WUED contains hyperlinks of other WUED utilities that a browser can access; Submit, Statistics and About. The description of these utilities is found in the following sections.
<table>
<thead>
<tr>
<th>User Input</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click the Home button</td>
<td>Display Home page</td>
</tr>
<tr>
<td>Click the History button</td>
<td>Display History page</td>
</tr>
<tr>
<td>Click the Random button</td>
<td>Display Random page</td>
</tr>
<tr>
<td>Click the Contents button</td>
<td>Display Contents page</td>
</tr>
<tr>
<td>Click the Edit button</td>
<td>Display Edit page</td>
</tr>
<tr>
<td>Click the Help button</td>
<td>Display Help page</td>
</tr>
<tr>
<td>Type a query in Text Box</td>
<td>Display the query in Text Box</td>
</tr>
<tr>
<td>Move mouse-pointer onto Drop-down List</td>
<td>Display search options</td>
</tr>
<tr>
<td>Click search options</td>
<td>Display the search option in Drop-down List</td>
</tr>
<tr>
<td>Click the Search button</td>
<td>Display Search results</td>
</tr>
</tbody>
</table>

Table 1. User inputs and corresponding actions in Menu
1.4.2 Home

Home explains the purpose of the online encyclopedia/dictionary and gives the browser a summary of how to use the online encyclopedia/dictionary (Figure 4).

![Home of World Dictionary of Computing](image)

**Figure 4. Home**
1.4.3 History

History displays previous entries that a browser accessed (Figure 5). This utility enables browsers to skip unnecessary procedure to search for the same information that they previously accessed.

![History](image)

Figure 5. History

WUED uses a technology called cookies for History. Browsers that support this technology include Netscape Navigator 3.0, Communicator 4.0 PR2 (at least) and Internet Explorer 3.0. A browser most likely will not be able to access this utility with earlier versions of these browsers.
The entries accessed are added in Set-Cookie header that is to be stored by the client for later retrieval. WUED sets two days of the expiration date that defines the valid time of the entries.
1.4.4 Random

Random provides a way to explore WUED’s browsable database. When a browser doesn’t have a specific term in question, Random is the best approach. A single entry of information is randomly selected through the entire database and displayed on browsers. Browsing Random is much like flipping the pages of an encyclopedia/dictionary (Figure 6).

Random displays the entry by randomly generating two characters and executing the SQL command that contains the characters and the percent sign wildcards (%). The SQL command used is as follows,

```
Select term from mydb where term like 'Char%Char%';
```

Figure 6. Random
1.4.4 Contents

Contents provide an alphabetical list of entries in the database (Figure 7). By simply clicking one of the letters in the alphabet, a browser can access the list of entries starting with the alphabetic letter.

![Web-based Universal Encyclopedia/Dictionary - Netscape](image)

**Content**

By simply clicking one of the following letters in the alphabet, a browser can access the list of entries starting with the alphabet letter.

A B C D E F G H I J K L M

Figure 7. Contents

Only 50 entries are displayed on each request, and a browser is allowed to move on to the next 50 entries by clicking "Next 50 Terms" button (Figure 8).
Contents of "f"

- f-code
- face time
- facsimile
- failure
- f-logic
- face-to-face
- fact
- failure-directed testing
- f2c
- facile
- factor
- fairchild f2c

Figure 8. Display of Entries
1.4.6 Edit

Edit is provided for modification of data in database. Only an administrator of the encyclopedia/dictionary is authorized to access this utility. The administrator is prompted to enter a password; entering an invalid password results in denial of access.

When a valid password is provided, edit options are displayed (Figure 9). By selecting an option, the administrator is allowed to import, insert, delete and modify data in database. Table 2 summaries the effects of the various user inputs.

![Edit Options](image)

Figure 9. Edit Options
The Apache hypertext transport protocol daemon was used to secure edit functionality. The directory containing the script for Edit has access protection by creating "htpasswd" and "htaccess" files. "htaccess" describes how to restrict access to the contents of the directory, and "htpasswd" contains the useids and passwords.

<table>
<thead>
<tr>
<th>User Input</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move mouse-pointer onto Drop-down List</td>
<td>Display options</td>
</tr>
<tr>
<td>Import data from Web submission</td>
<td>Display data from DBMS</td>
</tr>
<tr>
<td>Import data from a data file</td>
<td>Display a input request form</td>
</tr>
<tr>
<td>Insert new data to DBMS</td>
<td>Display a input request form</td>
</tr>
<tr>
<td>Delete data from DBMS</td>
<td>Display a input request form</td>
</tr>
<tr>
<td>Modify data in DBMS</td>
<td>Display data to modify</td>
</tr>
<tr>
<td>Delete comments from Guestbook</td>
<td>Display a input request form</td>
</tr>
<tr>
<td>Delete the most frequently missed term</td>
<td>Display a input request form</td>
</tr>
</tbody>
</table>

Table 2. User inputs and corresponding actions in Edit
1.4.7 Statistics

Statistics displays the most frequently requested missing terms and the most frequently requested terms by browsers (Figure 10). Every time a browser requests for a term, the number of requests of the term is incremented and recorded in the database. The maximum number of request is retrieved and used to display the entries that have the top ten highest numbers of the requests.

Figure 10. Statistics
1.4.8 Help

Help provides tips for navigating through an online encyclopedia/dictionary and offers detailed information on such topics as searching and features (Figure 11).

Figure 11. Help
1.4.9 Submit

Browsers are allowed to submit their information to an administrator through Submit (Figure 12). Their information is stored in the database and can be used to enhance an online encyclopedia/dictionary.

Figure 12. Submit
1.4.10 Contact

Contact is a place for browsers to make their comments about an online encyclopedia/dictionary (Figure 13).

Contact

This is a place for browsers to make their comments about an online encyclopedia/dictionary. The comments will be stored in database and displayed by browsers through Guestbook.

Please provide the following information.

Figure 13. Contact

The comments are stored in database and displayed by browsers through Guestbook.
1.4.11 About

Browsers can find the information about WUED through About (Figure 14). About introduces a description of WUED and a pointer where WUED packages are available.

Figure 14. About
1.5 Functional Requirements

The WUED operational structures and their requirements are described in this section.
1.5.1 Overall Algorithm

Figure 15. Overall Algorithm
1.5.2 State Diagram

The following is the state diagram of WUED functions. Every State can go back to the Menu State to reach its final destination.

![State Diagram]

Figure 16. State Diagram
1.5.3 WUED Algorithm

Figure 17. WUED Algorithm
1.5.4 Directory Structure

The following is a file and directory structure of WUED.

```
Library Directory

Wued Directory
  Wued.cgi
  README
  Admin Subdirectory
    Installation File
  Sound Subdirectory
  Image Subdirectory
    Database Subdirectory
    Wued Subdirectory
```

Figure 18. Directory Structure
1.5.5 Database Design

This section shows the database design structure of WUED.

1.5.5.1 Conceptual Model Diagram

[Diagram showing database tables with terms and definitions]

Figure 19. Conceptual Model Diagram
1.5.5.2 Logical Model Table Schema

Each of the following shows a database table, description of attributes and examples.

Table Name: wued

Term: Name of an entry (text)
Definition: Definition of a term (text)
Key: Keywords used for search (text)
Reqnumb: Number of times that a term is requested (int)

<table>
<thead>
<tr>
<th>term</th>
<th>definition</th>
<th>key</th>
<th>reqnumb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bird</td>
<td>Warm-blooded ...</td>
<td>{bird},{wings} ...</td>
<td>1</td>
</tr>
<tr>
<td>Lion</td>
<td>Feline mammal ...</td>
<td>{lion},{mammal} ...</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 3. Database Table: wued
Table Name: v_wued

Term: Entry (text)
Definition: Definition of a term (text)
Key: Keywords used for search (text)

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bird</td>
<td>Warm-blooded ...</td>
<td>{bird},{wings} ...</td>
</tr>
<tr>
<td>Lion</td>
<td>Feline mammal ...</td>
<td>{lion},{mammal} ...</td>
</tr>
</tbody>
</table>

Table 4. Database View Table: v_wued

Table Name: guest

Name: Name of a visitor (text)
Address: Address of a visitor (text)
Comments: Comments submitted (text)

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Kemp</td>
<td>Jkemp@cs...</td>
<td>It is a very ...</td>
</tr>
<tr>
<td>Steven Thomas</td>
<td><a href="mailto:Sthomas@mkt.com">Sthomas@mkt.com</a></td>
<td>WUED helps me ...</td>
</tr>
</tbody>
</table>

Table 5. Database Table: guest
Table Name: stat

Term: Name of an entry (text)
Reqnumb: Number of times that a term is missed (int)

<table>
<thead>
<tr>
<th>Term</th>
<th>Reqnumb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ant</td>
<td>1</td>
</tr>
<tr>
<td>Horse</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 6. Database Table: stat

Table Name: submit

Term: Name of entry (text)
Definition: Definition of a term (text)

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
<td>A specification of how ...</td>
</tr>
<tr>
<td>Function</td>
<td>A mapping of each of ...</td>
</tr>
</tbody>
</table>

Table 7. Database Table: submit
1.5.5.3 Database Management System

WUED is designed to be a database independent software application, which allows a user to select a DBMS during installation. However, the type of DBMS that DBI and DBD support limits WUED's DBMS selection. The following is the list of DBMSs that DBI and DBD support.

- Oracle7 RDBMS
- mSQL-1.x or mSQL-2.x
- mysql
- Ingres 6.4 or OpenIngres
- Informix 5.00 through to Informix 7.22
- Sybase
- Empress 6.8
- Fulcrum SearchServer 2.0, 3.x SDK
- DB2 v2.1 or beyond
- Quickbase
- Interbase
- Solid
- Postgres
1.5.5.4 SQL Commands

In order for WUED to be independent of DBMS, the SQL commands used in WUED should be available to all of DBMSs that WUED supports. The following is a list of the SQL commands used in WUED implementation.

- CREATE DATABASE db
- DROP DATABASE db
- CREATE TABLE table (column datatype [,column datatype] ...)
- DROP TABLE table
- CREATE INDEX index ON table (column [,column] ...)
- DROP INDEX index
- CREATE VIEW view AS select_subset
- DROP VIEW view
- SELECT select_list FROM table [WHERE conditions][ORDER BY column][ASC|DESC]
- INSERT [INTO] table VALUES (values_list)
- DELETE FROM table [WHERE conditions]
- UPDATE table SET column = expr [,column = expr, ...]
1.5.6 Data File Format

WUED allows an administrator of the encyclopedia/dictionary to import data directly into DBMS from a formatted data file. The format of an input data file should be carefully observed and is as follows. The example below shows the sample of the data file format.

1. A data file should start with the name of an entry. A blank line follows the entry.
2. Each line of a description should start with the tab character. A blank line follows the last line of definition.
3. Hyperlinks of each resource should use the following convention. Image files and sound files should be located in an image subdirectory (...wued/image/database) and a sound subdirectory.

Image: {image: (Name of an image file)}
Sound: {sound: description (Name of a sound file)}
URL: {description (url)}
Newsgroup: newsgroup: {news: url}
Email: {description (mailto: email address)}
i.e.

WUED

Web-base Universal Encyclopedia/Dictionary
developed at {CSUSB (http://www.csci.csusb.edu)}. A {online} searchable encyclopedia/{dictionary}.
WUED LOGO {image:(logo.gif)}
{sound:Click here to hear Jungle Sound (wued.wav)}
{For more information E-mail to Francis Lee
(mailto:flee@csci.csusb.edu)}
Get WUED Package {Download(ftp://www.csci.csusb.edu)}
Newsgroup: {news:csci.csusb}

CGI

The Common Gateway Interface. It is the part of Web Server that can communicate with other programs running on the server.
1.5.7 Operation with Database Table

1.5.7.1 Search Operation

Full-text Search matches all of keywords specified in key attribute of WUED database table. Any entries that contain the query as keywords will be displayed when Full-text Search is conducted. Exact-match Search matches an entry in term attribute of the database table. When data is inserted in the database, the loading script extracts the name of term, keywords and definition and inserts them in the database.

Keywords consist of the entry itself and any cross-references specified in definition. The cross-reference is {online} and {dictionary}. Each of these is stored in database and retrieved by Search Engine (Figure 20).
Figure 20. Operation with Database for Search and Statistics

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
<th>Key</th>
<th>Regnumb</th>
</tr>
</thead>
<tbody>
<tr>
<td>WUED</td>
<td>Web-based ...</td>
<td>{WUED}{online}</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>A {online} searchable ....</td>
<td>{dictionary}</td>
<td></td>
</tr>
<tr>
<td>WWW</td>
<td>World-wide {Web}</td>
<td>{WWW}{Web}</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
1.5.7.2 Statistics

The number of requests of each entry and the number of requests of non-existent entries are stored in the database so that Statistics displays the entries that have been requested most and missed most.

1.5.7.3 Web submission and Contact

The Web submissions from browsers are stored in the database and retrieved by the administrator for the database modification. The Web submissions can be either inserted in the main database table or deleted by using Edit. Browsers' comments are stored in the database and displayed through Guestbook. These comments can be also deleted by using Edit. Figure 21 is the diagram that shows how each of the above functions operates.
Figure 21. Operation with Database for Contact, Submit, and Statistics
1.6 Performance Requirements

1.6.1 Reliability

The reliability of WUED was verified via extensive testing of all features.

1.6.2 Efficiency

Since this is an interactive program, the response time for the next display is very important. In order to make this product more efficient, there was careful consideration in the design to reduce number of connection from CGI's script to the database.

1.6.3 Testability

Each requirement was identified and tested in the final product.
1.7 Exception Handling

Error messages are displayed on the browser when the system detects an error.
1.8 Conclusion

A complete and functional product was according to specification. WUED offers various functions that make it easy to access information and maintain data in the database.

Overall WUED is an improved online dictionary by providing the useful utilities such as the cross-referencing capabilities, the simple data retrieval functions, the database independence and user friendly GUI. It includes what we believe are best features of other existing online encyclopedia/dictionary.

The flexibility of the design of WUED allows easy modification for transforming it to specific applications such as Parts Catalog or Shopping Cart.
1.9 Proposed Future Development

1. The system, which was developed and implemented on Redhat Linux 5.0, may be modified to run on Windows environments. A possible environment would be Windows 95, Perl, DBI, DBD, and MS SQL Server.

2. WUED uses simple database tables to store data. There was a limitation on using database commands since WUED was designed to be database independent. If careful consideration of the design of a particular database is made at the expense of database independence, the performance of WUED may be improved.
1.10 Acceptance Criteria

1.10.1 Test Acceptance Criteria

The final product has met all requirements stated in this document (Section 1.5, Section 1.6).

1.10.2 Testing Methodology

The final product was tested in conformance with the following test modules.

1. Product Testing: The operation of each function and GUI was tested.

2. Usability Testing: As a sample of online encyclopedia, the FOLDOC [1] was loaded. Comments from various users were obtained, which were used to improve the functionality of WUED.

3. Performance Testing: The CPU execution time of the search engine was measured for various sizes of data in database.
1.11 Glossary of Terms

Browser

A client program (software loaded on your machine) that enables you browse documents on the Internet.

Hypertext link, or hyperlink

A highlighted word in a hypertext document which when selected leads the browser to another location.

URL, or Uniform Resource Locator

Specifies the location or address of documents on the World Wide Web.
CHAPTER 2 DETAILED DESIGN

This chapter presents the refinements of the architecture of WUED. What follows is the description of each main function of WUED. For each of the main function algorithm is given in pseudo language.

<table>
<thead>
<tr>
<th>1. Function Name</th>
<th>WUED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Main cgi of the WUED program</td>
</tr>
<tr>
<td>Subitems</td>
<td>Extract user commands and call subroutines</td>
</tr>
<tr>
<td>Note</td>
<td>Main frame of WUED</td>
</tr>
</tbody>
</table>

Procedure WUED
Begin
Declare Global Variables
Extract Command
If there is Command
    Execute Command
Else
    Display Home Page
End If
End

<table>
<thead>
<tr>
<th>2. Function Name</th>
<th>Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Conduct Search</td>
</tr>
<tr>
<td>Subitems</td>
<td>Full-text Search, Exact-match Search</td>
</tr>
<tr>
<td>Note</td>
<td></td>
</tr>
</tbody>
</table>

Procedure Search
Begin
Extract Term
Extract Search Command
Unless Term is extracted
Display Error Message
Else
Execute Search Command
End If
End

3. Function Name | Search_Full
Where Used | Search
Purpose | Conduct Full-text Search
Subitems | List Matching Term
Note |

Procedure Search_Full
Begin
Extract Term
Split if not Single Word Term
Establish DB Connection
While there is Term for search
  While Entry is Found
    Retrieve Summary of Definition
  End While
  If Term Found
    Display Search Results
  Else
    Display Error Message
    Update Statistics DB Table
  End
End If
Close DB Connection
End

4. Function Name | Search_Exact
Where Used | Search
Purpose | Conduct Exact-match Search
Subitems |
Note |

Procedure Search_Exact
Begin
Extract Term
Establish DB Connection
Retrieve Definition
If Definition is retrieved
  Call Set_Cookie
Display Term and Definition
Increment Number of Request
Close DB Connection
Else
Display Error Message
End If
End

<table>
<thead>
<tr>
<th>5. Function Name</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Random Display of Term</td>
</tr>
<tr>
<td>Subitem</td>
<td></td>
</tr>
<tr>
<td>Note</td>
<td></td>
</tr>
</tbody>
</table>

Procedure Random
Begin
Establish DB Connection
Until Term and Definition are retrieved
Generate Random Character
Retrieve Term and Definition
End While
Display Term and Definition
Close DB Connection
End

<table>
<thead>
<tr>
<th>6. Function Name</th>
<th>History</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Display Previous Accessed Term</td>
</tr>
<tr>
<td>Subitem</td>
<td>Parse_Client_Cookie</td>
</tr>
<tr>
<td>Note</td>
<td></td>
</tr>
</tbody>
</table>

Procedure History
Begin
Retrieve Terms from Cookies
If Terms exist
Display Terms accessed previously
Else
Display Message for the First Time Visitor
End if
End
7. Function Name | Content
---|---
Where Used | WUED
Purpose | Display List of Terms in Alphabetic Order
Subitems | Note
Note | Each Display contains 50 Terms

Procedure Content
Begin
Extract Character
Extract Term
Establish DB Connection
If Term is extracted
Display the next 50 Terms after the Term retrieved
Else
Display the first 50 Terms lead by the Char.
End IF
Close DB Connection
End

8. Function Name | Edit_Menu
---|---
Where Used | WUED
Purpose | Display Edit Menu
Subitems | Note

Procedure Edit_Menu
Begin
Extract Request Method
If Request Method is POST
Display Edit Menu
Else
Display Error Message
End IF
End

9. Function Name | Edit_Proceed
---|---
Where Used | WUED
Purpose | Extract Edit Command and Call Edit Function
Subitems | Note
Note

50
Procedure Edit_Proceed
Begin
  Extract Request Method
  IF Request Method is POST
    Extract and Execute Edit Command
  Else
    Display Error Message
  End IF
End

10. Function Name | Import
Where Used       | WUED
Purpose          | Retrieve Data from a Data File

Procedure Import
Begin
  Extract Request Method
  If Method is POST
    Establish DB Connection
    Retrieve Data from DB
    If Data is retrieved
      Display Data
      Display Import Selection
    Else
      Display No Data Available Message
    End IF
  Else
    Display Error Message
  End If
End

11. Function Name | Import_Proceed
Where Used       | WUED
Purpose          | Import Data in Database

Procedure Import_Proceed
Begin
  Extract Request Method
  If Method is POST
Extract Import Selection
If Import Selection is Delete
   Establish DB Connection
   Delete Data from Submit DB Table
   Close DB Connection
   Display Confirmation Message
Else If Import Selection is Insert
   Extract Term
   Unless Term is extracted
      Display Error Message
Else
   Establish DB Connection
   Insert Data in DB
   Close DB Connection
   Display Confirmation Message
End If
Else
   Display Error Message
End If
Else
   Display Error Message
End If
End

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Insert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Request for User Input</td>
</tr>
<tr>
<td>Subitems</td>
<td>Note</td>
</tr>
</tbody>
</table>

Procedure Insert
Begin
   Extract Request Method
   If Method is POST
      Request for User Input
   Else
      Display Error Message
   End
End
13. Function Name: Insert_Proceed
Where Used: WUED
Purpose: Insert Data in Database
Subitems
Note:

Procedure Insert_Proceed
Begin
  Extract Request Method
  If Method is POST
    Unless Term or Definition is extracted
      Display Error Message
    Else
      Establish DB Connection
      Insert Data in DB
      Display Confirmation Message
      Close DB Connection
  End If
  Else
    Display Error Message
  End IF
End

14. Function Name: Delete
Where Used: WUED
Purpose: Request for User Input
Subitems
Note:

Procedure Delete
Begin
  Extract Request Method
  If Method is POST
    Request for User Input
  Else
    Display Error Message
  End IF
End
15. Function Name: Delete Proceed
   Where Used: WUED
   Purpose: Delete Data from Database
   Subitems:
   Note:

   Procedure Delete_Proceed
   Begin
   Extract Request Method
   If Method is POST
     Extract Term
     If Term is extracted
       Establish DB Connection
       Delete Data from DB
       Display Confirmation Message
       Close DB Connection
     Else
       Display Error Message
   Else
     Display Error Message
   End

16. Function Name: Modify
    Where Used: Edit
    Purpose: Request User Input
    Subitems:
    Note:

    Procedure Modify
    Begin
    Extract Request Method
    If Method is POST
      Request for User Input
    Else
      Display Error Message
    End IF
    End

17. Function Name: Modify_Retrieve
    Where Used: WUED
    Purpose: Retrieve Data from Database
    Subitems:
    Note:

54
Procedure Modify_Proceed
Begin
    Extract Request Method
    If Method is POST
    Extract Term
        If Term is extracted
            Retrieve and Display Data and Request for User Modification
        Else
            Display Error Message
        End If
    Else
        Display Error Message
    End If
End

18. Function Name | Modify_Proceed
Where Used      | WUED
Purpose         | Modify Data in Database
Subitems        | Note

Procedure Modify_Proceeddd
Begin
    Extract Request Method
    If Method is POST
        Unless Term or Definition is extracted
            Display Error Message
    Else
        Establish DB Connection
        Modify Data in DB
        Close DB Connection
        Display Confirmation Message
    End If
    Else
        Display Error Message
    End If
End
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Where Used</th>
<th>Purpose</th>
<th>Subitems</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteGuest</td>
<td></td>
<td>WUED</td>
<td>Request User Input</td>
<td></td>
</tr>
<tr>
<td>DeleteSTerm</td>
<td></td>
<td>Edit</td>
<td>Delete Term in Statistics Database Table</td>
<td></td>
</tr>
<tr>
<td>DeleteStat</td>
<td></td>
<td>WUED</td>
<td>Request User Input</td>
<td></td>
</tr>
</tbody>
</table>

### Procedure DeleteGuest

```plaintext
Begin
   Extract Request Method
   If Method is POST
       Request for User Input
   Else
       Display Error Message
   End If
End
```

### Procedure DeleteSTerm

```plaintext
Begin
   Extract Request Method
   If Method is POST
       Extract Name of Visitor
       Establish DB Connection
       Delete Comments from DB
       Display Confirmation Message
       Close DB Connection
   Else
       Display Error Message
   End If
End
```

### Procedure DeleteStat

```plaintext
Begin
End
```
Begin
   Extract Request Method
   If Method is POST
       Request for User Input
   Else
       Display Error Message
   End If
End

### 22. Function Name: DeleteComment

<table>
<thead>
<tr>
<th>Where Used</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit</td>
<td>Delete Browser Comments in Database</td>
</tr>
</tbody>
</table>

**Subitems**

---

Procedure DeleteComment
Begin
   Extract Request Method
   If Method is POST
       Extract Term
       Establish DB Connection
       Delete Term from Statistics DB Table
       Display Confirmation Message
       Close DB Connection
   Else
       Display Error Message
   End If
End

### 23. Function Name: Contact

<table>
<thead>
<tr>
<th>Where Used</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>WUED</td>
<td>Insert Visitor's Information in Database</td>
</tr>
</tbody>
</table>

**Subitems**

---

Procedure Contact
Begin
   Unless there is browser's Message
       Display Error Message
   Else
       Extract Name, Address, Message
       Establish DB Connection
       Insert Data in Guest DB Table
       Display Confirmation Message
End
Close DB Connection
End If
End

<table>
<thead>
<tr>
<th>24. Function Name</th>
<th>Guestbook</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Display Browser's Comments</td>
</tr>
<tr>
<td>Subitems</td>
<td>Note</td>
</tr>
</tbody>
</table>

Procedure Guestbook
Begin
   Establish DB Connection
   Retrieve Comments from Guest DB Connection
   Display Comments
   Close DB Connection
End

<table>
<thead>
<tr>
<th>25. Function Name</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Display Statistical Results of Browser Access</td>
</tr>
<tr>
<td>Subitems</td>
<td>Note</td>
</tr>
</tbody>
</table>

Procedure Statistics
Begin
   Establish DB Connection
   Retrieve the Most Frequently Requested Term
   Retrieve the Most Frequently Missed Term
   Display the above Terms
   Close DB Connection
End

<table>
<thead>
<tr>
<th>26. Function Name</th>
<th>Submit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where Used</td>
<td>WUED</td>
</tr>
<tr>
<td>Purpose</td>
<td>Insert Visitor's Submission in Database</td>
</tr>
<tr>
<td>Subitems</td>
<td>Note</td>
</tr>
</tbody>
</table>

Procedure Submit
Begin
   Unless Term or Definition is extracted
      Display Error Message

58
Else
  Extract Term and Definition
  Establish DB Connection
  Insert Data in Statistics DB Table
  Display Confirmation Message
  Close DB Connection
End If
End
CHAPTER 3 PRODUCT TEST

This chapter focuses verification effort on the software design. Using the detail design description as a guide, important control paths of each Graphical User Interface are tested.

<table>
<thead>
<tr>
<th>GUI</th>
<th>Input</th>
<th>Desired Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu</td>
<td>Click WUED Icon</td>
<td>Display Term and Definition of WUED X</td>
</tr>
<tr>
<td></td>
<td>Click Home Icon</td>
<td>Display Home Page X</td>
</tr>
<tr>
<td></td>
<td>Click History Icon</td>
<td>Display Previously Accessed Terms X</td>
</tr>
<tr>
<td></td>
<td>Click Random Icon</td>
<td>Display Term and Definition Selected Randomly X</td>
</tr>
<tr>
<td></td>
<td>Click Contents Icon</td>
<td>Display Contents Page X</td>
</tr>
<tr>
<td></td>
<td>Click Edit Icon</td>
<td>Display Edit Page X</td>
</tr>
<tr>
<td></td>
<td>Click Help Icon</td>
<td>Display Help Page X</td>
</tr>
<tr>
<td></td>
<td>Type Term, Select Full and Click Search Icon</td>
<td>Display List of Terms Found X</td>
</tr>
<tr>
<td></td>
<td>Type Term, Select Exact and Click Search Icon</td>
<td>Display Term and Definition X</td>
</tr>
<tr>
<td>Home</td>
<td>Click Hypertext Link</td>
<td>Display Related Page X</td>
</tr>
<tr>
<td></td>
<td>Click Entry</td>
<td>Display Term and Definition X</td>
</tr>
<tr>
<td></td>
<td>Click Contact Icon</td>
<td>Display Contact Page X</td>
</tr>
<tr>
<td></td>
<td>Click About Icon</td>
<td>Display About Page X</td>
</tr>
<tr>
<td></td>
<td>Click Submit Icon</td>
<td>Display Submit Page X</td>
</tr>
<tr>
<td></td>
<td>Click Statistics Icon</td>
<td>Display Statistics Page X</td>
</tr>
<tr>
<td>Random</td>
<td>Click Hypertext Links</td>
<td>Display Related Page X</td>
</tr>
<tr>
<td></td>
<td>Click Entry</td>
<td>Display Term and Definition X</td>
</tr>
<tr>
<td>History</td>
<td>Click Hypertext Links</td>
<td>Display Related Page X</td>
</tr>
<tr>
<td></td>
<td>Click Entry</td>
<td>Display Term and Definition X</td>
</tr>
<tr>
<td>Contents</td>
<td>Click Hypertext Links</td>
<td>Display Related Page X</td>
</tr>
<tr>
<td>Help</td>
<td>Display List of Term in Alphabetical Order</td>
<td>X</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------------------------------------------</td>
<td>---</td>
</tr>
<tr>
<td>Edit</td>
<td>Display Related Page</td>
<td>X</td>
</tr>
<tr>
<td>Edit Menu</td>
<td>Display Edit Menu</td>
<td>X</td>
</tr>
<tr>
<td>Import</td>
<td>Display Import Page with Term and Definition Retrieved</td>
<td>X</td>
</tr>
<tr>
<td>Import from Data File</td>
<td>Display Import from Data File Page</td>
<td>X</td>
</tr>
<tr>
<td>Modify</td>
<td>Display Import from Data File Page</td>
<td>X</td>
</tr>
<tr>
<td>Modify</td>
<td>Display Insert Page</td>
<td>X</td>
</tr>
<tr>
<td>Delete</td>
<td>Display Delete Page</td>
<td>X</td>
</tr>
<tr>
<td>Delete Comments</td>
<td>Display Modify Page</td>
<td>X</td>
</tr>
<tr>
<td>Delete Statistics Term</td>
<td>Display Delete Statistics Term Page</td>
<td>X</td>
</tr>
<tr>
<td>Delete Comments</td>
<td>Display Delete Comments Page</td>
<td>X</td>
</tr>
<tr>
<td>Import</td>
<td>Import Data into Database</td>
<td>X</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete Data from Database</td>
<td>X</td>
</tr>
<tr>
<td>Insert</td>
<td>Insert Data into Database</td>
<td>X</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete Data from Database</td>
<td>X</td>
</tr>
<tr>
<td>Modify</td>
<td>Display Modify Data Page</td>
<td>X</td>
</tr>
<tr>
<td>Modify Data</td>
<td>Modify Data in Database</td>
<td>X</td>
</tr>
<tr>
<td>Delete Statistics Term</td>
<td>Delete Data from Database</td>
<td>X</td>
</tr>
<tr>
<td>Delete Comments</td>
<td>Delete Comments from Database</td>
<td>X</td>
</tr>
<tr>
<td>About</td>
<td>Display Related Page</td>
<td>X</td>
</tr>
<tr>
<td>Statistics</td>
<td>Click Hypertext Links</td>
<td>Display Related Page</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Click Entry</td>
<td></td>
<td>Display Term and Definition</td>
</tr>
</tbody>
</table>

| Submit              | Type Name and Definition and Click Submit | Insert Data into Database | X |

<table>
<thead>
<tr>
<th>Contact</th>
<th>Type Name, Address and Comments and Click Submit</th>
<th>Insert Browsers’ Submission into Database</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Click Guestbook Button</td>
<td></td>
<td>Display Guest Book</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 8. Product Test
CHAPTER 4 PERFORMANCE TEST

The performance test was conducted by measuring the response time of every search execution. Execution time of each search engine (Full-text Search, Exact-match Search) was measured to obtain performance information.

Initially, FOLDOC dictionary which size is 4.25 MB was loaded in WUED, and then the data size was incremented by 4.25 MB in each step. The keyword was artificially altered by appending numbers up to a max. of 29.8 MB. The test subjects (entries) are randomly picked as follows,

1. tilde  2. era  3. Network
   4. CASE  5. WWW  6. CGI
   7. Perl   8. HTML  9. DOS

For each entry, the average of three runs was computed and tabulated. The following table shows measurements of CPU time with different data sizes for each entry.

<table>
<thead>
<tr>
<th>Measurements</th>
<th>8.5 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Size</td>
<td>8.5 MB</td>
</tr>
<tr>
<td>No. 1 2 3 4 5 6 7 8 9</td>
<td>AVG</td>
</tr>
<tr>
<td>Full text</td>
<td>AVG</td>
</tr>
<tr>
<td>AVG</td>
<td>AVG</td>
</tr>
<tr>
<td>Exact match</td>
<td>AVG</td>
</tr>
</tbody>
</table>

Table 9. Performance Test: Data Size 8.5 MB
### Table 10. Performance Test: Data Size 12.8 MB

<table>
<thead>
<tr>
<th>Data Size</th>
<th>12.8 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>Full text</td>
<td>0.11 0.08 0.99 0.38 0.17 0.16 0.43 0.4 0.18</td>
</tr>
<tr>
<td>text</td>
<td>0.11 0.08 0.97 0.34 0.19 0.12 0.45 0.41 0.16</td>
</tr>
<tr>
<td>AVG</td>
<td>0.11 0.07 0.99 0.37 0.17 0.14 0.43 0.4 0.17</td>
</tr>
<tr>
<td>Exact match</td>
<td>0.09 0.06 0.10 0.07 0.07 0.08 0.11 0.06 0.04</td>
</tr>
<tr>
<td>match</td>
<td>0.07 0.04 0.08 0.04 0.05 0.05 0.10 0.05 0.04</td>
</tr>
<tr>
<td>AVG</td>
<td>0.07 0.06 0.08 0.06 0.06 0.06 0.11 0.06 0.05</td>
</tr>
</tbody>
</table>

### Table 11. Performance Test: Data Size 17.0 MB

<table>
<thead>
<tr>
<th>Data Size</th>
<th>17.0 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>Full text</td>
<td>0.13 0.08 1.18 0.54 0.17 0.20 0.51 0.48 0.19</td>
</tr>
<tr>
<td>text</td>
<td>0.12 0.06 1.20 0.45 0.21 0.18 0.59 0.47 0.24</td>
</tr>
<tr>
<td>AVG</td>
<td>0.13 0.06 1.24 0.49 0.19 0.18 0.56 0.47 0.21</td>
</tr>
<tr>
<td>Exact match</td>
<td>0.11 0.05 0.09 0.06 0.05 0.06 0.11 0.05 0.08</td>
</tr>
<tr>
<td>match</td>
<td>0.07 0.08 0.08 0.05 0.08 0.08 0.10 0.06 0.06</td>
</tr>
<tr>
<td>AVG</td>
<td>0.08 0.06 0.07 0.07 0.06 0.07 0.11 0.06 0.07</td>
</tr>
<tr>
<td>Data Size</td>
<td>21.3 MB</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>No.</td>
<td>1</td>
</tr>
<tr>
<td>Full text</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td>0.15</td>
</tr>
<tr>
<td></td>
<td>0.16</td>
</tr>
<tr>
<td>AVG</td>
<td>0.15</td>
</tr>
<tr>
<td>Exact match</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>0.08</td>
</tr>
<tr>
<td>AVG</td>
<td>0.08</td>
</tr>
</tbody>
</table>

Table 12. Performance Test: Data Size 21.3 MB

<table>
<thead>
<tr>
<th>Data Size</th>
<th>25.5 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>1</td>
</tr>
<tr>
<td>Full text</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td>0.15</td>
</tr>
<tr>
<td>AVG</td>
<td>0.14</td>
</tr>
<tr>
<td>Exact match</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>0.07</td>
</tr>
<tr>
<td>AVG</td>
<td>0.08</td>
</tr>
</tbody>
</table>

Table 13. Performance Test: Data Size 25.5 MB
<table>
<thead>
<tr>
<th>Data Size</th>
<th>29.8 MB</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No.</strong></td>
<td>1</td>
</tr>
<tr>
<td>Full</td>
<td>0.19</td>
</tr>
<tr>
<td>text</td>
<td>0.17</td>
</tr>
<tr>
<td></td>
<td>0.15</td>
</tr>
<tr>
<td><strong>AVG</strong></td>
<td>0.17</td>
</tr>
<tr>
<td>Exact</td>
<td>0.06</td>
</tr>
<tr>
<td>match</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>0.10</td>
</tr>
<tr>
<td><strong>AVG</strong></td>
<td>0.08</td>
</tr>
</tbody>
</table>

Table 14. Performance Test: Data Size 29.8 MB
Comparison

The following two comparison tables (Table 15, Table 16) show CPU execution times measured with different data sizes for each entry. The execution time for Full-text Search gradually increases with incrementing data size. The execution time for Exact-match Search remains constant.

<table>
<thead>
<tr>
<th>No.</th>
<th>Subject</th>
<th>8.5MB</th>
<th>12.8MB</th>
<th>17MB</th>
<th>21.3MB</th>
<th>25.5MB</th>
<th>29.8MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>tilde</td>
<td>0.09</td>
<td>0.11</td>
<td>0.13</td>
<td>0.15</td>
<td>0.14</td>
<td>0.17</td>
</tr>
<tr>
<td>2</td>
<td>era</td>
<td>0.07</td>
<td>0.07</td>
<td>0.06</td>
<td>0.04</td>
<td>0.05</td>
<td>0.07</td>
</tr>
<tr>
<td>3</td>
<td>network</td>
<td>0.66</td>
<td>0.99</td>
<td>1.24</td>
<td>1.61</td>
<td>1.91</td>
<td>2.21</td>
</tr>
<tr>
<td>4</td>
<td>CASE</td>
<td>0.26</td>
<td>0.37</td>
<td>0.49</td>
<td>0.56</td>
<td>0.65</td>
<td>0.75</td>
</tr>
<tr>
<td>5</td>
<td>WWW</td>
<td>0.13</td>
<td>0.17</td>
<td>0.19</td>
<td>0.23</td>
<td>0.28</td>
<td>0.31</td>
</tr>
<tr>
<td>6</td>
<td>CGI</td>
<td>0.11</td>
<td>0.14</td>
<td>0.18</td>
<td>0.2</td>
<td>0.22</td>
<td>0.27</td>
</tr>
<tr>
<td>7</td>
<td>perl</td>
<td>0.31</td>
<td>0.43</td>
<td>0.56</td>
<td>0.67</td>
<td>0.82</td>
<td>0.94</td>
</tr>
<tr>
<td>8</td>
<td>HTML</td>
<td>0.25</td>
<td>0.4</td>
<td>0.47</td>
<td>0.62</td>
<td>0.72</td>
<td>0.81</td>
</tr>
<tr>
<td>9</td>
<td>DOS</td>
<td>0.13</td>
<td>0.17</td>
<td>0.21</td>
<td>0.24</td>
<td>0.29</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td>AVG</td>
<td>0.22</td>
<td>0.32</td>
<td>0.39</td>
<td>0.48</td>
<td>0.56</td>
<td>0.65</td>
</tr>
</tbody>
</table>

Table 15. Performance Comparison for Full-text Search

<table>
<thead>
<tr>
<th>No.</th>
<th>Subject</th>
<th>8.5MB</th>
<th>12.8MB</th>
<th>17MB</th>
<th>21.3MB</th>
<th>25.5MB</th>
<th>29.8MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>tilde</td>
<td>0.1</td>
<td>0.07</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>2</td>
<td>era</td>
<td>0.04</td>
<td>0.06</td>
<td>0.06</td>
<td>0.04</td>
<td>0.06</td>
<td>0.06</td>
</tr>
<tr>
<td>3</td>
<td>network</td>
<td>0.07</td>
<td>0.08</td>
<td>0.07</td>
<td>0.07</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>4</td>
<td>CASE</td>
<td>0.07</td>
<td>0.06</td>
<td>0.07</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
</tr>
<tr>
<td>5</td>
<td>WWW</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.05</td>
<td>0.06</td>
<td>0.07</td>
</tr>
<tr>
<td>6</td>
<td>CGI</td>
<td>0.09</td>
<td>0.06</td>
<td>0.07</td>
<td>0.05</td>
<td>0.06</td>
<td>0.07</td>
</tr>
<tr>
<td>7</td>
<td>perl</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
<td>0.11</td>
</tr>
<tr>
<td>8</td>
<td>HTML</td>
<td>0.05</td>
<td>0.06</td>
<td>0.06</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>9</td>
<td>DOS</td>
<td>0.07</td>
<td>0.05</td>
<td>0.07</td>
<td>0.06</td>
<td>0.05</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>AVG</td>
<td>0.07</td>
<td>0.07</td>
<td>0.07</td>
<td>0.06</td>
<td>0.07</td>
<td>0.07</td>
</tr>
</tbody>
</table>

Table 15. Performance Comparison for Exact-match Search
Figure 22. Performance Comparison Graph

The result of experiments for Full-text Search was expected. The execution time of Exact-match Search remains constant due to hashing of keys that is inherent in the database.
APPENDIX A SOURCE CODE OF WUED

#!/usr/bin/perl -w

# File Name : wued.cgi
# Description : Main cgi. Used to direct proper function call.

$REQUIRE_DIR = '../lib';
push(0INC, $REQUIRE_DIR) if $REQUIRE_DIR;

use CGI qw(:standard);
use DBI;

# brings the library subroutines
require 'config.pl';
require 'html.pl';
require 'search.pl';
require 'random.pl';
require 'content.pl';
require 'contact.pl';
require 'error.pl';
require 'parse_form.pl';
require 'submit.pl';
require 'edit.pl';
require 'statistics.pl';
require 'history.pl';

&configure_var; # declare global variables

&parse_form_data(*FORM);

my $command = $FORM{'command'}; # extract commands
if ($command) {
  &$command;
} else {
  &home;
}

$html_footer;

exit (0);
File Name : contact.pl
Description : Used to handle browsers' comments
Function : contact - Stores comments in DBMS
guestbook - Display comments from DBMS

sub contact { 
if (!$FORM{message}) { 
$html___header("incomplete");
print "<BR><P>Your information is not complete. Please try again.";
} else { 
# extract browsers' information
my $name = $FORM{name};
my $address = $FORM{address};
my $message = $FORM{message};

my $drh = DBI->install_driver($DBDRIVER);
my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
$message s/'/(quote)/g;
# insert into database
my $rc = $dbh->do("insert into $DBGTABLE
values ('$name', '$address', '$message')");
# display a thank message
if ($rc) {
$html___header("thank you");
print "<BR><P>Thanks for comments. Following is the
comments that you just submitted. Your comments will
be reviewed and inserted into guestbook soon.";
print 
"<OL><LI>Your Name : $name <LI>Address : $address
<LI>Message : $message</OL><BR>";
} else { 
return__error(500, "DB Error",
"Connection to DB was not established");
}
$dbh->disconnect;
}

sub guestbook { 
# display Guestbook
}
my @messages;
my $drh = DBI->install_driver($DBDRIVER);
my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

my $cursor = $dbh->prepare("select name, address, message from $DBGTABLE");
$cursor->execute;

$html_header("guestbook");
print "<BR>

# retrieve messages
while (@messages = $cursor->fetchrow) {
}

# if no comments available then display messages
unless (@message) {
    print "<P>There are no comments available at this time</P>";
}

$cursor->finish;
$dbh->disconnect;

}
sub content {
    my $term;
    my $count = 0;
    my $drh = DBI->install_driver($DBDRIVER);
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

    # display the next 50 entries
    # this can be done by browser's click on next 50 button
    # extract the last entry from previous list of entries
    if ($term = $FORM{'term'}) {
        my $cursor = $dbh->prepare("select term from $DBVTABLE where term > " $term ' order by term; ");
        $cursor->execute;
        &html_header("c,ontents");
        print "<TABLE cellspacing = 10">
        my $count_term = 0;
        while (($count_term < 50) and ($contents = $cursor->fetchrow)) {
            if ($count%3 == 0) { print "<TR>");
            print "<TD><IMG SRC="$MAIN___DIR/image/wued/rarrow.gif">", &anchor($contents);
            $count = $count + 1;
            $count_term = $count_term + 1;
        }
    }

    # display the first 50 entries starting with the char selected by browser
    else {
        # extract the character
        my $char = $FORM['char'];
        my $cursor = $dbh->prepare("select term from $DBVTABLE where term like "$char" order by term; ");
        $cursor->execute;
        &html_header("c,ontents of "$char");
        print "<TABLE WIDTH=100% cellspacing = 10">
        my $count_term = 0;
        while (($count_term < 50) and ($contents = $cursor->fetchrow)) {
            if ($count%3 == 0) { print "<TR>");
            print "<TD WIDTH=30%>"; }
<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif">
$anchor($contents);
$count = $count + 1;
$count_term = $count_term + 1;

}
}

print "</TABLE>
print "<CENTER><FORM ACTION="$CGI_DIR/wued.cgi" METHOD = 
"POST">
print "<INPUT TYPE="hidden" NAME="command" VALUE="content">
print "<INPUT TYPE="hidden" NAME="term" VALUE=$contents>
print "<INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/next.gif" 
border=0></FORM></CENTER><BR>

$cursor->finish;
$dbh->disconnect;

1;
# File Name : edition.pl
# Description : Used to edit DBMS

# Function:
#   edit_process - Direct edit options
#   import - Retrieve import data from DBMS
#   import_proceed - Perform import data to DBMS
#   insert - Request data for insertion
#   insert_proceed - Perform insertion
#   delete - Request term for deletion
#   delete_proceed - Perform deletion
#   modify - Request term for modification
#   modify_retrieve - Retrieve data for modification
#   modify_proceed - Perform modification
#   deleteGuest - Request a name for deletion
#   deleteComment - Perform deletion of comments
#   deleteStat - Request a term for deletion
#   deleteSTerm - Perform deletion (statistics)

sub edit_proceed {
    my $method = $ENV{'REQUEST_METHOD'};

    if ($method eq "POST") {
        # extract edit option
        my $select = $FORM{'edit_select'};

        # call proper subroutines
        if ($select) {
            &$select;
        } else {
            &html_header("i,complete");
            print "<BR><P>You have not selected edit option. Please click
              Menu to go back to Main Edit Menu.</P>";
            &edit_footer;
        }
    } else {
        &return_error (500, "Wued Error",
            "Wued does not support your request");
    }
}

sub import {
    my $method = $ENV{'REQUEST_METHOD'};

    if ($method eq "POST") {

my $drh = DBI->install_driver($DBDRIVER);
my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

# check if there is Web submission
my $cursor = $dbh->prepare("select * from $DBSUBTABLE");
$cursor->execute;

my @data_field = $cursor->fetchrow;
$cursor->finish;
$dbh->disconnect;

# if there is Web submission then display
if (@data_field) {

$html_header("import");

print qq!
<BR><P>Following information is provided by a visitor. Modify the
information if necessary, mark one of import options, then
click button on the right.</P>

<form action="$CGI_DIR/wued.cgi" method="POST">
<table border=0 align=center cellpadding=10>
<tr valign=middle>
<td><h2 class="miniS">Term :</h2></td>
<td><input type="text" name="term" size=30 value="$data_field[0]"></td>
</tr>
<tr valign=middle>
<td><h2 class="miniS">Keywords :</h2></td>
<td><input type="text" name="keywords" size=30></td>
</tr>
<tr valign=top>
<td><h2 class="miniS">Definition :</h2></td>
<td><textarea rows=4 cols=30 name="definition">: $data_field[1]</textarea></td>
</tr></table>
<table border=0 align=center cellpadding=10>
<tr valign=middle>
<td><input type="radio" name="import_select" value="insert"></td>
<td><input type="image" src="$MAIN_DIR/image/wued/insert.gif" width=70 height=20 border=0></td>
<td><input type="radio" name="import_select" value="delete"></td>
<td><input type="image" src="$MAIN_DIR/image/wued/delete.gif" width=70 height=20 border=0></td>
</tr>
<tr align=center valign=middle>
<td colspan=4></td>
</tr>
</table>
</form>!

} else {

$html_header("empty");

}
print "<BR><P>Data Storage is empty. There is no data available in DBMS to import.</P>";

&edit_footer;
}
} else {

$return_error (500, "Wued Error",
               "Wued does not support your request");
}

# delete or insert Web submission
sub import_proceed {
    my $method = $ENV{'REQUEST_METHOD'};
    if ($method eq "POST") {
        my $term_up;
        # extract import command (insert or delete)
        my $toDo = $FORM{'import_select'};

        # delete
        if ($toDo eq "delete") {
            my $term = $FORM{'term'};
            $term_up = uc $term;
            unless ($term_up eq $term) {
                $term =~ tr/A-Z/a-z/;
            }

            my $drh = DBI->install_driver($DBDRIVER);
            my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

            my $rc = $dbh->do("delete from $DBSUBTABLE where term = '$term'");

            $dbh->disconnect;
            if ($rc != "0EO") {
                &html_header("complete");
                print "<BR><P>Information has been deleted from DBMS!</P>";
                &edit_footer;
            } else {

    76
return_error(500, "DB Error", "Connection to DB was not established");
}

# insert
} elsif ($todo eq "insert") {

if (!defined($FORM{'term'})) {
    &html_header("incomplete");
    print "<BR><P>The information is not complete. Please try again.</P>";
    &edir_footer;
} else {
    my $rc;
    my $term = $FORM{'term'};
    my $def = $FORM{'definition'};
    my $key = $FORM{'keywords'};
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

    $term_up = uc $term;
    unless ($term_up eq $term) {
        $term =~ tr/A-Z/a-z/;
    }

    $key =~ tr/A-Z/a-z/;

    $rc = $dbh->do("insert into $DBTABLE 
      values ( '$term', '$key', '$def')");

    $rc = $dbh->do("delete from $DBSUBTABLE where term = '$term'");

    if ($rc != "0EO") {
        &html_header("complete");
        print "<BR><P>Following information has been inserted.</P>";
        print "<OL>";
        print "<LI>Term : ", $term;
        print "<LI>Key : ", $key;
        print "<LI>Definition : ", $def;
        print "</OL>";
        &edir_footer;
    } else {

    }
}
return_error(500, "DB Error",
    "Connection to DB was not established");
}
}
} else {
    $html_header("incomplete");
    print "<BR><BR> You have not selected import option. Please go back and choose your request.";
    
    $edit_footer;

} else {

    return_error (500, "Wued Error",
        "Wued does not support your request");
}
}

# get the name of data file
sub import_file {
    my $method = $ENV{'REQUEST_METHOD'};
    if ($method eq "POST") {
        $html_header("import from a data file");
        print qq!
            <FORM ACTION="$CGI_DIR/wued.cgi" METHOD="POST">
                <INPUT TYPE="hidden" NAME="command" VALUE="import_file_proceed">
                <P>You can import data from a data file into database. The data file should be in a data subdirectory of WUED.</P>
                <P>Please enter a name of file, then click submit or press enter.</P>
                <TABLE border=0 align=center>
                    <TR><TD>File :</TD><TD><INPUT TYPE="text" NAME="file" SIZE=30>
                    <TD><INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/submitb.gif" VALUE="submit" border=0>
                </TABLE>
            </FORM>!
    } else {
        return_error (500, "Wued Error",
            "Wued does not support your request");
    }
}

# load data from data file
sub import_file_proceed {
    my $method = $ENV{'REQUEST_METHOD'};
    if ($method eq "POST") {
        my $unlock = 8;
        my $line;
    } else {
my $term;
my $term_up;
my $def;
my $ele;
my $rc;
my $input_file = $FORM{'file'};
my $path = "wued/admin/$input_file";

if (open(FILE, "<$path")) {
    my $drh = DBI->install_driver($DBDRIVER);
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

    my $exclusive_lock = 2;
    flock (FILE, $exclusive_lock);

    $term = <FILE>;

    # retrieve and insert data into dictionary untill reached end of file until (eof FILE) {
        $def = "";
        $keys = "";

        # retrieve definition - matches tab char or new line char
        while ( $line = <FILE>, ($line =~ /[^\t\n]/)) {
            $line =~ s/^\t//;
            $def = $def . $line;
        }

        $term =~ s/[^\t\n]/g;
        $term =~ s/\n/g;

        $term_up = uc $term;

        # if entry is all capitalized letters then check for entry with low case
        if ($term_up eq $term) {
            $term_up =~ tr/A-Z/a-z/;
            $keys = $keys . "($term)($term_up)";

            my $cursor = $dbh->prepare("select definition from $DBTABLE
                where term = '$term_up'");
            $cursor->execute;
            my $lowDef = $cursor->fetchrow;
            $cursor->finish;
        }
# if entry with low case exists then attach cross-ref
if ($lowDef) {
    $lowDef = $lowDef . "\n\nSee also \{$term\}\n";
    $rc = $dbh->do("update $DBTABLE set
definition ='$lowDef' where term = '$term_up'");
    $def = $def . "\n\nSee also \{$term_up\}\n";
}
} else {

    $term =~ tr/A-Z/a-z/;
    $keys = $keys . "\{$term\}";

}

# set up keywords
my @term_list = ($def =~ /{[^{}]*}/g);
foreach $element (@term_list) {
    unless (($element =~ \(\)/) or ($element =~ /:/)) {
        $term_up = uc $element;
        unless ($term_up eq $element) {
            $element =~ tr/A-Z/a-z/;
        }
        $element = "\{$element\}";
        unless ($keys =~ /\$element/) {$keys = $keys . $element; }
    }
}

$keys =~ s/'//g;
$def =~ s/'/(quote)/g;
$rc = $dbh->do("insert into $DBTABLE
values ('$term', '$keys', '$def')");

$term = $line;
}
close(FILE);
flock (FILE, $unlock);
$dbh->disconnect;
&html_header("import completed");
print "<BR><P>Data in \$input_file are imported into database"
completely";

) else {

&html__header("n,ot found");
print "<BR>&lt;P&gt;Cannot open a file for input. Please check if the
file exists in the directory.&lt;/P&gt;";
print $input_file;
}
} else {

$return_error (500, "Wued Error",
"Wued does not support your request");
}
}

# display insert page
sub insert {

my $method = $ENV{'REQUEST_METHOD'};
if ($method eq "POST") {

&html__header("i,nsert");

print qq!
&lt;FORM ACTION="$CGI_DIR/wued.cgi" METHOD= "POST">
&lt;INPUT TYPE="hidden" NAME="command" VALUE="insert_proceed">
&lt;P&gt;Please enter a term, keywords, and definition of the term.
You can enter more than one keyword, and each key word should
be surrounded by curry brakets (e.g. {key}). Please note that
you do not need to add the term as your keyword.&lt;/P&gt;
&lt;TABLE border=0 align=center&gt;
&lt;TR&gt;&lt;TD&gt;Term :&lt;/TD&gt;&lt;TD&gt;&lt;/TD&gt;&lt;/TR&gt;
&lt;TR&gt;&lt;TD&gt;&lt;INPUT TYPE="text" NAME="term" SIZE=40&gt;&lt;/TD&gt;&lt;/TR&gt;
&lt;TR&gt;&lt;TD&gt;&lt;INPUT TYPE="text" NAME="key" SIZE=40&gt;&lt;/TD&gt;&lt;/TR&gt;
&lt;TR&gt;&lt;TD&gt;&lt;INPUT TYPE="text" NAME="key" SIZE=40&gt;&lt;/TD&gt;&lt;/TR&gt;
&lt;TR&gt;&lt;TD&gt;&lt;TEXTAREA ROWS=5 COLS=40 NAME="definition">
&lt;/TEXTAREA&gt;&lt;/TD&gt;&lt;/TR&gt;
&lt;/TABLE&gt;&lt;BR&gt;
&lt;CENTER&gt;&lt;INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/insert.gif"
border=0&gt;&lt;/CENTER&gt;
&lt;/FORM&gt;!
)
} else {

$return_error (500, "Wued Error",
"Wued does not support your request");
}
}

# extract and insert data into database
sub insert_proceed {

81
my $method = $ENV{'REQUEST_METHOD'};
if ($method eq "POST") {

    my $term_up;
    # check for missing information
    if (!($FORM{'term'}) or !($FORM{'definition'})) {
        &html_header("incomplete");
        print "<BR><P>Your information is not complete. Please try
gain.</P>");
    } else {
    # if every information is provided
    my $rc;
    my $term = $FORM{ 'term'};
    my $key = $FORM{ 'key'};
    my $def = $FORM{ 'definition'};
    # connecting to database
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

    $term_up = uc $term;
    unless ($term_up eq $term) {
        $term =~ tr/A-Z/a-z/;
    }
    # insertion to wued
    $key = &convertKey($key);
    $key = $key . $term . $def;
    $def =~ s/./(quote)/;
    $rc = $dbh->do("insert into $DBTABLE
    values ('$term', '$key', '$def')");
    if ($rc) {
        &html_header("complete");
        print qq!
        <BR><P>Following information was successfully inserted.</P><BR>
        <OL>
        <LI>Term : $term
        <LI>key : $key
        <LI>Definition : $def
        </OL>!
    } else {
        return_error(500, "DB Error",
                     "Connection to DB was not established");
    }
}
$dbh->disconnect;
&edit_footer;
} else {
&return_error (500, "Wued Error",
    "Wued does not support your request");
}

# display delete page
sub delete {

my $method = $ENV{'REQUEST_METHOD'};
if ($method eq "POST") {

$html_header("d,elete");

print qq!
    <FORM ACTION="$CGI_DIR/wued.cgi" METHOD="POST">
    <P>Please enter a term for deletion, then click submit or press enter.<P>
    <TABLE border=0 align=center>
    <TR><TD>Term :
    <TD><INPUT TYPE="text" NAME="term" SIZE=30>
    <TD><INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/submitb.gif" VALUE="submit" border=0>
    </TABLE>
    </FORM>!

} else {

&return_error (500, "Wued Error",
    "Wued does not support your request");
}

# extract entry and delete it from database
sub delete_proceed {
my $method = $ENV{'REQUEST_METHOD'};
if ($method eq "POST") {

my $term = $FORM{'term'};
if ($term) {

my $drh = DBI->install_driver($DBDRIVER);
my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
my $rc = $dbh->do("delete from $DBTABLE where term = '$term'");
$dbh->disconnect;


# return value 0E0 - data not found in database
if ($rc ! eq "0E0") {
    &html_header("complete");
    print "<BR><P>Deletion of $term has been made successfully.
    </P>";
} else {
    return_error(500, "DB Error",
        "Your information is not found at database or
        Connection to DB was not established");
}
} else {
    &html_header("incomplete");
    print "<BR><P>Your request is not complete. Please try
    again.</P>";
}
&edit_footer;
} else {
    &return_error (500, "Wued Error",
        "Wued does not support your request");
}

# display modify page
sub modify {
    my $method = $ENV{'REQUEST_METHOD'};
    if ($method eq "POST") {
        &html_header("modify");
        print qq!
        <FORM ACTION="$CGI_DIR/wued.cgi" METHOD="POST">
        <P>Please enter a term for data modification</P>
        <TABLE border=0 align=center cellspacing=10 cellpadding=0>
        <TR><TD><B>Term :</B>
        <TD><INPUT TYPE="text" NAME="term" SIZE=30>
        <TD><INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/submitb.gif"
            border=0>
        </TABLE>
        </FORM>!
    } else {

}
&return_error (500, "Wued Error", "Wued does not support your request");
}

# retrieve and display entry and definition to allow modification
sub modify_retrieve {
  my $method = $ENV{'REQUEST_METHOD'};
  if ($method eq "POST") {
    # extract entry
    my $key_term = $FORM{'term'};
    if ($key_term) {
      my $drh = DBI->install_driver($DBDRIVER);
      my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
      my $cursor = $dbh->prepare("select * from $DBTABLE where term = " . "'$key_term'";
      $cursor->execute;
      my @data_field = $cursor->fetchrow;
      $cursor->finish;
      $dbh->disconnect;
      # display information
      if (@data_field) {
        &html_header("modify");
        print qq!
          <FORM ACTION="$CGI_DIR/wued.cgi" METHOD= "POST">
            <INPUT TYPE="hidden" NAME="command" VALUE="modify_proceed">
            <INPUT TYPE="hidden" NAME="keyTerm" VALUE="$key_term">
            Modify the following information, then click submit or press enter.
            <TABLE border=0 align=center cellpadding=10>
              <TR valign=middle>
                <TD><B>Term :</B> <TD><INPUT TYPE="text" NAME="term" VALUE="" . "$data_field[0]">
              <TR valign=middle>
                <TD><B>Keyword :</B> <TD><INPUT TYPE="text" NAME="key" VALUE="" . "$data_field[1]">
              <TD valign=top><B>Definition :</B> <TEXTAREA ROWS=4 COLS=30 NAME="definition">" . "$data_field[2]"</TEXTAREA>
            </TABLE>
          </FORM>
        </form>
      }
    }
  }
}

85
modify data in database

sub modify_proceed {
  my $method = $ENV{'REQUEST_METHOD'};

  if ($method eq "POST") {
    if (!defined $FORM{'term'} or !defined $FORM{'definition'}) {
      html_header("incomplete");
      print "<BR><P>Your information is not complete. Please try again.</P>";
    } else {
      my $rc;
      my $term = $FORM{'term'};
      my $key = $FORM{'key'};
      my $def = $FORM{'definition'};
      my $keyTerm = $FORM{'keyTerm'};

      my $dbh = DBI->install_driver($DBDRIVER);
      my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

      ...
my $term_up = uc $term;

unless ($term_up eq $term) {
    $term =~ tr/A-Z/a-z/;
}

$def =~ s/'/(quote)/;
$key = &convertKey($key);

$term_up = uc $keyTomodify;

unless ($term_up eq $keyTomodify) {
    $keyTomodify =~ tr/A-Z/a-z/;
}

$rc = $dbh->do("update $DBTABLE set term = '$term', key = '$key',
definition ='$def' where term = '$keyTomodify'");

$dbh->disconnect;

if ($rc != "0EO") {
    &html__header("c^omplete");
    print qq!
    <BR><P>Modification has been made successfully.<br>
    Following information is new data.</P>
    <OL>
    <LI>Term : $term
    <LI>Key : $key
    <LI>Definition : $def
    </OL>!
} else {
    return_error(500, "DB Error",
    "Data provided is not found at database or
    Connection to DB was not established");
}

&edit_footer;
} else {
    &return_error (500, "Wued Error",
    "Wued does not support your request");
}

# display delete comments page
sub deleteGuest {
    my $method = $ENV{'REQUEST_METHOD'};
if ($method eq "POST") {

    &html_header("d,elete comments");
    print qq!<BR><BR>
    <P>Please enter the exact name of person (case sensitive):</P>
    <FORM ACTION="$CGI__DIR/wued.cgi" METHOD="POST">
    <TABLE border=0 align=center>
    <TR valign=middle>
    <TD><INPUT TYPE="hidden" NAME="command" VALUE="deleteComment">
    <TD><INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/submitb.gif" VALUE="submit" border=0>
    </TABLE>
    </FORM>!

} else {

    &return_error (500, "Wued Error",
                   "Wued does not support your request");
}

# delete comments from database
sub deleteComment {

    my $method = $ENV{'REQUEST_METHOD'};

    if ($method eq "POST") {

        my $name = $FORM{'name'};
        my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
        my $rc = $dbh->do("delete from $DBGTABLE where name = '$name'");
        $dbh->disconnect;

        if ($rc) {

            &html__header("d,eletion completed");
            print "<BR><P>Comments from $name was deleted from Guestbook</P>";
        } else {

            return__error(500, "DB Error",
                          "Your information is not found at database or Connection to DB was not established");
        }
    } else {

        &return_error (500, "Wued Error",
                       "Wued does not support your request");

}
sub deleteStat {
    my $method = $ENV{'REQUEST_METHOD'};
    if ($method eq "POST") {
        &html_header("delete");
        print qq!
            <BR><BR>
            <P>Please enter the name of term :</P>
            <FORM ACTION="$CGI_DIR/wued.cgi" METHOD= "POST">
                <INPUT TYPE="hidden" NAME="command" VALUE="deleteSTerm">
                <TABLE border=0 align=center>
                    <TR valign=middle>
                        <TD><INPUT TYPE="text" NAME="term" SIZE=20 ALIGN=center>
                        <TD><INPUT TYPE="image" SRC="$MAIN__DIR/image/wued/submitb.gif" VALUE="submit" border=0>
                </TABLE>
            </FORM>!
    } else {
        &return_error (500, "Wued Error",
                        "Wued does not support your request");
    }
}

# delete term and requested number from database
sub deleteSTerm {
    my $method = $ENV{'REQUEST_METHOD'};
    if ($method eq "POST") {
        my $term = $FORM{'term'};
        my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
        $rc = $dbh->do("delete from $DBTABLE where term = '$term'");
        $dbh->disconnect;
        if ($rc) {
            &html_header("deletion completed");
            print "<BR><P>Term, $term was deleted</P>";
        } else {
            return_error(500, "DB Error",
                        "Your information is not found at database or"
Connection to DB was not established});

} else {
    &return_error (500, "Wued Error",
        "Wued does not support your request");
}
1;
sub return_error {
    local ($status, $keyword, $message) = @_; 
    print "Content-type: text/html", "\n";
    print "Status: ", $status, " ", $keyword, "\n\n";
    print qq!
    <TITLE>Unexpected Error</TITLE>
    <link rel=stylesheet href="$MAIN_DIR/html/style.css" type="text/css">
    <BODY>
    <H1 class = "heading">
    <SPAN style="font-size:50pt; color: black; font-weight: normal">
    E</SPAN>rror</H1>
    <H1>$keyword</H1>
    <HR>$message</HR>!
    
    1;
}
sub history {

    my $count = 0;

    &html_header("history");
    &parse_client_cookies(*COOKIE);

    # extract terms from cookies
    my $cookie = $COOKIE{'term'};
    my @cookie = split (/,,/, $cookie);

    print "<BR><P>History displays previous entries that a browser
    accessed. This utility enables browsers to skip unnecessary
    procedure to search for the same information that they
    previously accessed</P>";

    # if entries exist then display
    if (@cookie) {
        print "<P>Following terms are previously accessed.</P>";
        print "<TABLE cellspacing = 10>";

        # display entries accessed previously
        foreach $cookie (@cookie) {
            if ($count % 4 == 0) { print "<TR>"; }

            if ($cookie) {
                print "<TD>", &textToUrl($cookie);
                $count = $count + 1;
            }
        }

        print "</TABLE><BR>";
    } else {
        print "<BR><P>Terms that you browse will be stored for your
        later browse. </P><BR>
        
    }

    # parse coolies from clients
    sub parse_client_cookies {
        local (*COOKIE_DATA) = @_;
local (@key_value_pairs, $key_value, $key, $value);

@key_value_pairs = split (/;\s/, $ENV{'HTTP_COOKIE'});

foreach $key_value (@key_value_pairs) {
    ($key, $value) = split (/=\s/, $key_value);

    $key =~ tr/+/ /;
    $value =~ tr/+/ /;

    $key =~ s/%{[\dA-Fa-f][\dA-Fa-f]}/pack ("C", hex ($1))/eg;
    $value =~ s/%{[\dA-Fa-f][\dA-Fa-f]}/pack ("C", hex ($1))/eg;

    if (defined($COOKIE_DATA{$key})) {
        $COOKIE_DATA{$key} = join ("\0", $COOKIE_DATA{$key}, $value);
    } else {

        $COOKIE_DATA{$key} = $value;
    }
}

1;
sub parse_form_data {

    local (*FORM_DATA) = @_;  
    local ($request_method, $query_string, @key_value_pairs, $key, $value);

    $request_method = $ENV{'REQUEST_METHOD'};

    if ($request_method eq "GET") {
        $query_string = $ENV{'QUERY_STRING'};
    } elsif ($request_method eq "POST") {
        read (STDIN, $query_string, $ENV{'CONTENT_LENGTH'});
    } else {
        return_error (500, "Server Error",
                       "Server uses unsupported method");
    }

    @key_value_pairs = split (/&/, $query_string);

    foreach $key_value (@key_value_pairs) {
        ($key, $value) = split (/=/, $key_value);
        $value =~ tr/+/- /;  
        $value =~ s/%([0-9a-f][0-9a-f])/$1/g;

        if (defined($FORM_DATA{$key})) {
            $FORM_DATA{$key} = join ("\0", $FORM_DATA{$key}, $value);
        } else {
            $FORM_DATA{$key} = $value;
        }
    }
}

1;
sub random {
    my $term;
    my $def;
    my @data;
    my $number;
    my $charOne;
    my $charTwo;
    my $cursor;
    my $count = 0;

    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
    srand;
    @number = (97 .. 122);

    # iterate 500 times until term is found, otherwise exit and display a message
    until ($term or ($count > 500)) {
        # generate two characters randomly
        $charOne = chr($number[rand(@number)]);
        $charTwo = chr($number[rand(@number)]);

        # retrieve a term that matches following form
        # (random character + any characters + random character + any characters)
        $cursor = $dbh->prepare("select term, definition from $DBVTABLE where term like '$charOne%$charTwo%';");
        $cursor->execute;
        @data = $cursor->fetchrow;
        if (@data) {
            $term = $data[0];
            $def = $data[1];
        }
        $cursor->finish;
        $count = $count + 1;
    }
    &html__header("random");

    # displays the term retrieved
    if ($term) {

    }
print "<DL><DT><H2 class="term">$term</H2>";
print "<DD>", textToUrl($def), "</DL>";
} else {

print "<P>Cannot Display Random Selection. Data Sets are too small.";

}

$dbh->disconnect;


sub search {
    if (!$FORM{'term'}) { # if no input
        &html__header("i,complete");
        html("<BR><P>Search cannot be perfomed. Please enter a word or phrase.</P>");
    } else {
        my $option = $FORM{'search_option'};
        $option;
    }
}

# full-text search, connect to v_wued db
sub search_full {
    my $term;
    my $term_up;
    my $term_found;
    my $termAcr;
    my $cursor;
    my $rc;
    my $exist = 0;
    my $key_terms = $FORM{'term'};
    # split if not a single word entry
    my @terms = split('/\+/', $key_terms);
    # connect to database
    my $drh = DBI->install_driver($DBDRIVER);
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);
    # full text search
    foreach $term (@terms) {
        # handle capitalized term
        $term_up = uc $term;
        unless ($term_up eq $term) {
            $term =~ tr/A-Z/a-z/;
        }
        # handle space in front and at the end
        $term =~ s/^\s+//;
        $term =~ s/\s+$//;
    }
$cursor = $dbh->prepare("select term, definition from $DBVTABLE
where key like '%($term)%' order by term");
$cursor->execute;

# store terms found
while (@term_found = $cursor->fetchrow) {
    # to display 100 chars of def
    $term_found[1] = substr($term_found[1], 0, 100);
    $term_set{$term_found[0]} = $term_found[1];
}

$cursor->finish;

# if there are relevant terms found, then display with summaries
if (keys(%term_set)) {
    &html_header("search results");
    my $num_term = keys(%term_set);
    print "<CENTER><H2 class="miniS">WUED contains $num_term items
    relevant to your query.</H2><CENTER>";
    foreach $key (keys (%term_set)) {
        print "<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif"",
        &anchor($key);
        print "<BR><P">, &textToUrl($term_set{$key}), "</P>";
    }
    else {
        # if not, update the stat table
        &html_header("not found.");
        print "<BR><P>There is no relevant data available to your query.
        Check the spelling and try removing suffixes like
        -ing and -s.</P>";
        # for each term check if the term is listed in the stat table
        foreach $term (@terms) {
            $cursor = $dbh->prepare("select reqnumb from $DBSTABLE
            where term = '$term'");
            $cursor->execute;
            my $exist = $cursor->fetchrow;
            $cursor->finish;
            if($exist) {
                # if listed, increment reqnumb
                $exist = $exist + 1;
                $rc = $dbh->do("update $DBSTABLE set reqnumb = $exist
                where term = '$term'");
            }
        }
    }
}
else {
    $rc = $dbh->do("insert into $DBSTABLE values ('$term', 1)");
}

$dbh->disconnect;

# exact matching search, connect to wued db
sub search_exact {
    my $term_up;
    my $search_term = $FORM{'term'};
    my $drh = DBI->install_driver($DBDRIVER);
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

    # handle capitalized term
    $term_up = uc $search_term;

    unless ($term_up eq $search_term) {
        $search_term =~ tr/A-Z/a-z/;
    }

    # retrieve definition
    my $cursor = $dbh->prepare("select definition from $DBVTABLE where term = '$search_term'");
    $cursor->execute;
    my $resource = $cursor->fetchrow;
    my $def = $resource[0];
    $cursor->finish;

    # if term found
    if($def) {
        $def = textToUrl($def); # set up cross-reference
        # set cookies
        &set_cookies($search_term);
        &html_header("i,nformation");

        # display term and definition requested
        print "<DL><DT><H2 class="term">$search_term</H2><DD><P>$def</P></DL>"

        # update statistics
        $cursor = $dbh->prepare("select reqnumb from $DBTABLE where term = '$search_term'");
    }
$cursor->execute;

my $reqnumb = $cursor->fetchrow;

$reqnumb = $reqnumb + 1;
$cursor->finish;

my $rc = $dbh->do("update $DBTABLE set reqnumb = $reqnumb
where term = '$search_term'");

$dbh->disconnect;

} else {

$html_header("not found");
print "<BR><P>There is no matching word or phrase to your query.
Check spelling and try removing suffixes like -ing
or -s.</p>");

}

sub set_cookies {
my $cookie_term = $_[0];
parse_client_cookies(*COOKIE);

# extract terms accessed previously from cookies
my $preCookie = $COOKIE{'term'};

$cookie_term = "\" . $cookie_term . "\"

# if there is term found in cookies
unless ($preCookie =~ /$cookie_term/) {
  $preCookie = $preCookie . "," . $cookie_term;
}

# set cookies
my $cookie = "term=$preCookie; expires=Fri, Nov-14-2061 00:00:00 GMT;";
print "Set-cookie: $cookie", "\n";

}
sub statistics {
    my $num_stra = 10;
    my $termAcr;
    my $drh = DBI->install_driver($DBDRIVER);
    my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

    # most frequently requested terms
    my $cursor = $dbh->prepare("select max(reqnumb) from $DBTABLE");
    $cursor->execute;
    my $max_requested = $cursor->fetchrow;
    $cursor->finish;

    $cursor = $dbh->prepare("select term, reqnumb from $DBTABLE where
            reqnumb > $low_bound order by reqnumb DESC");
    $cursor->execute;
    &html_header("statistics");
    print "<BR><P>Statistics displays the most frequently requested
            missing term and the most frequently requested term by
            browsers. At every time when a browser requests a term
            in question, the number of requests of the term is
            incremented and recorded in database.</P>";
    print "<BR><HR><BR>
    <P>Most Frequently Requested Terms</P><BR>";
    print "$max_requested, 
    # retrieved every terms from max. # of requested to the lower
    # boundary # of requested.
    while (@contents = $cursor->fetchrow) {
        if ($contents[1] == $max_requested) {
            print $contents[0], "&nbsp;&nbsp;&nbsp;&nbsp;";
        } else {
            $max_requested = $max_requested -1;
            print "$max_requested, 
        }
    }
}
$cursor->finish;

print "<BR><HR><BR>";

# most frequently missed terms
print "<P>Most Frequently Missed Terms</P><BR>";
$cursor = $dbh->prepare("select max(reqnumb) from $DBSTABLE");
$cursor->execute;

$max_missed = $cursor->fetchrow;

# set lower boundary of the # of requested time
$low_bound = $max_missed - $num_stra;
$cursor->finish;

$cursor = $dbh->prepare("select term, reqnumb from $DBSTABLE where reqnumb > $low_bound order by reqnumb DESC");
$cursor->execute;

print "# of requests : ", $max_missed, "<BR><BR>";

# retrieved every terms from max. # of requested to the lower # boundary # of requested.
while (@contents = $cursor->fetchrow) {

    if ($contents[1] == $max_missed) {
        print $contents[0], "&nbsp;&nbsp;&nbsp;&nbsp;";
    } else {
        $max_missed = $max_missed -1;
        if ($contents[0]) {
            print "<BR><BR># of requests : ", $max_missed, "<BR>";
            print "<BR>", $contents[0], "&nbsp;&nbsp;&nbsp;&nbsp;";
        }
    }
}

$cursor->finish;
$dbh->disconnect;
print "<BR><HR><BR>";
sub submit {

my $term_up;
my $query = $ENV{'QUERY_STRING'};

# if there is no information submitted then display error message
if (!$FORM{'term'} or !$FORM{'definition'}) {
  &html_header("i,ncomplete");
  print "<BR><P>Your information is not complete. Please try again.");

} else {

  # extract information
  my $term = $FORM{'term'};
  my $def = $FORM{'definition'};

  my $drh = DBI->install_driver($DBDRIVER);
  my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS);

  $term_up = uc $term;
  unless ($term_up eq $term) {
    $term =~ tr/A-Z/a-z/;
  }

  $def =~ s/'/(quote)/g;

  my $rc = $dbh->do("insert into $DBSUBTABLE
    values ('$term', '$def')");

  if ($rc) {
    &html_header("t,hank you");
    print "<BR><P>Thanks for your contribution. Following is the
    information that you just submitted. Your
    information will be reviewed and used to enhance
    $WUED_NAME<BR><BR>**;
    print "<OL><LI>Term : $term <LI>Definition : $def</OL><BR>";
  } else {
    return_error(500, "DB Error",
      "Connection to DB was not established");
  }
}
$dbh->disconnect;
}
#!usr/bin/perl

# File Name : LOAD
# Description: Load a data file into database

'$REQUIRE_DIR = '..'/lib';

push(0, $REQUIRE_DIR) if $REQUIRE_DIR;
use CGI qw(:standard);
require 'config.pl';

use DBI;
&configure_var;

my $exclusive_lock = 2;
my $unlock = 8;
my $line;
my $term;
my $term_up;
my $def;
my $ele;
my $rc;

# get name of a data file
print "Enter the name of Data File : 
";
chomp ($input_file = <STDIN>);

my $dbh = DBI->install_driver($DBDRIVER);
my $dbh = DBI->connect("dbi:$DBDRIVER:$DBNAME", $DBUSER, $DBPASS)
or die $DBI::errstr;

# create database view tables and index
($dbh->do( "DROP VIEW v_wued"));
($dbh->do( "CREATE VIEW v_wued AS SELECT key, term, definition FROM wued"));
($dbh->do( "DROP VIEW v_term"));
($dbh->do( "CREATE VIEW v_term AS SELECT term from wued"));
($dbh->do( "DROP index termid"));
($dbh->do( "CREATE index termid on v_wued (term, definition, key)" ));
($dbh->do( "DROP index id"));
($dbh->do( "CREATE index id on wued (term) " ));
($dbh->do( "DROP index tid"));
($dbh->do( "CREATE index tid on v_term (term) " ));

104
if (open(DIC, "<$input_rfile")) {
    flock (DIC, $exclusive_lock);

    $term = <DIC>;

    # loading
    until (eof DIC) {
        $def = "";
        $keys = "";

        # read definition (definition starts with tab char or new line char
        while ( $line = <DIC>, ($line =~ /^\[\t\n\]/)) {
            $line =~ s/\t//;
            $def = $def . $line;
        }

        $term =~ s/\[\t\n\]//g; # remove tab and newline chars
        $term =~ s/\'///g;

        $term_up = uc $term;

        if ($term_up eq $term) {
            $term_up =~ tr/A-Z/a-z/;
            $keys = $keys . "{$term}{$term_up}";
        } else {
            $term =~ tr/A-Z/a-z/;
            $keys = $keys . "{$term}"
        }

        # set up keywords
        my @term_list = ($def =~ /{([^{^}]*})/g);
        foreach $element (@term_list) {
            unless (($element =~ /\([^\)]\)/) or ($element =~ /:/)) {
                $term_up = uc $element;

                unless ($term_up eq $element) {
                    $element =~ tr/A-Z/a-z/;
                }

                $element = "{$element}";

                unless ($keys =~ /$element/) { $keys = $keys . $element; }
            }
        }

        $keys =~ s/\'///g;
        $def =~ s/\'(quote)/g;
    }
$rc = $dbh->do("insert into $DBTABLE
values ( '$term', '$keys', '$def' )");

$term = $line;
}

close(DIC);
flock (DIC, $unlock);
} else {
print "cannot open a file for input\n";
}

# do the same thing but for capitalized entries only
# attach cross ref. if same entries with lower case exist
if (open(DIC, "<$input_file")) {

close(DIC);
flock (DIC, $exclusive_lock);
$term = <DIC>;
until (eof DIC) {
$def = "";
$keys = "";

while ( $line = <DIC>, ($line =~ /^\[\t\n\]/)) {
   $line =~ s/\t//;
   $def = $def . $line;
}

$term = s/\[\t\n\]/g; # remove tab and newline chars
$term = s//g;
$term_up = uc $term;
if (($term_up eq $term) and ($term =~ /\D/) and ($term =~ /\w/) ) {
   $term_up =~ tr/A-Z/a-z/;
   $cursor = $dbh->prepare("select definition from $DBTABLE
where term = '$term_up' ");
   $cursor->execute;
   $lowDef = $cursor->fetchrow;
   $cursor->finish;

   if ($lowDef) {
      $lowDef = $lowDef . "\n\nSee also {$term} \n"
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");

      $cursor = $dbh->prepare("select definition from $DBTABLE
         where term = '$term' ");

      $cursor->execute;
      $def = $cursor->fetchrow;

      if ($lowDef) { $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term' ");
      
   } else { print "cannot open a file for input";
   }

   close(DIC);
flock (DIC, $unlock);
} else {
print "cannot open a file for input";
}

# do the same thing but for capitalized entries only
# attach cross ref. if same entries with lower case exist
if (open(DIC, "<$input_file")) {

close(DIC);
flock (DIC, $exclusive_lock);
$term = <DIC>;
until (eof DIC) {
$def = "";
$keys = "";

while ( $line = <DIC>, ($line =~ /^\[\t\n\]/)) {
   $line =~ s/\t//;
   $def = $def . $line;
}

$term = s/\[\t\n\]/g; # remove tab and newline chars
$term = s//g;
$term_up = uc $term;
if (($term_up eq $term) and ($term =~ /\D/) and ($term =~ /\w/) ) {
   $term_up =~ tr/A-Z/a-z/;
   $cursor = $dbh->prepare("select definition from $DBTABLE
where term = '$term_up' ");
   $cursor->execute;
   $lowDef = $cursor->fetchrow;
   $cursor->finish;

   if ($lowDef) {
      $lowDef = $lowDef . "\n\nSee also {$term} \n"
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");

      $cursor = $dbh->prepare("select definition from $DBTABLE
         where term = '$term' ");

      $cursor->execute;
      $def = $cursor->fetchrow;

      if ($lowDef) { $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term' ");
      
   } else { print "cannot open a file for input";
   }

   close(DIC);
flock (DIC, $unlock);
} else {
print "cannot open a file for input";
}

# do the same thing but for capitalized entries only
# attach cross ref. if same entries with lower case exist
if (open(DIC, "<$input_file")) {

close(DIC);
flock (DIC, $exclusive_lock);
$term = <DIC>;
until (eof DIC) {
$def = "";
$keys = "";

while ( $line = <DIC>, ($line =~ /^\[\t\n\]/)) {
   $line =~ s/\t//;
   $def = $def . $line;
}

$term = s/\[\t\n\]/g; # remove tab and newline chars
$term = s//g;
$term_up = uc $term;
if (($term_up eq $term) and ($term =~ /\D/) and ($term =~ /\w/) ) {
   $term_up =~ tr/A-Z/a-z/;
   $cursor = $dbh->prepare("select definition from $DBTABLE
where term = '$term_up' ");
   $cursor->execute;
   $lowDef = $cursor->fetchrow;
   $cursor->finish;

   if ($lowDef) {
      $lowDef = $lowDef . "\n\nSee also {$term} \n"
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");

      $cursor = $dbh->prepare("select definition from $DBTABLE
         where term = '$term' ");

      $cursor->execute;
      $def = $cursor->fetchrow;

      if ($lowDef) { $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term' ");
      
   } else { print "cannot open a file for input";
   }

   close(DIC);
flock (DIC, $unlock);
} else {
print "cannot open a file for input";
}

# do the same thing but for capitalized entries only
# attach cross ref. if same entries with lower case exist
if (open(DIC, "<$input_file")) {

close(DIC);
flock (DIC, $exclusive_lock);
$term = <DIC>;
until (eof DIC) {
$def = "";
$keys = "";

while ( $line = <DIC>, ($line =~ /^\[\t\n\]/)) {
   $line =~ s/\t//;
   $def = $def . $line;
}

$term = s/\[\t\n\]/g; # remove tab and newline chars
$term = s//g;
$term_up = uc $term;
if (($term_up eq $term) and ($term =~ /\D/) and ($term =~ /\w/) ) {
   $term_up =~ tr/A-Z/a-z/;
   $cursor = $dbh->prepare("select definition from $DBTABLE
where term = '$term_up' ");
   $cursor->execute;
   $lowDef = $cursor->fetchrow;
   $cursor->finish;

   if ($lowDef) {
      $lowDef = $lowDef . "\n\nSee also {$term} \n"
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");

      $cursor = $dbh->prepare("select definition from $DBTABLE
         where term = '$term' ");

      $cursor->execute;
      $def = $cursor->fetchrow;

      if ($lowDef) { $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term_up' ");
      $rc = $dbh->do("update $DBTABLE set
         definition = '$lowDef' where term = '$term' ");
      
   } else { print "cannot open a file for input";
   }

   close(DIC);
flock (DIC, $unlock);
} else {
print "cannot open a file for input";
}
$cursor->finish;
$def = $def . "\n\nSee also {$term__up}\n";
$rc = $dbh->do("update $DBTABLE set
definition='{$def' where term = '{$term'}");
}
}

$term = $line;
}
close(DIC);
flock (DIC, $unlock);
$dbh->disconnect;
} else {
print "cannot open a file for input\n";
}
sub html_header {
  my $str = $$.0;
  my @head = split(/,/, $str);

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

  print "<HTML><HEAD>"

  print qq!
    <TITLE>Web-based Universal Encyclopedia/Dictionary</TITLE></HEAD>
    <link rel=stylesheet href="$MAIN_DIR/style.css" type="text/css">
    <BODY BGCOLOR="#FFFFFF">
      <TABLE border=0>
        <TR valign=middle>
          <TD><A HREF="$CGI_DIR/wued.cgi?command=about">
            <IMG SRC="$MAIN_DIR/image/wued/wued.gif" BORDER=0 ALT="Wued" WIDTH=42
            HEIGHT=19 VSPACE=0></A>
          <TD><A HREF="$CGI_DIR/wued.cgi?command=home">
            <IMG SRC="$MAIN_DIR/image/wued/home.gif" BORDER=0 ALT="Home" WIDTH=30
            HEIGHT=20 VSPACE=0></A>
          <TD><A HREF="$CGI_DIR/wued.cgi?command=history">
            <IMG SRC="$MAIN_DIR/image/wued/history.gif" BORDER=0 ALT="History"
            WIDTH=30 HEIGHT=20 VSPACE=0></A>
          <TD><A HREF="$CGI_DIR/wued.cgi?command=random">
            <IMG SRC="$MAIN_DIR/image/wued/random.gif" BORDER=0 ALT="Random"
            WIDTH=30 HEIGHT=20></A>


108
<TD><A HREF="$CGI_DIR/wued.cgi?command=content_front">
<IMG SRC="$MAIN_DIR/image/wued/content.gif" BORDER=0 ALT="Help"
WIDTH=23 HEIGHT=20></A></TD>

<TD><A HREF="$CGI_DIR/wued.cgi?command=edit">
<IMG SRC="$MAIN_DIR/image/wued/edit.gif" BORDER=0 ALT="Edit"
WIDTH=23 HEIGHT=20></A></TD>

<TD><A HREF="$CGI_DIR/wued.cgi?command=help">
<IMG SRC="$MAIN_DIR/image/wued/help.gif" BORDER=0 ALT="Help"
WIDTH=23 HEIGHT=20></A></TD>

<TD align=middle>
<FORM ACTION="$CGI_DIR/wued.cgi" METHOD="GET">
<INPUT TYPE="hidden" NAME="command" VALUE="search">
<TD><INPUT TYPE="text" NAME="term" SIZE=20 ALIGN=center>
<TD>
<SELECT name="search_option">
<OPTION value="search_full">Full
<OPTION value="search_exact">Exact
</SELECT>
</FORM>
</TD>

<TABLE border=0 align=right valign=top>
<TD><A HREF="$CGI_DIR/wued.cgi?command=submit_front">Submit</A></TD>
<TD><A HREF="$CGI_DIR/wued.cgi?command=statistics">Statistics</A></TD>
<TD><A HREF="$CGI_DIR/wued.cgi?command=about">About</A></TD>
</TABLE>

<H1 class = "heading">
<SPAN style="font-size:50pt; color:black; font-weight:normal">
$head[0]</SPAN>$head[1]</H1>!

sub html_footer {
print qq!
<BR><BR>
<CENTER>
<A HREF="$CGI_DIR/wued.cgi?command=home">Home |</A>
<A HREF="$CGI_DIR/wued.cgi?command=history">History |</A>
<A HREF="$CGI_DIR/wued.cgi?command=random">Random |</A>
<A HREF="$CGI_DIR/wued.cgi?command=content_front">Contents |</A>
<A HREF="$CGI_DIR/wued.cgi?command=contact_front">Contact |</A>
<A HREF="$CGI_DIR/wued.cgi?command=edit">Edit |</A>
<A HREF="$CGI_DIR/wued.cgi?command=help">Help</A>
</CENTER>
</BODY>
</HTML>!}
sub edit_footer {

print qq!
<br>
<form action="$CGI_DIR/wued.cgi" method="post">
<input type="hidden" name="command" value="edit">
<input type="hidden" name="pass" value="$PASSWORD">
<center>
<input type="image" src="$MAIN_DIR/image/wued/menu.gif" width=70 height=20 border=0>
</center>
</form>!
}

sub home {

$html__header("home of");
print qq!
<h2><center>$NAME</center></h2>
<table>
<tr><td valign=top align=center>
<a href="$CGI__DIR/wued.cgi?command=about">WUED</a><br>
A great tool to create an online encyclopedia/dictionary.<br>
<img src="$MAIN_DIR/image/wued/earth.gif">
</td> <td valign=top>
<p>$DESCRIPTION</p>
<p>A browser can conduct a full-text search or exact-match search by typing a search query and hitting the search button or Enter. The search will run on the database and display related information. The full-text search displays the number of entries found and a list of entries along with brief summaries of each entry. The exact-match search displays the definition of an entry, if present in the database. <a href="$CGI__DIR/wued.cgi?command=help">How to search</a> for more information.</p>
</td>
</tr>
</table>
<hr><br>
<a href="$CGI__DIR/wued.cgi?command=submit_front">Submit</a><br>
You are allowed to submit your information to an administrator through submit. Your information is stored in the database and can be used to enhance an online encyclopedia/dictionary.<p>

<a href="$CGI__DIR/wued.cgi?command=statistics">Statistics</a><br>
Statistics displays the most frequently requested missing term and the most frequently requested term by browsers.<p>

<a href="$CGI__DIR/wued.cgi?command=about">About WUED</a><br>
Browsers can find the information about WUED through About. About introduces a description of WUED and a pointer where WUED package are available.</p>
Contact is a place for browsers to make their comments about an online encyclopedia/dictionary. The comments are stored in database and displayed by browsers through Guestbook.

Abstract

Web-based Universal Encyclopedia/Dictionary (WUED) is a software application that provides a database independent user tool that places online encyclopedia or dictionary type data. WUED creates a searchable encyclopedia/dictionary of user provided data through a user-friendly Graphical User Interface via an Internet browser. It allows a user to display terms and related resources online so that information of interest can be easily browsed. It provides a one-stop source of information about stored data, and includes cross-referencing and hyperlinks to related resources elsewhere.

Introduction

The Internet is rapidly becoming mainstream media conduit for communication between individuals, companies, and global dwellers. As part of the Internet, the Web has been growing extremely fast in recent years, and its applications served as the major tools of exchanging the information. Many of the Web applications have used a simple ASCII text file to store data. However, more efficient and convenient ways of storing data are demanded since applications are becoming more flexible and complicated, and requiring storing larger amounts of data. The Web-based database was carefully considered to solve the problem and can be used to develop other Web applications.

One of the Web applications that easily provides useful information to browsers is the online dictionary. There are already several hundreds of online dictionaries on the Web allowing browsers to access the information of interest. For example, the Free On-line Dictionary of Computing (FOLDOC) is an online searchable dictionary designed by Denis Howe in 1985. The dictionary is stored as a single text file and contains...
over 11,000 definitions totaling more than four megabytes. It allows
user to enter queries and displays the related information.</p>

WUED was designed to provide a user tool that creates something with
similar functionality to the existing online dictionaries and uses a
database instead of using ad hoc scripts and the flat files. WUED uses
generic database functions in order to be independent of the database
and can be easily deployed when data is available.</p>

WUED provides INSTALL script that runs all the necessary
installation procedures. This interactive procedure of INSTALL script
defines a name of encyclopedia/dictionary, a type of database
management system and others. During Installation of WUED a user is
allowed to select any one of running database system that Database
Interface (DBI) and Database Driver (DBD) support. The user's
selection will be ported to the selected database driver and database
system.</p>

WUED gives a browser information that contains the related links on
Web, media of Audio and Picture type - musical examples, pronunciation
of words, pictures, maps, etc.</p>

WUED's GUI, EDIT, provides an easy way to edit the information
stored in encyclopedia/dictionary. It allows a user to import, insert,
delete and modify data in database. This utility is only available to
the administrator of encyclopedia/dictionary, and browsers are not
allowed to access this GUI for modification of information in
database.</p>

Development of WUED will make use of the following hardware and
software.</p>

IBM based Compatible PC with Redhat Linux5.0
Common Gateway Interface (CGI) using Perl v5.004_04
Database Interface (DBI) v0.63
Database Driver for PostgreSQL (DBD:Pg) v0.90
POSTGRES95

WUED is designed to be a database independent software application,
which allows a user to select DBMS during installation. However, the
type of DBMS that DBI and DBD support linkits WUED's DBMS selection.
The following is the list of DBMSs that DBI and DBD support.</p>

Oracle7 RDMS
mSQL-1.x or mSQL-2.x
mysql
Ingres 6.4 or OpenIngres
Informix 5.00 through to Informix 7.22
Empress 6.8
Fulcrum SearchServer 2.0, 3.x SDK
DB2 v2.1 or beyond
Quickbase
In order for WUED to be independent on DBMS, the SQL commands used in WUED should be available to all of DBMSs that WUED supports. The following is a list of the SQL commands used in WUED implementation.

- CREATE DATABASE db
- DROP DATABASE db
- CREATE TABLE table (column datatype [,column datatype] ...)
- DROP TABLE table
- CREATE INDEX index ON table (column [,column] ...)
- DROP INDEX index
- CREATE VIEW view AS select_subset
- DROP VIEW view
- SELECT select_list FROM table [WHERE conditions][ORDER BY column][ASC|DESC]
- INSERT [INTO] table VALUES (values_list)
- DELETE FROM table [WHERE conditions]
- UPDATE table SET column = expr [,column = expr, ...]

sub contact_front {

$html_header("c,ontact");

print qq!

This is a place for browsers to make their comments about an online encyclopedia/dictionary. The comments will be stored in database and displayed by browsers through Gueestbook.

Please provide the following information.

FORM ACTION="$CGI_DIR/wued.cgi" METHOD= "POST">
<input type="hidden" name="command" value="contact">
<table border=0 align=center>
<tr valign=middle>
<td valign=bottom>Full Name ...
<td><input type="text" name="name" size=30>
<tr valign=middle>
<td valign=bottom>Email Address ...
<td><input type="text" name="address" size=30>
<tr valign=top>
<td colspan=2>Please enter the information that you'd like to add:
<tr valign=top>
<td colspan=2><textarea rows=5 cols=50 name="message"></textarea></table>

</FORM>
By simply clicking one of the following letters in the alphabet, a browser can access the list of entries starting with the alphabet letter.

```html
<table border=0 align=center valign=middle cellspacing=20 cellpadding=0>
  <tr valign=middle><td><a href="$CGI_DIR/wued.cgi?command=content&char=a">A</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=b">B</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=c">C</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=d">D</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=e">E</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=f">F</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=g">G</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=h">H</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=i">I</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=j">J</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=k">K</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=l">L</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=m">M</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=n">N</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=o">O</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=p">P</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=q">Q</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=r">R</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=s">S</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=t">T</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=u">U</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=v">V</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=w">W</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=x">X</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=y">Y</a></td>
  <td><a href="$CGI_DIR/wued.cgi?command=content&char=z">Z</a></td>
</tr>
</table>
```
<TABLE align=center><TR>
<TD><SELECT NAME="edit_select">
<OPTION value="import">Import data from visitor's submission
<OPTION value="import_file">Import data from a data file
<OPTION value="insert">Insert new data to DBMS
<OPTION value="delete">Delete data from DBMS
<OPTION value="modify">Modify data in DBMS
<OPTION value="deleteGuest">Delete comments from Guestbook
<OPTION value="deleteStat">Delete the most frequently missed term</SELECT>
</TD><TD><INPUT TYPE="image" SRC="$MAIN_DIR/image/wued/submitb.gif" WIDTH=70 HEIGHT=20 border=0>
</TABLE>!

sub help {
$html_header("h,elp");

print qq!
<P>This site provides tips for navigating through an online encyclopedia/dictionary and offers detailed information on such topics as searching and features.</P>

<UL type="disc">
<LI><A HREF="#tools">Menu</A>
<LI><A HREF="#home">Home of Ericyclopedia/Dictionary</A>
<LI><A HREF="#history">History</A>
<LI><A HREF="#random">Random Display of Information</A>
<LI><A HREF="#content">Contents in Database</A>
<LI><A HREF="#edit">Edit for a WUED Administrator Only</A>
<LI><A HREF="#search">How to Search</A>
<LI><A HREF="#contact">Contact Us</A>
<LI><A HREF="#submit">Submit and Contribute</A>
<LI><A HREF="#statistics">Statistics</A>
<LI><A HREF="#about">About WUED</A>
</UL>

</TABLE>!
A menu is displayed at the top of each window to enable a browser to navigate the encyclopedia/dictionary by a simple click of a tool button. The menu consists of six tool buttons, a text box and a drop-down list. Tool buttons allow a browser to move on to the other WUED utilities of interest. The utilities that each tool button allows a browser to access are Home, History, Random, Contents, Edit, Help and Search.

Home explains the purpose of the online encyclopedia/dictionary and gives the browser a summary of how to use online encyclopedia/dictionary.

History displays previous entries that a browser accessed. This utility enables browsers to skip unnecessary procedure to search for the same information that they previously accessed. WUED uses a technology called cookies for History. Browsers that support this technology include Netscape Navigator 3.0, Communicator 4.0 PR2 (at least) and Internet Explorer 3.0. A browser most likely will not be able to access this utility with earlier versions of these browsers. The entries accessed are added in Set-Cookie header that is to be stored by the client for later retrieval. WUED sets two days of the expiration data that defines the valid time of the entries.

Random provides a way to explore WUED's browsable database. When a browser doesn't have a specific term in question, Random is the best approach. A single entry of information is randomly selected through the entire database and displayed on browsers. Browsing Random is much like flipping the pages of an encyclopedia/dictionary. Random displays the entry by randomly generating two characters and executing the SQL command that contains the characters and the permissible wildcards (%). The SQL command used is "Select term from wued where term like 'RandomCharacter%RandomCharacter%'".

Contents provide an alphabetical list of entries in the database. By simply clicking one of the letters in the alphabet, a browser can access the list of entries starting with the alphabetic letter. Only 50 entries are displayed on each request, and a browser is allowed to move on to the next 50 entries by clicking "Next 50" button at the bottom of page.

Edit is provided for modification of data in database. Only an administrator of the encyclopedia/dictionary is authorized to access this utility. The administrator is prompted to enter a password; entering an invalid password results in denial of access. When a valid
password is provided, edit options are displayed. By selecting an option, the administrator is allowed to import, insert, delete and modify data in database.</p>

<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif"><A NAME=search>How to Search</A>

The search will run on the database and display related information. The full-text search displays the number of entries found and a list of entries along with brief summaries of each entry. Full-text search searches through all keywords and terms that contain the keywords specified in the search are displayed as hyperlinks. The exact-match search displays the definition of an entry, if present in the database. If the definition contains other resources such as related URLs, images or sounds, these resources are converted to hyperlinks for easy access by the browser. </p>

<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif"><A NAME=contact>Contact</A>

Contact is a place for browsers to make their comments about an online encyclopedia/dictionary. The comments are stored in database and displayed by browsers through Guestbook.</p>

<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif"><A NAME=submit>Submit and Contribute</A>

Browsers are allowed to submit their information to an administrator through Submit. Their information is stored in database and used to enhance an online encyclopedia/dictionary.</p>

<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif"><A NAME=statistics>Statistics</A>

Statistics displays the most frequently requested missing term and the most frequently requested term by browsers. Every time a browser requests for a term, the number of requests of the term is incremented and recorded in database.</p>

<IMG SRC="$MAIN_DIR/image/wued/rarrow.gif"><A NAME=about>About WUED</A>

Browsers can find the information about WUED through About. About introduces a description of WUED and a pointer where WUED packages are available.</p>

sub submit_front {

$html_header("s,ubmit");

print qq!

You are allowed to submit your information to an administrator through this page. Your information will be stored in the database and can be used to enhance an online encyclopedia/dictionary. </p>

<Form ACTION="$CGI_DIR/wued.cgi" METHOD= "POST">

<TABLE border=0 align=center>

<TR valign=top>

<TD valign=top>Term ...
# convert keywords into cross-references
sub textToUrl {
    $n = $n[0];
    my $element;
    s/'+/plus/g;
    my @term_list = /{([^{}^:]*))/g;
    foreach $element (@term_list) {
        $url = &anchor($element);
        s/\{\$element\}/\$url/;
    }

    # Convert resources in hypertext links
    s/\n\n/<P>/g;
    s/\{(sound:\([^{}^:]*\)\s*\([^{}^:]*\)\}\}|<IMG SRC="$MAIN_DIR/image/wued/sound.gif">
        <EM><A HREF="$MAIN_DIR/sound/$1">$1</A></EM>|gi;
    s/\{(image:\([^{}^:]*\)\s*\([^{}^:]*\)\}\}|<BR><CENTER>
        <IMG SRC="$MAIN_DIR/image/database/$1"></CENTER><BR>|gi;
    s/\{FTP(ftp|g;
    s/\{MORE\}|<IMG SRC="$MAIN_DIR/image/wued/link.gif">
        <EM><A HREF="$2">$1</A></EM>|gi;
    s/\{quote\}|'/g;
    $/:plus/+;/g;
    return $n;
}

# return the term with an anchor
sub anchor { $t = $n[0];
    my $term_search = "";
    if ($term =~ /\s/) {
    }
\@str = split (/\s/, $term);

foreach $ele (@str) {
  if ($term_search eq "") {
    $term_search = $ele;
  } else {
    $term_search = $term_search . "$" . $ele;
  }
}

my $term_up = uc $term_search;

unless ($term_up eq $term_search) {
  $term_search =~ tr/A-Z/a-z/;
}

$term = "<A HREF="$CGI_DIR/wued.cgi?command=search_exact&term
  =$term_search">$[0]<\/A>";
}

else {

  my $term_up = uc $term;

  unless ($term_up eq $term_search) {
    $term_search =~ tr/A-Z/a-z/;
  }

  $term = "<A HREF="$CGI_DIR/wued.cgi?command=search_exact&term
    =$term">$[0]<\/A>";
}

return $term;
}

# add curly brakets around the keywords
sub convertKey { my $str = $_[0]; my $key; my $term_up;

  $str =~ s/*/g;

  my @key_list = split (/\s+/, $str);

  $str = "";

  foreach $key (@key_list) {
    $term_up = uc $key;
    unless ($term_up eq $key) { $key =~ tr/A-Z/a-z/; }
    $str = $str . $key;
  }

  return $str;
}
sub convertText {
    my $str = $_[0];

    $str =~ s/{//g;
    $str =~ s/}//g;
    return $str;
}
REFERENCES CITED

http://wombat.cod.ic.ac.uk/, Editor Denis Howe