

California State University, San Bernardino

**CSUSB ScholarWorks**

---

Theses Digitization Project

John M. Pfau Library

---

2005

## A portable traveler's weblog

Feng-Chun Lung

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



Part of the [Databases and Information Systems Commons](#)

---

### Recommended Citation

Lung, Feng-Chun, "A portable traveler's weblog" (2005). *Theses Digitization Project*. 2832.  
<https://scholarworks.lib.csusb.edu/etd-project/2832>

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](mailto:scholarworks@csusb.edu).

A PORTABLE TRAVELER'S WEBLOG

---

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  
Feng-Chun Lung  
March 2005

A PORTABLE TRAVELER'S WEBLOG

---

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

---


by  
Feng-Chun Lung  
March 2005

Approved by:

  
\_\_\_\_\_  
Dr. David Turner, Chair, Computer Science

3/3/2005  
Date

  
\_\_\_\_\_  
Dr. Richard Botting

  
\_\_\_\_\_  
Dr. Kerstin Voigt

## ABSTRACT

Portable Traveler's Web Log is a project aimed at providing people the ability to record and publish their travel experiences. The users can modify and view the site either from a normal web browser, or from a mobile device with WML capabilities. The visitors can view the site either from a normal web browser, or from a mobile device with WML capabilities. Furthermore, this site offers a secure environment to keep all the users' private information. So data security is important.

The project includes two user interfaces: one is a normal web browser, which uses JSP (Java Server Page), HTML (Hyper Text Markup Language) as the basic language. The other one is the cellular device, which uses JSP (Java Server Page) WML (Wireless Markup Language) as the basic language. The users can record, modify and store their travel experience anytime or anywhere by using these two tools.

A Portable Traveler's Web Log is an "easy to use" application. Everyone knows how to use a web browser or cellular device, so they can easily login to record travel experiences or view the travel web log of other users of the system.

## ACKNOWLEDGMENTS

I would not have finished my graduate project successfully without the support and encouragement of my advisor, my family and my friends. I am so grateful to Dr. Turner, who is such a wonderful and perfect advisor, and he not only offers me this project but also directs me through this entire effort. I am also grateful to Dr. Botting and Dr. Kerstin Voigt to be my committee and provide helpful comments on the project.

Also, I appreciate my family for all their love, understanding and tolerance. My husband not only supports me throughout my study in many ways, but also provides full financial aid; without them this project cannot be finished on time. I would like to share the honor with them.

Finally, I thank the faculty of Computer Science department for giving me an opportunity to pursue my M.S. in Computer Science at California State University. I also want to thank all my friends who helped me during my studying at CSUSB.

## TABLE OF CONTENTS

ABSTRACT .....	iii	
ACKNOWLEDGMENTS .....	iv	
LIST OF TABLES .....	ix	
LIST OF FIGURES .....	x	
CHAPTER ONE: INTRODUCTION		
1.1 Purpose of This Project .....	1	
1.2 Scope of Project .....	2	
1.2.1 Deliverables .....	2	
1.2.2 Function of Software Products .....	2	
1.3 Significance of the Project .....	2	
1.4 Definition and Abbreviations .....	3	
1.5 Organization of the Documentation .....	4	
CHAPTER TWO: A PORTABLE TRAVELER'S WEBLOG ARCHITECTURE .....		5
2.1 Software Interfaces .....	7	
CHAPTER THREE: SOFTWARE REQUIREMENTS SPECIFICATION		
3.1 Introduction .....	9	
3.2 Overall Description .....	9	
3.2.1 Product Perspective .....	9	
3.2.2 Product Functions .....	10	
3.2.3 Product Architecture .....	10	
3.2.4 State Chart of A Traveler Web Log Session .....	12	
3.2.5 Design Different of Web Browser and Cellular Phone .....	12	

CHAPTER FOUR: DATABASE DESIGN

4.1 Data Analysis ..... 16

4.2 Database Schema Logical Model -  
Relational Schema ..... 17

4.3 Data Type and Details ..... 18

CHAPTER FIVE: PROJECT IMPLEMENTATION

5.1 Graphical User Interface Normal Browser  
Website ..... 20

5.1.1 Welcome Page - Registered Users ..... 20

5.1.2 Login Page (For Logged in User) ..... 22

5.1.3 Private Home Page (For Logged in  
User) ..... 23

5.1.4 Private Home - View Log Record  
(For Logged in User) ..... 25

5.1.5 Private Home - Edit Log Record  
(For Logged in User) ..... 27

5.1.6 Private Home - New Edit Page (For  
Logged in User) ..... 29

5.1.7 Welcome Page - View Public Logs ..... 31

5.1.8 Public Home Page (For the  
Visitor) ..... 32

5.1.9 Public Home - View Log Record (For  
the Visitor) ..... 33

5.1.10 Welcome Page - Click Here to  
Register ..... 34

5.1.11 New User Form (For the Visitor) ..... 34

5.2 Graphical User Interface Wireless Make  
Language-Ready Phone Simulator ..... 35

5.2.1 Wireless Make Language Welcome  
Page - Registered Users ..... 36

5.2.2	Wireless Make Language Login Page (For Logged in User) .....	37
5.2.3	Wireless Make Language Private Home Page (For Logged in User) .....	38
5.2.4	Wireless Make Language Private Home - View Record (For Logged in User) .....	39
5.2.5	Wireless Make Language Private Home - Edit Record Page (For Logged in User) .....	41
5.2.6	Wireless Make Language Private Home - New Page (For Logged in User) .....	42
5.2.7	Wireless Make Language Welcome Page - Search .....	44
5.2.8	Wireless Make Language Public Home Page (For the Visitor) .....	44
5.2.9	Wireless Make Language Public Home - View Record (For the Visitor) .....	45
5.2.10	Wireless Make Language Welcome Page - New User (For the Visitor) .....	46
5.2.11	Wireless Make Language New User Form (For the Visitor) .....	47

CHAPTER SIX: MAINTENANCE MANUAL

6.1	Software Installation .....	49
6.1.1	JAVA 2 Platform, Standard Edition .....	49
6.1.2	Structured Query Language Installation .....	49
6.1.3	JAVA Database Connectivity .....	50
6.1.4	Tomcat .....	51



CHAPTER SEVEN: CONCLUSION AND FUTURE DIRECTIONS

7.1 Conclusion .....	52
7.2 Future Directions .....	52
APPENDIX: SOURCE CODE .....	54
REFERENCES .....	66

LIST OF TABLES

Table 1. Structure of Table Log .....	18
Table 2. Structure of Table User .....	19

## LIST OF FIGURES

Figure 1.	System Architecture .....	6
Figure 2.	Use Case Diagram .....	10
Figure 3.	System Architecture .....	11
Figure 4.	State Chart of A Traveler Web Log Session .....	12
Figure 5.	E-R Diagram .....	17
Figure 6.	Database Relational Schema .....	18
Figure 7.	Welcome Page - Registered Users .....	21
Figure 8.	Login Page (For Logged in User) .....	23
Figure 9.	Private Home Page (For Logged in User) .....	25
Figure 10.	Private Home - View Log Record (For Logged in User) .....	27
Figure 11.	Private Home - Edit Log Record (For Logged in User) .....	29
Figure 12.	Private Home - New Edit Page (For Logged in User) .....	31
Figure 13.	Public Home Page (For the Visitor) .....	33
Figure 14.	Public Home - View Log Record (For the Visitor) .....	34
Figure 15.	New User Form (For the Visitor) .....	35
Figure 16.	Wireless Markup Language Welcome Page - Registered Users .....	37
Figure 17.	Wireless Markup Language Login Page (For Logged in User) .....	38
Figure 18.	Wireless Markup Language Private Home Page (For Logged in User) .....	39
Figure 19.	Wireless Markup Language Private Home - View Record (For Logged in User) .....	40

Figure 20.	Wireless Markup Language Private Home - Edit Log Page (For the Logged in User) .....	42
Figure 21.	Wireless Markup Language Private Home - New Page (For Logged in User) .....	43
Figure 22.	Wireless Markup Language Welcome Page - Search .....	44
Figure 23.	Wireless Markup Language Public Home Page (For the Visitor) .....	45
Figure 24.	Wireless Markup Language Public Home - View Record (For the Visitor) .....	46
Figure 25.	Wireless Make Language Welcome Page - New User (For the Visitor) .....	47
Figure 26.	Wireless Markup Language New User Form (For the Visitor) .....	48

## CHAPTER ONE

### INTRODUCTION

#### 1.1 Purpose of This Project

This project is a web-based application, that provides a friendly and simple interface to let users easily record and modify their travel experiences anytime or anywhere. All the information will be stored in a MYSQL database and retrieved by JSP using JDBC. The records are shared with not only authorized users but also with visitors to the site.

The project includes two parts: one is a normal web browser, which uses HTML (Hyper Text Markup Language) as the basic interface language. The other one is the cellular device, which uses WML (Wireless Markup Language) as the basic interface language. The users can record, modify and store their travel experiences anytime or anywhere by using these two tools.

A Portable Traveler's Web Log is an "easy to use" web-based application. Everyone knows how to use a web browser or cellular device, so they can login to record travel experiences or view the traveling web log of the other people who use the system.

## 1.2 Scope of Project

### 1.2.1 Deliverables

The project has produced the following artifacts:

1. A web application project directory organized according to the standard layout described in Tomcat (<http://jakarta.apache.org/tomcat>).
2. Java build file that compiles all java code and generates javadoc.
3. Javadoc for source code developed for system.
4. SQL creation scripts that create the database.
5. Project report containing various UML diagrams, such as class diagrams, use case diagrams, deployment diagrams, etc.

### 1.2.2 Function of Software Products

This system allows the users to modify the site either from a normal web browser, or from a mobile device with WML capabilities.

## 1.3 Significance of the Project

We live in a global society where communications with family and business associates must be maintained across time zone and national borders. This project meets the challenge of providing two distinct user friendly interfaces to shared data. A Portable Traveler's Web Log

system makes use of current technologies to increase reliability and efficiency in the creation and execution of real-life projects.

#### 1.4 Definition and Abbreviations

APTWS - A Portable Traveler's Web Log System.

HTML - Hyper Text Markup Language.

HTTP - Hyper Text Transfer Protocol, the client/Server protocol that define how messages are formatted and transmitted on the World Wide Web.

Java - An object oriented language developed by Sun Microsystems Java programs.

JavaScript - A scripting language that is widely supported in Web browsers and other web tools.

Java Servlet - A Java application that runs in a Web Server and provide server-side processing, typically to access a database.

JDBC - Java database Connectivity. A programming interface that lets Java applications access a database via the SQL language.

JSP - Java Server Page, An extension of the Java servlet technology from Sun, displays dynamic content on the Web Page.

MySQL - Structured Query Language.

WML - The Wireless Markup Language

## 1.5 Organization of the Documentation

The remaining sections of this document will be organized as follows: Chapter 2 introduces the architecture of A Portable Traveler's Web Log. Chapter 3 describes the software requirement specification (SRS). Chapter 4 illustrates database design. Chapter 5 presents project implementation. Chapter 6 discusses the maintenance manual. Chapter 7 contains conclusion and future directions.



## CHAPTER TWO

### A PORTABLE TRAVELER'S WEBLOG ARCHITECTURE

This chapter I will briefly introduce this project, which is to implement a web system that provides an environment for users to record their travel experiences. The system is a 3-tier distributed architecture that displays the user interface in the web browser using HTML, and that displays the user interface in the cellular phone using WML. The middle tier is the Apache Tomcat web server that handles requests from the client browser and provides access to the third tier MySQL via JDBC.

The web application executes a user command:

- User types a URL in web browser.
- Request is transmitted to web server via HTTP protocol.
- Web server response to the request and executes from a JSP page and loaded by the JSP engine.
- Java business logic communicates with database via JDBC.
- JSP generates custom HTML documents or generates custom WML documents and sends them back to the user via the HTTP protocol.

- User's web browser displays HTML page or WML page is displayed by user's cellular phone.

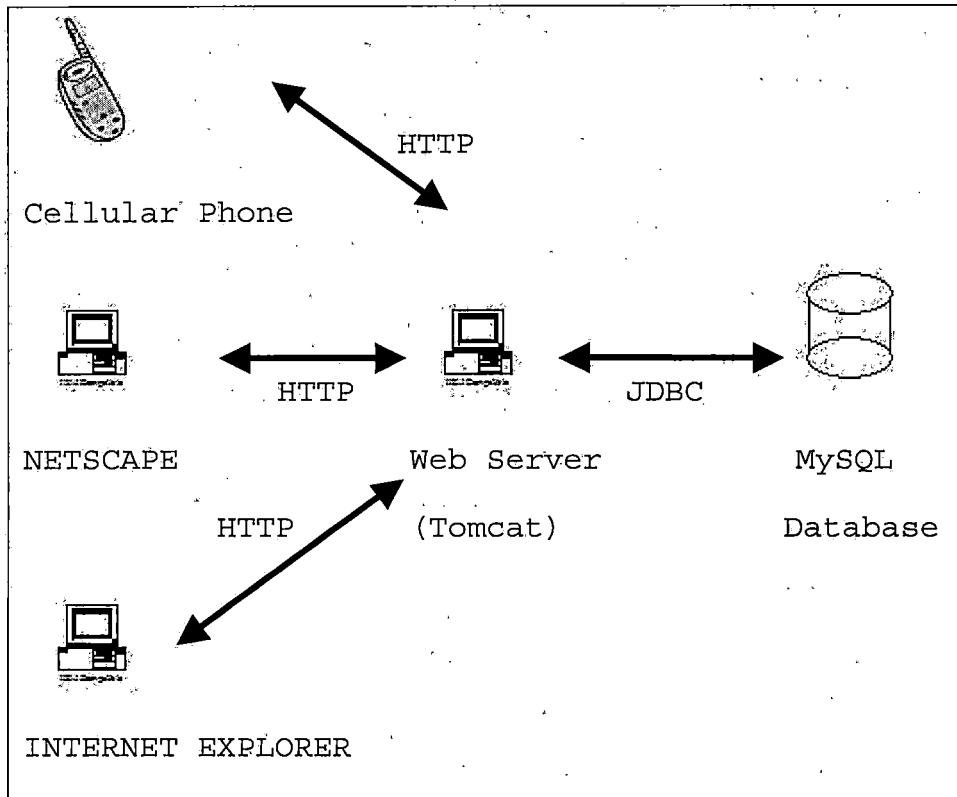


Figure 1. System Architecture

The components used to build APTWS (A Portable Traveler's Web Log System) were chosen with the following criteria: (i) the components should be shareware, i.e., available freely for non-commercial purposes, (ii) they do not depend on a specific operating system and hence are easily portable across systems, (iii) database server independent, so that new and different versions of the server can be plugged in easily.

The user interface components are built by using HTML 6.0 forms, WML, frames and JavaScript. The application is implemented using Java Server Pages (JSP). JSP was used because it can use java business logic and provides a common way for programs to interface with java containers. JSP is an extension of Java Servlet technology. Typically, a Java Servlet can do the same tasks as JSP; however, JSP makes it easy to mix static HTML with Java code.

The database choice available to APTWS is MySQL. MySQL is a real multi-user database and free. Also, the availability of the JDBC driver for MySQL is the most important reason to choose it. Moreover, the same code could be used to link with another version of MySQL database by changing the JDBC driver, thereby making it database independent.

## 2.1 Software Interfaces

- Internet browser: Netscape or Internet Explorer, Cellular Phone.
- Operating system: Windows 98/Me/2000/XP, or Unix/Linux.
- Database: MySQL.
- Compiler: JDK 1.4.
- Language: HTML / JAVA / JavaScript / JSP/ WML.

- Database connector: JDBC.
- JSP Container/Web server: Jakarta Tomcat.

## CHAPTER THREE

### SOFTWARE REQUIREMENTS SPECIFICATION

#### 3.1 Introduction

A Portable Traveler's Web Log is a project aimed at providing people the ability to record and publish their travel experiences. These users can modify and view the site either from a normal web browser, or from a mobile device with WML capabilities. Other visitors can view the site either from a normal web browser, or from a mobile device with WML capabilities.

#### 3.2 Overall Description

##### 3.2.1 Product Perspective

A Portable Traveler's Web Log is web based. The interfaces are via Internet.

The hardware interface requirement is that it must run on the existing web servers. The software interface requirement is that it must support current versions of Netscape, Internet Explorer and WML-ready phones. The communications interface requires support for Hyper-Text Transfer Protocol (HTTP).

### 3.2.2 Product Functions

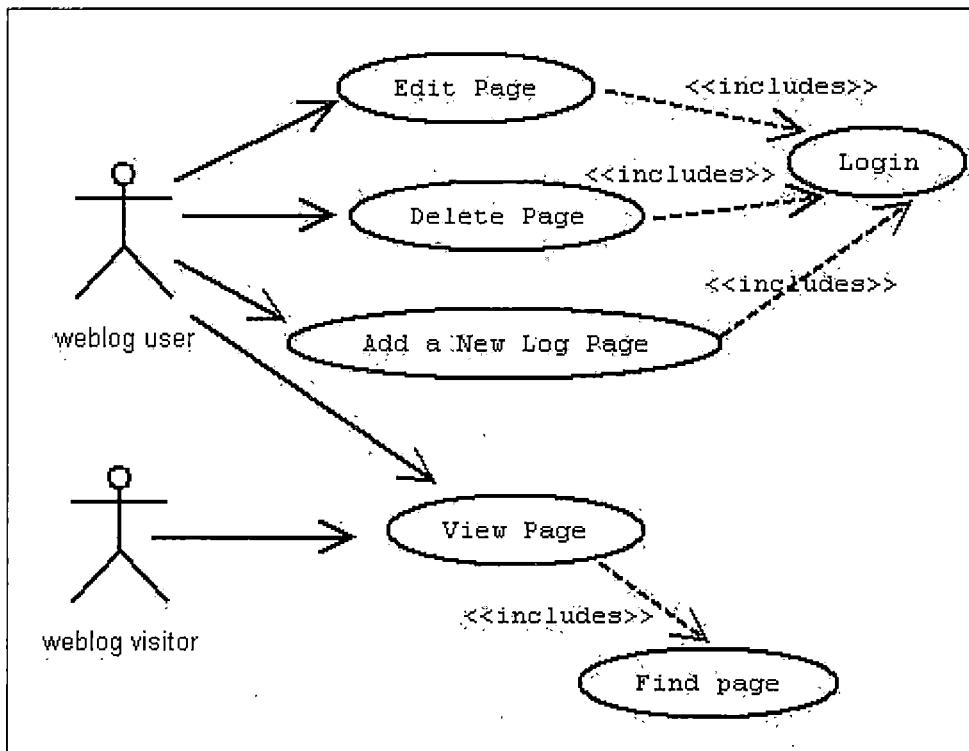


Figure 2. Use Case Diagram

### 3.2.3 Product Architecture

A user task will be conducted in the following steps:

- User types a URL in web browser.
- Request is transmitted to web server via HTTP protocol.
- Web server response to the request and executes from a JSP page and loaded by the JSP engine.
- Java business logic communicates with database via JDBC.

- JSP generates custom HTML documents or generates custom WML documents and sends them back to the user via the HTTP protocol.
- User's web browser displays HTML page or WML page is displayed by user's cellular phone.

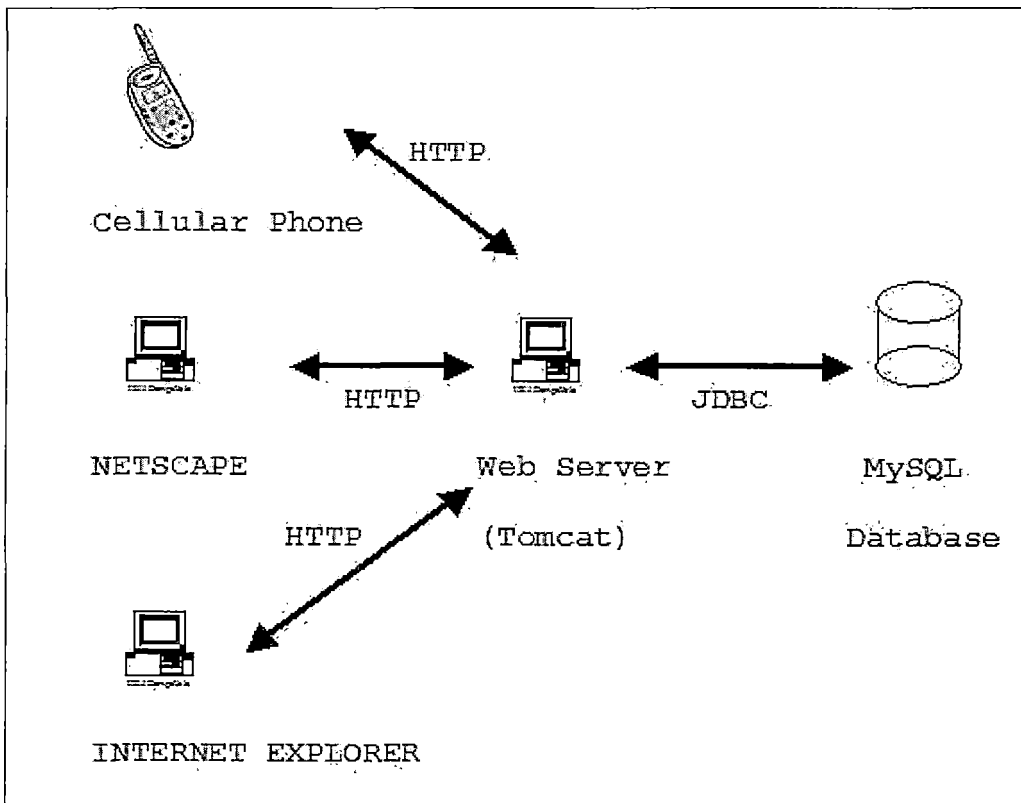


Figure 3. System Architecture

### 3.2.4 State Chart of A Traveler Web Log Session

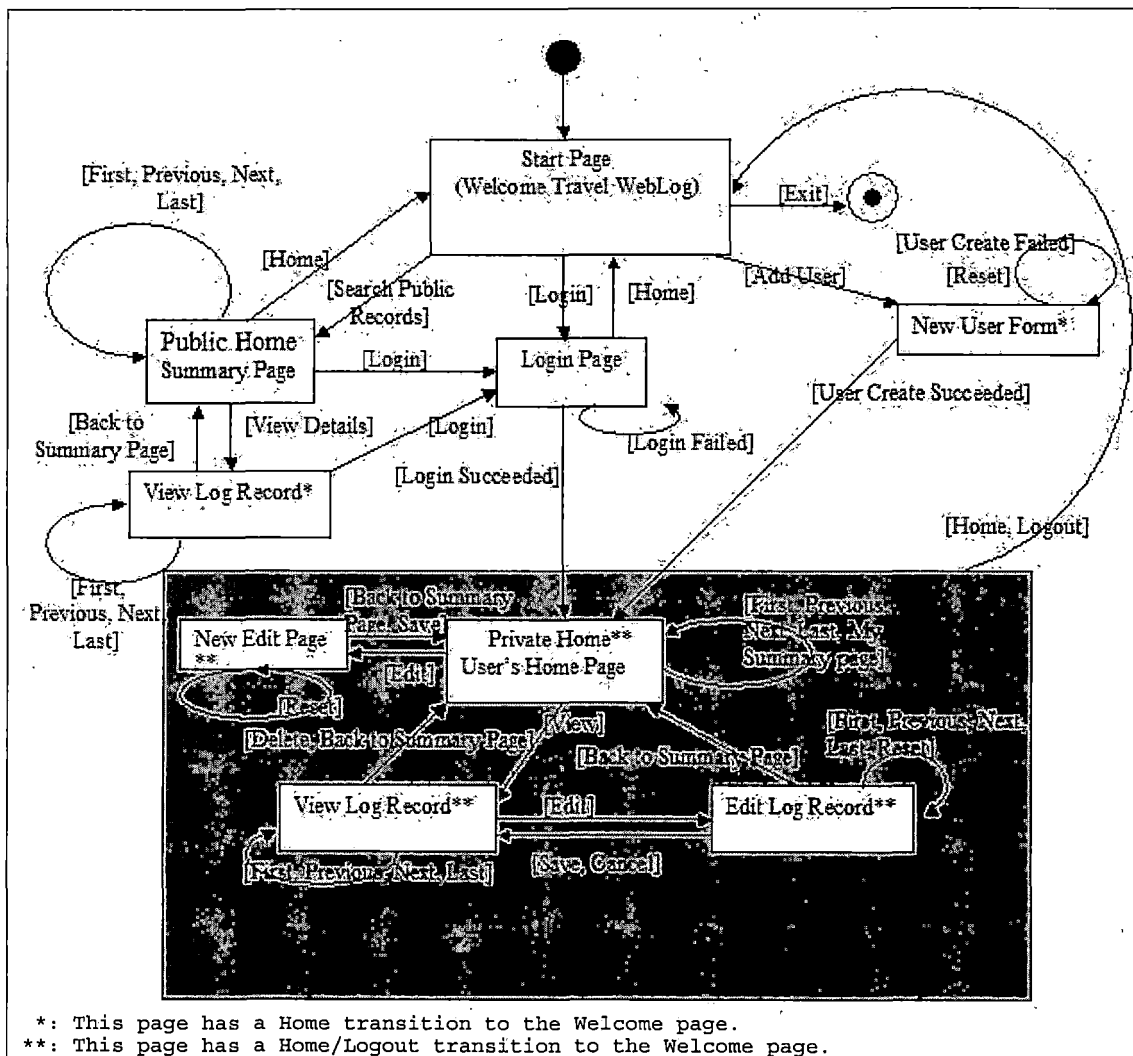


Figure 4. State Chart of A Traveler Web Log Session

### 3.2.5 Design Different of Web Browser and Cellular Phone

1. Screen size: A normal browser can display all necessary in a single page such as log description, create time, modify time, the title of each log entry and other links. Screen Limit



of cellular phone only displays the title of each log entry.

2. Web language: A normal browser uses HTML as the basic language. Cellular phone uses WML as the basic language. For example:

Login page in HTML -

```
<form action = "authenticate.jsp" method="post">
Userid:  <input type = "text" name="userid"
size="24">
Password: <input type="password" name="password"
size="24">
<input type="submit" value="Login">
<input type= "reset" value="reset">
</form>
```

Login page in WML -

```
<card id="c0" title="Login">
<onevent type="onenterforward">
<go href="#login">
<setvar name="userid" value=""></setvar>
<setvar name="password" value=""></setvar>
</go>
</onevent>
<onevent type="onenterbackward">
<prev>
```

```

</onevent>
</card>
<card id="login" title="Login">
User name: <input name='userid' format="*m"/>
Passwörd: <input type='password' name='password'
format="*m"/>
<anchor title="Send">Login
<go method="post" href="wml_authenticate.jsp">
<postfield name="userid"
value="$ (userid) "></postfield>
<postfield name="password"
value="$ (password) "></postfield>
</go>
<anchor>
</card>

```

3. The method of keeping information: A normal browser uses session JSP object to keep information. Cellular phone uses parameter to keep information. For example:

Visit to user's home page in HTML -

```
<a href="wml_publichome.jsp">
```

JSP code -

```
user =
```

```
(travelweblog.User)session.getValue("user");
```

Visit to user's home page in WML -

```
<a
```

```
href="wml_publichome.jsp?ut=3208508_109345882394
```

```
5">
```

JSP code -

```
user =
```

```
(travelweblog.User)application.getAttribute(user  
Token);
```

## CHAPTER FOUR

### DATABASE DESIGN

#### 4.1 Data Analysis

The data for designing and implementing the schema of the database depends on properties of user and logs. In designing the schema for the APWTS database, two distinct parts have been identified. The first includes the log part, which includes log id, log userid, log title, log details, log visibility, log create\_time, and log modify\_time. The second includes user id, user name, and user password which would always be encrypted before storage. All the entities and attributes are detailed in Figure 5.

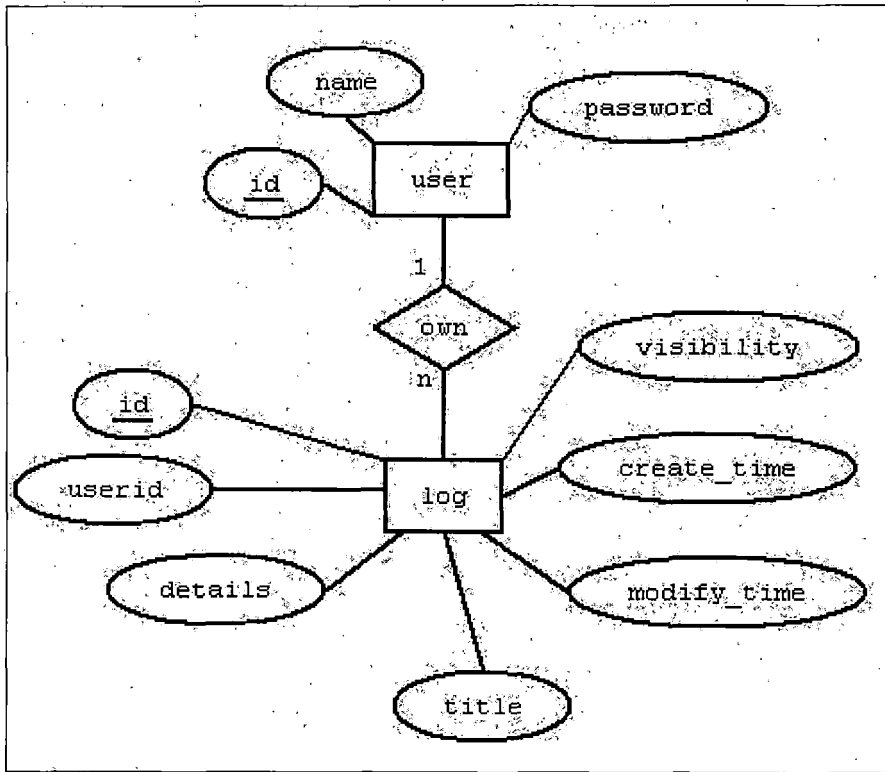


Figure 5. E-R Diagram

#### 4.2 Database Schema Logical Model - Relational Schema

The conceptual model ER diagram maps into the following relational table design. In the following tables, underlined fields indicate the primary key.

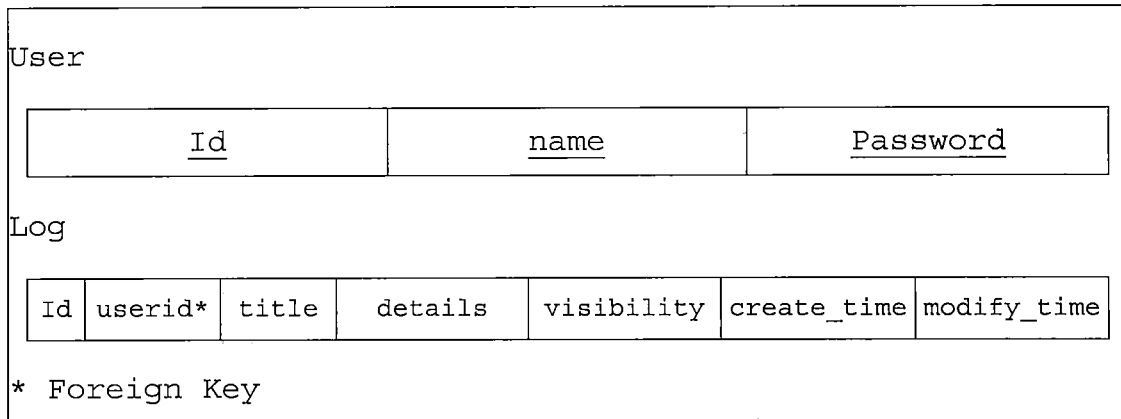


Figure 6. Database Relational Schema

### 4.3 Data Type and Details

The logical model established the following detailed design in MySQL database. The following tables describe data type, length, primary key, null or non-null keys, and extra information such as auto\_increment.

Table 1. Structure of Table Log

Filed	Type	Null	Key	Default	Extra
id	int(11)		PRI	NULL	auto_increment
userid	int(11)	Yes	FOR	NULL	
title	text	Yes		NULL	
details	text	Yes		NULL	
visibility	tinyint(1)	Yes		NULL	
create_time	bigint(20)	Yes		NULL	
modify_time	bigint(20)	Yes		NULL	

Table 2. Structure of Table User

Field	Type	Null	Key	Default	Extra
Id	int(11)		PRI	NULL	auto_increment
name	varchar(50)	Yes	MUL	NULL	
password	varchar(50)	Yes		NULL	

## CHAPTER FIVE

### PROJECT IMPLEMENTATION

APTWS is designed to perform six different functions for 2 different users. Refer to Figure 2 is the Use case Diagram of this project.

#### 5.1 Graphical User Interface Normal Browser Website

User interfaces for the Portable Traveler's Web Log System are designed as HTML pages. The contents are generated dynamically by JSP in response to the user's requests. APTWS GUI is an "easy to use" system. The GUI is written using Hyper Text Markup Language (HTML) Version 6.0 forms. The APTWS GUI is executable under Internet Explorer 5.0 or greater. The following sub-section explains the GUI functions and details.

##### 5.1.1 Welcome Page - Registered Users

This page is the first page that all the users see when they enter APTWS. This page offers the traveling web log information including three links: (1) Click Here to Register: This link is for a visitor who want to register in APTWS. (2) Registered Users: This link is for the existed users to edit and record their web log. (3) View



Public Logs: This link is for all people to browse the pages that are published as public.

In travel page 1, if the user clicks "Registered Users", then the browser automatically goes to Page 2 "Login" page. The visitors see only links to pages with the public attribute that is set to true.

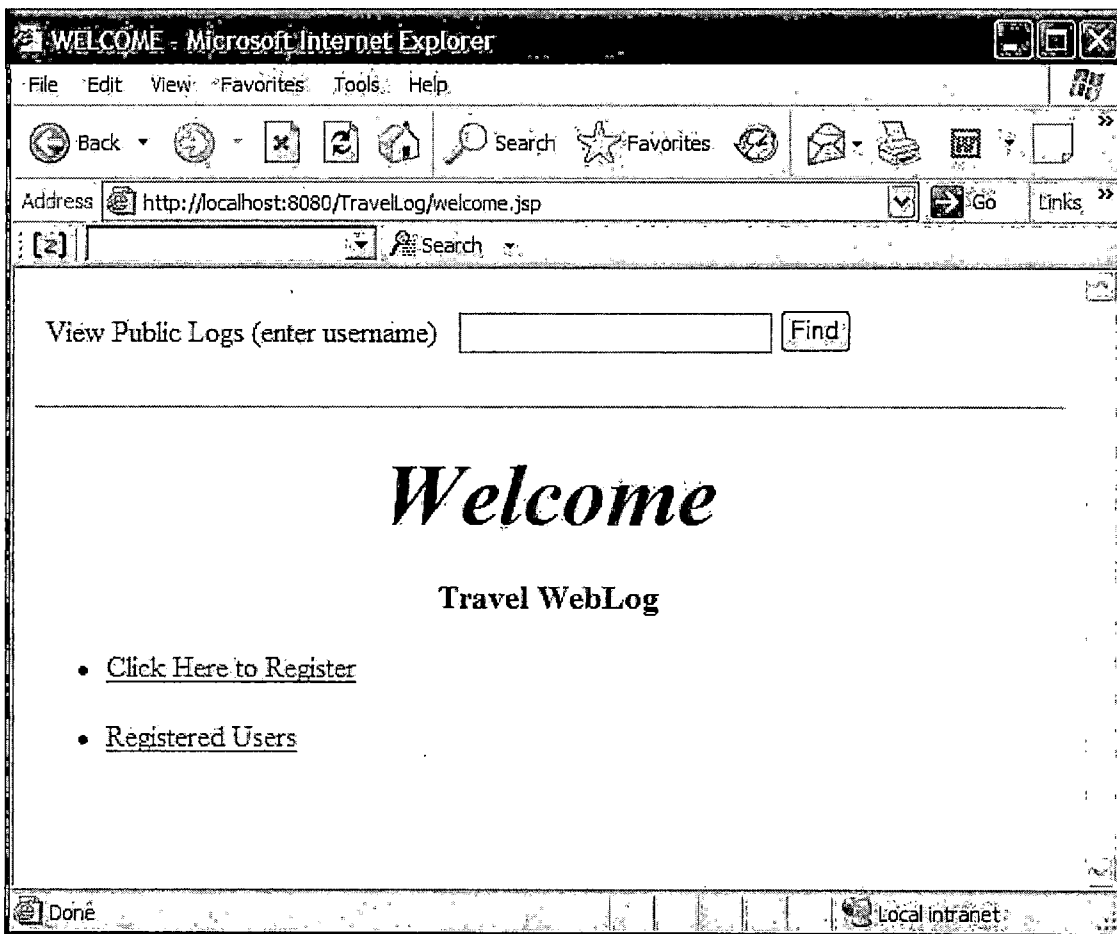


Figure 7. Welcome Page - Registered Users

### 5.1.2 Login Page (For Logged in User)

The user logs in by providing a user id and a password. If the user types in user id and the password, and clicks "Login" button, then the server checks to see if user id and password is correct. When the user id and password is correct, the server sets an attribute in the session object to indicate that the user is authorized to access the user's web log page of "Private Home." If a user fails to provide the correct user id and password in the login page, then she is sent back to the page she came from, without any user's web log page of "Private Home" functions visible.

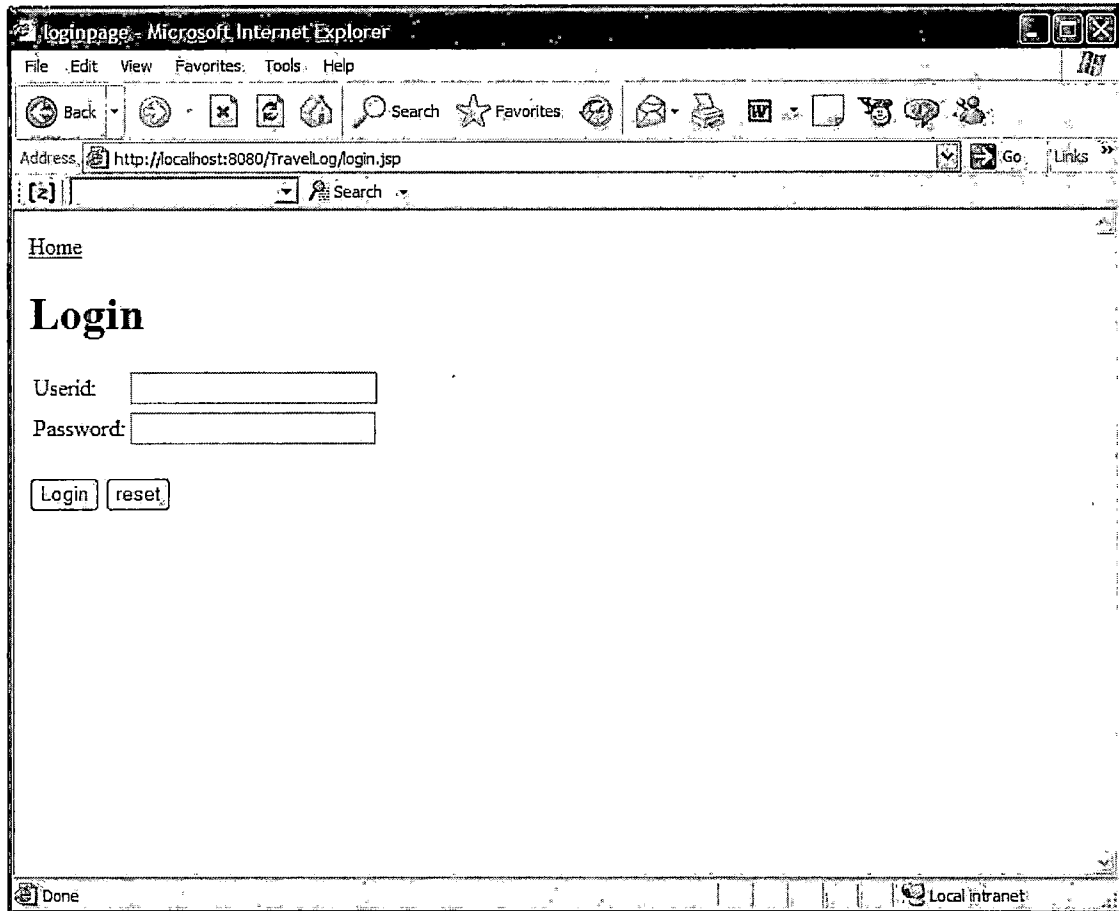


Figure 8. Login Page (For Logged in User)

### 5.1.3 Private Home Page (For Logged in User)

The logged in user can link to all pages, public and private, and add a new log or edit an old page.

After the user clicks "Login" button on login page, she sees the old page. Every page has three records.

Because she is logged in, she also sees eight links:

- (1) Home: The browser automatically goes to page 1 (Welcome Page), but the user is still logged in.
- (2) Logout: The browser automatically goes to page 1

(Welcome Page), but the user is not logged in. (3) My Summary Page: The browser automatically goes to "Private Home" page (for logged in user). (4) Add a new log: The browser automatically goes to "New Edit Page." (5) First: Press this link will display the first page of the user's web log. If the current page is first page, the First link and Previous link will be disable. (6) Previous: Press this link will display the previous page of the current page. (7) Next: Press this link will display the next page of the current page. (8) Last: Press this link will display the last page of the user's web log. If the current page is last page, the Next link and Last link will be disabled.

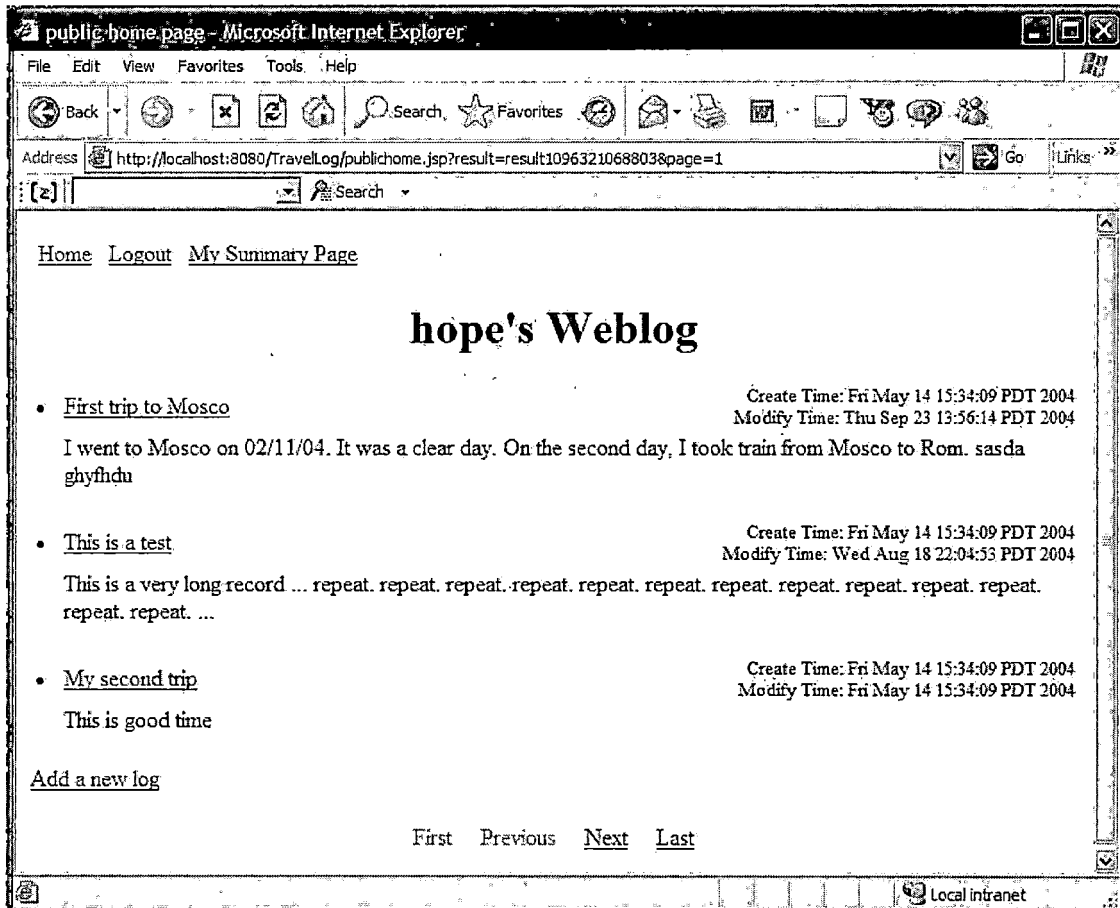


Figure 9. Private Home Page (For Logged in User)

#### 5.1.4 Private Home - View Log Record (For Logged in User)

After logged in user clicks one record of the user's web log of the "Private Home" page, the browser automatically goes to "View Log Record" page. This page includes seven links and two buttons.

Seven links are (1) Home: The browser automatically goes to page 1 (Welcome Page), but the user is still logged in. (2) Logout: The browser automatically goes to

page 1 (Welcome Page), but the user is not logged in.

(3) Back to Summary Page: The browser automatically goes to "Private Home" page (for logged in user). (4) First: Press this link will display the first record of the user's web log. If the current record is first record, the First link and Previous link will be disable. (5) Previous: Press this link will display the previous record of the current record. (6) Next: Press this link will display the next record of the current record. (7) Last: Press this link will display the last record of the user's web log. If the current record is last record, the Next link and Last link will be disable.

Two buttons are (1) Edit: The browser automatically goes to "Edit Log Record" page. (2) Delete: The current page is deleted, and then the browser automatically goes to "Private Home" page.

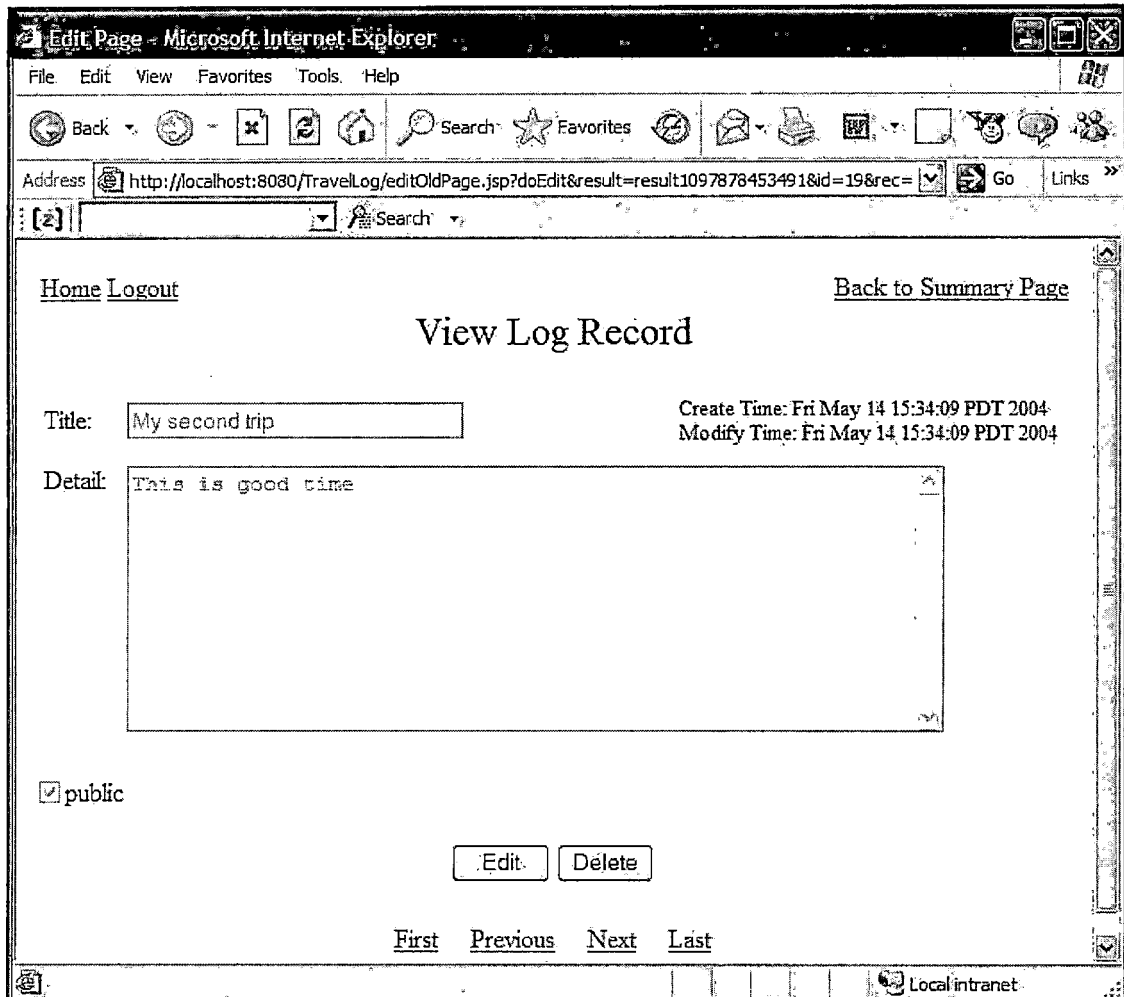


Figure 10. Private Home - View Log Record (For Logged in User)

#### 5.1.5 Private Home - Edit Log Record (For Logged in User)

After the user clicks "Edit" button on "View Log Record" page, the user goes to the "Edit Log Record" page. Because the user is logged in, this page is used to modify the contents of a travel experience page. This page includes three links, three buttons and a check item.

Three links are (1) Home: The browser automatically goes to page 1 (Welcome Page), but the user is still logged in. (2) Logout: The browser automatically goes to page 1 (Welcome Page), but the user is not logged in. (3) Back to Summary Page: The browser automatically goes to "Private Home" page (for logged in user).

Three buttons are (1) Save: A new revised record is stored in the database and the browser automatically goes to "View Log Record" page. (2) Reset: A new revised record is erased on current page. (3) Cancel: The browser automatically goes to "View log Record" page without saving any revised record.

A check item is "public": This page (assuming it is marked public) displays both the "Public Home" page for the visitor and "Private Home" page for logged in user. If this page (assuming it is not marked public) only displays the "Private Home" page for logged in user.



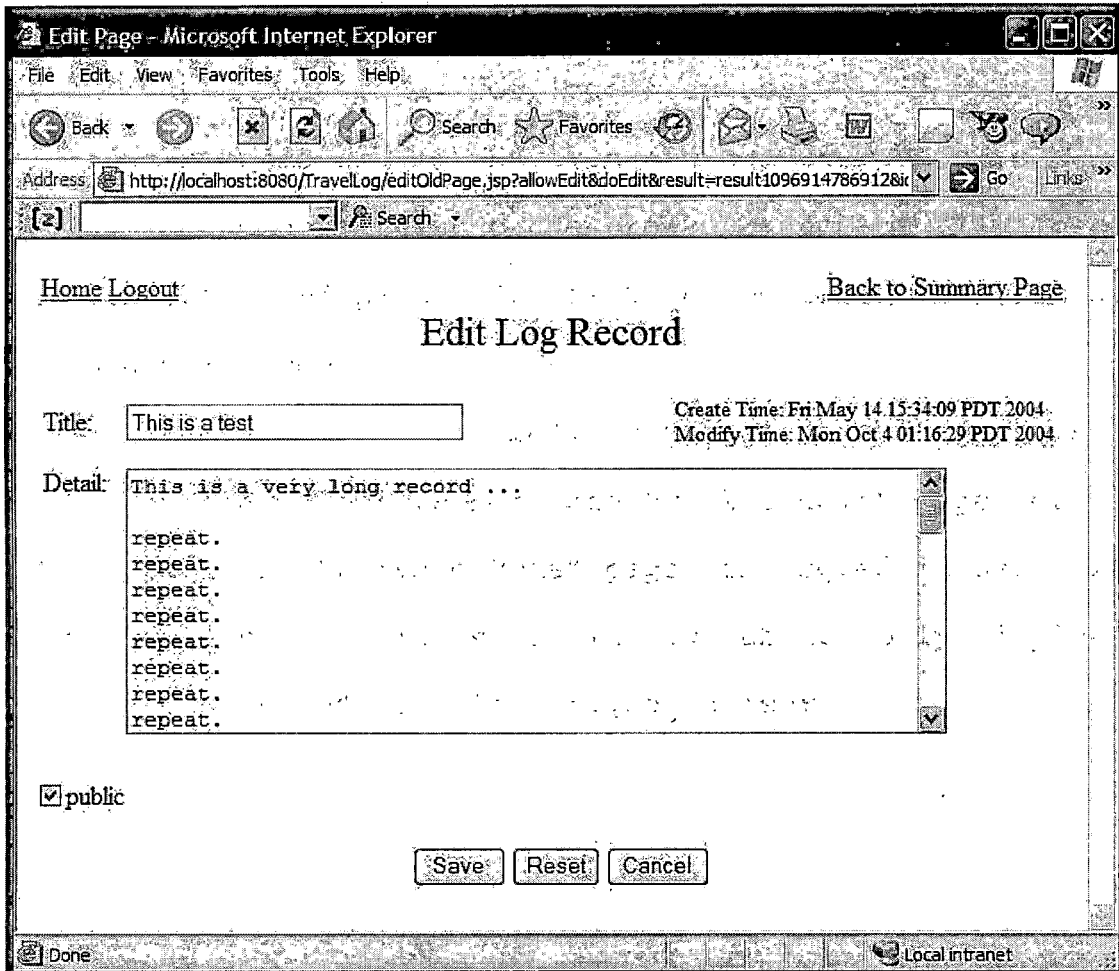


Figure 11. Private Home - Edit Log Record (For Logged in User)

#### 5.1.6 Private Home - New Edit Page (For Logged in User)

A new entry page. This page allows the user to record a new travel experience. This page includes three links, two buttons and a check item.

Three links are (1) Home: The browser automatically goes to page 1 (Welcome Page), but the user is still logged in. (2) Logout: The browser automatically goes to page 1 (Welcome Page), but the user is not logged in.

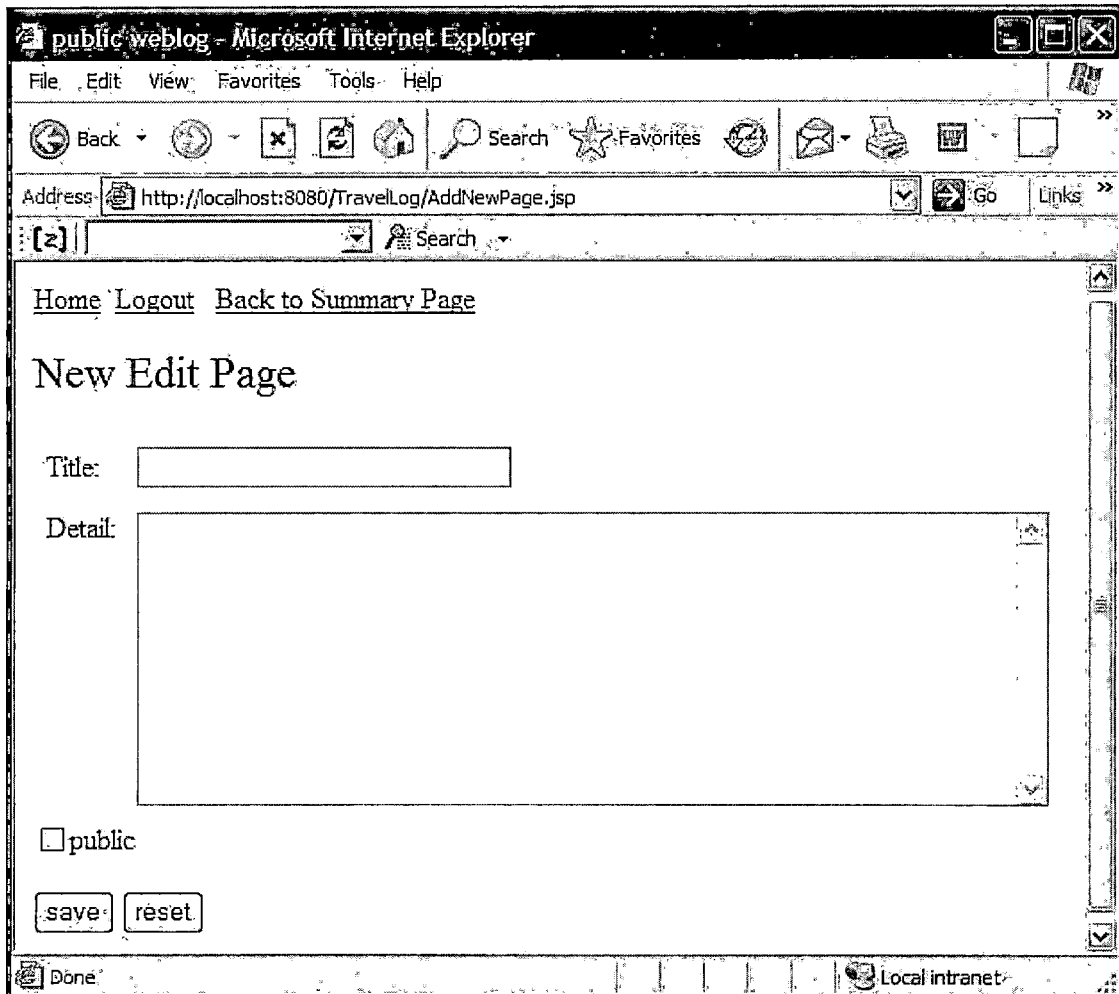


Figure 12. Private Home - New Edit Page (For Logged in User)

#### 5.1.7 Welcome Page - View Public Logs

In travel page 1, the user clicks "View Public Logs," then the browser automatically goes to the "Public Home" page, which is for the visitor. The visitor sees only links to pages with the public attribute that is set to true (see Figure 7).

#### 5.1.8 Public Home Page (For the Visitor)

The visitor sees only links to pages with the public attribute that is set to true.

After the visitor clicks "View Public Logs," then the browser goes to the user's web log. This page includes six links are (1) Home: This link is for the visitor to go back "Welcome" page. (2) Login: The link is for the visitor to go "Login" page. (3) First: Press this link will display the first page of the user's web log. If the current page is first page, the First link and Previous link will be disable. (4) Previous: Press this link will display the previous page of the current page. (5) Next: Press this link will display the next page of the current page. (6) Last: Press this link will display the last page of the user's web log. If the current page is last page, the Next link and Last link will be disable.

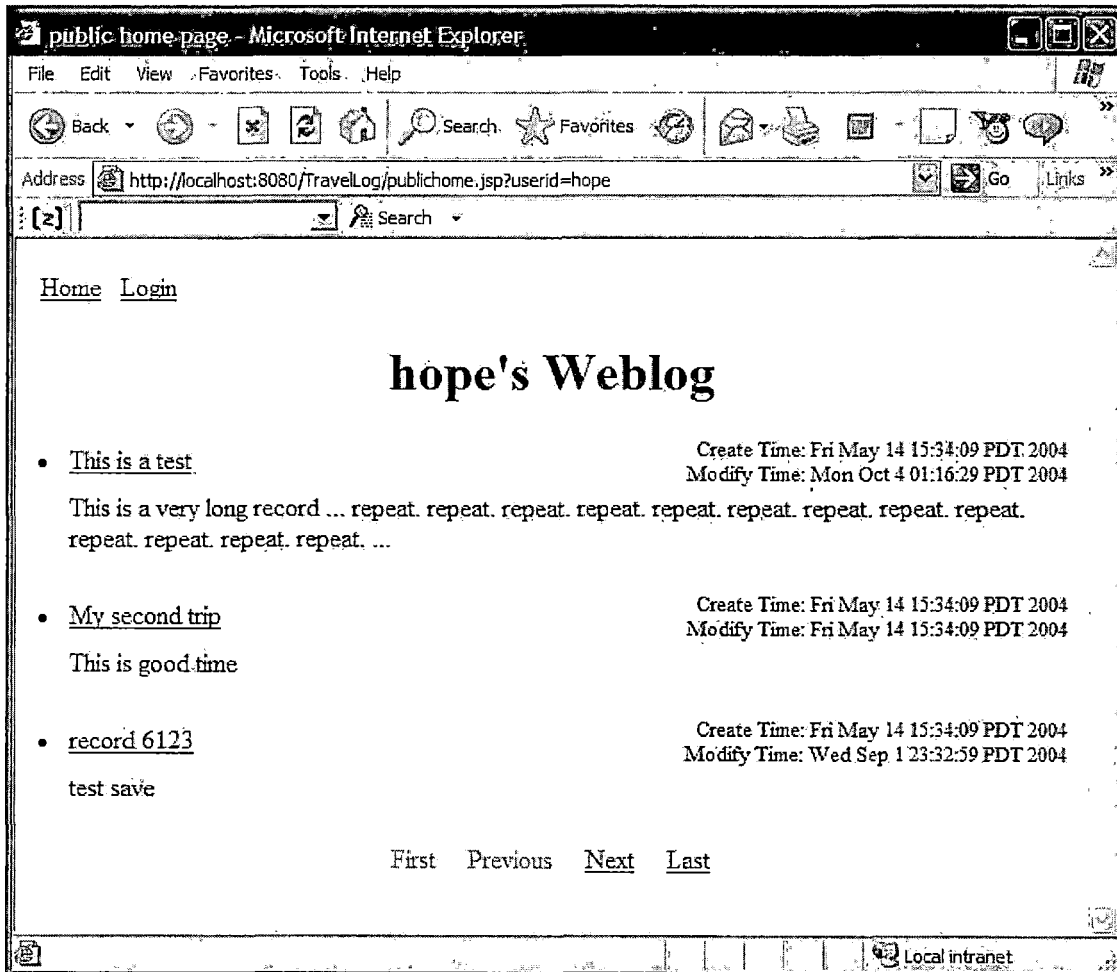


Figure 13. Public Home Page (For the Visitor)

#### 5.1.9 Public Home - View Log Record (For the Visitor)

After the visitor clicks one record of the user's web log in "Public Home" page, the browser automatically goes to "View Log Record" page. This page includes seven links.

Three links are (1) Home (2) Login(3) Back to Summary page(4) First(5) Previous(6) Next(7) Last.

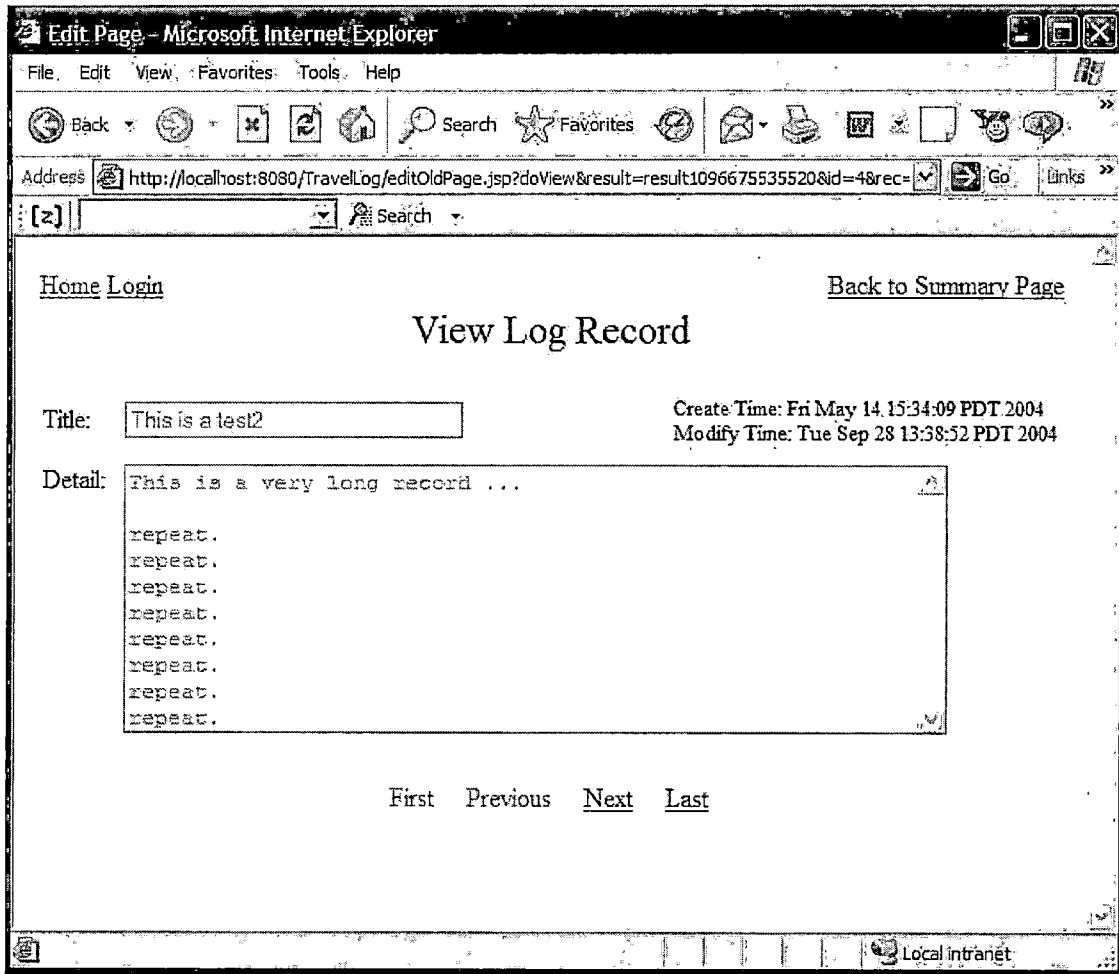


Figure 14. Public Home - View Log Record (For the Visitor)

#### 5.1.10 Welcome Page - Click Here to Register

In travel page 1, when the visitor clicks the "Click Here to Register," the browser automatically goes to the "New User Form" page, which is for the visitor to register an account in the APTW system (see Figure 7).

#### 5.1.11 New User Form (For the Visitor)

The visitor wants to register a new account in the APTW system to record their travel experiences.

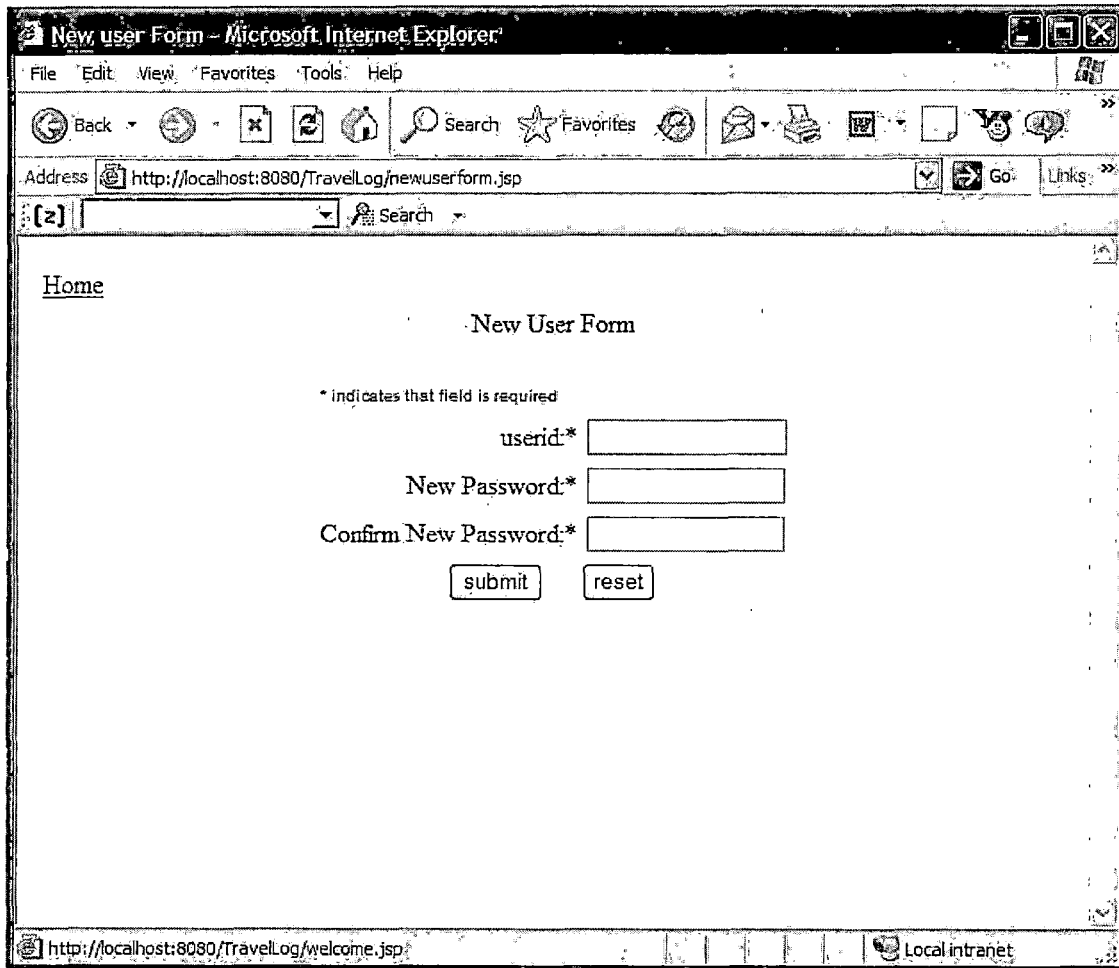


Figure 15. New User Form (For the Visitor)

## 5.2 Graphical User Interface Wireless Make Language-Ready Phone Simulator

User interfaces for the Portable Traveler's Web Log System are designed as WML pages. The contents are generated dynamically by JSP, JavaScript in response to the user's requests. The APTWS GUI is "easy to use" system. The GUI is written using wireless Make Language (WML) Version 1.1 forms. The APTWS GUI is executable under

Openwave SDK6.2 or greater. The following sub-section explains the GUI functions and details.

#### 5.2.1 Wireless Make Language Welcome Page - Registered Users

This page will be the first page that all the users will see when they enter the APTWS. This page offers the traveling web log information including three links:

(1) New User: This link is for non-logged in user who wants to register in the APTWS. (2) Registered Users: This link is for the existing users to edit and record their web log. (3) Search: This link is for all people to browse the pages that are published as public.

In travel page 1, when the user clicks "Registered Users", the browser automatically goes to Page 2 "Login." The visitor sees only links to pages with the public attribute that is set to true.



Figure 16. Wireless Markup Language Welcome Page - Registered Users

#### 5.2.2 Wireless Markup Language Login Page (For Logged in User)

The user logs in by providing a user id and a password. If the user types in user id and the password, and clicks "Login", then the server checks to see if user id and password is correct. When the user id and password is correct, the server sets an attribute in the parameter to indicate that the user is authorized to access the user's web log of "Private Home." If a user fails to provide the correct user id and password in the login page,



then she is sent back to the page she came from, without any user's web log pages of "Private Home" visible.



Figure 17. Wireless Markup Language Login Page (For Logged in User)

### 5.2.3 Wireless Markup Language Private Home Page (For Logged in User)

After login, the logged in user can link to all pages, public and private, and add a new log or edit an old page. The user can click "Menu" button, the screen displays four links: (1) Home: The browser automatically goes to "Welcome" page, but the user still login in. (2) Logout: The browser automatically goes to "Welcome" Page, but the

user is not logged in. (3) Add New Log: The user want to write a new log and store it in the database. (4) Summary Page: The browser automatically goes to "Private Home" page (for logged in user).

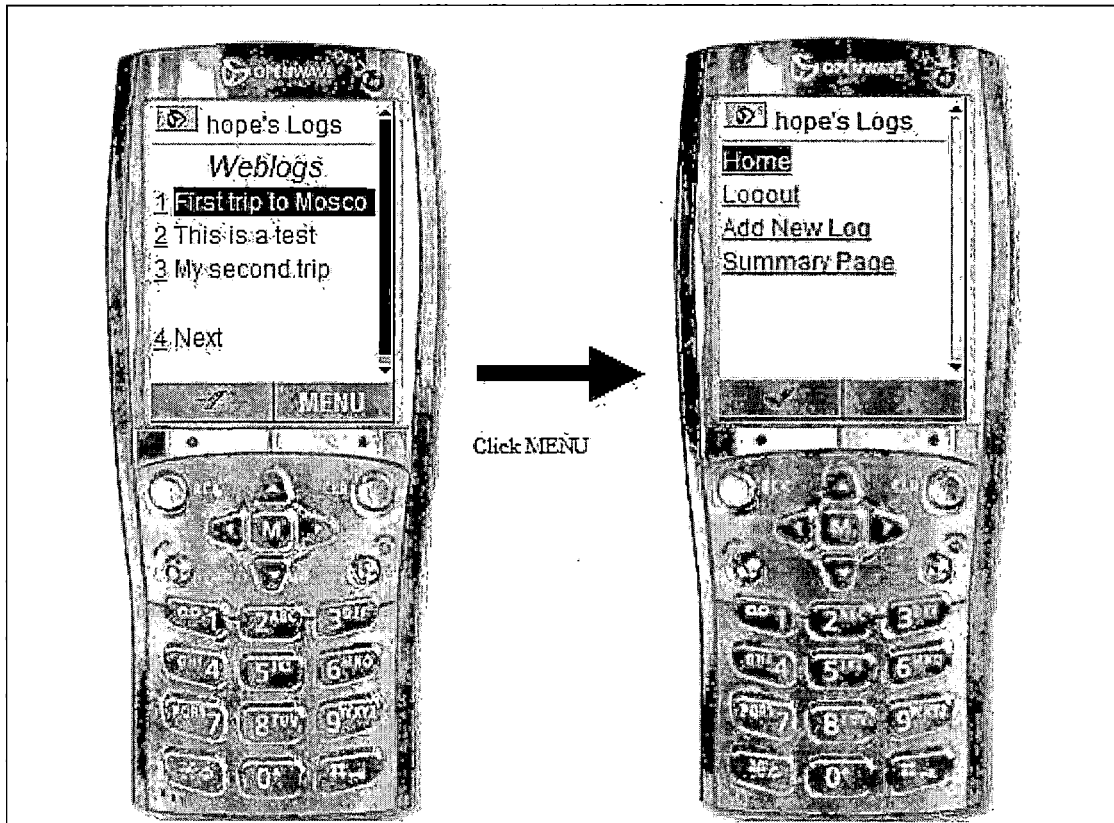


Figure 18. Wireless Markup Language Private Home Page (For Logged in User)

#### 5.2.4 Wireless Make Language Private Home - View Record (For Logged in User)

After logged in user clicks one record of the user's web log of "Private Home" page, the browser automatically goes to "View Log Record" page. When logged in user clicks "Menu" button, the screen displays five links: (1) Edit:

The browser automatically goes to "Edit Record" page.

(2) Delete: A record is deleted in the database, and then the browser automatically goes to "Private Home" page.

(3) Home: The browser automatically goes to "Welcome" page, but the user is still logged in. (4) Logout: The browser

automatically goes to "Welcome" Page, but the user is not logged in. (5) Summary Page: The browser automatically

goes to "Private Home" page (for logged in user).

If the logged in user clicks the edit link, then she goes to the "Edit Record" page.

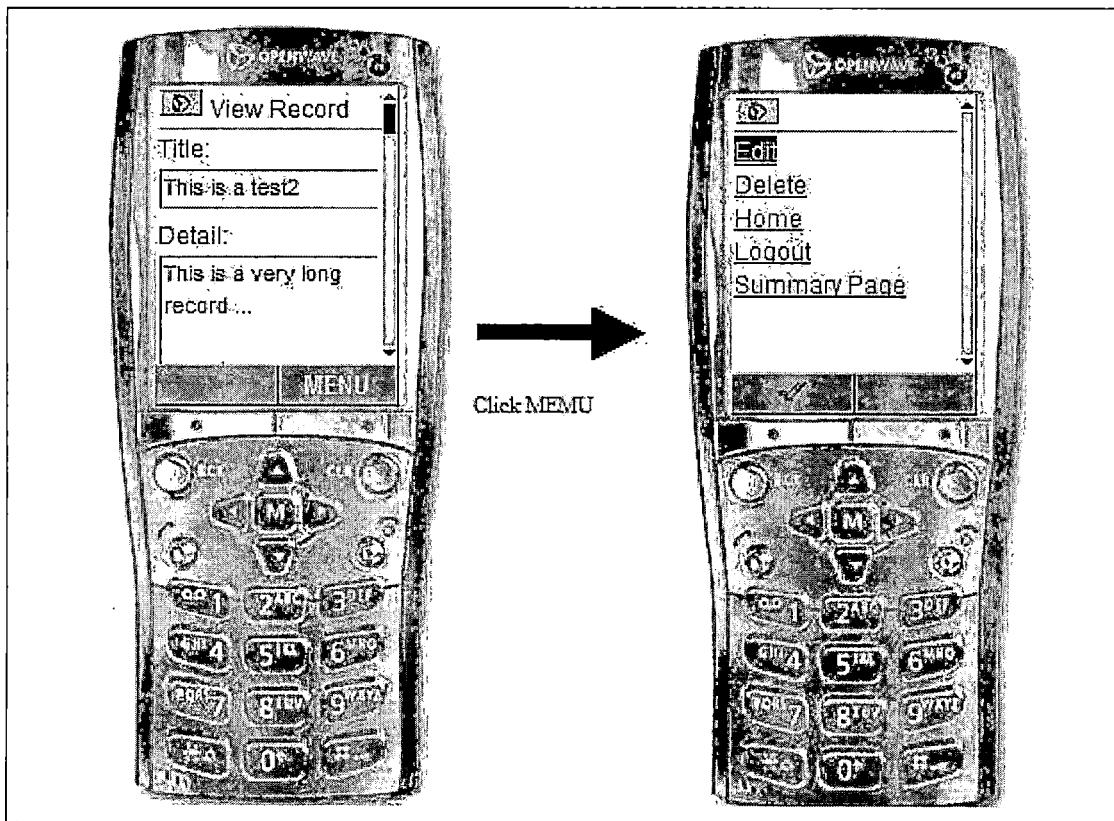


Figure 19. Wireless Markup Language Private Home - View Record (For Logged in User)

#### 5.2.5 Wireless Make Language Private Home - Edit Record Page (For Logged in User)

After the user clicks "Edit" of MEMU button of the "View Log Record" page, the user goes to the "Edit Record" page. Because the user is logged in, this page is used to modify the contents of a travel experience page. A check item is "public": This page (assuming it is marked public) displays both the "Public Home" page for the visitor and "Private Home" page for logged in user. If this page (assuming it is not marked public) only displays the "Private Home" page for logged in user. When the logged in user clicks the "MENU" button, the screen displays four links: (1) Save: A revised travel record is stored in the database and the browser automatically goes to "View Record" page. (2) Home: The browser automatically goes to page 1 (Welcome Page), but the user is still logged in. (3) Logout: The browser automatically goes to page 1 (Welcome Page), but the user is not logged in. (4) Summary Page: The browser automatically goes to "Private Home" page (for logged in user).

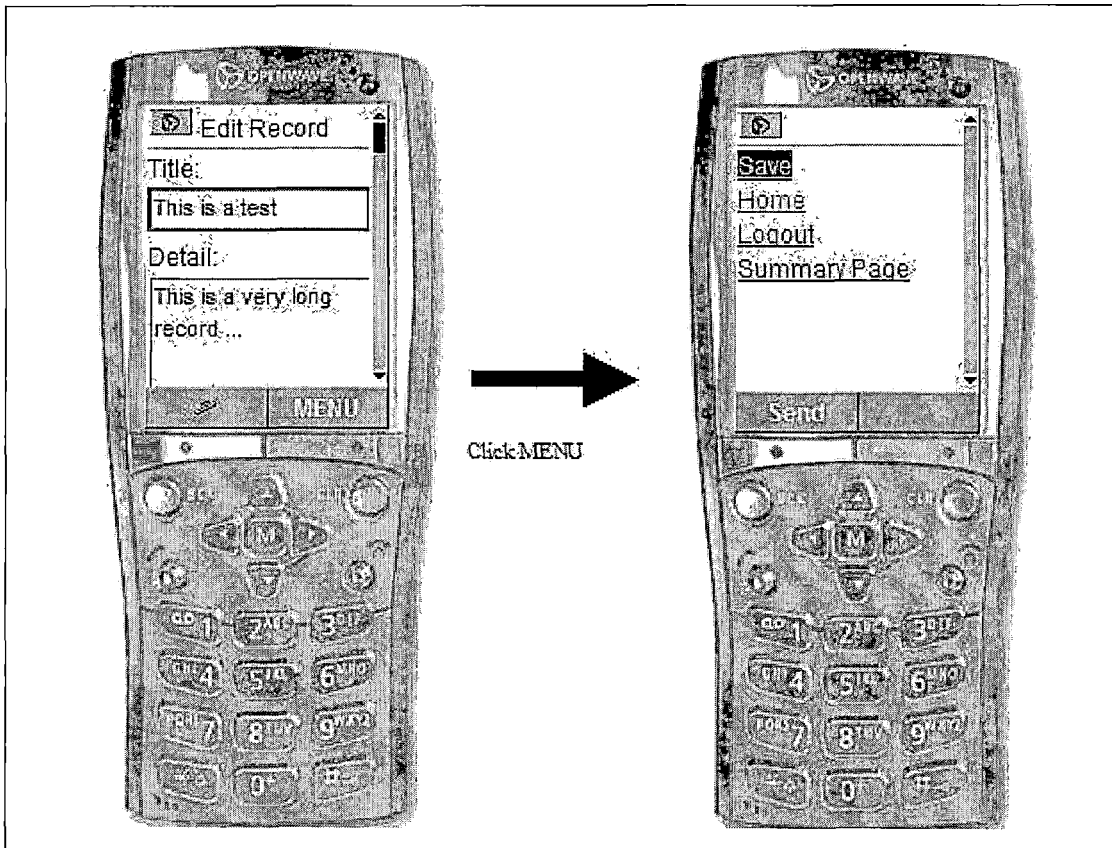


Figure 20. Wireless Markup Language Private Home - Edit Log Page (For the Logged in User)

#### 5.2.6 Wireless Make Language Private Home - New Page (For Logged in User)

A new entry page. This page allows the user to record a new travel experience. This page is similar to the "Edit Record" page in functionality. A check item is "public": This page (assuming it is marked public) displays both the "Public Home" page for the visitor and "Private Home" page for logged in user. If this page (assuming it is not marked public) only displays the "Private Home" page for logged in user.

When logged in user clicks "MENU" button, the screen displays four links: (1) Save: A new travel record is stored in the database and the browser automatically goes to "Private Home" page. (2) Home: The browser automatically goes to page 1 (Welcome Page), but the user is still logged in. (3) Logout: The browser automatically goes to page 1 (Welcome Page), but the user is not logged in. (4) Summary Page: The browser automatically goes to "Private Home" page (for logged in user).

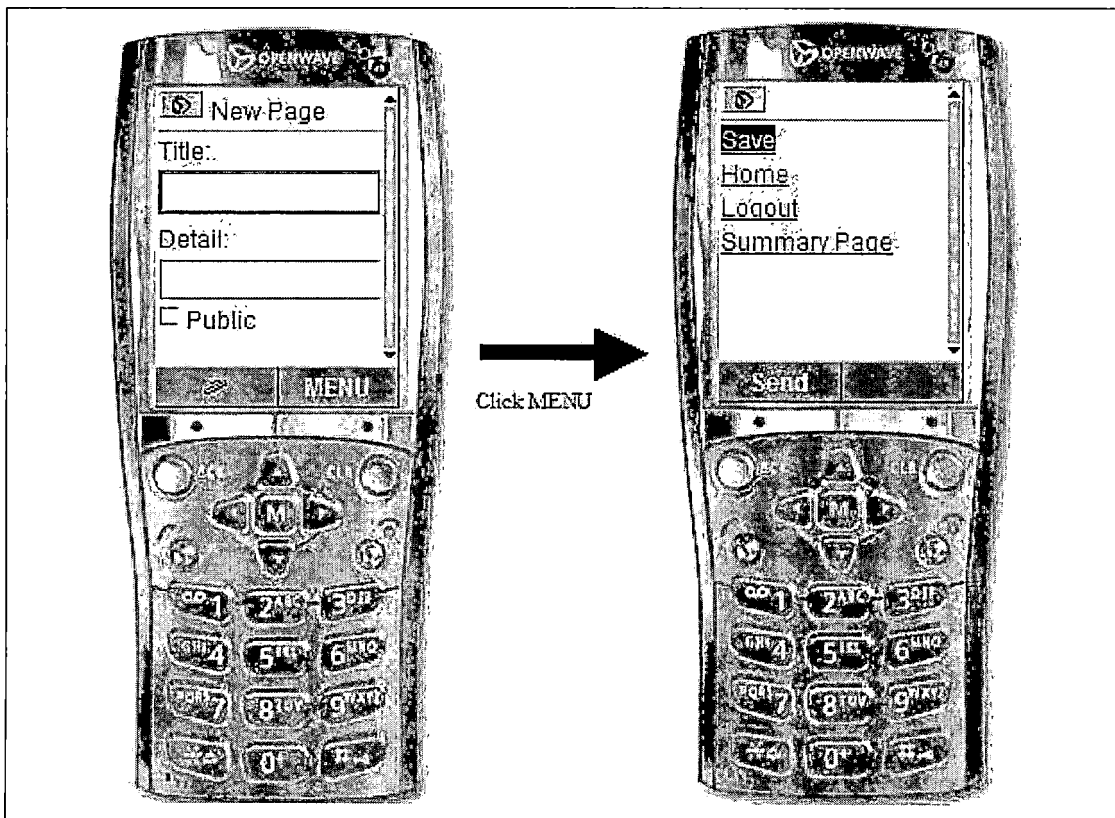


Figure 21. Wireless Markup Language Private Home - New Page (For Logged in User)

### 5.2.7 Wireless Markup Language Welcome Page - Search

In travel page 1, when the user clicks "Search" on the "Welcome" page, the browser automatically goes to "View Public Logs" page. The visitor sees only links to pages with the public attribute that is set to true.

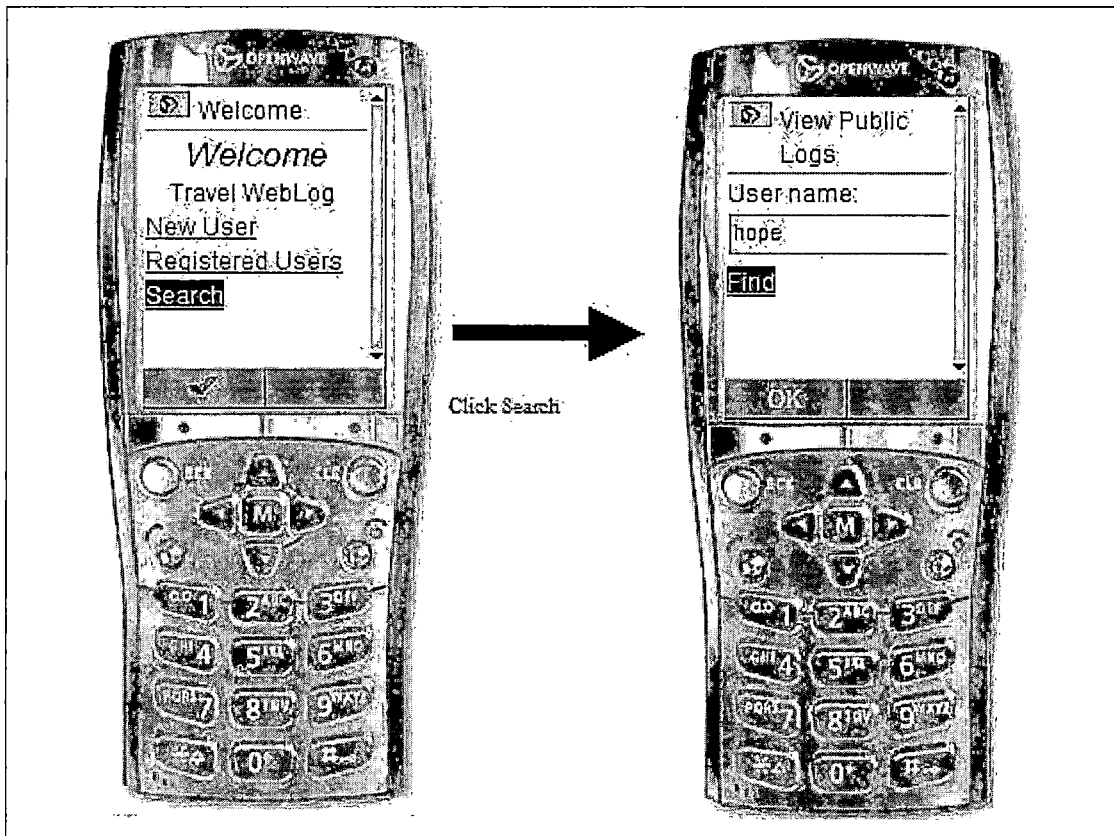


Figure 22. Wireless Markup Language Welcome Page - Search

### 5.2.8 Wireless Markup Language Public Home Page (For the Visitor)

The visitor sees only links to pages with the public attribute that is set to true. In the "user's Logs" of the "Public Home" page, the visitor can click "MENU" button,

the screen displays two links: (1) Home: The browser automatically goes to page 1 (Welcome Page). (2) Login: The browser automatically goes to "Login" page.

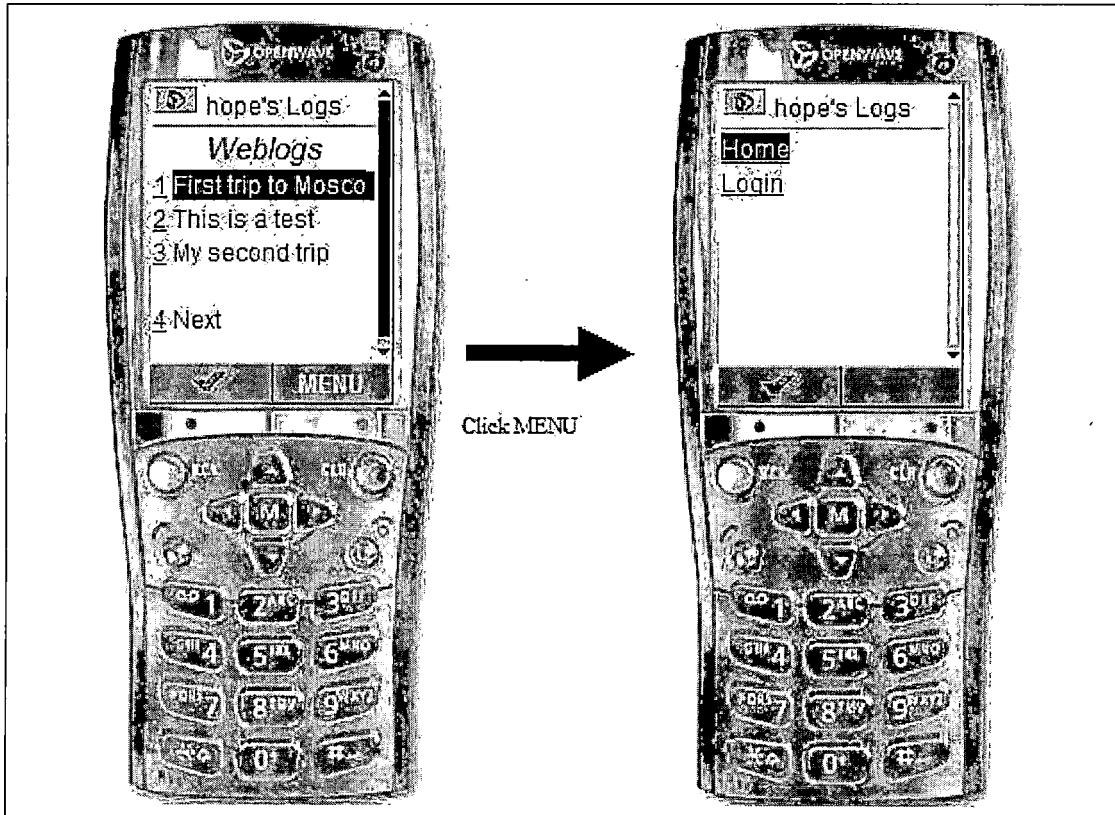


Figure 23. Wireless Markup Language Public Home Page (For the Visitor)

#### 5.2.9 Wireless Markup Language Public Home - View Record (For the Visitor)

After the visitor clicks one record of the user's web log of "Public Home" page, the browser automatically goes to "View Record" page. When the visitor can click "MENU" button, the screen displays three links: (1) Home: The browser automatically goes to page 1 (Welcome Page).



- (2) Login: The browser automatically goes to "Login" page.
- (3) Summary Page: The browser automatically goes to "Public Home" page (for the visitor).

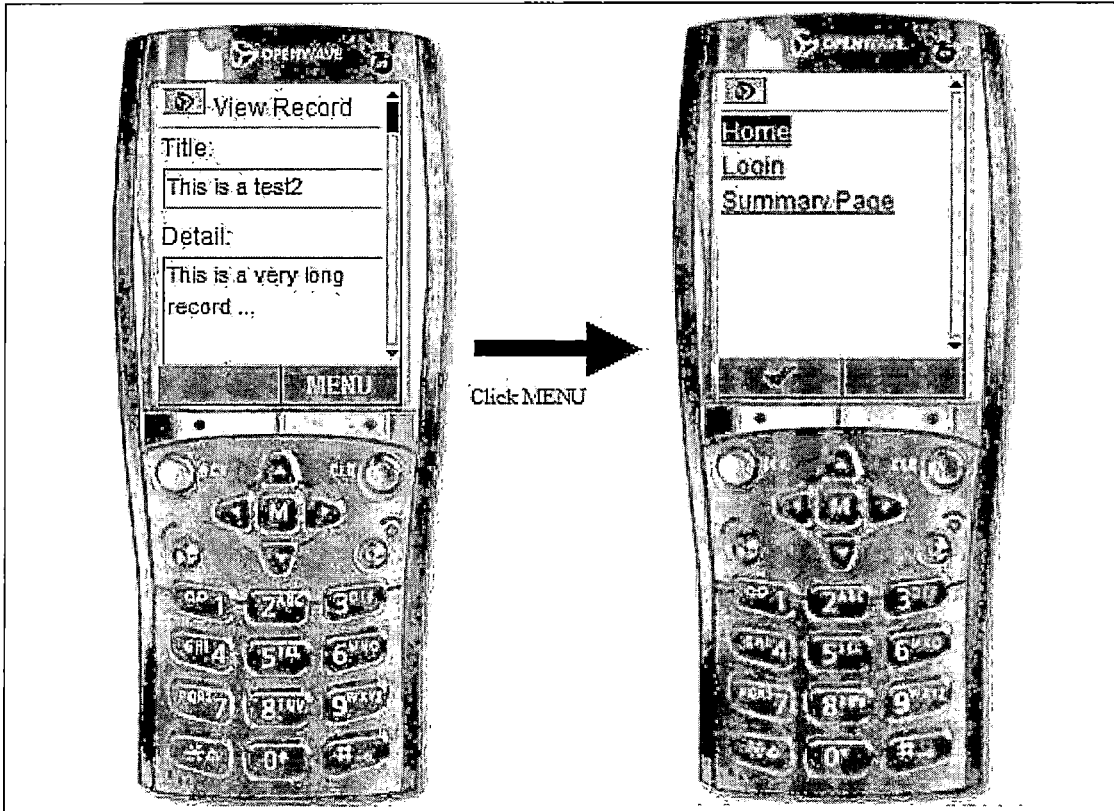


Figure 24. Wireless Markup Language Public Home - View Record (For the Visitor)

#### 5.2.10 Wireless Markup Language Welcome Page - New User (For the Visitor)

In travel page 1, when the user clicks "New User", the browser automatically goes to the "New User" page, which is for the visitor to register an account in the APTW system.



Figure 25. Wireless Make Language Welcome Page - New User  
(For the Visitor)

#### 5.2.11 Wireless Make Language New User Form (For the Visitor)

The visitor wants to register a new account in the APTW system to record their travel experiences.



Figure 26. Wireless Markup Language New User Form (For the Visitor)

## CHAPTER SIX

### MAINTENANCE MANUAL

System maintenance is an important step to ensure that the system runs smoothly and meets the expectation of the users. In this project, there are 3 major issues: Software Installation, Variables Modification, and APTWS Installation.

#### 6.1 Software Installation

APTWS requires MYSQL, TOMCAT, and JDBC to run. The following will detail the installation of these software packages.

##### 6.1.1 JAVA 2 Platform, Standard Edition

JAVA 2 Platform, Standard Edition (J2SE) is the compiler program for JSP programs and the TOMCAT Container requires it. First of all, we go to <http://java.sun.com/j2se/1.4.1/download.html> to download SDK Windows (all languages, including English), and then install it.

##### 6.1.2 Structured Query Language Installation

Structured Query Language (MySQL) is the database system we use in the APTWS and it is free. Because it also provides JDBC to easily connect by JAVA program, thus it is good choice for designing this project. First of all,

we need to download MYSQL 3.23 for windows 95/98/2000/XP at <http://www.mysql.com/downloads/mysql-3.23.html>. After downloading the compress file, please unzip the file and install it. Second, in DOS command, we type

```
C:\Document and Setting>cd \mysql.sav\bin
```

Then we install the MYSQL service in our server. Third, we type

```
C:\mysql.sav\bin>mysqlshow
```

To start our service. Forth, we have to setup the user and password, the user is named javauser, and so we have to set its password to javadude.

```
C:\mysql.sav\bin>>mysql javatest -u javauser -p
```

```
Enter password:javadude
```

After that, we can try to input following command:

```
mysql> select * from user;
```

```
mysql> exit;
```

```
C:\mysql.sav\bin>
```

Then, we have already installed MYSQL and it's working as expected.

### 6.1.3 JAVA Database Connectivity

The API used to execute SQL statement is different for each database engine. Java programmers, however, are because they free from such database portability issues. They have a single API, the Java Database Connectivity API

(JDBC), that's portable between database engines. The JDBC library provides an interface for executing SQL statements. It provides the basic functionality for data access. A number of drivers are available for MySQL, and information about this can be obtained at the MySQL homepage at <http://www.mysql.com/downloads>, under JDBC. For our purpose, we will use the MySQL driver, which is a Type-4 JDBC driver that is under the GNU Library License.

#### 6.1.4 Tomcat

TOMCAT is one of the Jakarta apache projects, it is a JAVA container to process JSP programs and construct a web server for static web pages. First of all, we go to <http://apache.mirrorcentral.com/dist/jakarta/tomcat-4/binaries/> to download the file tomcat-4.1.18.zip and extract it to hard driver. Also, we copy C:\tomcat\bin\startup.bat and shutdown.bat to the desktop as shortcut in order to easily start and shut down tomcat.

## CHAPTER SEVEN

### CONCLUSION AND FUTURE DIRECTIONS

#### 7.1 Conclusion

In the project, APTWS provides a perfect environment for user to record their travel experiences anytime over web browsers and cellular phones. The system uses Tomcat Web Server running on windows platform with MYSQL database. I implemented the system using JavaScript, JSP, HTML and WML languages and I implemented the advanced presentation feature within the browser using JavaScript. All dynamic contents are handled by JSP and a few static contents use HTML and WML files directly. Business logic was written in Java, and a data Source was used to connect to the database.

A Portable Traveler's Weblog is an "easy to use" system of a Web-based application. Everyone knows how to use a web browser or cellular device, so they can login to record travel experiences or view the travel web log of the other people who exist in the system.

#### 7.2 Future Directions

The possible improvements that can be made for APTWS include the following.

To make the graphical user interface more friendly. APTWS aims to provide a friendly user interface, but there are still many possibilities for improvements. For example, adding a "Photo" button to display the user's travel photos would add much to the system. Adding "voice log" ability could record voice log using cellular phone. A "Print" button could be added to make printing easier. Allow "SMS" input from specified phones to display simple text messaging from normally phone to phone. Most of the current APTWS source code is reusable, so it will be easy for the designer to make these changes.

In future developments, the system will be more flexible to be installed for various different mobile devices.



APPENDIX  
SOURCE CODE

```

Filename: Welcome.jsp
<% String error = request.getParameter("error_message");
// Testing client types
String userAgent = request.getHeader("user-agent");
if (userAgent == null || !userAgent.startsWith("Mozilla"))
{
    // If the client is not a browser, assume it is from cell phone
    if (error != null)
    {
        response.sendRedirect("wml_welcome.jsp?error_message=" + error);
    }
    else
    {
        response.sendRedirect("wml_welcome.jsp");
    }
}
%>
<html>
<head>
<title> WELCOME</title>
</head>
<body>
<form action = "publichome.jsp?" method="get">
<table cellpadding="5">
<tr>
<td>
View public logs (enter username)
</td>
<td>
<input type='text' name='userid' size ='24' >
<input type='submit' value='Find'>
</td>
</tr>
Filename: login.jsp
<html>
<title>loginpage</title>
<body>
<tr>
<td>
<a href ="welcome.jsp">Home</a>
</td>
</tr>
<h1> Login </h1>
<form action ="authenticate.jsp" method="post">
<table>
<tr>
<td>
Userid:
</td>
<td>
<input type = "text" name="userid" size="24">
</td>
</tr>
<tr>

```

```

<td>
Password:
</td>
<td>
<input type="password" name="password" size="24">
</td>
Filename: publichome.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
travelweblog.User user;
java.util.Vector records;
boolean showPrivate;
int currentPage;
String pageParam = request.getParameter("page");
String resultID = request.getParameter("result");
if (resultID != null)
{
// If page parameter is given, use is trying to look at
// log records in multiple pages.
currentPage = Integer.parseInt(pageParam);
travelweblog.QueryResult result = (travelweblog.QueryResult)session.getValue(resultID);
user = result.user;
records = result.records;
showPrivate = result.showPrivate;
}
else
{// If page parameter is not given, it is a query request.
currentPage = 1;
String username = request.getParameter("userid");
if (username == null)
{// If the username is not given, user is displaying
// his or her own records
showPrivate = true;
user = (travelweblog.User)session.getValue("user");
if (user == null)
{
Filename: editOldPage.jsp
<% boolean allowEdit = (request.getParameter("allowEdit") != null);
travelweblog.User user = (travelweblog.User)session.getValue("user");
if (allowEdit && user == null)
{
response.sendRedirect("login.jsp?error_message=User%20needs%20to%20login%20to%20a
ccess%20this%20information");
return;
}
}
try
{
int id = Integer.parseInt(request.getParameter("id"));
travelweblog.LogEntry logEntry = travelweblog.LogEntry.findLogEntry(id);

```





```

<input type="reset" value="reset">
</form>
</body>
</html>

```

Filename: createuser.jsp

```

<%
String name = request.getParameter("userid");
String password = request.getParameter("password");
String confirmpassword = request.getParameter("confirmpassword");
if (name.equals("") && password.equals("") && confirmpassword.equals(""))
{
    response.sendRedirect("newuserform.jsp?error_message=User%20create%20fail.");
}
else if (!password.equals(confirmpassword))
{
    response.sendRedirect("newuserform.jsp?error_message=Passwords%20do%20not%20match");
}
else
{
    try
    {
        travelweblog.User user = travelweblog.User.createUser(name, password);
        session.putValue("user", user);
        response.sendRedirect("publichome.jsp");
    } catch (Exception e)
    {
        response.sendRedirect("newuserform.jsp?error_message=User%20create%20failed.");
        //out.println(e.getMessage());
    }
}
%>

```

Filename: wml\_welcome.jsp

```

<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="card11" title="Welcome">
<p align="center">
<font size="+4"><i>Welcome</i></font><br/>
Travel WebLog
</p>
<p>
<a href="wml_newuserform.jsp" >New user</a><br/>
<a href="wml_login.jsp">Registered users</a><br/>
<a href="wml_search.jsp">Search</a><br/>
</p>
</card>
</wml>

```

Filename: wml\_login.jsp

```

<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");

```

```

// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
%>
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="c0" title="Login">
<onevent type="onenterforward">
<go href="#login">
    <setvar name="userid" value=""></setvar>
<setvar name="password" value=""></setvar>
</go>
</onevent>
<onevent type="onenterbackward">
<prev>

</onevent>
</card>
<card id="login" title="Login">
<p>
<%
String error = request.getParameter("error_message");
if (error != null)
{
%>
<%=error%> <br>
<%
}
%>
User name:
<input name='userid' format="*m"/>
Password:
<input type='password' name='password' format="*m"/>
<anchor title="Send">Login
<go method="post" href="wml_authenticate.jsp">
<postfield name="userid" value="$(userid)"></postfield>
<postfield name="password" value="$(password)"></postfield>
</go>
<anchor>
</p>
</card>
</wml>
Filename: wml_publichome.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
travelweblog.User user;
java.util.Vector records;
boolean showPrivate = true;

```

```

int currentPage;
String userToken = request.getParameter("ut");
if (userToken == null)
{
userToken = "";
showPrivate = false;
}
String pageParam = request.getParameter("page");
String resultID = request.getParameter("result");
if (resultID != null)
{
// If page parameter is given, use is trying to look at
// log records in multiple pages.
currentPage = Integer.parseInt(pageParam);
travelweblog.QueryResult result = (travelweblog.QueryResult)application.getAttribute(resultID);
user = result.user;
records = result.records;
// showPrivate = result.showPrivate;
}
else
{
// If page parameter is not given, it is a query request.
currentPage = 1;
String username = request.getParameter("userid");
if (username == null)
{
// If the username is not given, user is displaying
// his or her own records
// showPrivate = true;
Filename: wml_editOldPage.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
boolean allowEdit = (request.getParameter("allowEdit") != null);
String userToken = request.getParameter("ut");
//travelweblog.User user = (travelweblog.User)session.getValue("user");
if (allowEdit && userToken == null)
{
response.sendRedirect("wml_login.jsp?error_message=User%20needs%20to%20login%20to%20access%20this%20information");
return;
}
}
try
{
//int id = Integer.parseInt(request.getParameter("id"));
//travelweblog.LogEntry logEntry = travelweblog.LogEntry.findLogEntry(id);
String action = "Unknown";
if (request.getParameter("doEdit") != null)
{
action = "doEdit";
}
else if (request.getParameter("doView") != null)
{

```



```

action = "doView";
}
else if (request.getParameter("doDelete") != null)
{
action = "doDelete";
}
if (action.equals("doEdit") || action.equals("doView"))
{
String resultID = request.getParameter("result");
travelweblog.QueryResult result = (travelweblog.QueryResult)application.getAttribute(resultID);
if (result == null)
{
Filename: wml_search.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
%>
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="search" title="View public logs">
<onevent type="onenterforward">
<refresh>
<setvar name="userid" value=""/>
</refresh>
</onevent>
<onevent type="onenterbackward">
<refresh>
<setvar name="userid" value=""/>
</refresh>
</onevent>
<p>
<%
String error = request.getParameter("error_message");
if (error != null)
{
%>
<%=error%> <br>
<%
}
%>
User name:
<input name='userid' value="" format="*m"/>
<anchor title="OK">Find
<go method="post" href="wml_publichome.jsp?userid=$(userid)">
<!--
<postfield name="userid" value="$(userid)"></postfield>
-->
</go>
</anchor>

```

```

</p>
</card>
</wml>
Filename:wml_newuserform.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
%>
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="newuser" title="New user">
<onevent type="onenterforward">
<refresh>

<setvar name="password" value=""/>
<setvar name="confirmpassword" value=""/>
</refresh>
</onevent>
<onevent type="onenterbackward">
<refresh>
<setvar name="password" value=""/>
<setvar name="confirmpassword" value=""/>
</refresh>
</onevent>
<p>
<%
String error = request.getParameter("error_message");
if (error != null)
{
%>
<%=error%> <br>
<%
}
%>
User name:
<input name='userid' value="" format="*m"/>
Password:
<input type='password' name='password' value="" format="*m"/>
Confirm password:
<input type='password' name='confirmpassword' value="" format="*m"/>
<anchor title="OK">Submit
<go method="post" href="wml_createuser.jsp">
<postfield name="userid" value="$(userid)"></postfield>
<postfield name="password" value="$(password)"></postfield>
<postfield name="confirmpassword" value="$(confirmpassword)"></postfield>
</go>
<anchor>
</p>
</card>

```

```

</wml>
Filename: wml_addNewPage.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
boolean allowEdit = (request.getParameter("allowEdit") != null);
String userToken = request.getParameter("ut");
if (allowEdit && userToken == null)
{
response.sendRedirect("wml_login.jsp?error_message=User%20needs%20to%20login%20to%20access%20this%20information");
return;
}
%>
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
<card id="c0" title="New Page">
<onevent type="onenterforward">
<go href="#edit">
    <setvar name="title" value=""></setvar>
<setvar name="detail" value=""></setvar>
</go>
</onevent>
<onevent type="onenterbackward">
<prev>
</onevent>
</card>
<card id="edit" title="New Page">
<p.do type="options" label="MENU">
<p.go href="#menu">
</p.go>
</p.do>
<p>
Title:
<input name="title" format="*m"></input>
Detail:
<input name="detail" format="*m"></input>
<select name="public" multiple="true" ivalue="0">
<option value="T">Public</option>
</select>
</p>
</card>
<card id="menu" title="">
<anchor title="Send">Save
<go method="post" href="wml_saveAction.jsp?doSave">
<postfield name="ut" value="<%=userToken%>"></postfield>
<postfield name="title" value="$(title)"></postfield>
<postfield name="detail" value="$(detail)"></postfield>
<postfield name="public" value="$(public)"></postfield>

```

```

</go>
<anchor><br/>
<a href="wml_welcome.jsp">Home</a><br/>
<a href="wml_logoff.jsp?ut=<%=userToken%>">Logout</a><br/>
<a href="wml_publichome.jsp?ut=<%=userToken%>">Summary Page</a><br/>
</card>
</wml>
Filename: wml_createuser.jsp
<%
// How to prevent caching on the client system
response.setHeader("Expires", "0");
// How to prevent caching at proxy servers
response.setHeader("Cache-Control", "no-store");
String name = request.getParameter("userid");
String password = request.getParameter("password");
String confirmpassword = request.getParameter("confirmpassword");
if (name.equals("") && password.equals("") && confirmpassword.equals(""))
{
response.sendRedirect("wml_newuserform.jsp?error_message=User%20create%20fail.");
}
else if (!password.equals(confirmpassword))
{
response.sendRedirect("wml_newuserform.jsp?error_message=Passwords%20do%20not%20match");
}
else
{
try
{
travelweblog.User user = travelweblog.User.createUser(name, password);
session.putValue("user", user);
response.sendRedirect("wml_publichome.jsp");
} catch (Exception e)
{
response.sendRedirect("wml_newuserform.jsp?error_message=User%20create%20failed.");
//out.println(e.getMessage());
}
}
%>

```

## REFERENCES

- [1] Jayson Falkner, et al. "Beginning JSP Web Development", First Edition, Wrox Press Inc, August 2001.
- [2] Jason Hunter and William Crawford. Java Servlet Programming Second Edition. O'Reilly and Associates, 2002.
- [3] P.J. Deitel. "How to Program JAVA", Fourth Edition Deitel & Associate, Inc, 2000.
- [4] P.J. Deitel. "How to Program Advanced Java 2 Platform", Deitel & Associate, Inc, 2002.
- [5] Martin Fowler and Kendall Scott. "UML Distilled- A brief guide to the standard object modeling language", Second Edition, Addison-Wesley, July 2001.
- [6] Ivor Horton. "Beginning Java 2", JDK 1.3 Edition, Wrox Press Ltd, 2000.
- [7] Ramez Elmasri and Shamkant B. Navathe. "Fundamentals of Database Systems", Third Edition Addison-Wesley, June 2000.
- [8] Shelly Cashman Woods. "HTML Complete Concepts and Techniques", second Edition, Thomson Course Technology, 2002.
- [9] Development and Research Center. "eWAP Wireless World - WML", Unalis, 2001.
- [10] William B. Sanders. "Javascript Design", New Riders, 2002.