

California State University, San Bernardino

CSUSB ScholarWorks

Theses Digitization Project

John M. Pfau Library

2005

Vacation system

Min-Wei Lee

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



Part of the [Computer Sciences Commons](#)

Recommended Citation

Lee, Min-Wei, "Vacation system" (2005). *Theses Digitization Project*. 2829.

<https://scholarworks.lib.csusb.edu/etd-project/2829>

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.

VACATION SYSTEM

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
Min-Wei Lee
September 2005

VACATION SYSTEM


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

by

Min-Wei Lee

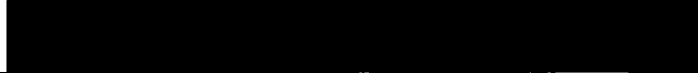
September 2005

Approved by:




Dr. David Turner, Chair, Computer Science

July 18, 2005
Date



Dr. George Georgiou



Dr. Keith Schubert

ABSTRACT

The purpose of this master's project is to explore the use of Web Services to solve enterprise computing problems. XML-based Web Services allow complex information systems to be subdivided into weakly-coupled distributed components built in different environments and running in different operating systems. To explore this architectural paradigm, two systems were built: an employee records management system, and a vacation system. The employee records management system allows an administrator to view, create, modify and delete employees from a database. The vacation system allows employees to request vacations, and for managers to process these requests.

The software system in this project is structured as a two-tier architecture. The first tier is the Web Service. The purpose of the Web Service is to provide a persistence service for various web applications of the company. That means the web applications do not interact directly with a database to store data.

In the second tier, there are web applications. The web applications interact with the persistence service. In this project, there are two applications: the vacation system and the employee records system.

ACKNOWLEDGMENTS

I would like to express my deep appreciation to my graduate advisor, Dr. David Turner, for his important and valuable contributions to me and my project. Discussions with Dr. Turner always give me significant help to resolve my problems and give me positive directions to do my research. Dr. Turner offered me a lot of precious information and suggestions during all the period of doing my project. I also would like to thank my two project committee professors, Dr. Georgiou and Dr. Schubert, for their supportive help and important advice.

I would like to thank my parents, deeply from my heart, who provide me the chance to fulfill my every dream in life. Their love warmly supports me in every day and their encouragement gives me the power to accomplish every success in my life.

TABLE OF CONTENTS

ABSTRACT	iii
ACKNOWLEDGMENTS	iv
LIST OF TABLES	viii
LIST OF FIGURES	ix
CHAPTER ONE: INTRODUCTION	1
1.1 Background	2
1.2 Purpose of this Project	2
1.3 What is a Web Service	3
1.4 Service-Oriented Architecture	5
1.5 Project Products	6
CHAPTER TWO: SYSTEM REQUIREMENTS	8
2.1 System Interfaces	9
2.2 Hardware Interfaces	10
2.3 Software Interfaces	10
2.4 User Roles	11
2.5 Use Case Diagram	12
2.6 Security	14
CHAPTER THREE: SYSTEM DESIGN	
3.1 Data Analysis	19
3.2 Database Schema Conceptual Model - Entity Relationship Diagram	19
3.3 Database Schema Logical Model - Relational Schema	20
3.4 Data Type and Detail	21
3.5 Page Flow Models	23

3.5.1	Vacation System	23
3.5.2	Employee Records System	60
3.6	Sequence Diagram	84
3.6.1	Vacation System	85
3.6.2	Employee Records System	91
CHAPTER FOUR:	SYSTEM IMPLEMENTATION	95
CHAPTER FIVE:	PROJECT IMPLEMENTATION	
5.1	User Interface Design	98
5.2	Graphical User Interface and Description	98
5.2.1	Secure Login Page	98
5.2.2	Apply Page	99
5.2.3	CheckRemainder Page	100
5.2.4	Track/Cancel Page	101
5.2.5	Approve Page	102
5.2.6	Create New User Page	103
5.2.7	Delete User Page	104
5.2.8	Modify User Page	105
5.2.9	Check User State Page	106
5.2.10	Forget Password Page	107
5.2.11	List Department Users Page	108
5.2.12	List All Users Page	109
CHAPTER SIX:	SYSTEM VALIDATION	
6.1	Unit Test	111
6.2	Subsystem Testing	114
6.3	System Test Plan	116

CHAPTER SEVEN: MAINTENANCE MANUAL

7.1 Software Installation	117
7.1.1 Gentoo Installation	117
7.1.2 Install Java 2 Platform, Standard Edition	118
7.1.3 Install Tomcat	118
7.1.4 MySQL Installation	119
7.1.5 Remote Procedure Calls Router Installation	119
7.2 System Variables	120
7.3 Vacation System Installation/Migration	120
7.4 Backup and Restore	121
7.4.1 System Backup	121
7.4.2 Database Backup	122
7.4.3 System Restore	122
7.4.4 Database Restore	122

CHAPTER EIGHT: CONCLUSION AND FUTURE DIRECTIONS

8.1 Conclusion	124
8.2 Future Directions	125

APPENDIX A: CREATEACCOUNT.SERVLET, DBDAO.SERVLET AND ACCOUNTDAO.SERVLET AS EXAMPLES TO EXPLAIN HOW THE SERVLET WORKS	126
--	-----

APPENDIX B: CODE SNIPPETS OF DBDAO.SERVLET AND ACCOUNTDAO.SERVLET FOR CREATEACCOUNT.SERVLET	131
---	-----

REFERENCES	134
------------------	-----

LIST OF TABLES

Table 1. Structure of Table Users	22
Table 2. Structure of Table LDays	22
Table 3. The Unit Test Results	112
Table 4. Subsystem Test Results	115
Table 5. System Test Results	116

LIST OF FIGURES

Figure 1.	The Web Service Concept	5
Figure 2.	The Service-Oriented Architecture Concept	6
Figure 3.	System Deployment Diagram	9
Figure 4.	User Role Diagram	12
Figure 5.	Vacation System Use Case Diagram	13
Figure 6.	Employee Records System Use Case Diagram	14
Figure 7.	Security Diagram	18
Figure 8.	Vacation System Entity Relationship Diagram	20
Figure 9.	Employee Records System Database Relational Schema	21
Figure 10.	Vacation System Database Relational Schema	21
Figure 11.	Login Page (General User)	23
Figure 12.	System Home Page (General User)	24
Figure 13.	Apply for Leave Page (General User)	25
Figure 14.	Check Remainder Page (General User)	26
Figure 15.	Track/Cancel Page (General User)	27
Figure 16.	Check User State Page (General User)	28
Figure 17.	Message Page (General User)	29
Figure 18.	Track Result Page (General User)	30
Figure 19.	Password Quote Page (General User)	31
Figure 20.	Login Page (Manager)	32
Figure 21.	System Home Page (Manager)	33

Figure 22. Apply for Leave Page (Manager)	34
Figure 23. Check Remainder Page (Manager)	35
Figure 24. Track/Cancel Page (Manager)	36
Figure 25. Approve Page (Manager)	37
Figure 26. Check User State Page (Manager)	38
Figure 27. Message Page (Manager)	39
Figure 28. Track Result Page (Manager)	40
Figure 29. Password Quote Page (Manager)	41
Figure 30. Login Page (Senior Manager)	42
Figure 31. System Home Page (Senior Manager)	43
Figure 32. Track/Cancel Page (Senior Manager)	44
Figure 33. Approve Page (Senior Manager)	45
Figure 34. Check User State Page (Senior Manager)	46
Figure 35. Message Page (Senior Manager)	47
Figure 36. Track Result Page (Senior Manager)	48
Figure 37. Password Quote Page (Senior Manager)	49
Figure 38. Login Page (System Administrator)	50
Figure 39. System Home Page (System Administrator)	51
Figure 40. Apply for leave Page (System Administrator)	52
Figure 41. Check Remainder Page (System Administrator)	53
Figure 42. Track/Cancel Page (System Administrator)	54
Figure 43. Approve Page (System Administrator)	55

Figure 44.	Checke User State Page (System Administrator)	56
Figure 45.	Message Page (System Administrator)	57
Figure 46.	Track Result Page (System Administrator)	58
Figure 47.	Password Quote Page (System Administrator)	59
Figure 48.	Login Page (General User)	60
Figure 49.	System Home Page (General User)	61
Figure 50.	Modify Profile Page (General User)	62
Figure 51.	Message Page (General User)	63
Figure 52.	Password Quote Page (General User)	64
Figure 53.	Login Page (Manager)	65
Figure 54.	System Home Page (Manager)	66
Figure 55.	List Dept Users Page (Manager)	67
Figure 56.	Modify Profile Page (Manager)	68
Figure 57.	Message Page (Manager)	69
Figure 58.	Password Quote Page (Manager)	70
Figure 59.	Login Page (Senior Manager)	71
Figure 60.	System Home Page (Senior Manager)	72
Figure 61.	Modify Profile Page (Senior Manager)	73
Figure 62.	Message Page (Senior Manager)	74
Figure 63.	Password Quote Page (Senior Manager)	75
Figure 64.	Login Page (System Administrator)	76
Figure 65.	System Home Page (System Administrator)	77

Figure 66. Create New User Page (System Administrator)	78
Figure 67. Delete User Page (System Administrator)	79
Figure 68. Modify User Page (System Administrator)	80
Figure 69. List All Users Page (System Administrator)	81
Figure 70. Message Page (System Administrator)	82
Figure 71. Modify Profile Page (System Administrator)	83
Figure 72. Password Quote Page (System Administrator)	84
Figure 73. Apply for Leave	86
Figure 74. Check Remainder	87
Figure 75. Track/Cancel	88
Figure 76. Approve	89
Figure 77. Check User State	90
Figure 78. Create New User	91
Figure 79. Delete User	92
Figure 80. Modify User	93
Figure 81. List Users	94
Figure 82. Code Architecture	95
Figure 83. Class Diagram for Vacation System	96
Figure 84. Class Diagram for Employee Records System	97
Figure 85. Secure Login Page	99
Figure 86. Apply Page	100

Figure 87. Check Remainder Page	101
Figure 88. Track/Cancel Page	102
Figure 89. Approve Page	103
Figure 90. Create New User Page	104
Figure 91. Delete User Page	105
Figure 92. Modify User Page	106
Figure 93. Check User Page	107
Figure 94. Password Page	108
Figure 95. List Department Users Page	109
Figure 96. List All Users Page	110

CHAPTER ONE

INTRODUCTION

The purpose of this project is to investigate how to do system integration within an organization using SOAP-based Web Services, and to investigate a new architectural design in which Web Services are used to provide a persistence service to enterprise applications. The prototypes developed in this project illustrate how to strongly decouple applications from databases by providing a persistence service through which applications manage their persistent data. The benefit of doing this is to simplify the development of applications within an enterprise by removing the need to develop code for accessing the database.

To perform these investigations, we developed two different systems: a system to manage employee leave and a system to manage employee records. Both of these systems access a common persistence service through the use of SOAP. The Vacation System (VS) is a web system that allows employees of a company to request leave, and allows managers to grant or deny leave. Employees can also use the system to check the number of days they currently have available for leave, and to track or delete pending

vacation requests. When an application is approved, the system sends a confirmation letter to inform the employee that the vacation request is successful.

1.1 Background

I worked for a company for two years, and I found out that the procedure to apply leave is inefficient. So I had the idea to construct a system to make the procedure more efficient. Also, I found out that there are many systems in the company; most of them are very old; some of them were built almost 10 years ago. So, it's very difficult to integrate the old systems with new systems.

Since I can't modify the old systems very easily, what I can do is to find out how to integrate them with the new systems. After taking a class on Web Services, I realized that Web Services provides a way to solve this integration problem efficiently. So, I decided to do my project on Web Services.

1.2 Purpose of this Project

The purpose of this master's project is to explore the use of Web services to solve enterprise computing problems. XML-based Web Services allow complex information systems to be subdivided into weakly-coupled distributed components built in different environments and running in

different operating systems. To explore this architectural paradigm, two systems were built: an employee records system, and a vacation system.

1.3 What is a Web Service

The web has developed rapidly, which has created new opportunities for system integration based on XML. Some people expect that software will be increasingly based on connecting distributed components using SOAP-based Web Services. Web Services are generally implemented over HTTP, which means they can operate through firewalls. Also, with standards-based SOAP specifications, you can develop distributed system interfaces on any operating system platform and in any language.

Web Services are a new kind of web application. They can be published, located, and used across the web. Web Services can be applied from very simple requests to very complicated procedures. Once a Web Service is issued on the UDDI by WSDL, other applications (and other Web Services) can discover and request to use the service.

There are so many definitions of XML Web Service, but almost all definitions have these things in common:

1. XML Web Services must communicate through a standard web protocol. In most cases, the

protocol used is SOAP. SOAP is a protocol specification which is used to pass XML-encoded data through HTTP.

2. XML Web Services are described by a description language, named Web Services Description Language (WSDL). And the descriptions are also saved as a XML document. WSDL offers a format to describe the Web Service.
3. XML Web Services are published, registered on the Universal Discovery Description and Integration (UDDI). Then other users can search and request to use them. UDDI provides users to search, connect, and use the registered Web Services.

Apparently, you can access any system through a Web Service interface. You can combine different systems into a single huge one. For example, you can combine a payment system, a stock trading system, an insurance system, and a banking system into a financial management system. It's easy to achieve this goal with Web Services.

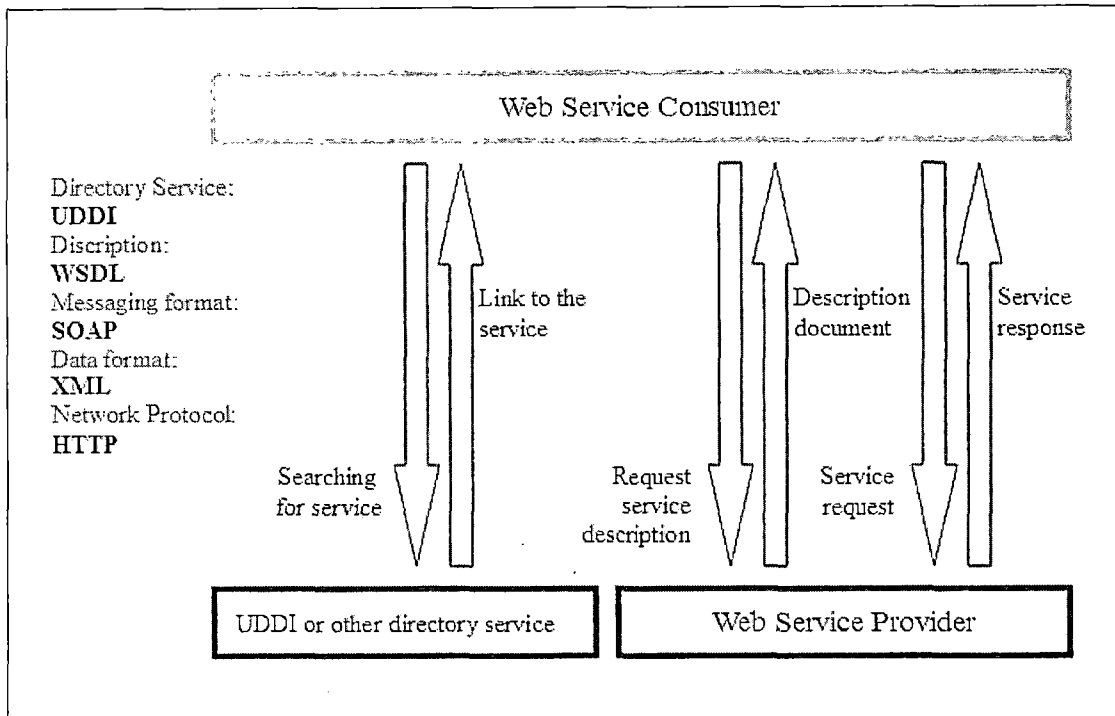


Figure 1. The Web Service Concept

1.4 Service-Oriented Architecture

Service-Oriented Architecture (SOA) is not a new thing. Several years, SOA had been applied successfully to build applications by information department. At that time, Web Service and XML haven't been brought up.

SOA is not a technology. SOA is one kind of constructing and organizing method for building operation environment of applications. SOA is a model for design, deploy, management in a company.

SOA is a design method which aims to maximize the reuse of application services to increase adaptability and

efficiency for information department. With SOA, a company can save many costs on constructing new systems.

Web Service is one kind of SOA. In this project, I use Web Service as a persistence service for building my system.

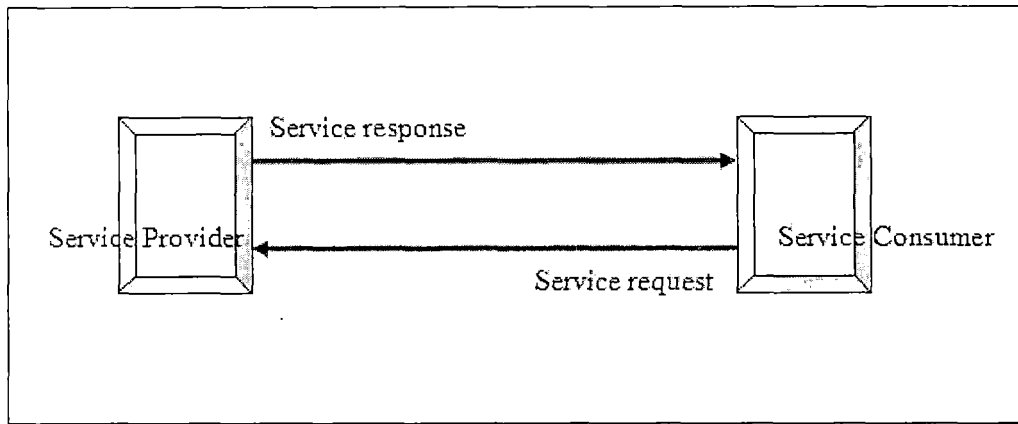


Figure 2. The Service-Oriented Architecture Concept

1.5 Project Products

The Vacation System project led to the following products:

- Implementation of Vacation System: a working web-based online application system with JAVA programs, XML configuration, JSP and MySQL database, which achieve the specific needs of Vacation System. All the forms follow the original paper application form in order to

accomplish convenient and familiar processes for the user.

- System documentation: a project documentation, which is available with system design, specifications, project implementation and testing reports.

CHAPTER TWO

SYSTEM REQUIREMENTS

The Vacation System (VS) project aims to be friendly and convenient for the users. The components of VS are a web server, a database server, and a client browser. Customer browsers use the Internet (TCP/IP) to apply vacation through the web server under HTTP/HTTPS. The web server connects to the email server with TCP/IP in order to send or retrieve email through SMTP. The web application connects to the database via TCP, and accesses database functionality through JDBC.

In order to choose implementation components conforming to the criteria of the shareware standard, this project uses Tomcat as web server, and MySQL as database server. The other components, such as the web browsers, are dependant on which kind of browsers the customers use. The email server was provided by the ISP (Internet Service Provider).

The architecture of this project is shown below in Figure 3.

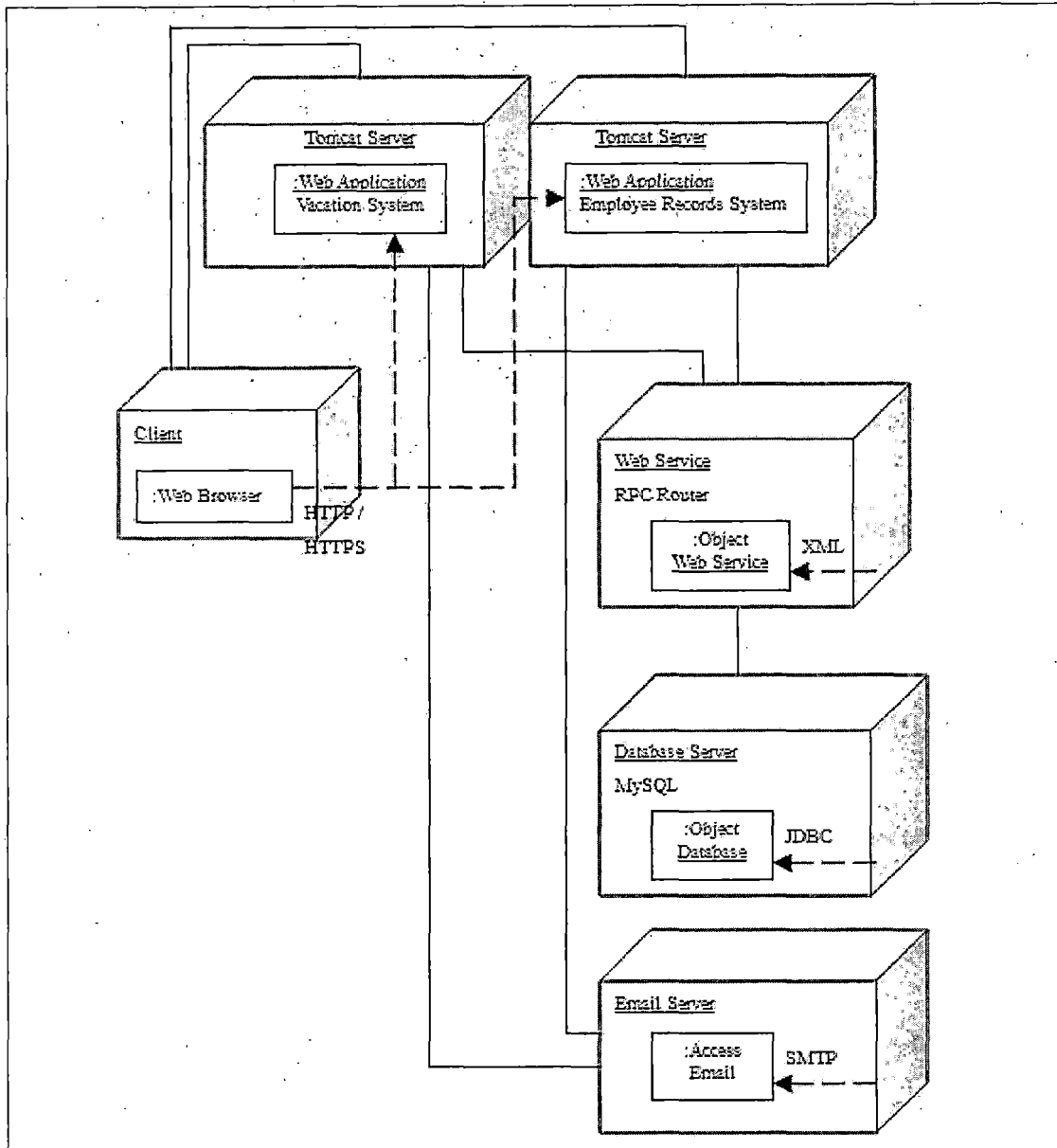


Figure 3. System Deployment Diagram

2.1 System Interfaces

The Vacation System (VS) is a 3-tier distributed architecture that displays the user interface in a web browser using HTML. The middle tier is a Java Servlet (automatically created from JSP pages) that handles

requests from the client browser and provides access to the third tier via JDBC, which is a MySQL database. The HTTP server is provided by Apache Tomcat, which also implements JSP and Java Servlet APIs.

2.2 Hardware Interfaces

Vacation System will not implement hardware interface directly. However, it will trust the underlying operating system (Windows, NT, ME, XP, Linux, UNIX, MAC) to handle the hardware interfaces.

2.3 Software Interfaces

As explained above, there will be two different software interfaces depending on the type of access that the user is demanding or the function that the user wants the software to perform. The reasons that why I choosed the software interfaces used in this project are summarized as following:

- Operating system (Gentoo Linux): a special flavor of Linux that can be automatically optimized and customized for just about any application or needed.
- Web Server/Container (Jakarta Tomcat Server): Tomcat server is a Java based web application

container that was created to run servlets and Java Server Pages (JSP) in web applications.

- JAVA 2 Platform, Standard Edition (J2SE): A Java-based, runtime platform that provides many features for developing web-based Java applications, including database access (JDBC API) interface technology, and security for both local network and Internet use and it's required in the Tomcat JAVA Container.
- Database Server (MySQL Server): MySQL is an open source database software, and it's popular. MySQL also provides a JDBC driver to be easily connected from a JAVA program.
- Java Database Connector (JDBC): MySQL connector.
- Web Service : Remote Procedure Calls Router
- Languages: HTML/JAVA/JavaScript/JSP/XML.

2.4 User Roles

Web site users fall into four categories: general user, manager, senior manager and system administrator. The following subsections describe their activities.

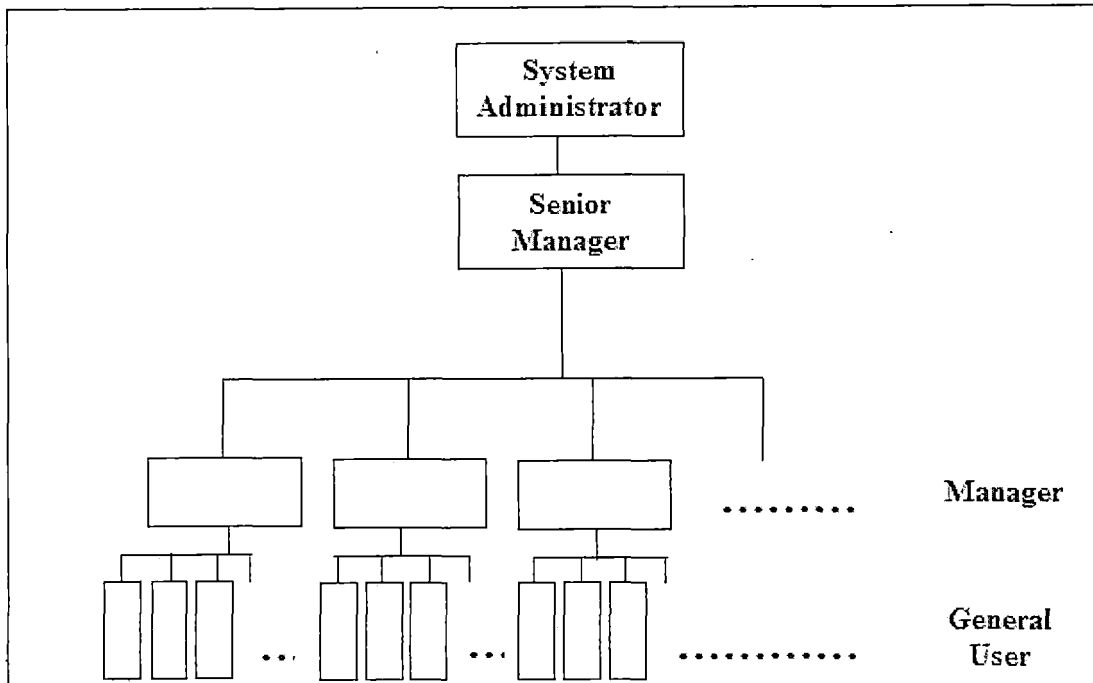


Figure 4. User Role Diagram

2.5 Use Case Diagram

The design of Vacation System aims to perform 9 main functions for 4 different prospective users. The following figure is the Use Case Diagram of this project.

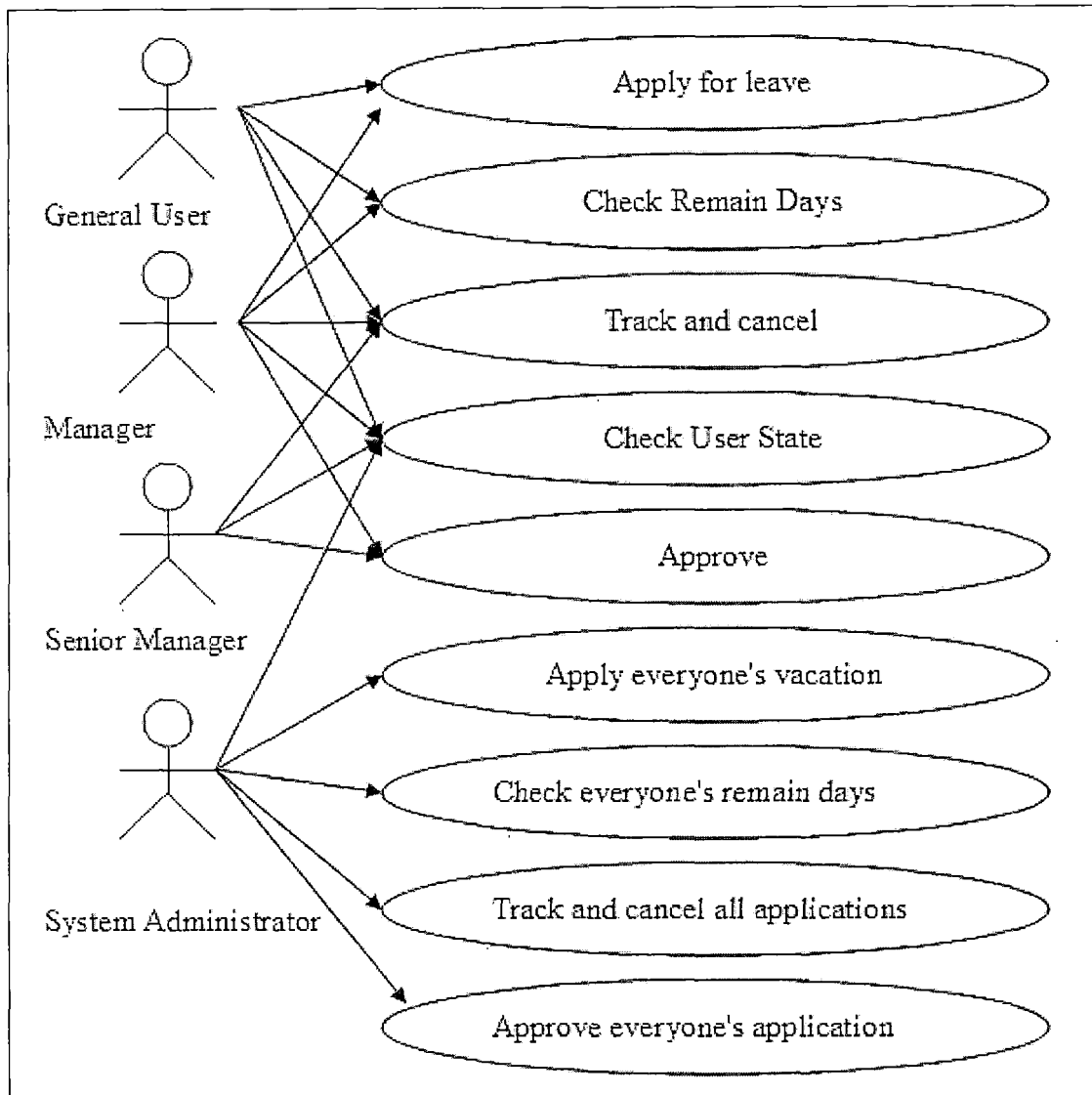


Figure 5. Vacation System Use Case Diagram

The design of Employee Records System aims to perform 6 main functions for 4 different prospective users. The following figure is the Use Case Diagram of this project.

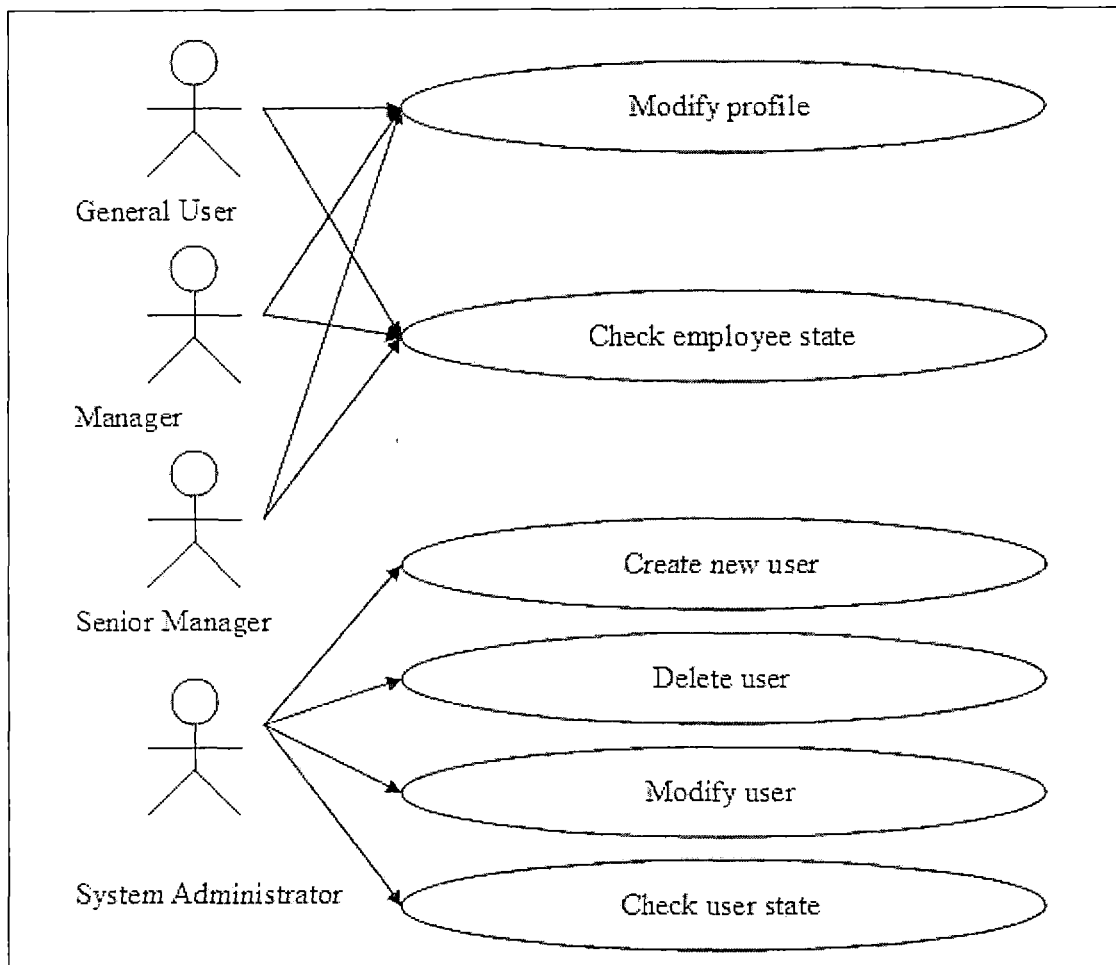


Figure 6. Employee Records System Use Case Diagram

2.6 Security

It is important to have a general understanding of information security prior to design a web application. There are six general security services that cover the various functions required of a secure application. These can also be considered as requirements to make the web applications in this project secure.

Authentication: Ensures that the sender and receiver are who they claim to be. Mechanisms such as username/password, smart cards, and Public Key Infrastructure (PKI) can be used to assure authentication.

Authorization or Access Control: Ensures that an authenticated entity can access only those services they are allowed to access. Access control lists are used to implement this.

Availability: Requires that uninterrupted services are provided to authenticated and authorized users.

Confidentiality: This assures that information in storage and in-transit are accessible only for reading by authorized parties. Encryption is used to assure message confidentiality.

Integrity: Ensures that information, either in storage or in-transit cannot be modified intentionally or unintentionally. Digital signatures are used to assure message integrity.

Nonrepudiation: Requires that neither the sender nor the receiver of a message be able to legitimately claim they didn't send/receive the message.

In order to make sure the validity and the safety, I will use the SSL (Secure Sockets Layer) between the web browser and the web server, and use a Web Service password

between the web applications and Web Service. SSL is used to make sure that the messages transmitted between the users and the web applications will not be changed or be stolen. SSL will maintain the security and the completeness. The "s" in "https" stands security for http over SSL. The Web Service password is passed into the Web Service system with a configuration file (web.xml). Each time the web applications access the Web Service, it supplies the password.

There are 3 access points where security needs to be implemented: the web interface that browsers access, the Web Service interface that the Vacation System and Employee Records System need to access, and the database interface that the Web Service needs to access (See Figure 7).

Authentication: Employee Records System (ERS) and Vacation System (VS) require authentication by logging in with a username and password. The ERS and VS authenticate with the Web Service by providing a password with each request. The Web Service authenticates with the database by providing a database username and password.

Authorization or Access Control: All pages are controlled by sessions; all access without valid sessions will be redirected to the login page.

Availability: The system runs on a single server. To increase availability, additional servers are required.

Confidentiality: SSL provides the confidentiality for the layer between browser and web applications. Between the web applications and the Web Service, data is transmitted in a private network.

Integrity: SSL provides integrity for the layer between browser and web applications; SOAP over TCP in the private network is in charge of the layer between web applications and persistence service; process to process communication provides integrity for the layer between persistence service and database.

Nonrepudiation: This property is not considered in this project.

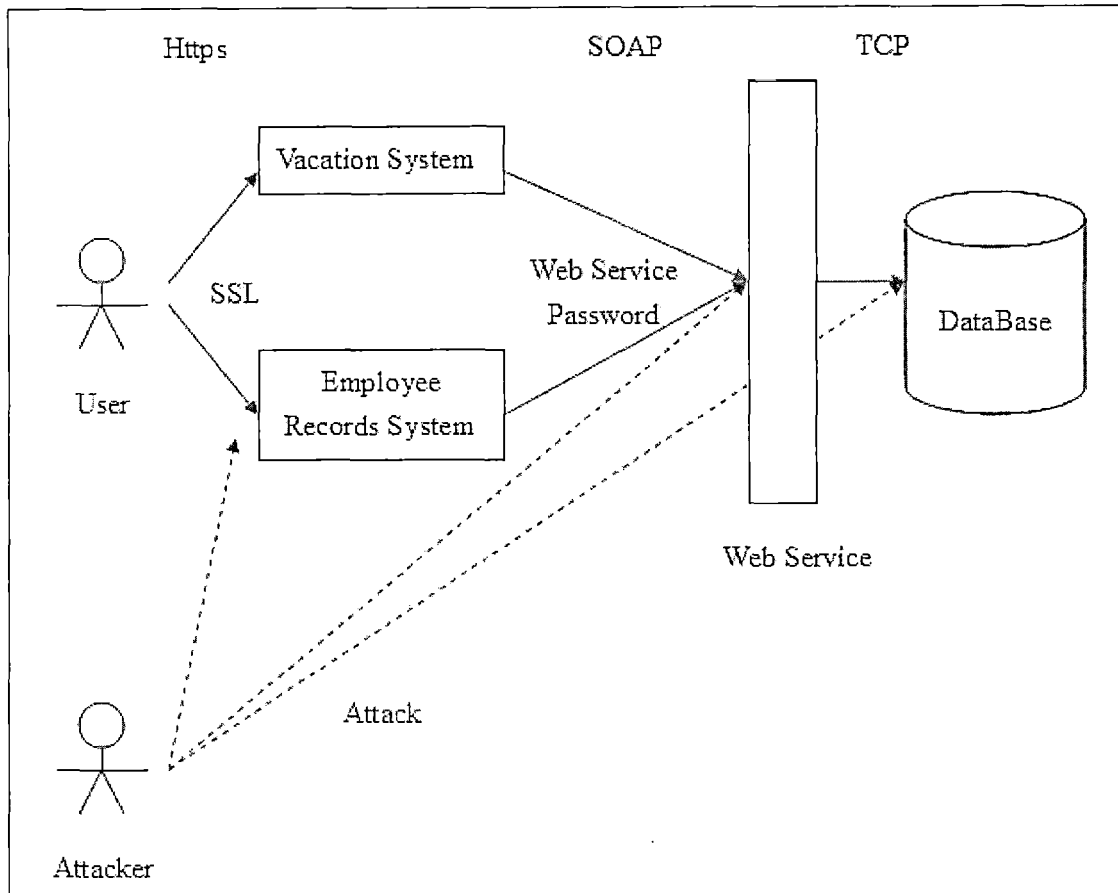


Figure 7. Security Diagram

CHAPTER THREE

SYSTEM DESIGN

The database server used by Vacation System is MySQL. Once the database has been created using MySQL, all interfacing with the database will be done from Java. One must note that all interfaces as seen by the users of the system will be through pages with HTML forms generated from JSP and JavaServlets.

3.1 Data Analysis

The data for designing and implementing the schema of the database depends on two entities: users table and ldays table. All the input data will be checked by using JavaServlet or JavaScript when the data is processed. The tables of users and ldays are connected by the relation of EmployeeID. To get the vacation application information, we use IdNumber to search it.

3.2 Database Schema Conceptual Model - Entity Relationship Diagram

All the entities and relations used in Vacation System are described in the ER Diagram in Figure 8.

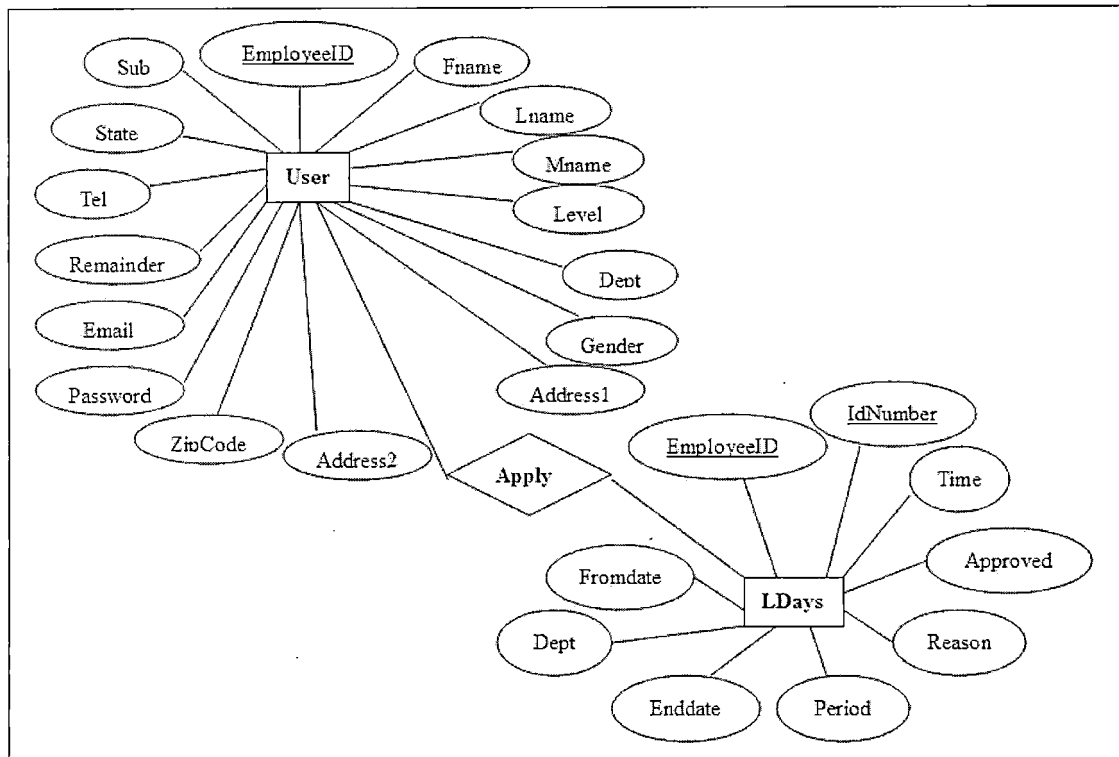


Figure 8. Vacation System Entity Relationship Diagram

3.3 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).

users					
<u>EmployeeID</u>	Fname	Lname	Mname	Level	Dept
Gender	Address1	Address2	ZipCode	Password	Email
Remainder	Tel	State	Sub		

Figure 9. Employee Records System Database Relational Schema

ldays					
<u>IdNumber</u>	EmployeeID	Fromdate	Enddate	Period	Reason
Approved	Dept	Time			

Figure 10. Vacation System Database Relational Schema

3.4 Data Type and Detail

The logical model establishes 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 Users

Field	type	null	key	default	extra
EmployeeID	varchar(10)	YES	PRI		
Fname	varchar(20)				
Lname	varchar(20)				
Mname	varchar(20)				
Level	varchar(2)				
Dept	varchar(5)				
Gender	varchar(2)				
Address1	varchar(40)				
Address2	varchar(40)				
ZipCode	varchar(10)				
Password	varchar(20)				
Email	varchar(40)				
Remainder	varchar(5)				
Tel	varchar(10)				
State	varchar(20)				
Sub	varchar(10)				

Table 2. Structure of Table LDays

field	Type	null	key	default	extra
IdNumber	varchar(20)	YES	PRI		
EmployeeID	varchar(10)		FOR		
Fromdate	varchar(16)				
Enddate	varchar(16)				
Period	varchar(5)				
Reason	varchar(40)				
Approved	varchar(2)				
Dept	varchar(5)				
Time	varchar(30)				

3.5 Page Flow Models

Page flow model shows the process of the pages, by reading these diagrams, you can easily understand where the page goes, and where does it from.

The square in the upper section of the diagram stands for the link or button to other page. The lower section stands for the content, the form field of the page.

3.5.1 Vacation System

3.5.1.1 General User. Figure 11 represents the Login Page which is used by general user, if the user successfully logins the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

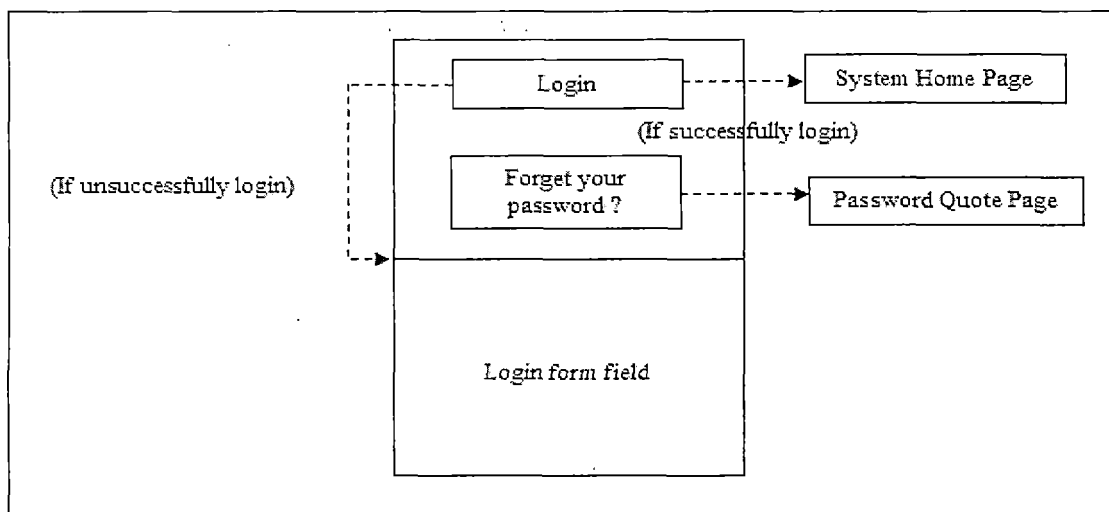


Figure 11. Login Page (General User)

Figure 12 represents the System Home Page which is used by a general user, a user can choose which system he is going to use, or logout to the Login Page.

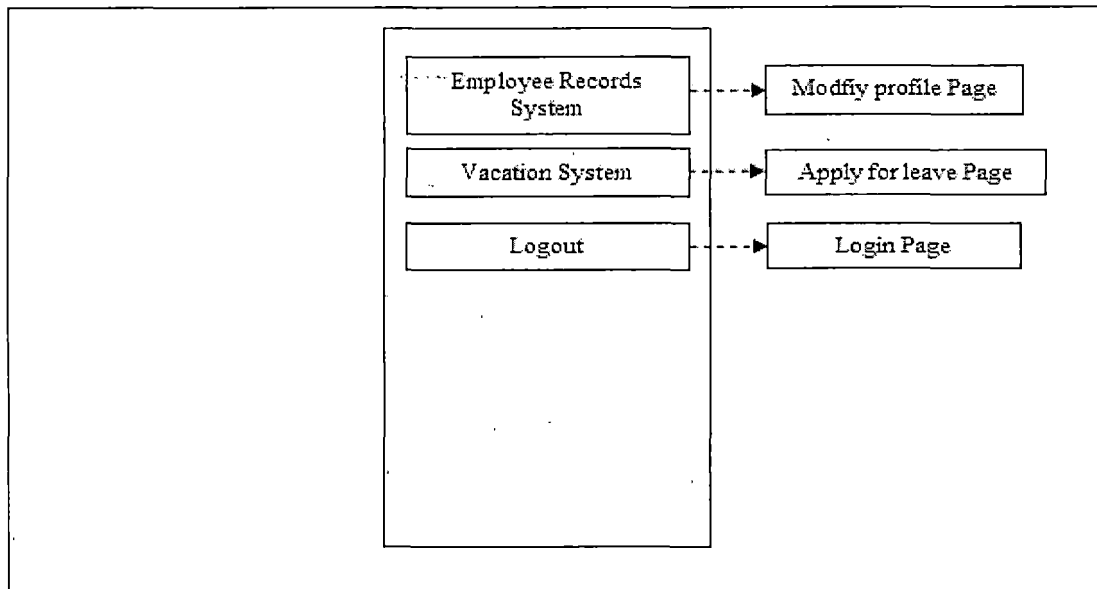


Figure 12. System Home Page (General User)

Figure 14 represents the Check Remainder Page which is used by a general user, a user can check how many days are available for leave through this page, or go to other pages through other links.

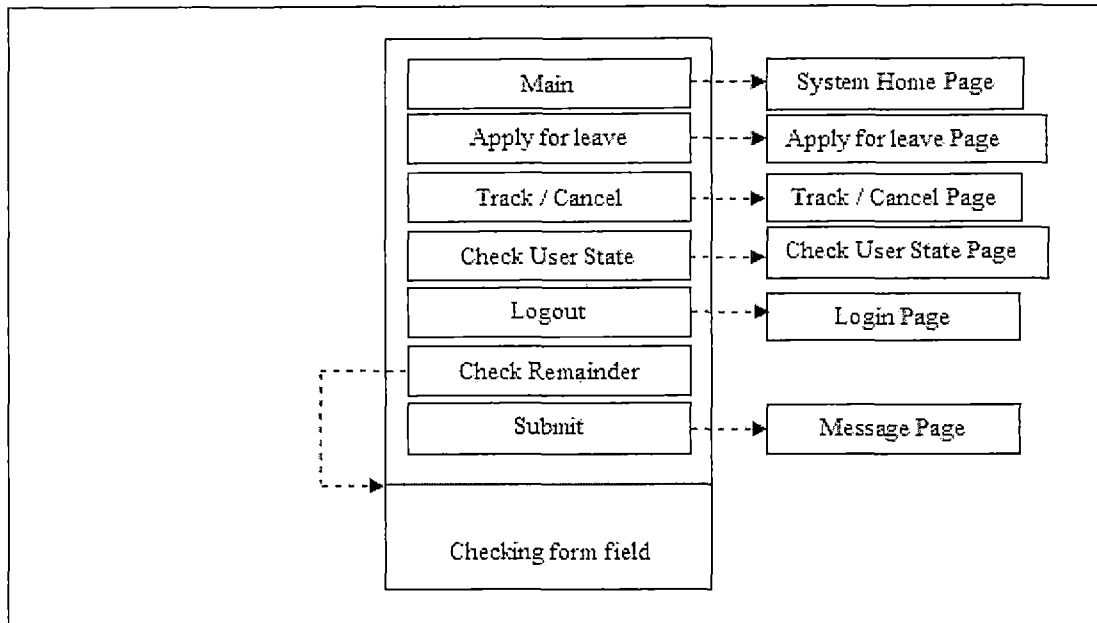


Figure 14. Check Remainder Page (General User)

Figure 15 represents the Track / Cancel Page which is used by a general user, a user can track his application through this page, or go to other pages through other links.

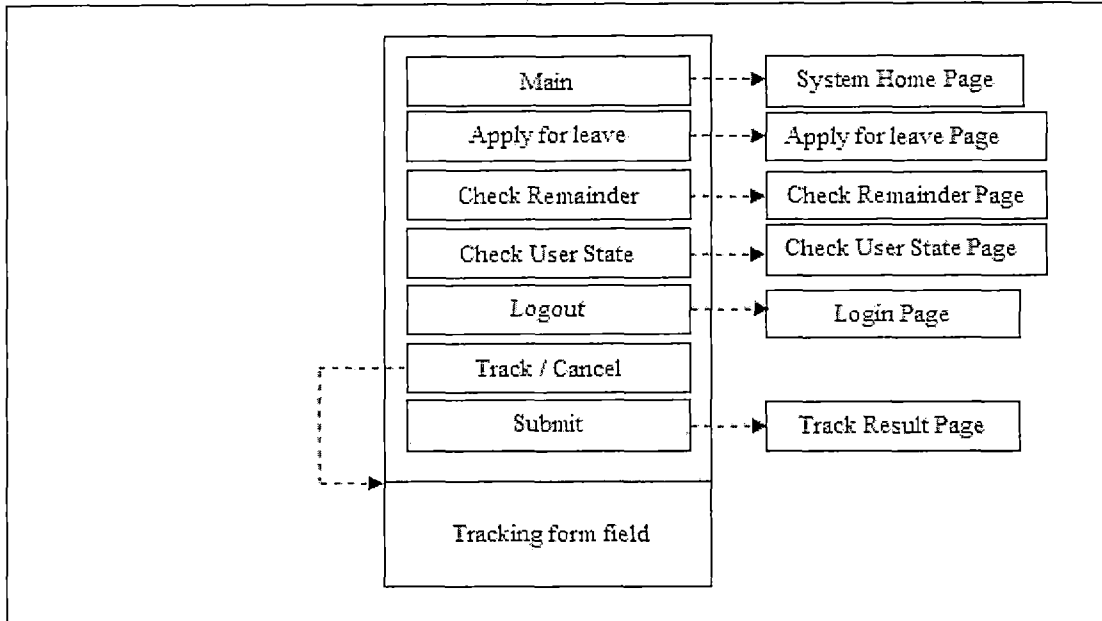


Figure 15. Track/Cancel Page (General User)

Figure 16 represents the Check User State Page which is used by a general user, a user can check the state or other employee's state through this page, present or on vacation, or go to other pages through other links.

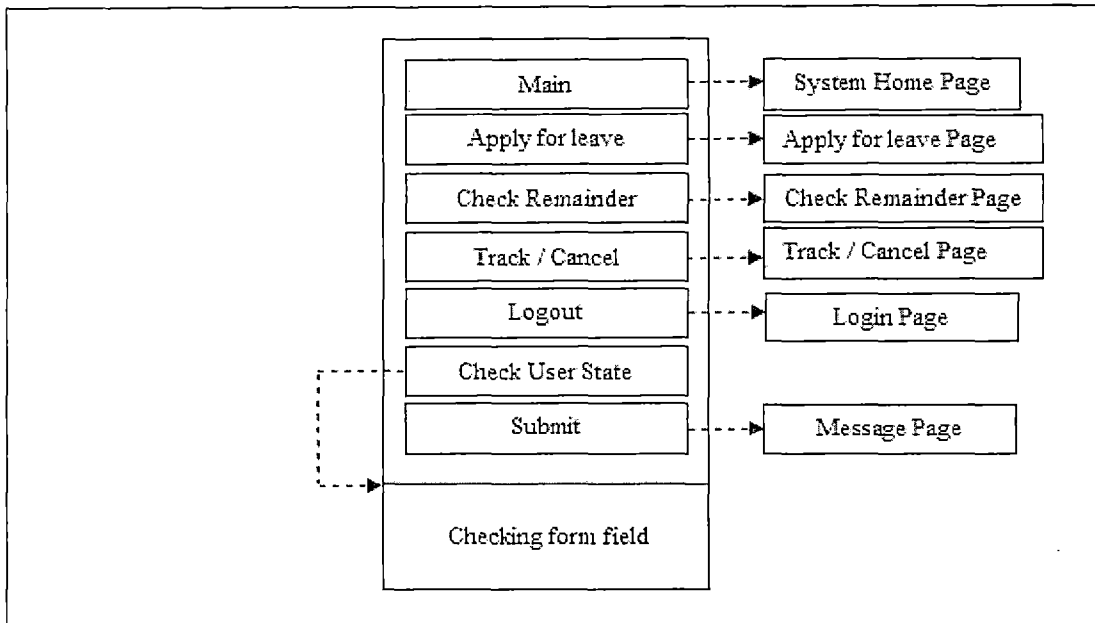


Figure 16. Check User State Page (General User)

Figure 17 represents the Message Page which is used by a general user, a user can read the system message through this page, or go to other pages through other links.

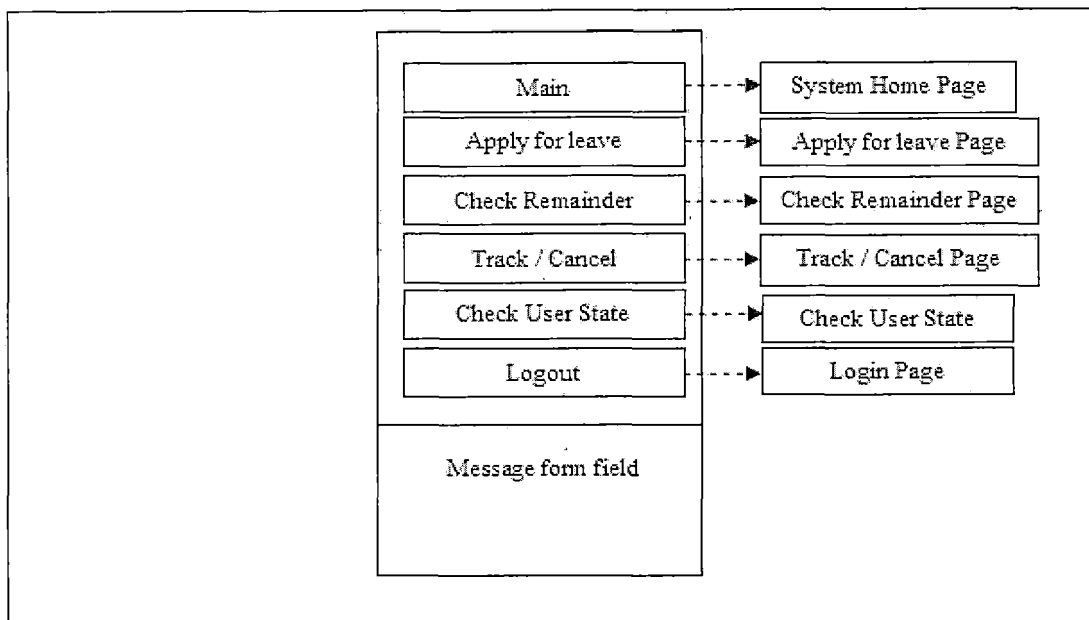


Figure 17. Message Page (General User)

Figure 18 represents the Track Result Page which is used by a general user, a user can see the track result through this page, and he can cancel applications which are not approved yet, or go to other pages through other links.

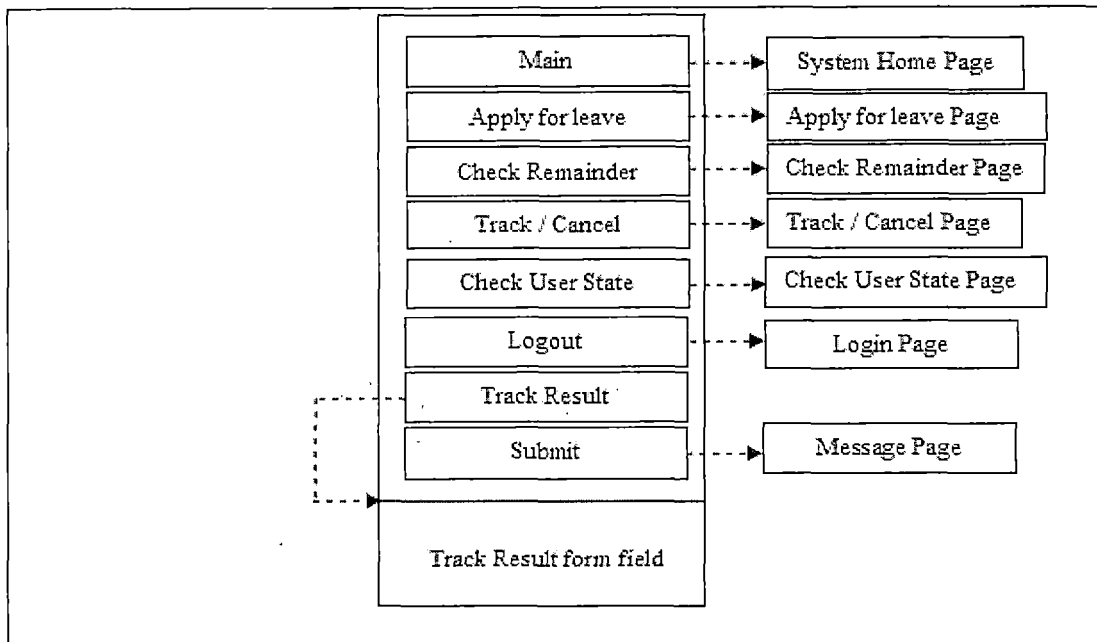


Figure 18. Track Result Page (General User)

Figure 19 represents the Password Quote Page which is used by a general user, a user can quote his password through this page, or go back to the Login Page.

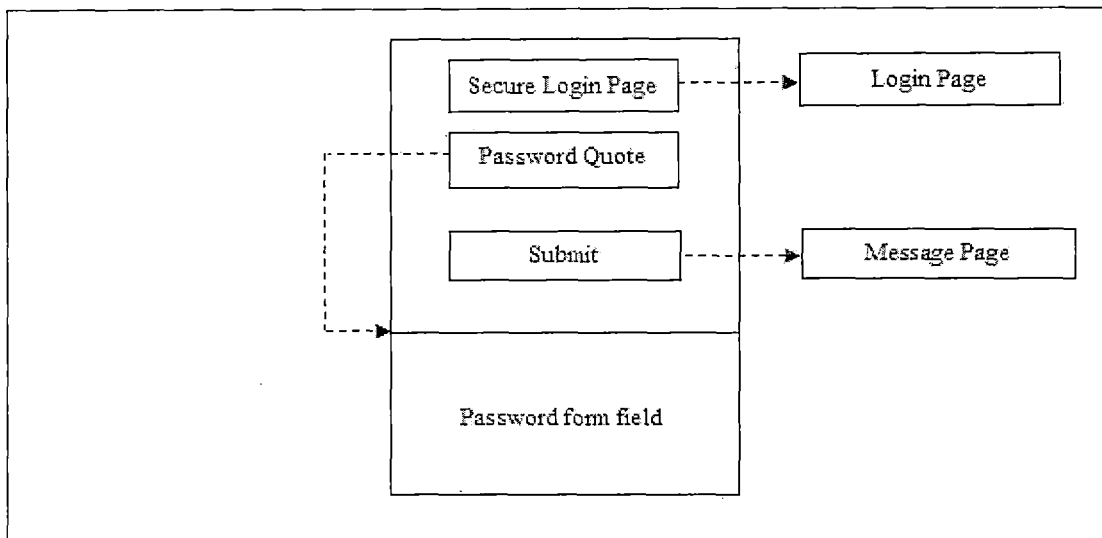


Figure 19. Password Quote Page (General User)

3.5.1.2 Manager. Figure 20 represents the Login Page which is used by a manager, if the user successfully logs the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password

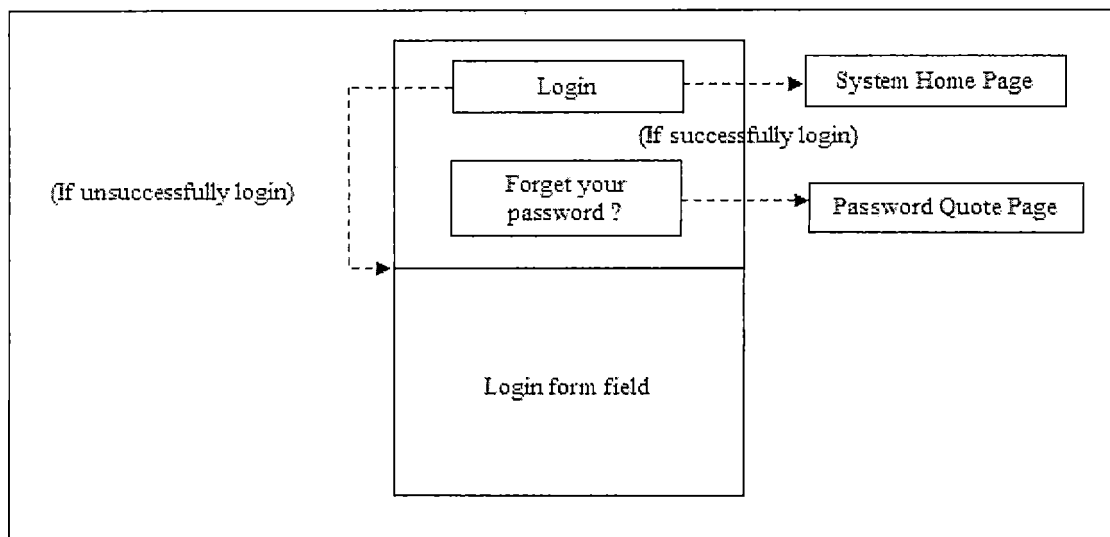


Figure 20. Login Page (Manager)

Figure 21 represents the System Home Page which is used by a manager, a user can choose which system he is going to use, or logout to the Login Page.

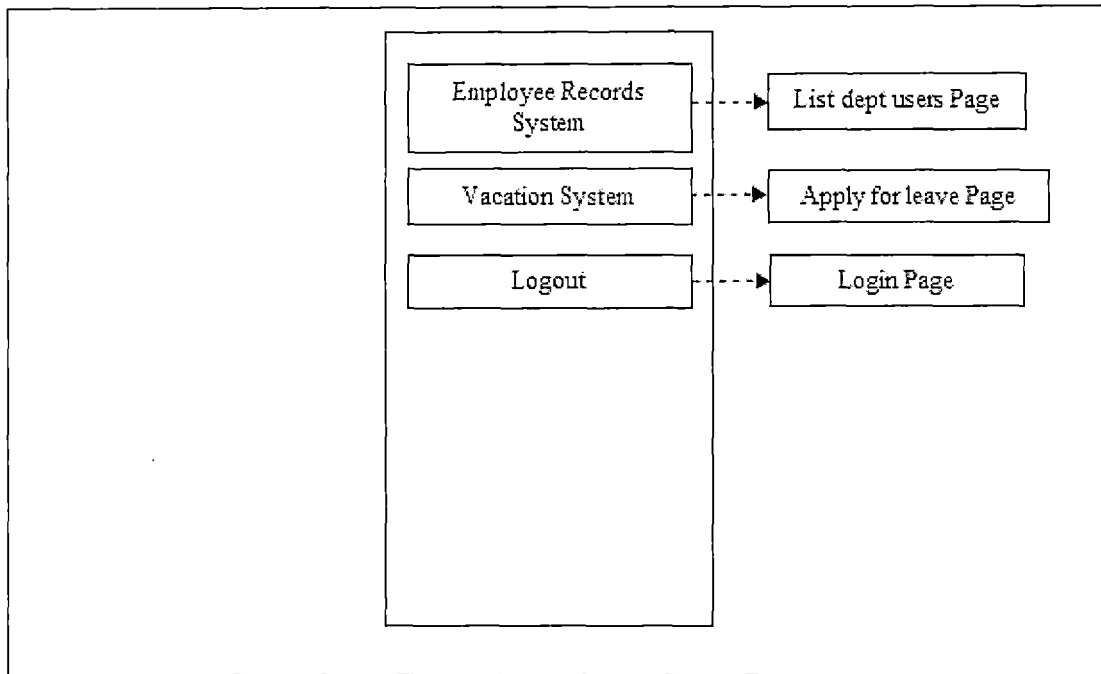


Figure 21. System Home Page (Manager)

Figure 22 represents the Apply for leave Page which is used by a manager, a user can apply for leave through this page, or go to other pages through other links.

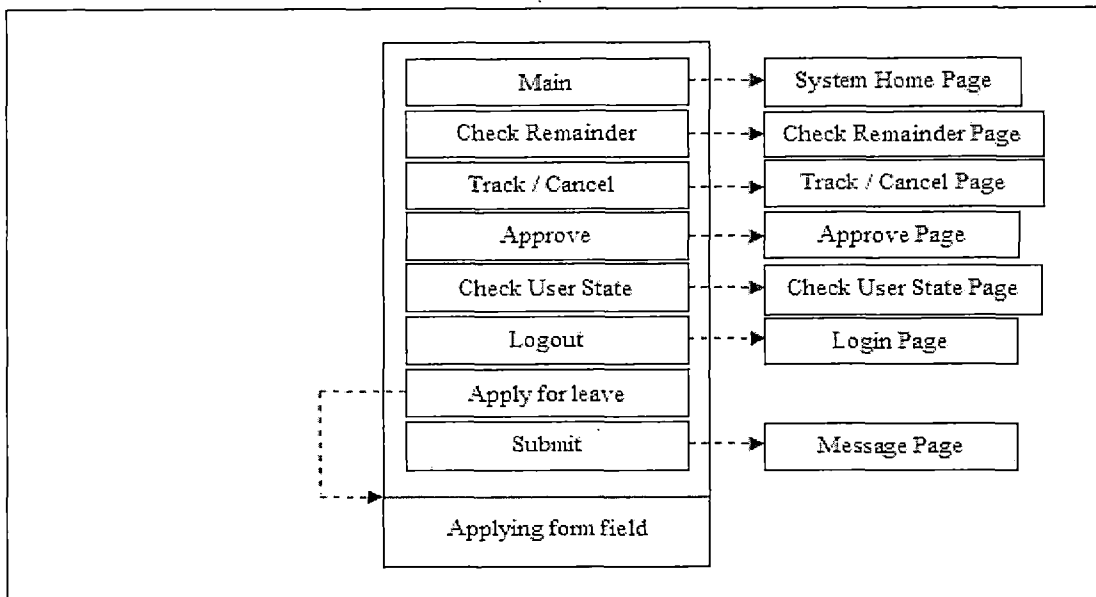


Figure 22. Apply for Leave Page (Manager)

Figure 23 represents the Check Remainder Page which is used by a manager, a user can check how many days are available for leave through this page, or go to other pages through other links.

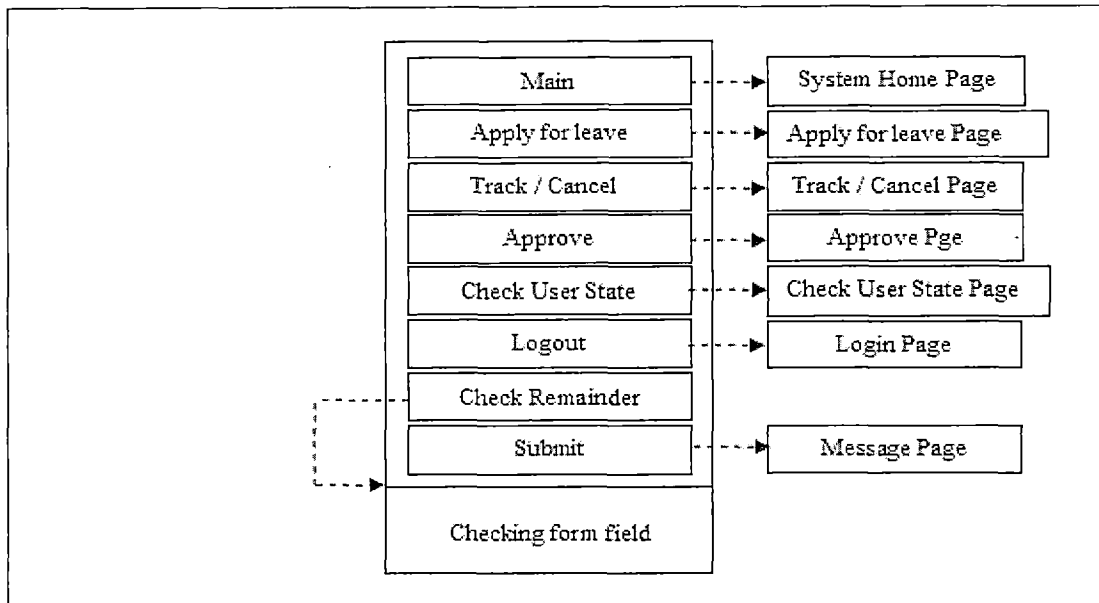


Figure 23. Check Remainder Page (Manager)

Figure 24 represents the Track / Cancel Page which is used by a manager, a user can track his application through this page, or go to other pages through other links.

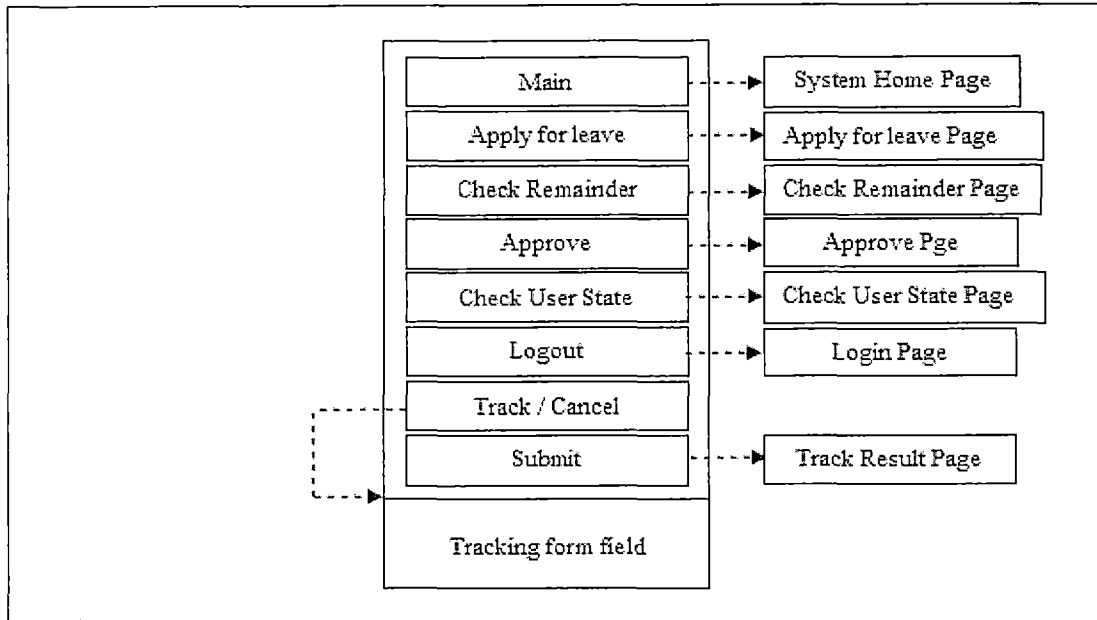


Figure 24. Track/Cancel Page (Manager)

Figure 25 represents the Approve Page which is used by a manager, a manager can approve applications in the department through this page, or go to other pages through other links.

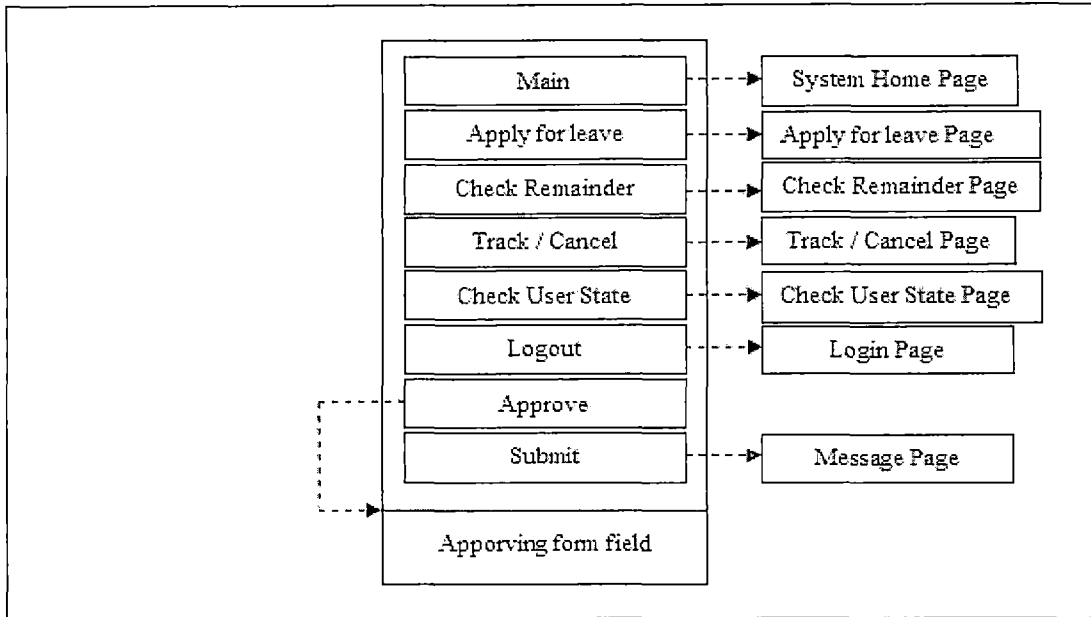


Figure 25. Approve Page (Manager)

Figure 26 represents the Check User State Page which is used by a manager, a user can check the state or other employee's state through this page, present or on vacation, or go to other pages through other links.

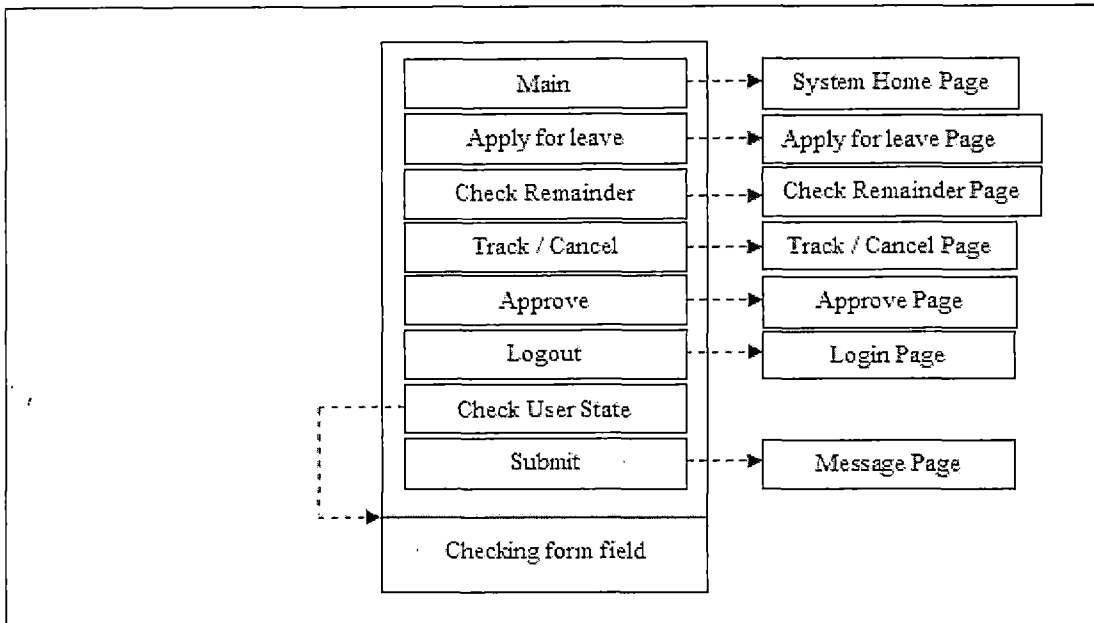


Figure 26. Check User State Page (Manager)

Figure 27 represents the Message Page which is used by a manager, a user can read the system message through this page, or go to other pages through other links.

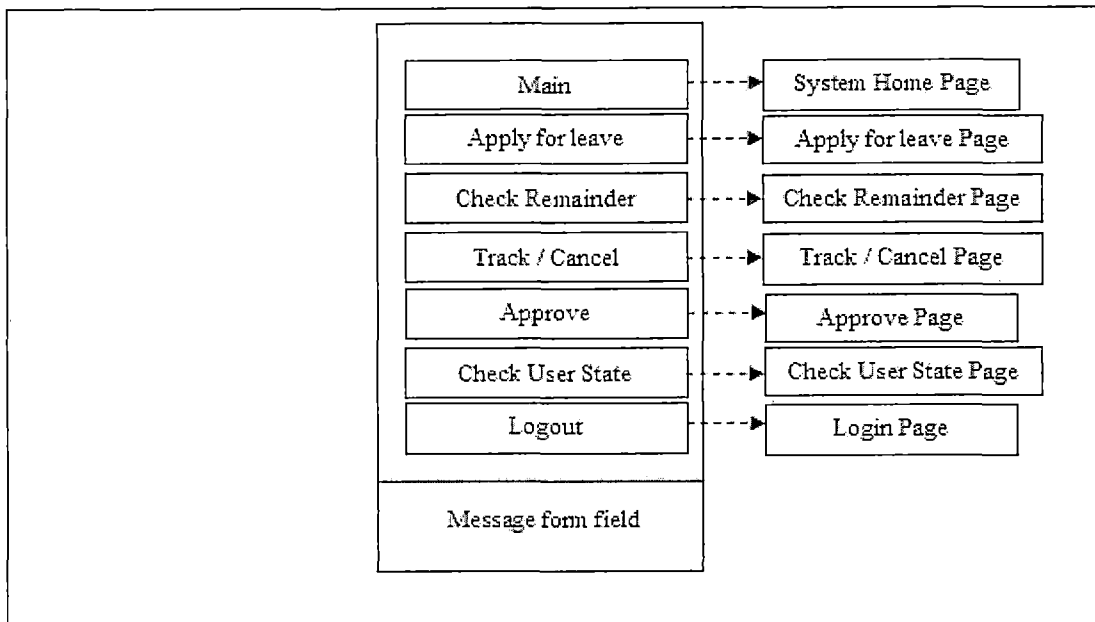


Figure 27. Message Page (Manager)

Figure 28 represents the Track Result Page which is used by a manager, a user can see the track result through this page, and he can cancel applications which are not approved yet, or go to other pages through other links.

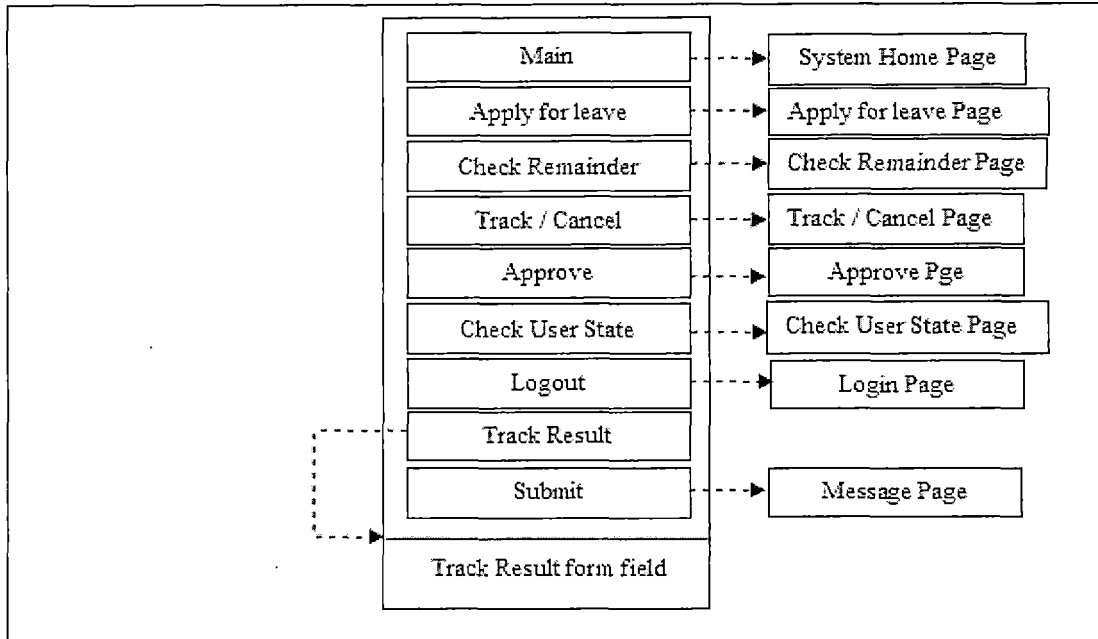


Figure 28. Track Result Page (Manager)

Figure 29 represents the Password Quote Page which is used by a manager, a user can quote his password through this page, or go back to the Login Page.

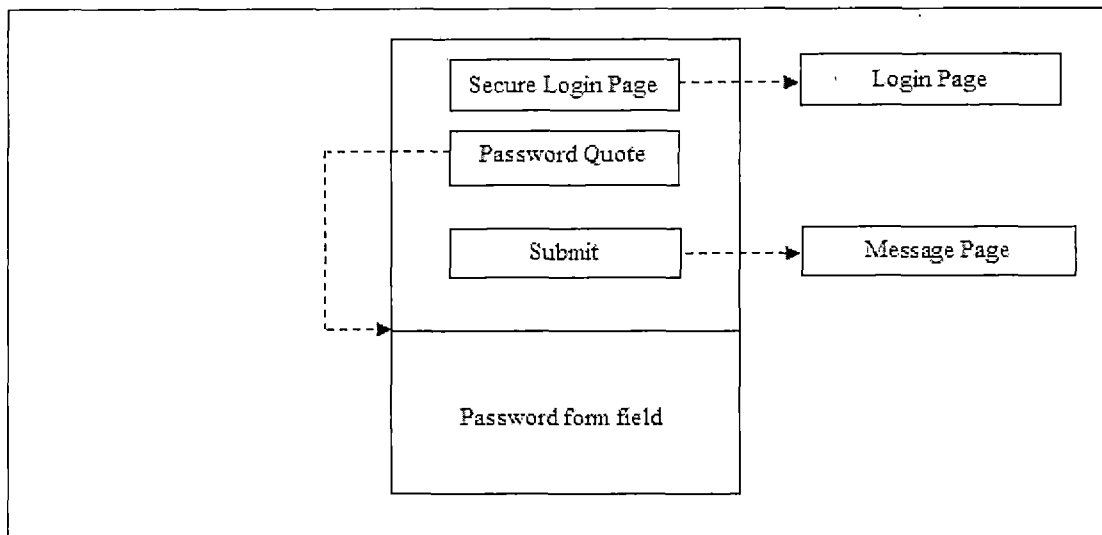


Figure 29. Password Quote Page (Manager)

3.5.1.3 Senior Manager. Figure 30 represents the Login Page which is used by a senior manager, if the user successfully logs in the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

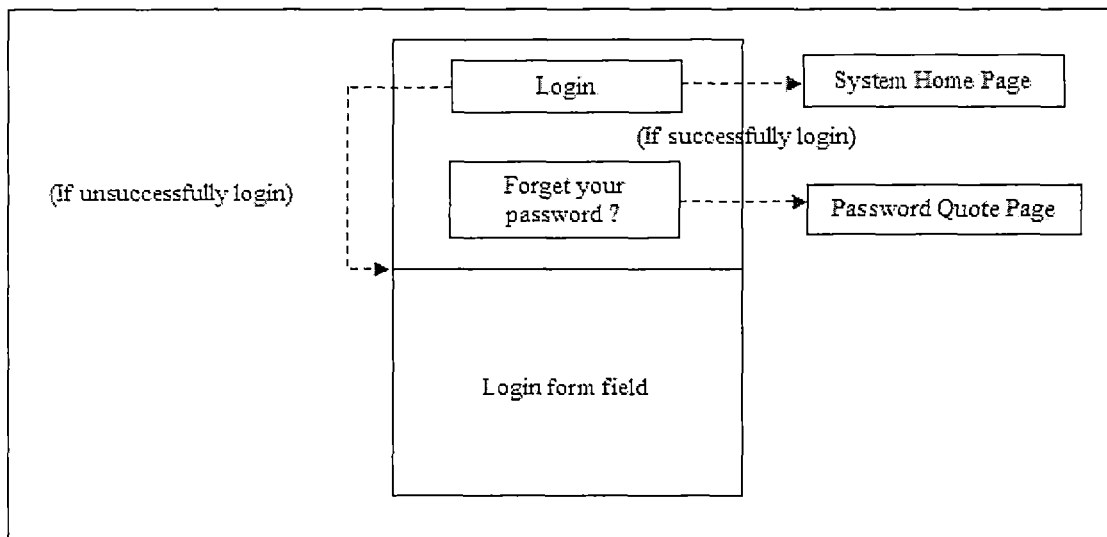


Figure 30. Login Page (Senior Manager)

Figure 31 represents the System Home Page which is used by a senior manager, a user can choose which system he is going to use, or logout to the Login Page.

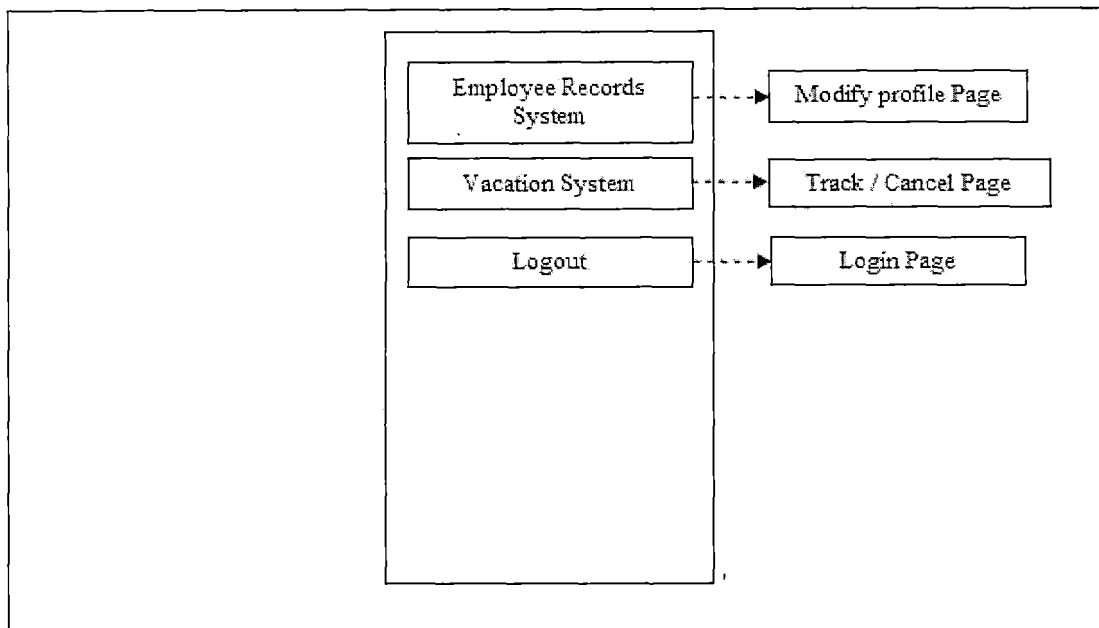


Figure 31. System Home Page (Senior Manager)

Figure 32 represents the Track / Cancel Page which is used by a senior manager, a user can track his application through this page, or go to other pages through other links.

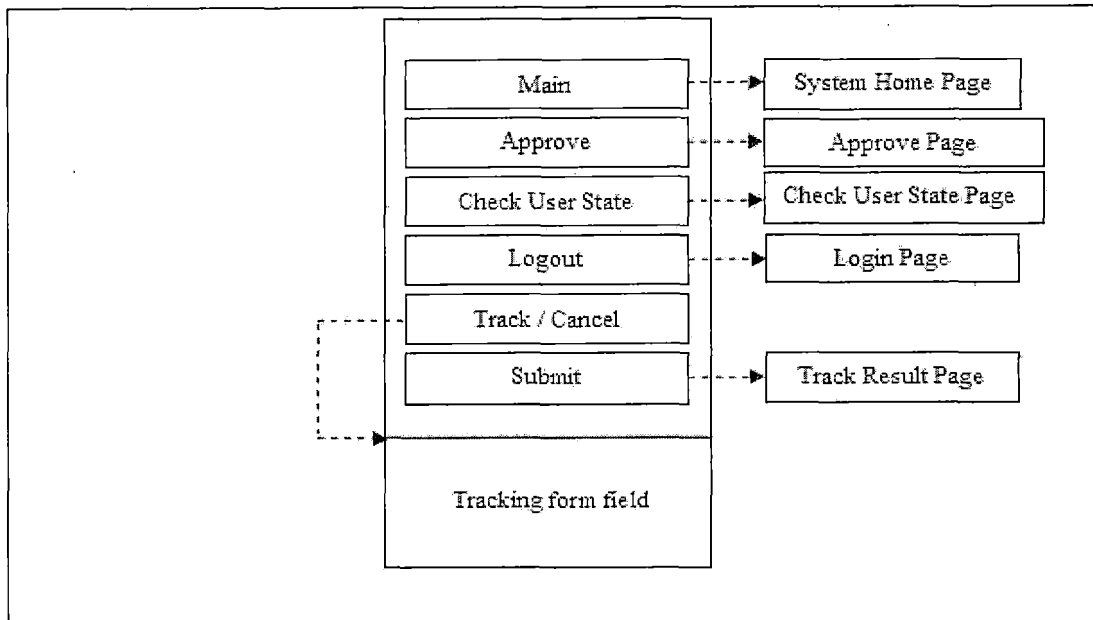


Figure 32. Track/Cancel Page (Senior Manager)

Figure 33 represents the Approve Page which is used by a senior manager, a senior manager can approve applications which are over 3 days through this page, or go to other pages through other links.

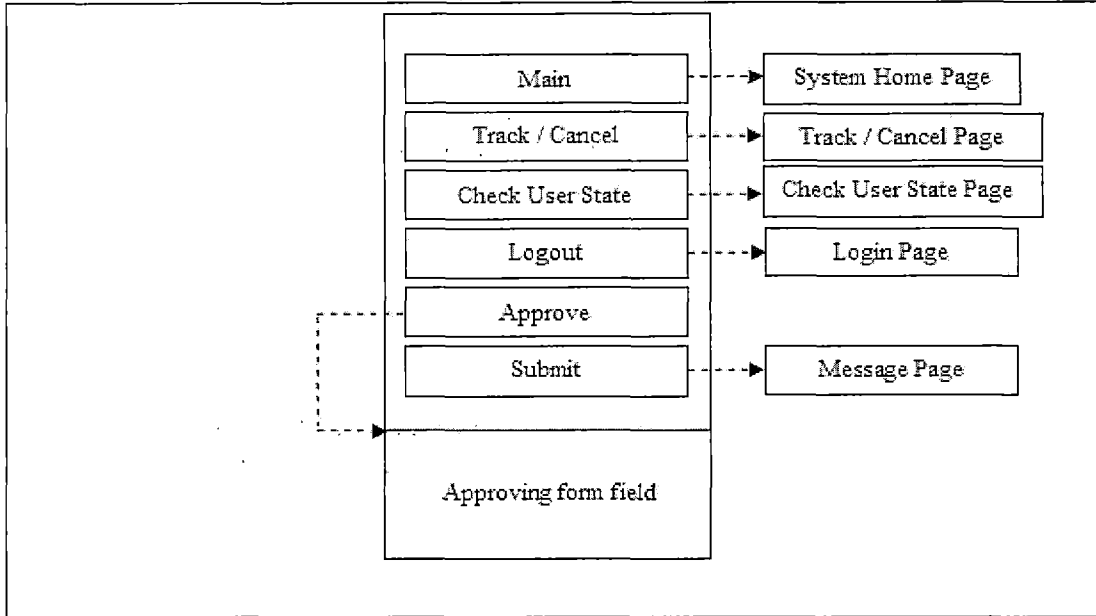


Figure 33. Approve Page (Senior Manager)

Figure 34 represents the Check User State Page which is used by a senior manager, a user can check the state or other employee's state through this page, present or on vacation, or go to other pages through other links.

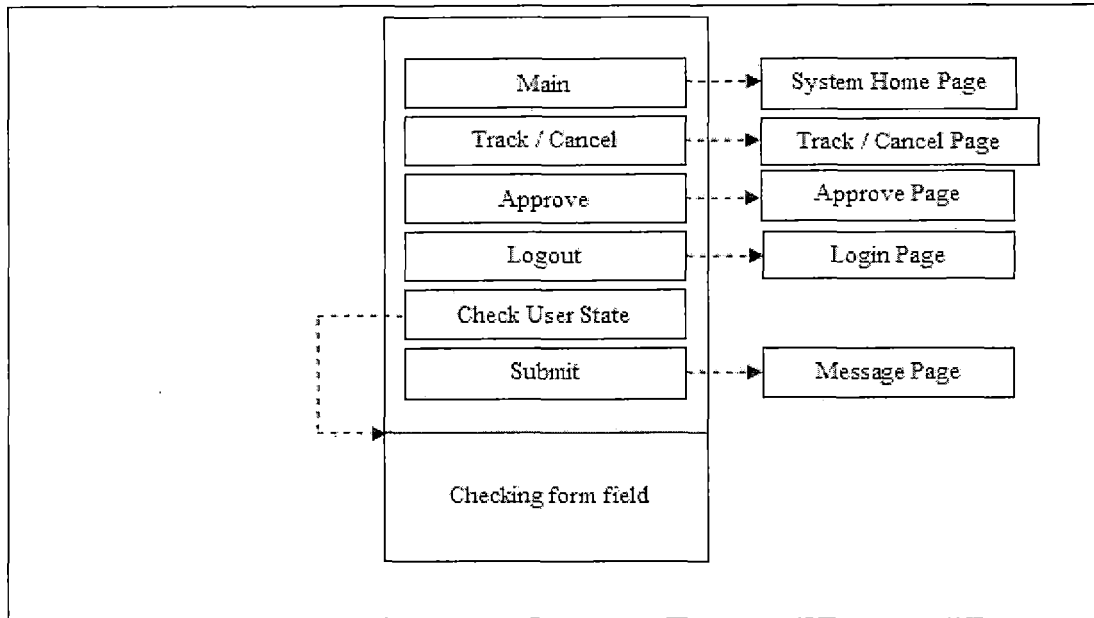


Figure 34. Check User State Page (Senior Manager)

Figure 35 represents the Message Page which is used by a senior manager, a user can read the system message through this page, or go to other pages through other links.

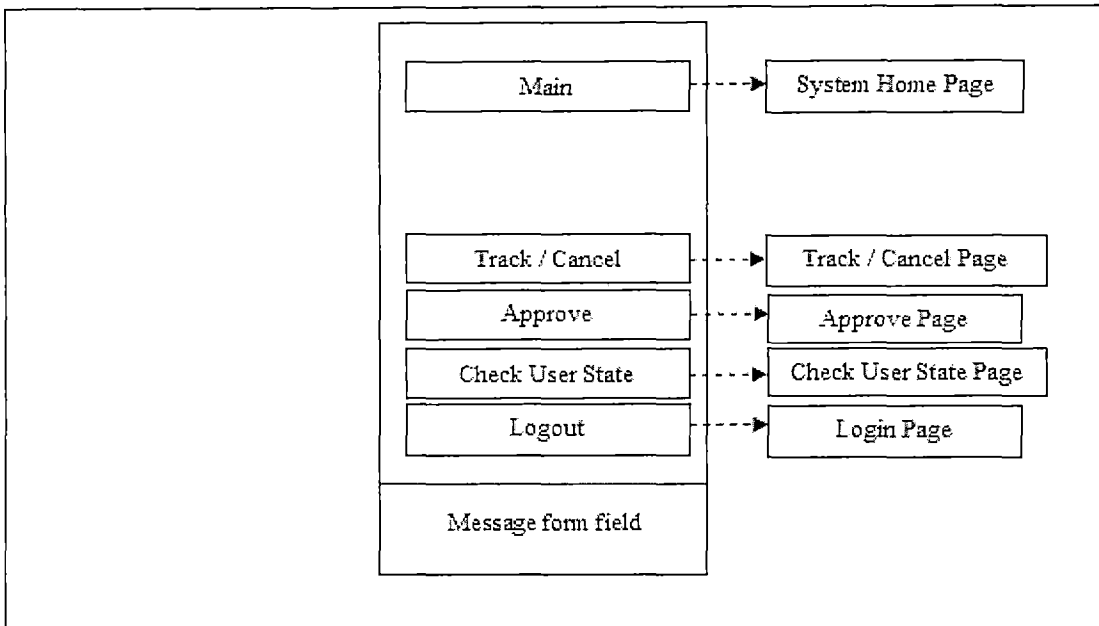


Figure 35. Message Page (Senior Manager)

Figure 36 represents the Track Result Page which is used by a senior manager, a user can see the track result through this page, and he can cancel applications which are not approved yet, or go to other pages through other links.

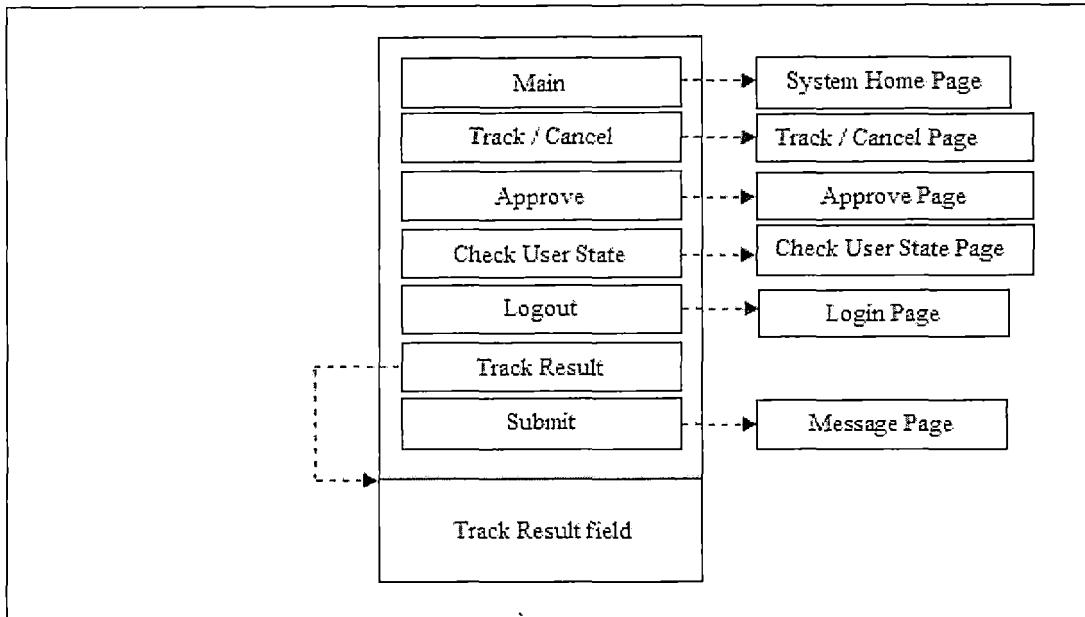


Figure 36. Track Result Page (Senior Manager)

Figure 37 represents the Password Quote Page which is used by a senior manager, a user can quote his password through this page, or go back to the Login Page.

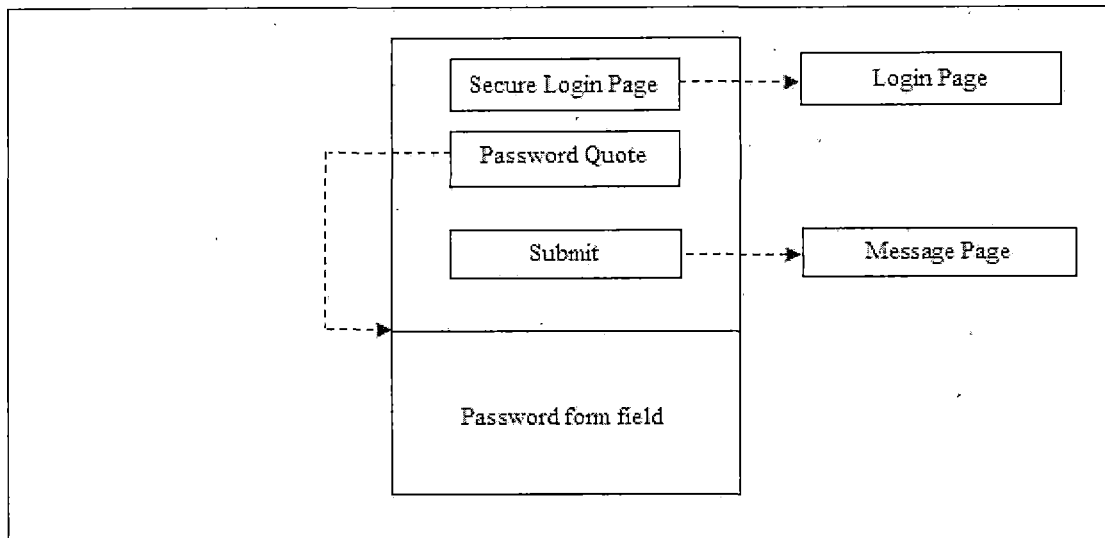


Figure 37. Password Quote Page (Senior Manager)

3.5.1.4 System Administrator. Figure 38 represents the Login Page which is used by a system administrator, if the user successfully logs in the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

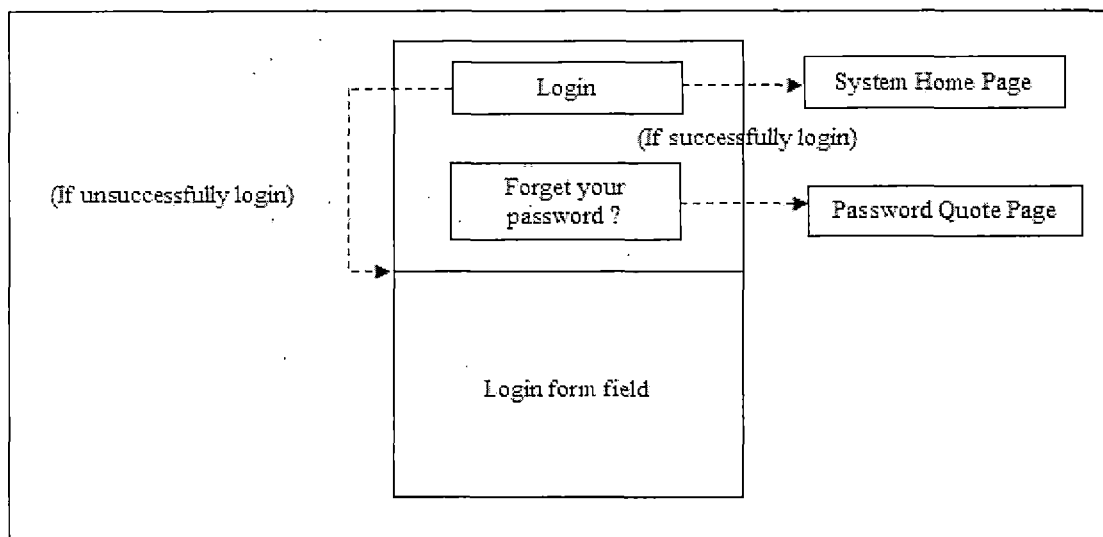


Figure 38. Login Page (System Administrator)

Figure 39 represents the System Home Page which is used by a system administrator, a user can choose which system he is going to use, or logout to the Login Page.

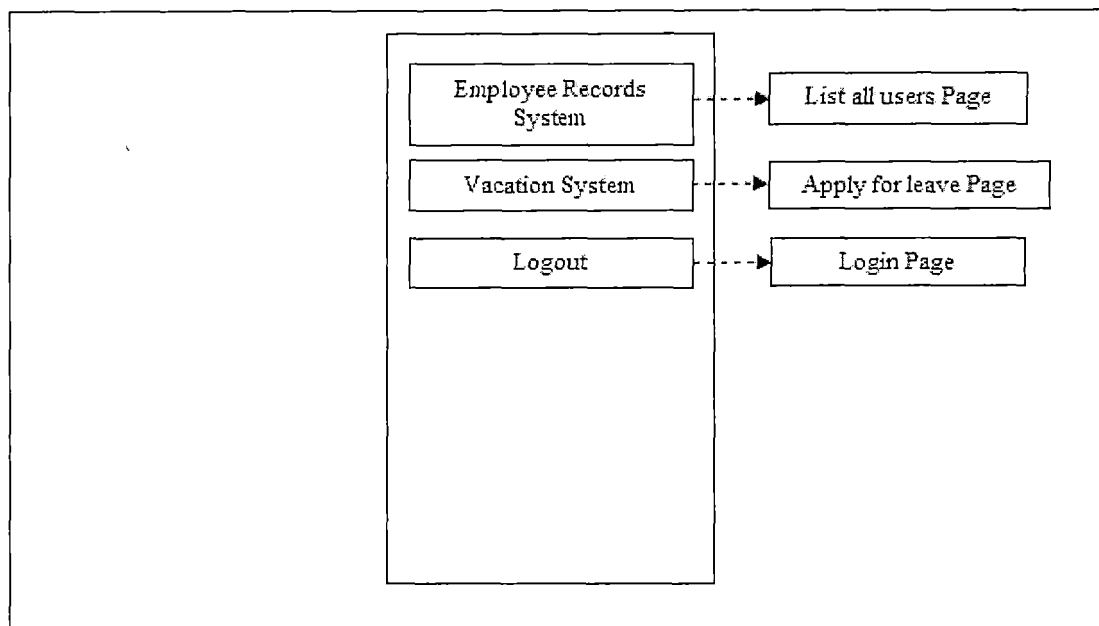


Figure 39. System Home Page (System Administrator)

Figure 40 represents the Apply for leave Page which is used by a system administrator, a system administrator can apply every user's application through this page, or go to other pages through other links.

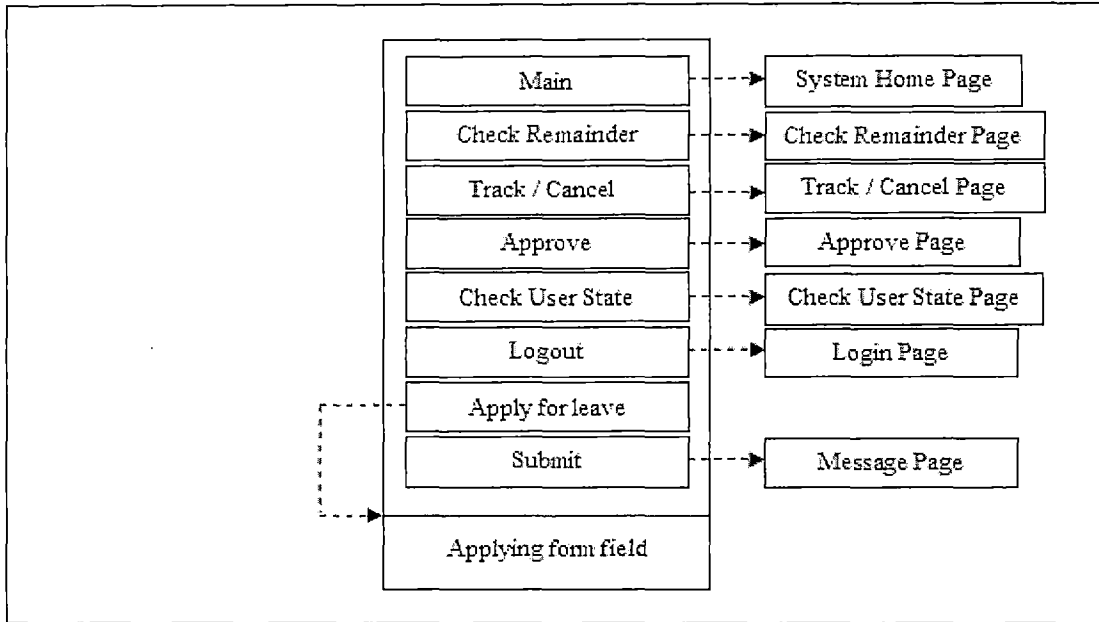


Figure 40. Apply for leave Page (System Administrator)

Figure 41 represents the Check Remainder Page which is used by a system administrator, a user can check every user's available days for leave through this page, or go to other pages through other links.

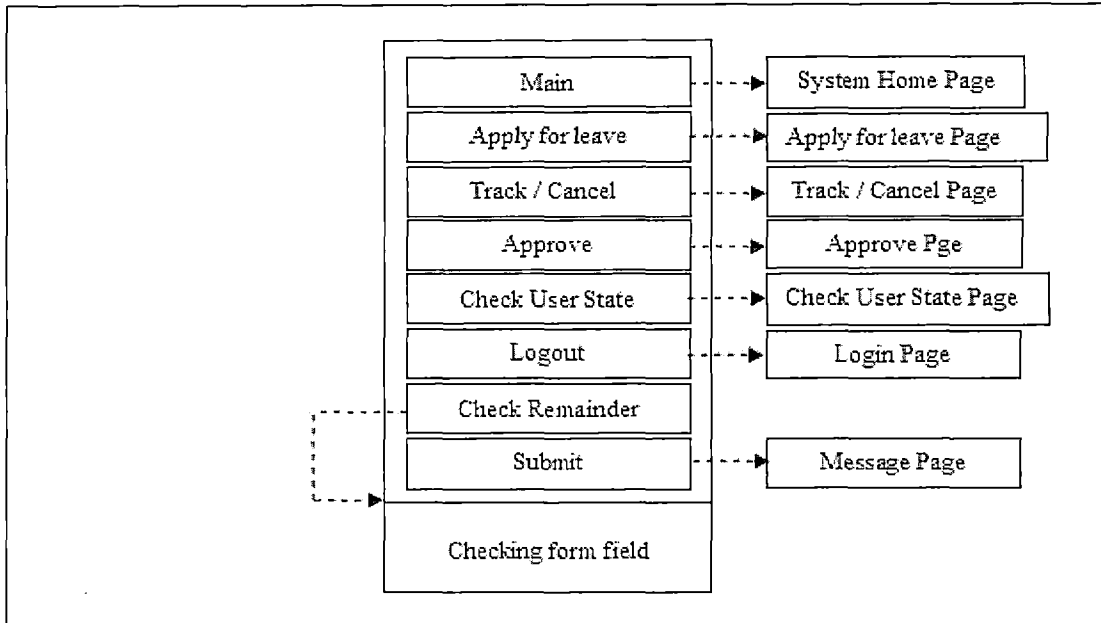


Figure 41. Check Remainder Page (System Administrator)

Figure 42 represents the Track / Cancel Page which is used by a system administrator, a system administrator can track all applications through this page, or go to other pages through other links.

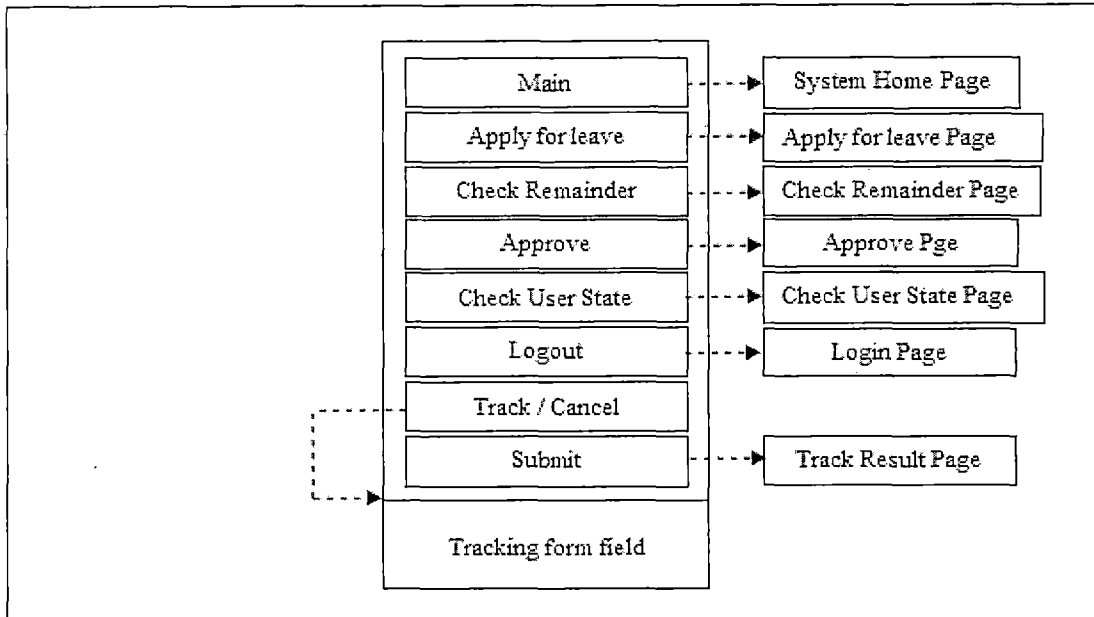


Figure 42. Track/Cancel Page (System Administrator)

Figure 43 represents the Approve Home Page which is used by a system administrator, a system administrator can approve all applications through this page, or go to other pages through other links.

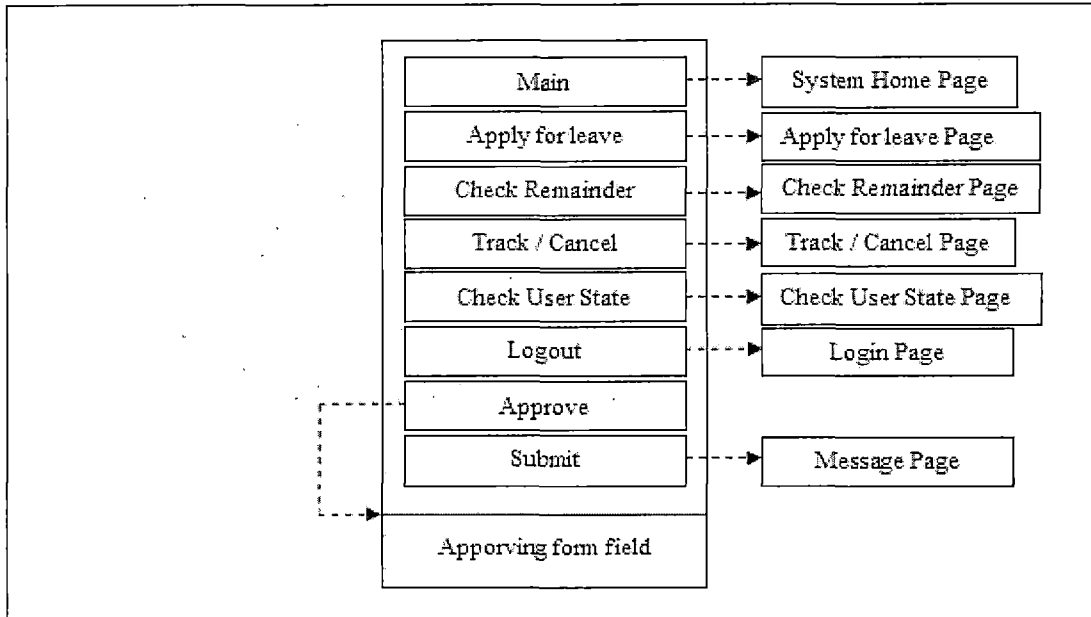


Figure 43. Approve Page (System Administrator)

Figure 44 represents the Check User State Page which is used by a system administrator, a user can check the state or other employee's state through this page, present or on vacation, or go to other pages through other links.

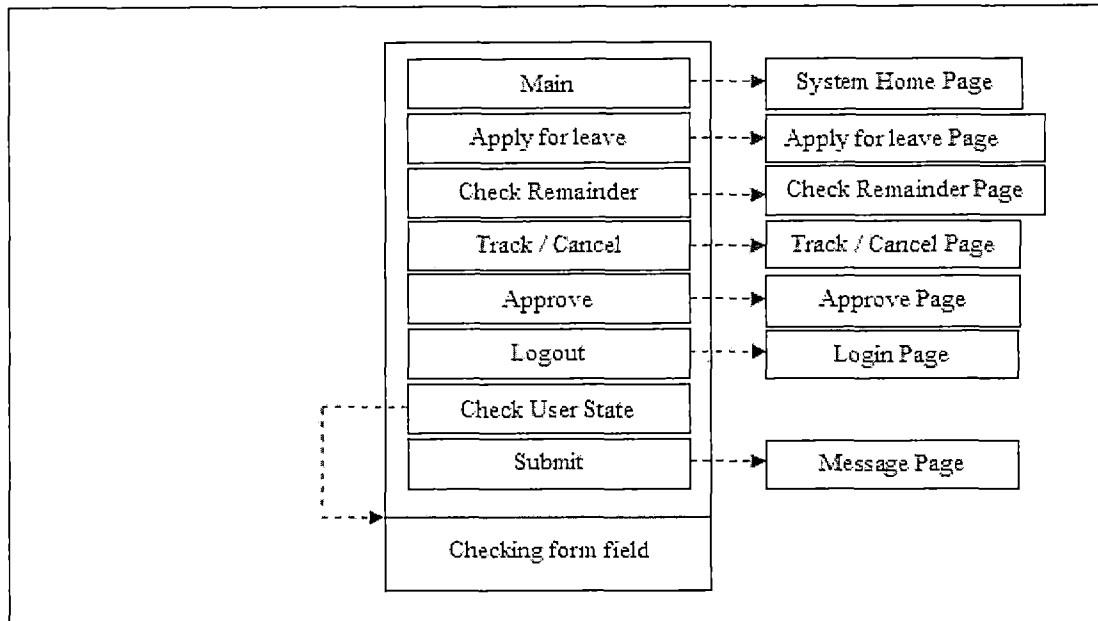


Figure 44. Checke User State Page (System Administrator)

Figure 45 represents the Message Page which is used by a system administrator, a user can read the system message through this page, or go to other pages through other links.

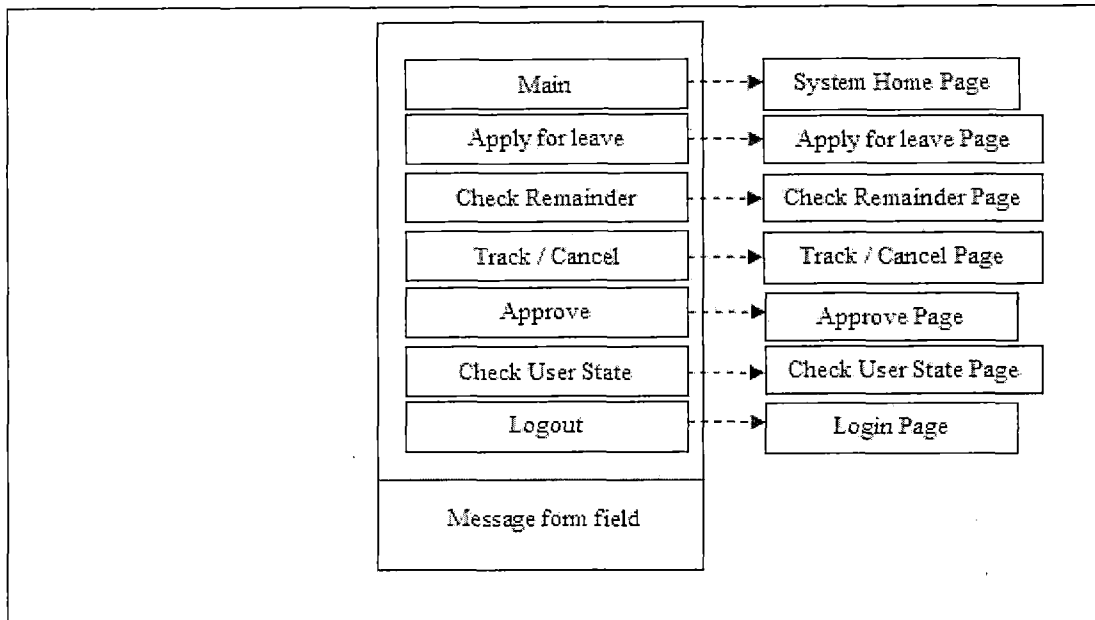


Figure 45. Message Page (System Administrator)

Figure 46 represents the Track Result Page which is used by a system administrator, a user can see the track result through this page, and he can cancel applications which are not approved yet, or go to other pages through other links.

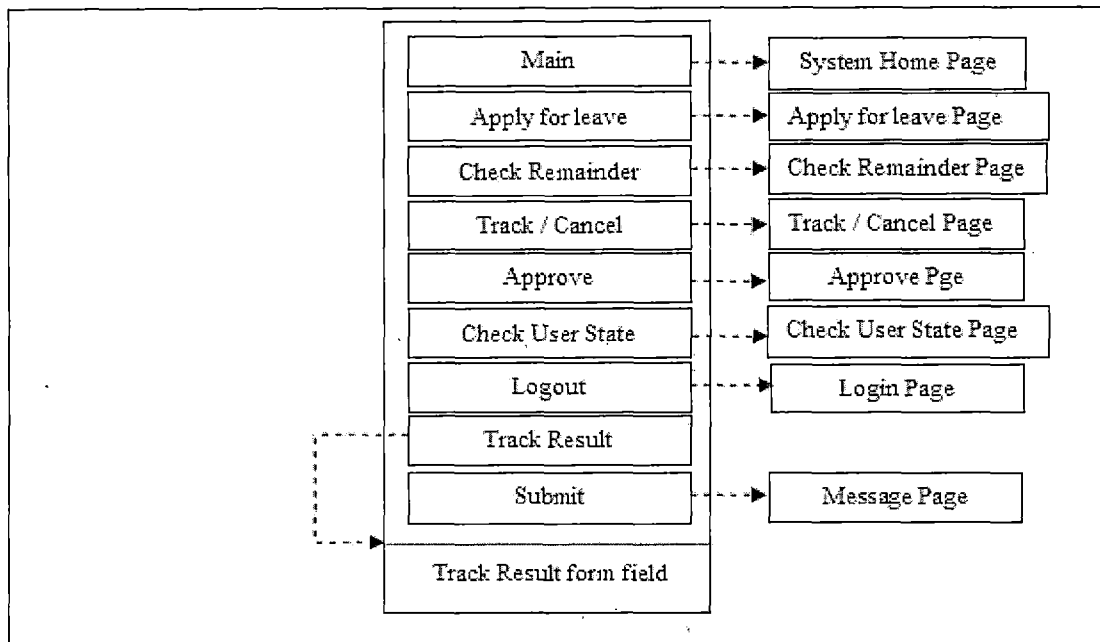


Figure 46. Track Result Page (System Administrator)

Figure 47 represents the Password Quote Page which is used by a system administrator, a user can quote his password through this page, or go back to the Login Page.

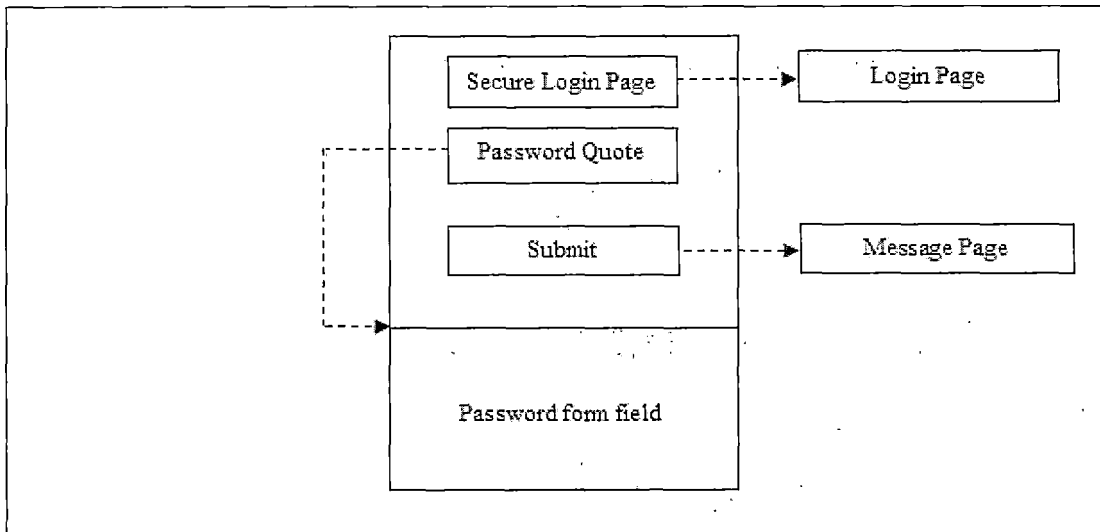


Figure 47. Password Quote Page (System Administrator)

3.5.2 Employee Records System

3.5.2.1 General User. Figure 48 represents the Login Page which is used by a general user, if the user successfully logs in the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

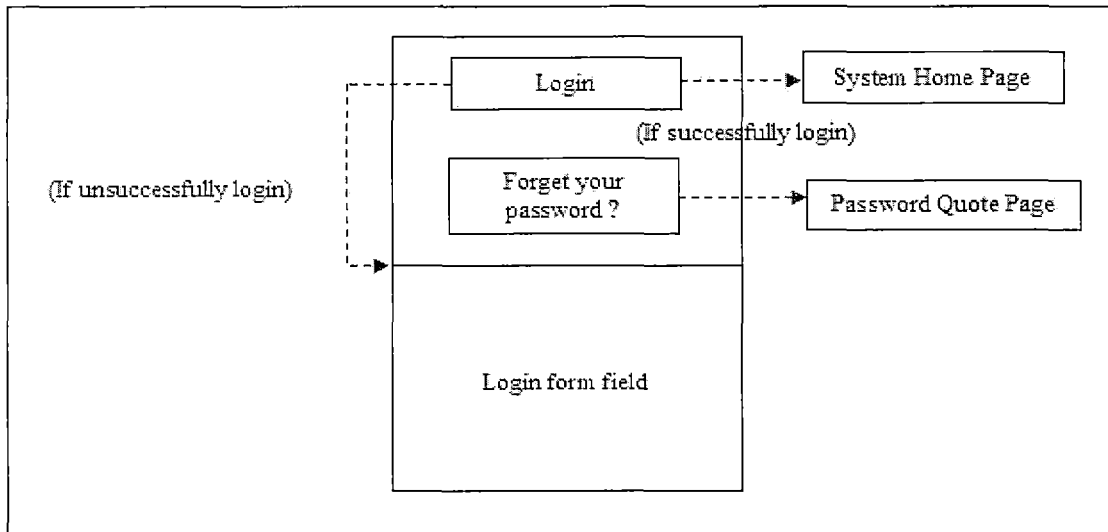


Figure 48. Login Page (General User)

Figure 49 represents the System Home Page which is used by a general user, a user can choose which system he is going to use, or logout to the Login Page.

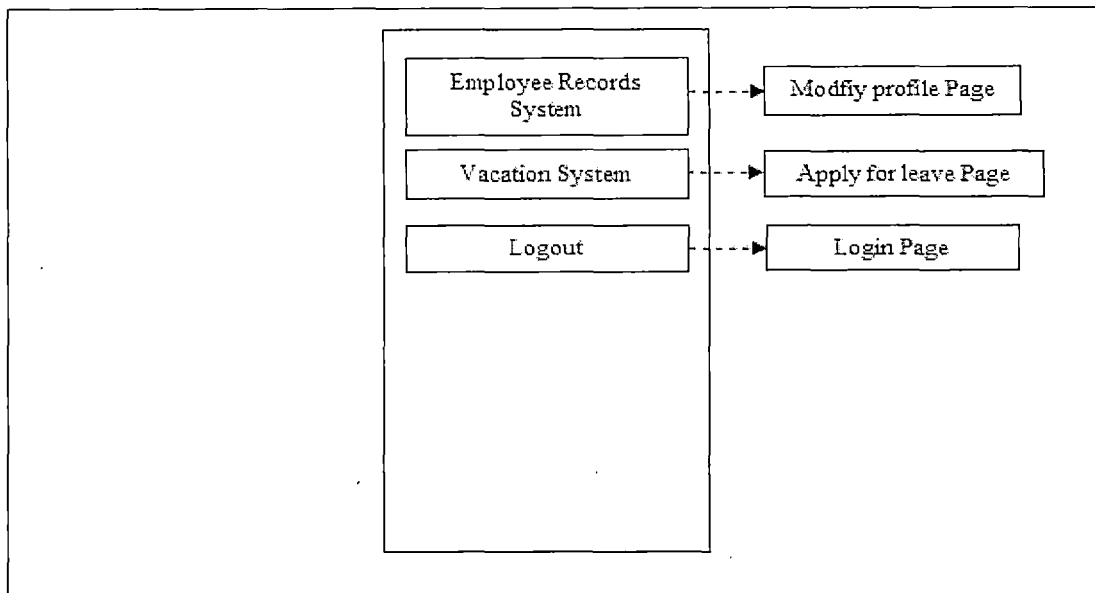


Figure 49. System Home Page (General User)

Figure 50 represents the Modify profile Page which is used by a general user, a user can modify his own profile through this page, go back to the System Home Page, or logout to the Login Page.

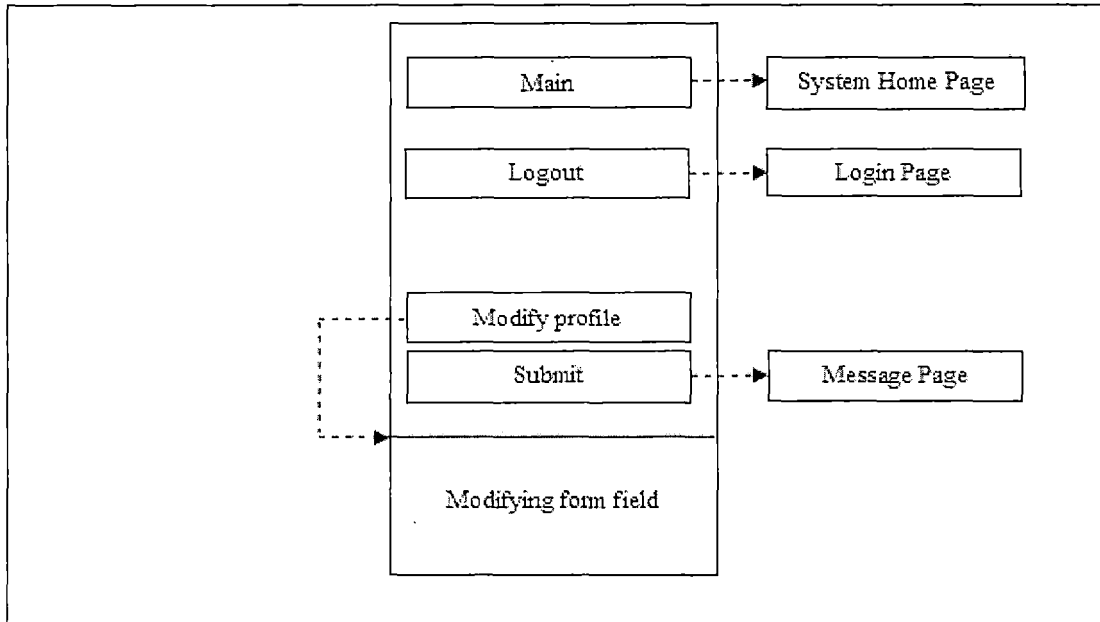


Figure 50. Modify Profile Page (General User)

Figure 51 represents the Message Page which is used by a general user, a user can read the system message through this page, or go to other pages through other links.

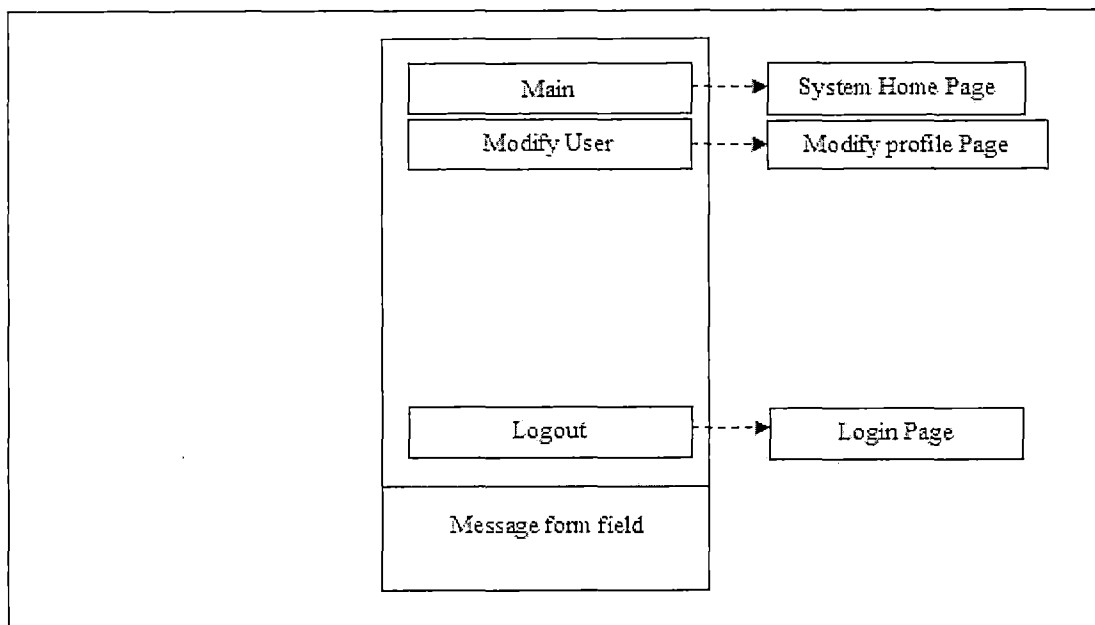


Figure 51. Message Page (General User)

Figure 52 represents the Password Quote Page which is used by a general user, a user can quote his password through this page, or go back to the Login Page.

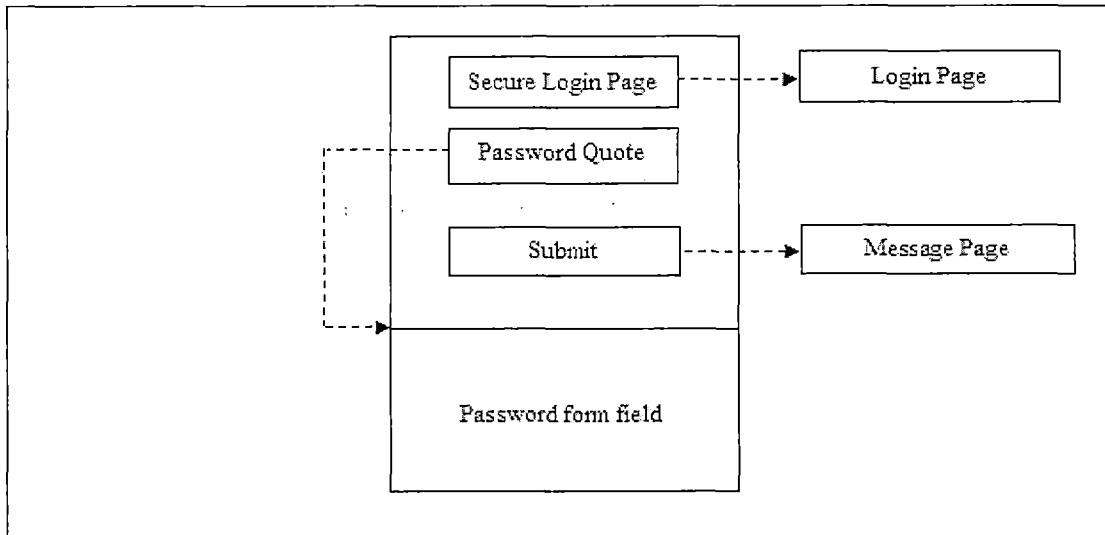


Figure 52. Password Quote Page (General User)

3.5.2.2 Manager. Figure 53 represents the Login Page which is used by a manager, if the user successfully logs the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

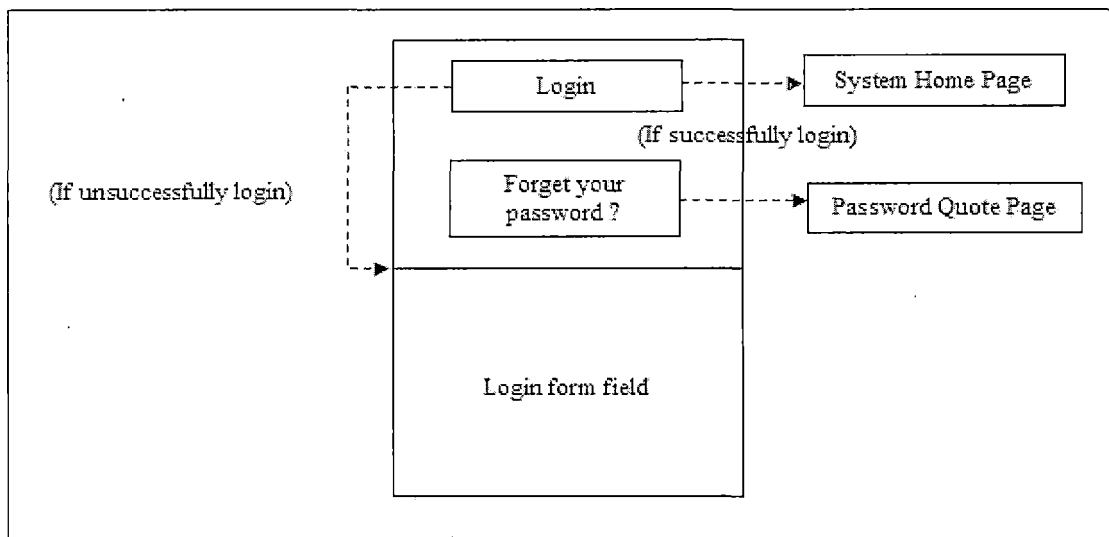


Figure 53. Login Page (Manager)

Figure 54 represents the System Home Page which is used by a manager, a user can choose which system he is going to use, or logout to the Login Page.

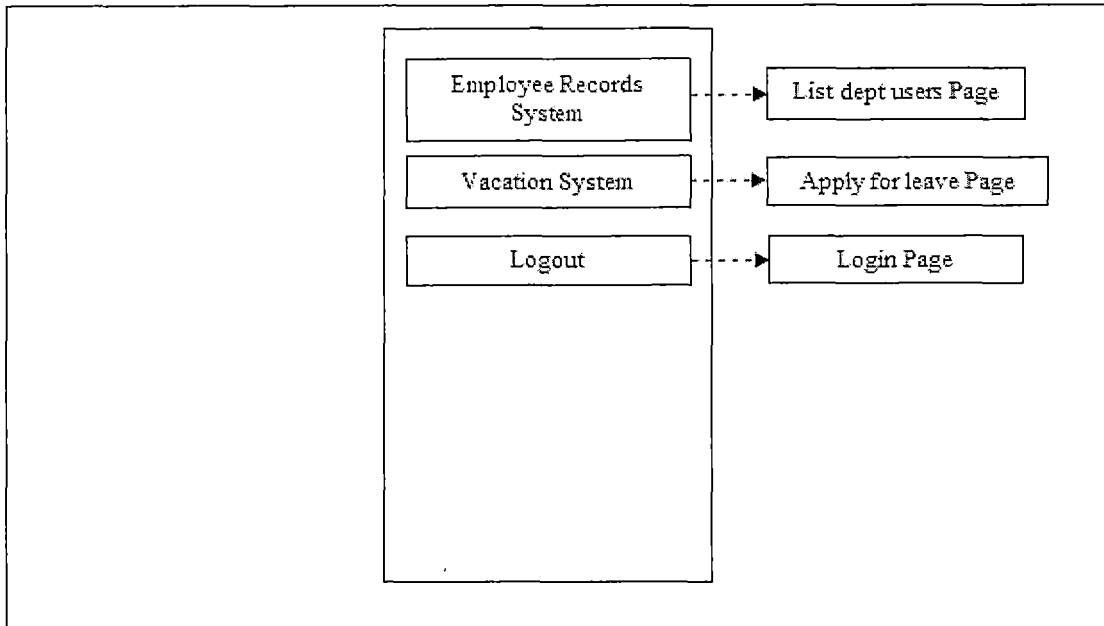


Figure 54. System Home Page (Manager)

Figure 55 represents the List dept users Page which is used by a manager, a manager can see all users in the department through this page, or go to other pages through other links.

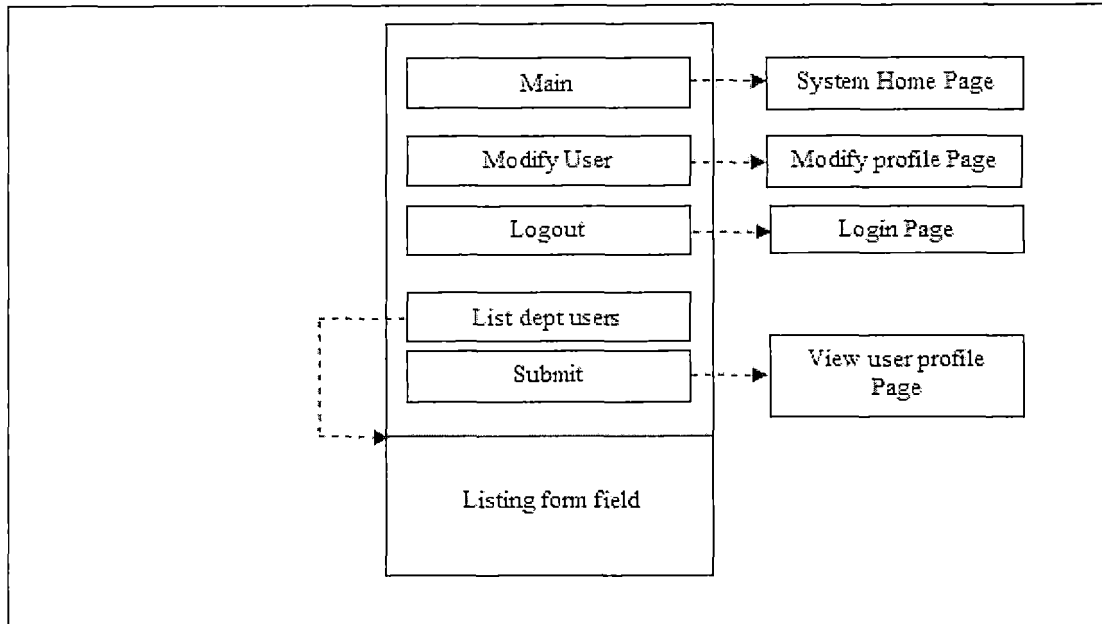


Figure 55. List Dept Users Page (Manager)

Figure 56 represents the Modify profile Page which is used by a manager, a user can modify his own profile through this page, or go to other pages through other links.

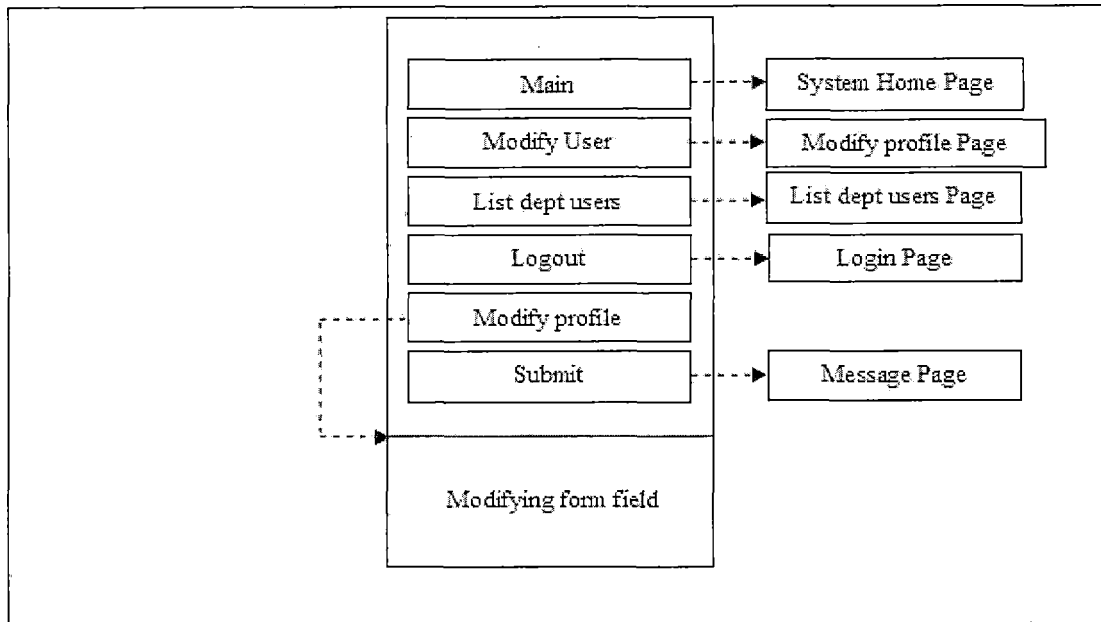


Figure 56. Modify Profile Page (Manager)

Figure 57 represents the Message Page which is used by a manager, a user can read the system message through this page, or go to other pages through other links.

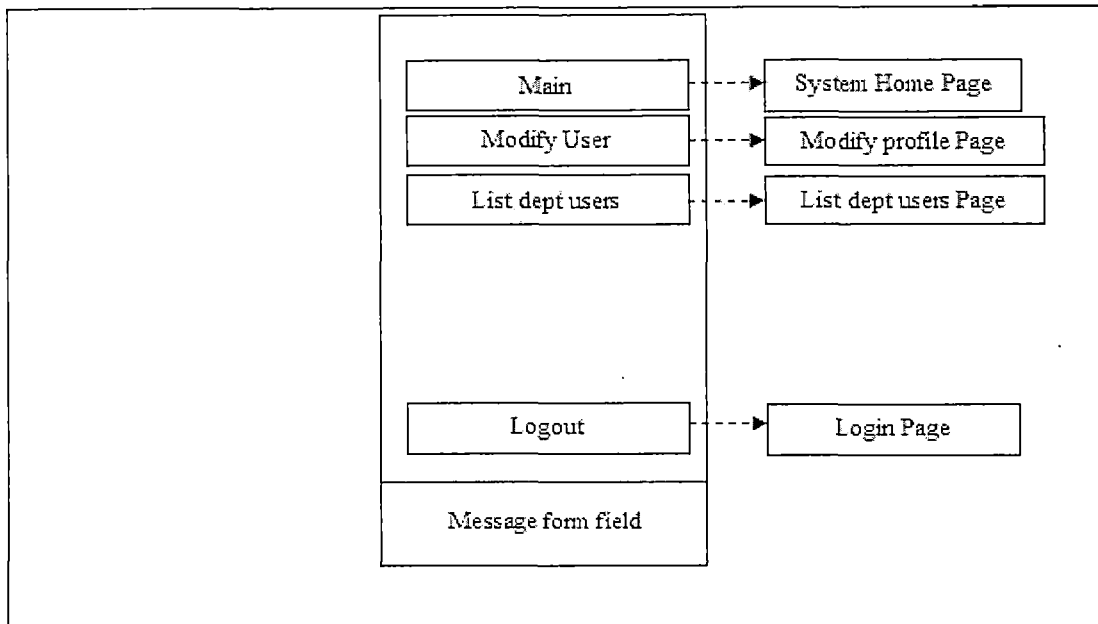


Figure 57. Message Page (Manager)

Figure 58 represents the Password Quote Page which is used by a manager, a user can quote his password through this page, or go back to the Login Page.

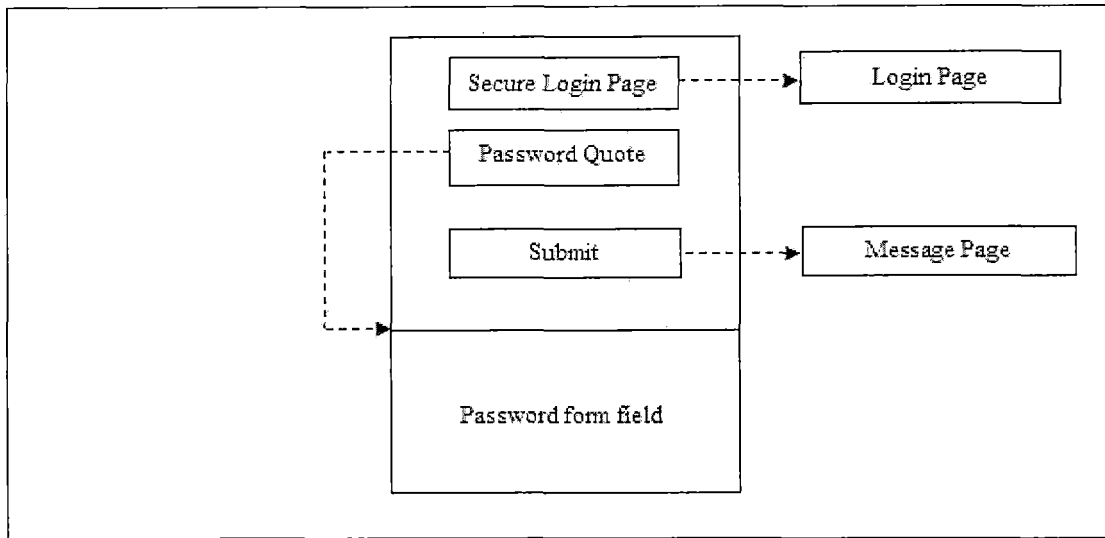


Figure 58. Password Quote Page (Manager)

3.5.2.3 Senior Manager. Figure 59 represents the Login Page which is used by a senior manager, if the user successfully logs in the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

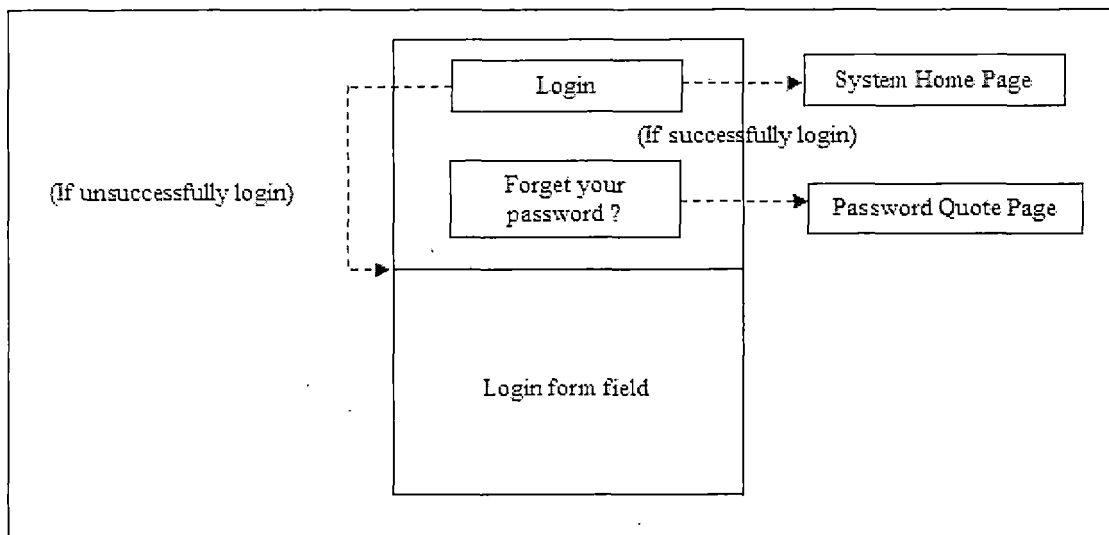


Figure 59. Login Page (Senior Manager)

Figure 60 represents the System Home Page which is used by a senior manager, a user can choose which system he is going to use, or logout to the Login Page.

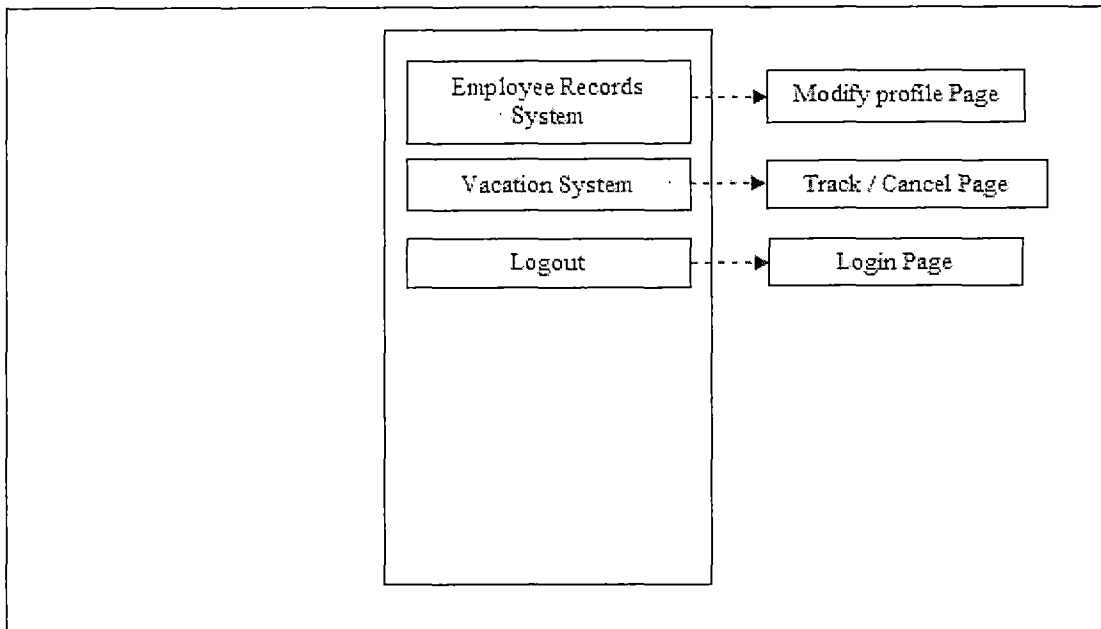


Figure 60. System Home Page (Senior Manager)

Figure 61 represents the Modify profile Page which is used by a senior manager, a user can modify his own profile through this page, go back to the System Home Page, or logout to the Login Page.

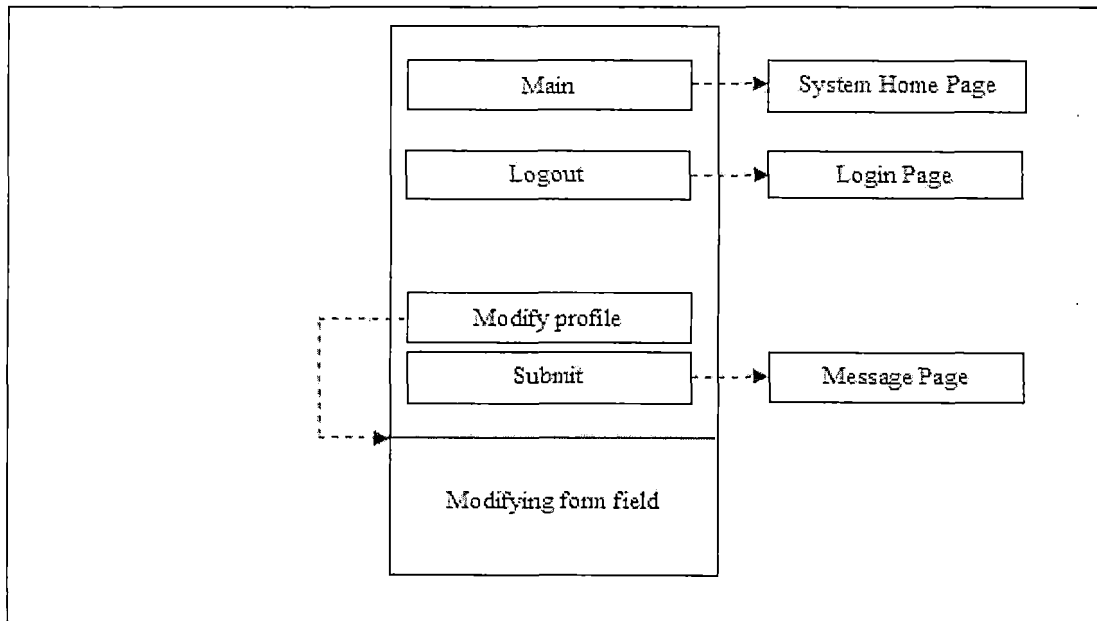


Figure 61. Modify Profile Page (Senior Manager)

Figure 62 represents the Message Page which is used by a senior manager, a user can read the system message through this page, or go to other pages through other links.

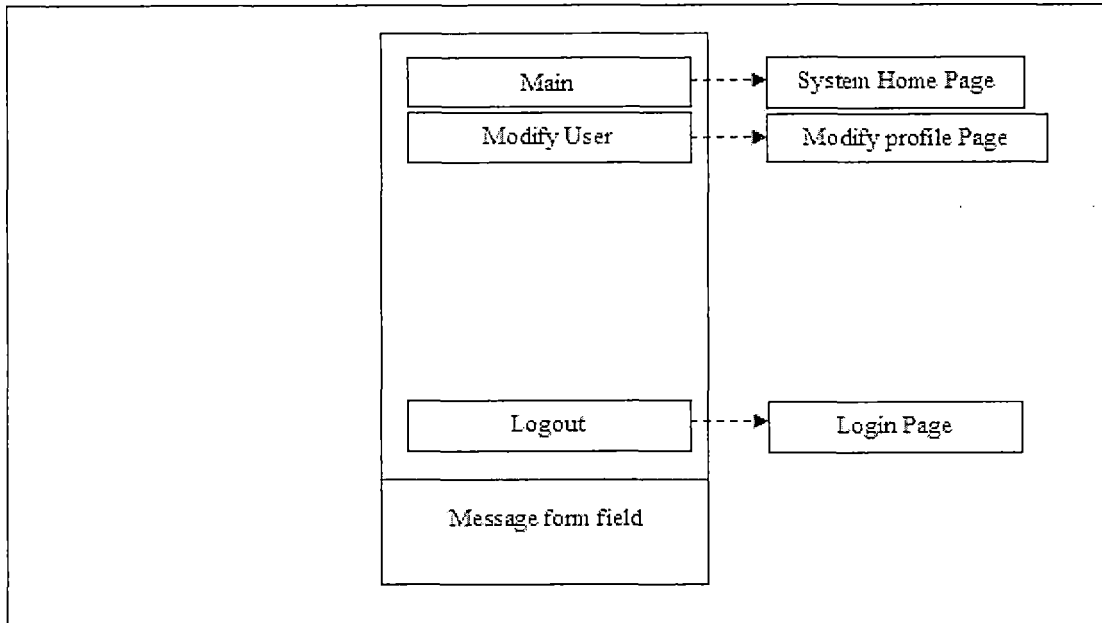


Figure 62. Message Page (Senior Manager)

Figure 63 represents the Password Quote Page which is used by a senior manager, a user can quote his password through this page, or go back to the Login Page.

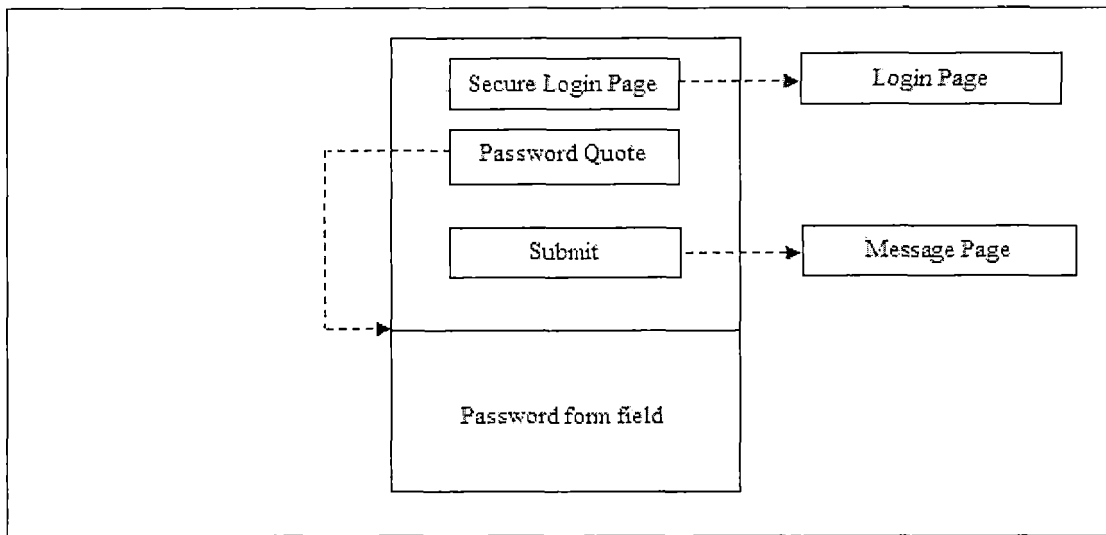


Figure 63. Password Quote Page (Senior Manager)

3.5.2.4 System Administrator. Figure 64 represents the Login Page which is used by a system administrator, if the user successfully logs in the system, the user will be redirected to the System Home Page; if not, the user will stay at the Login Page. If the user forgets his password, the user can go to the Password Page to request the password.

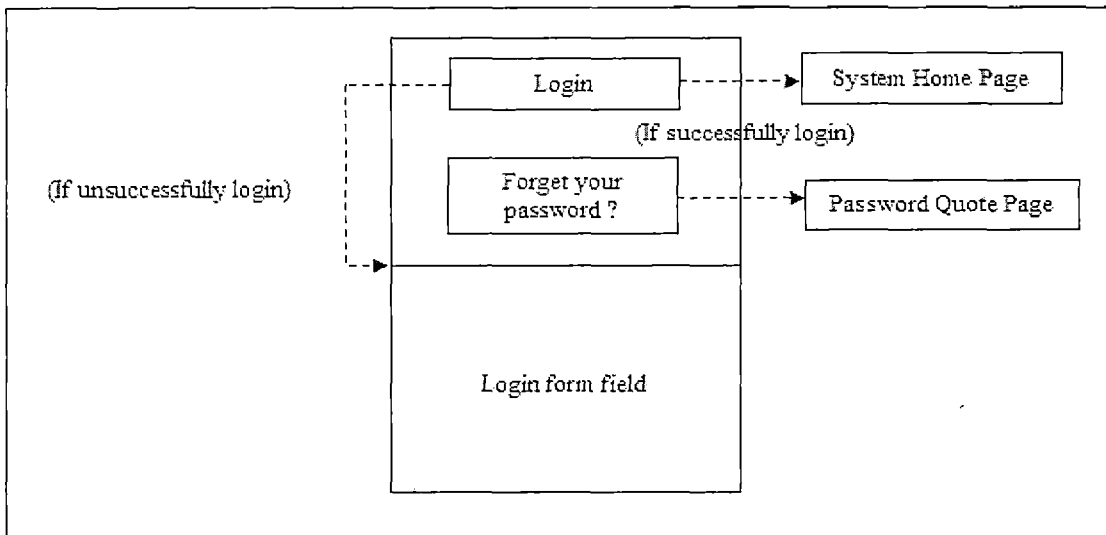


Figure 64. Login Page (System Administrator)

Figure 65 represents the System Home Page which is used by a system administrator, a user can choose which system he is going to use, or logout to the Login Page.

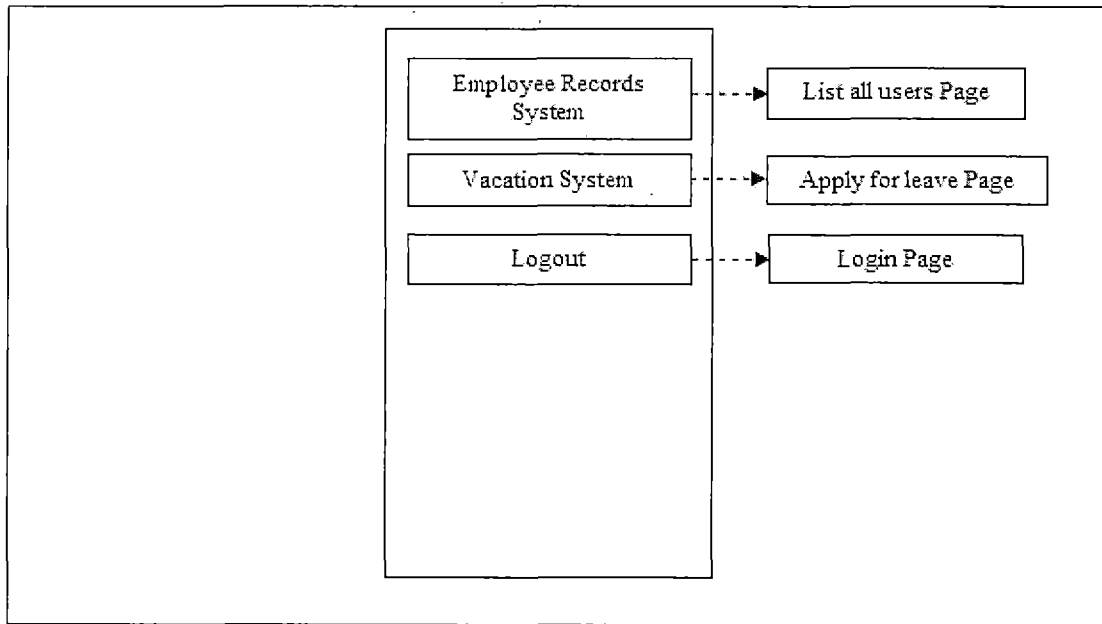


Figure 65. System Home Page (System Administrator)

Figure 66 represents the Create New User Page which is used by a system administrator, a system administrator can create new user through this page, or go to other pages through other links.

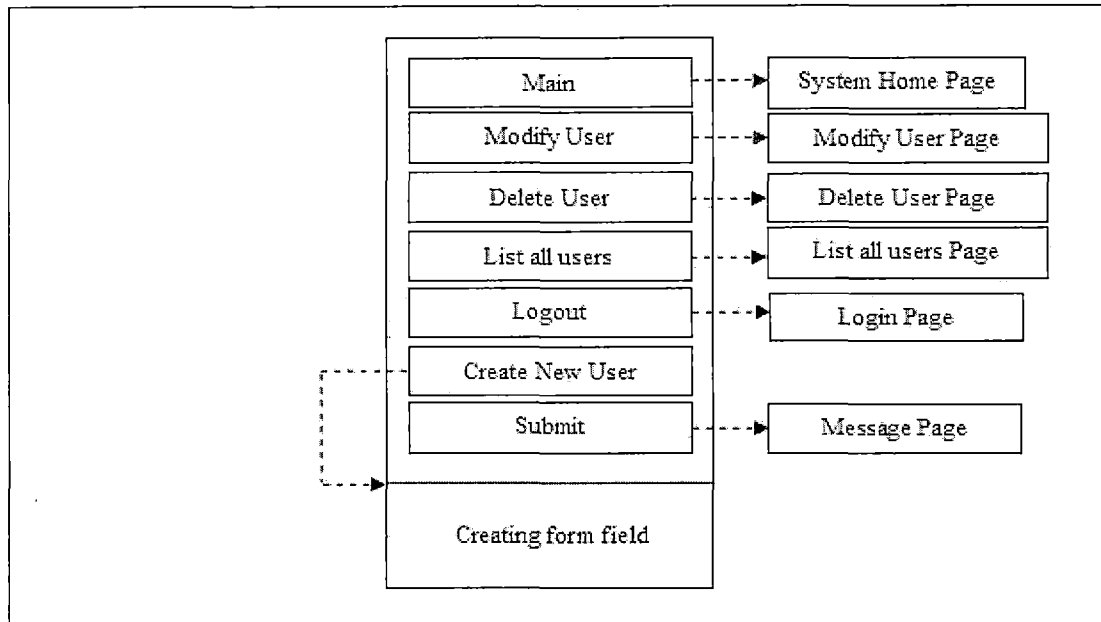


Figure 66. Create New User Page (System Administrator)

Figure 67 represents the Delete User Page which is used by a system administrator, a system administrator can delete user through this page, or go to other pages through other links.

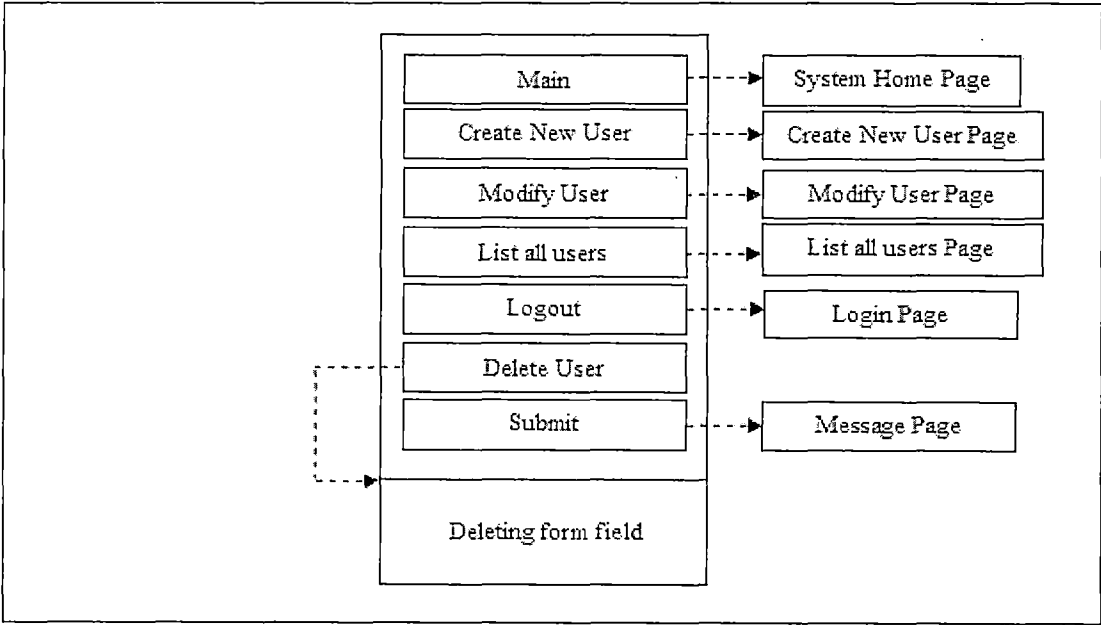


Figure 67. Delete User Page (System Administrator)

Figure 68 represents the Modify User Page which is used by a system administrator, a system administrator can choose to modify every user's profile through this page, or go to other pages through other links.

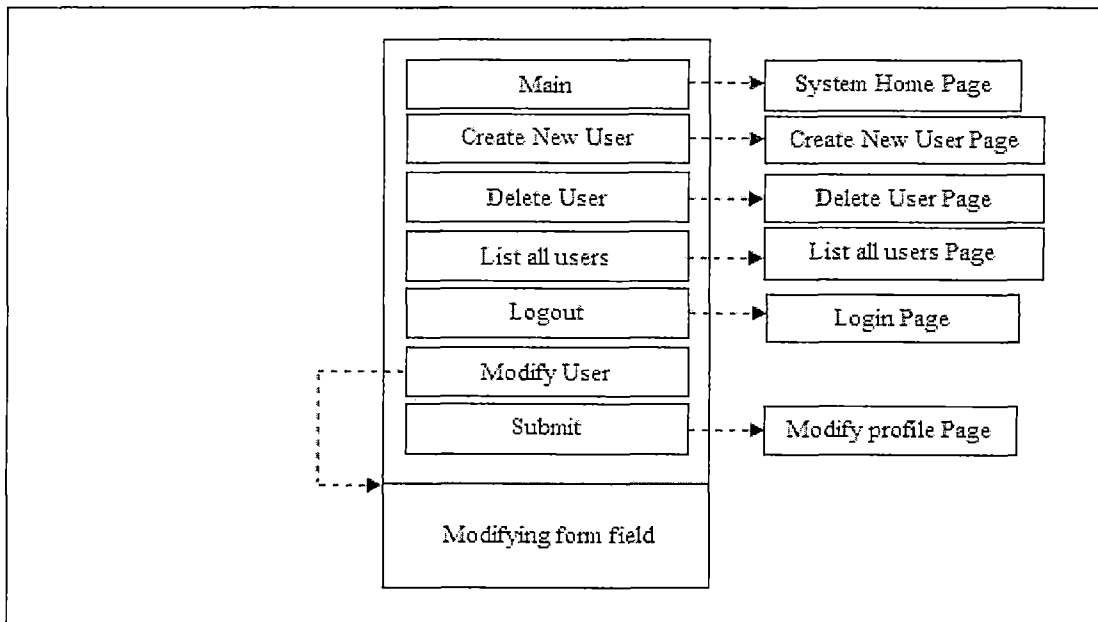


Figure 68. Modify User Page (System Administrator)

Figure 69 represents the List all users Page which is used by a system administrator, a system administrator can see all users through this page, or go to other pages through other links.

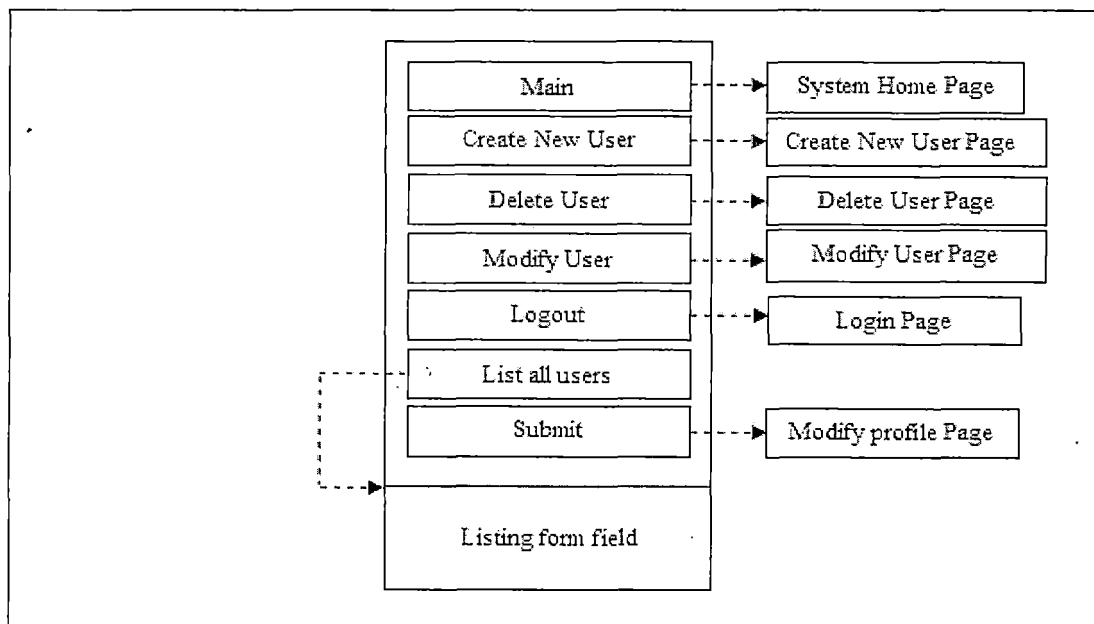


Figure 69. List All Users Page (System Administrator)

Figure 70 represents the Message Page which is used by a system administrator, a user can read the system message through this page, or go to other pages through other links.

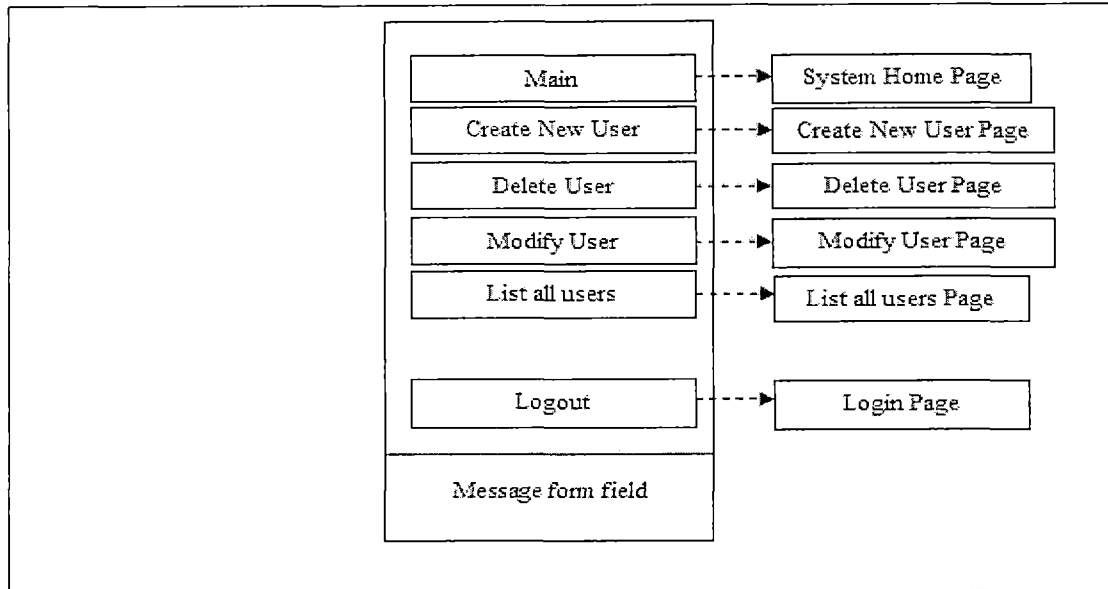


Figure 70. Message Page (System Administrator)

Figure 71 represents the Modify profile Page which is used by a system administrator, a system administrator can modify every user's profile through this page, or go to other pages through other links.

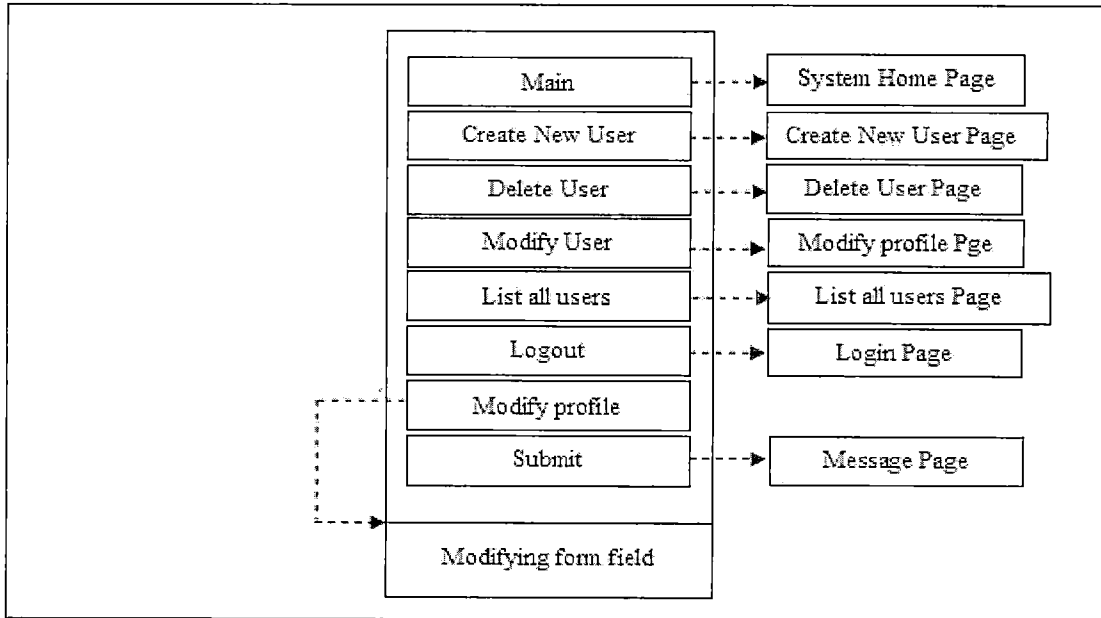


Figure 71. Modify Profile Page (System Administrator)

Figure 72 represents the Password Quote Page which is used by a system administrator, a user can quote his password through this page, or go back to the Login Page.

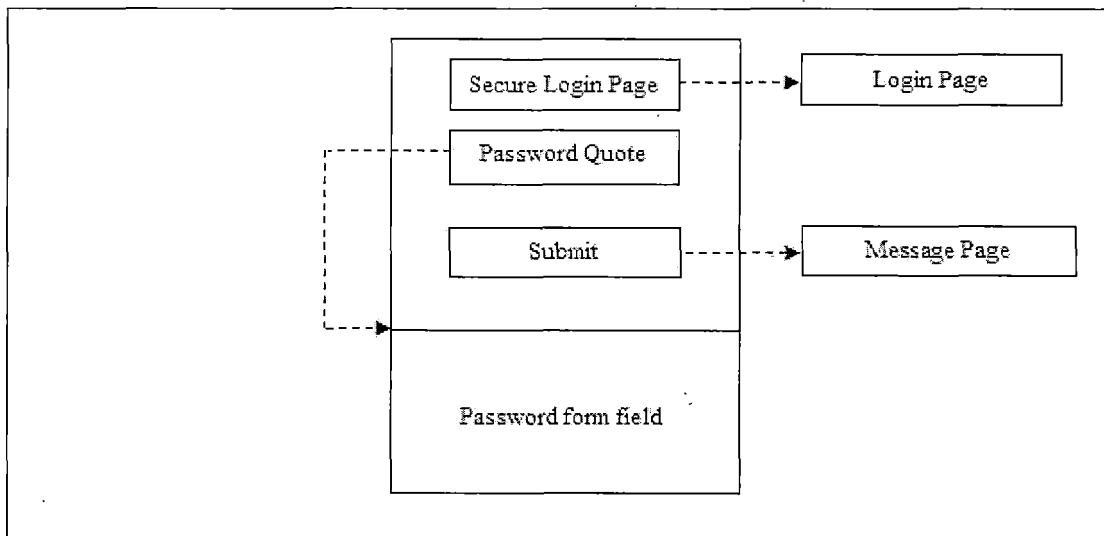


Figure 72. Password Quote Page (System Administrator)

3.6 Sequence Diagram

A Sequence diagram describes the sequence of actions that occur in a system. The methods used in each object, and the order in which the methods are used are described in a Sequence diagram. This makes the Sequence diagram a very useful tool to represent the dynamic behavior of a system.

A Sequence diagram is a two-dimensional diagram. On the horizontal axis, it shows the life of the object that it represents, while on the vertical axis, it shows the sequence of the creation or calls of these objects.

3.6.1 Vacation System

Figure 73 represents the process to apply for leave, while the user is trying to apply for leave, the system will call the function, named "CreateLDays," then the function will check if the remainder (the available days for leave) is larger or equal to the days which the user is trying to apply for leave. If it's true, the function transfer some parameters to the DBDAO (an interface of Web Service, it is used to access the database as a persistence service.), then check if the user has applied for leave on the designate date, if true, the user will get an error message; if false, the DBDAO will call insertLDay to insert the application, and call upDate_rem to update the remainder of the user.

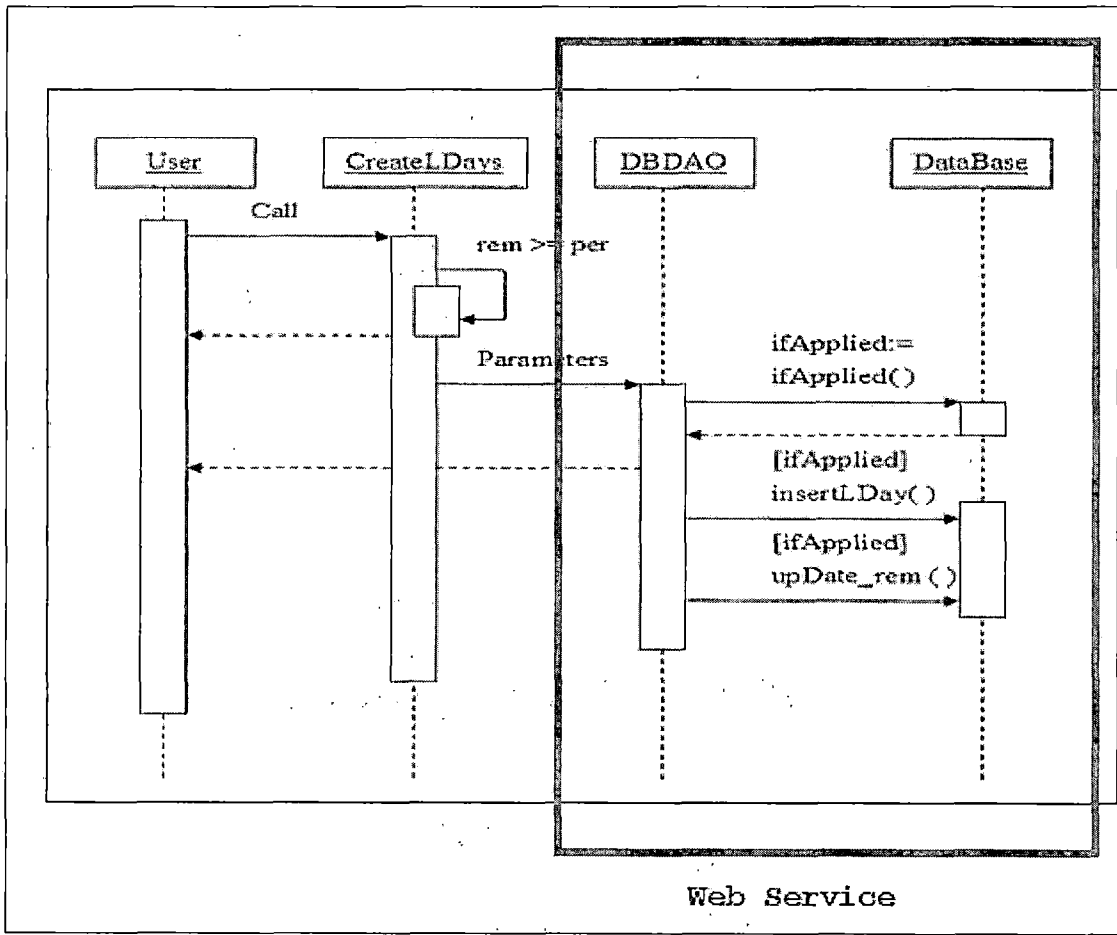


Figure 73. Apply for Leave

Figure 74 represents the process to check remainder, while the user is trying to check his remain days for leave, the system will call the function, named "CheckRemainder," then the function will transfer some parameters to the DBDAO, then check if these parameters are valid by "ifValiduser," if true, the DBDAO will call "findRemainder" to get the remain days; if false, the user will get an error message.

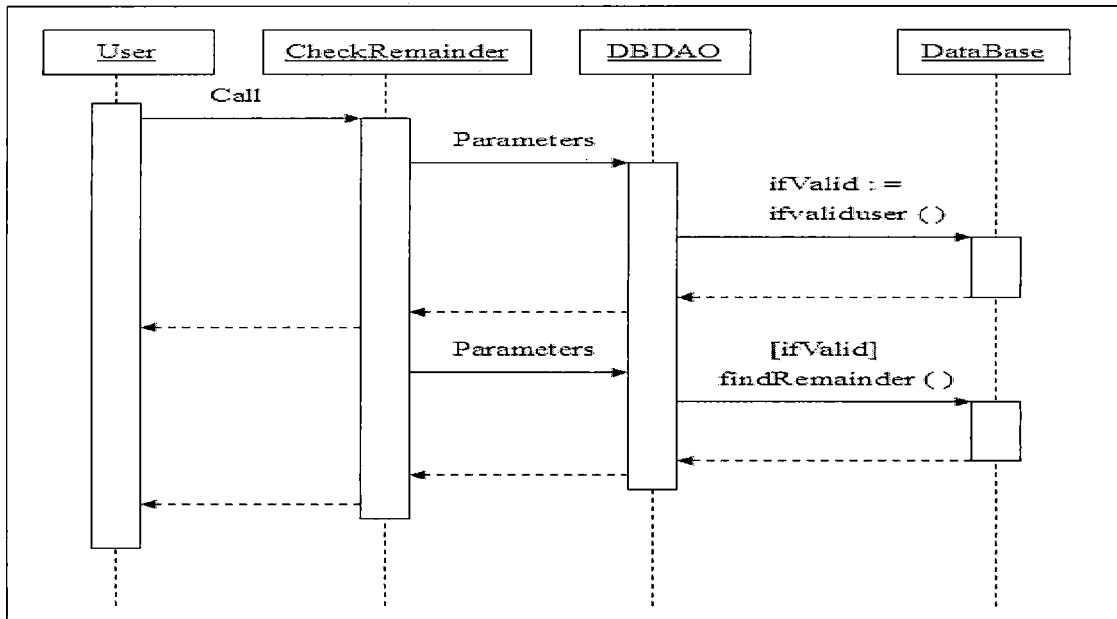


Figure 74. Check Remainder

Figure 75 represents the process to track/cancel, while the user is trying to track/cancel his applications, the system will call the function, named "Trackapp," then the function will transfer some parameters to the DBDAO, then check if these parameters are valid by "checkLdays," if true, the DBDAO will call "printLdays" to print out applications; if false, the user will get an error message.

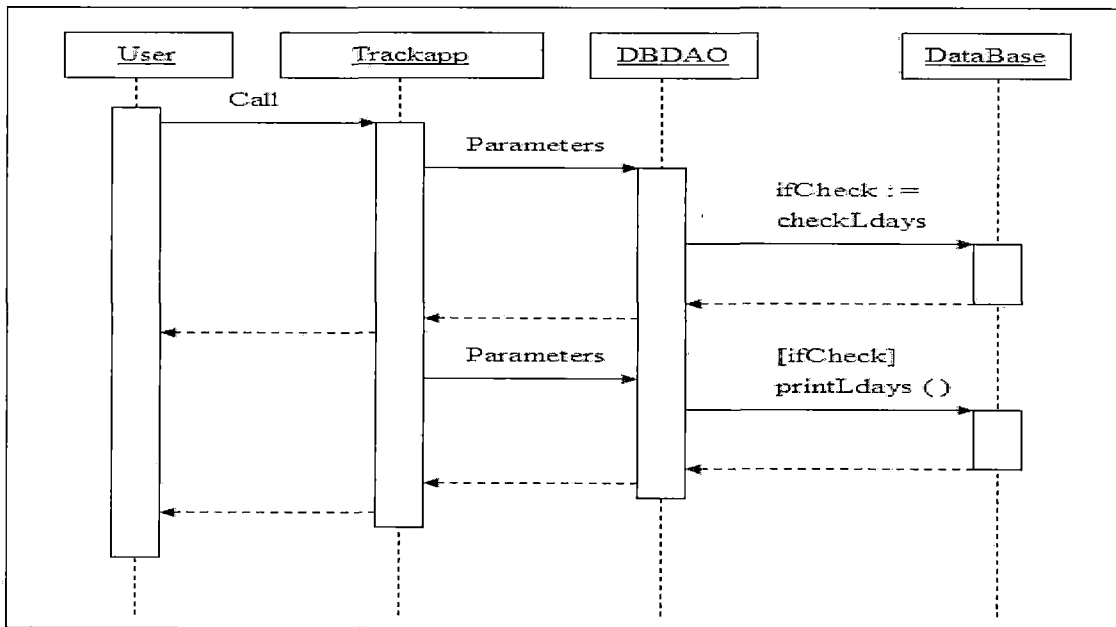


Figure 75. Track/Cancel

Figure 76 represents the process to approve, while the manager, senior manager or system administrator is trying to approve applications, the system will call the function, named "Approve," then the function will transfer some parameters to the DBDAO, then check if these parameters are valid by "findByid," if true, the DBDAO will call "updateLday" to update the application; if false, the user will get an error message.

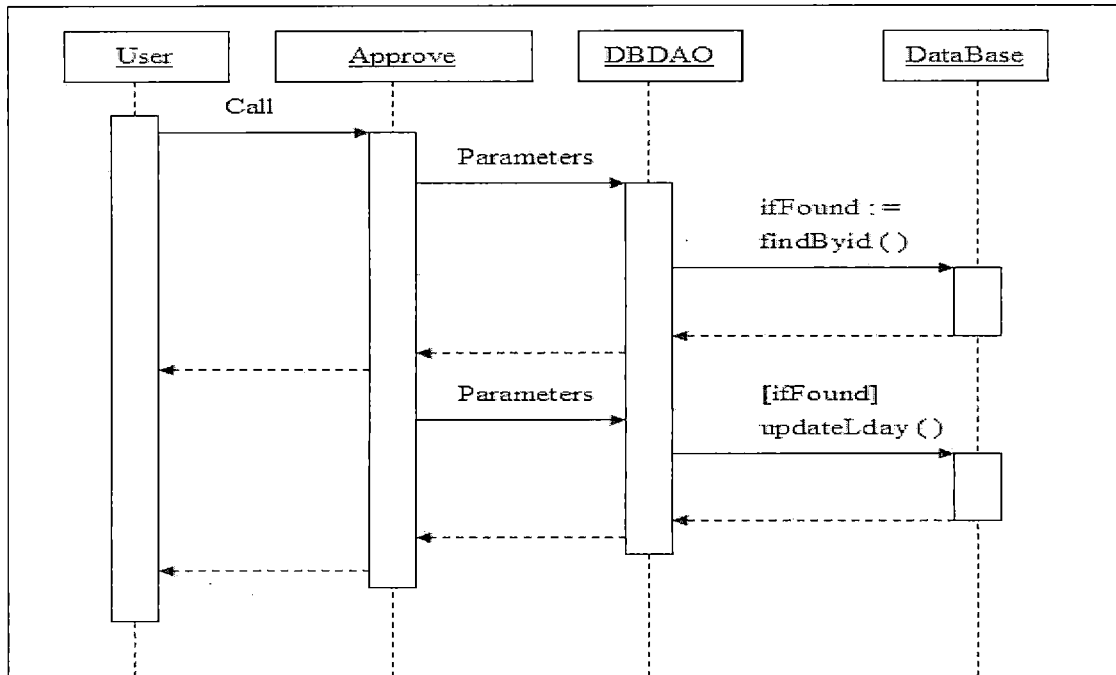


Figure 76. Approve

Figure 77 represents the process to check user state, while the user is trying to check his or other user's state, present or on vacation, the system will call the function, named "CheckAccountState," then the function will transfer some parameters to the DBDAO, the DBDAO will call "findState" to get the state of the user.

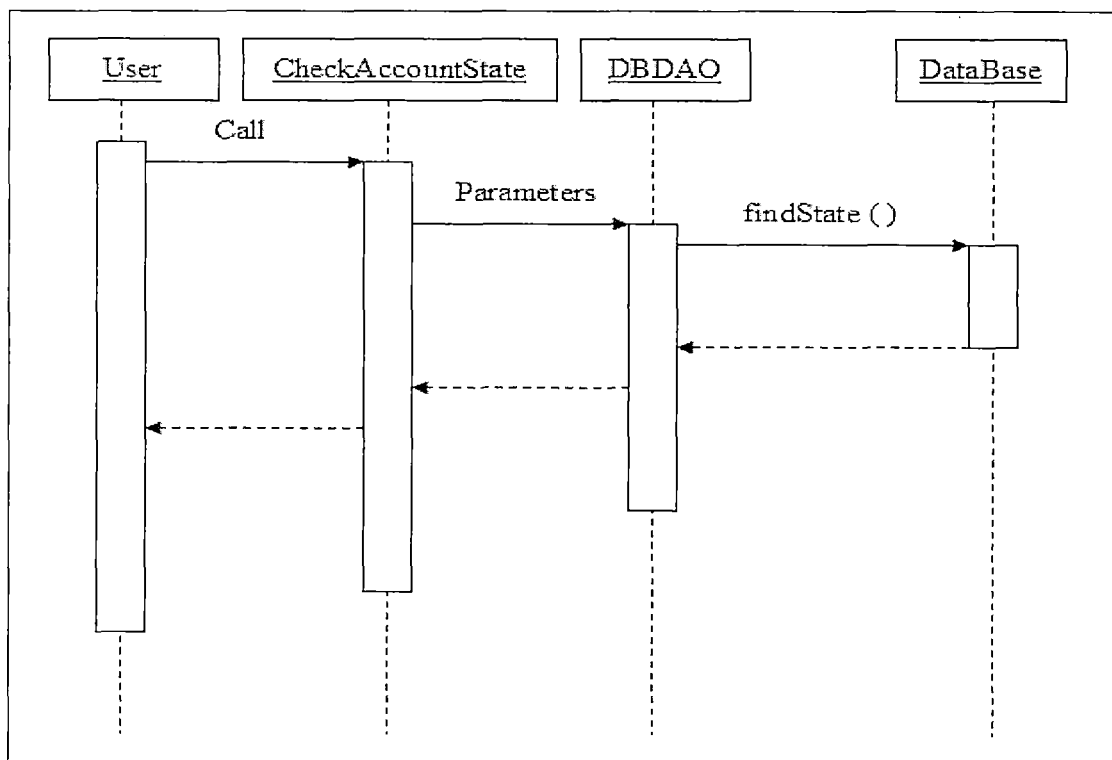


Figure 77. Check User State

3.6.2 Employee Records System

Figure 78 represents the process to create new user, while the system administrator is trying to create a new user, the system will call the function, named "CreateAccount," then the function will transfer some parameters to the DBDAO, the DBDAO will call "insertUser" to insert the user.

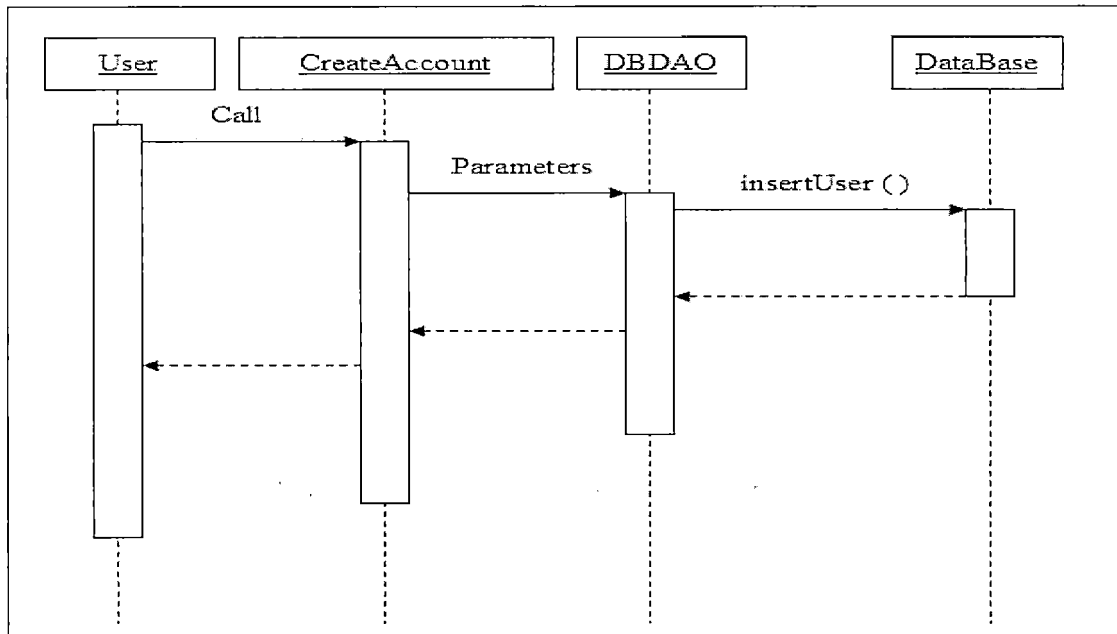


Figure 78. Create New User .

Figure 79 represents the process to delete user, while the system administrator is trying to delete an account, the system will call the function, named "DeleteAccount," then the function will transfer some parameters to the DBDAO, the DBDAO will call "deleteUser" to delete the user.

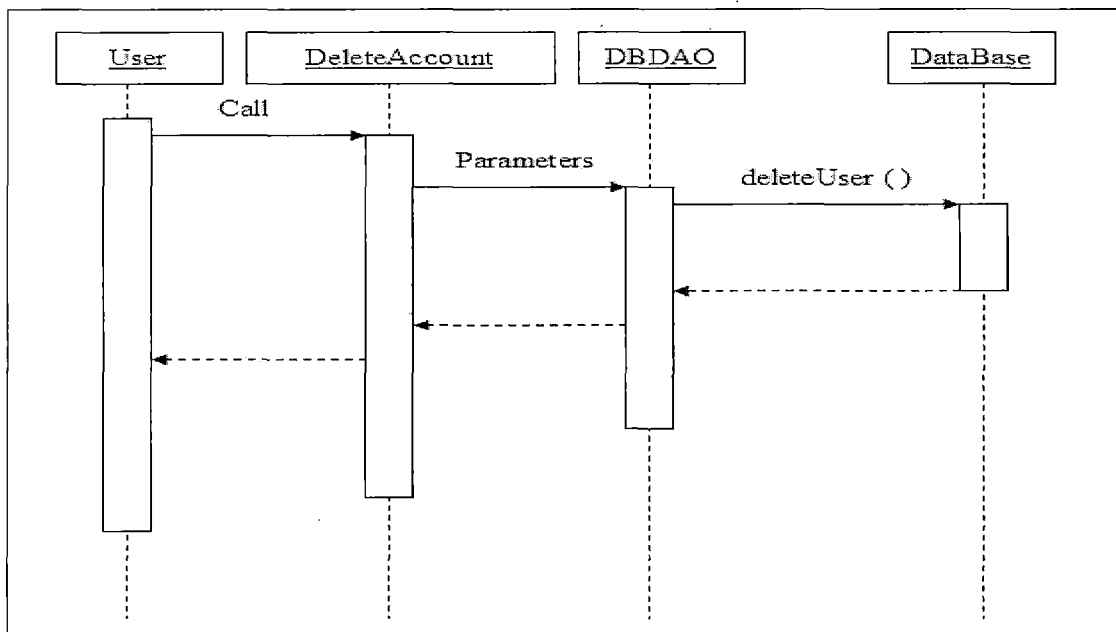


Figure 79. Delete User

Figure 80 represents the process to modify user, while the user is trying to modify the profile, the system will call the function, named "ModifyAccount," then the function will transfer some parameters to the DBDAO, the DBDAO will call "updateUser" to update the user's profile.

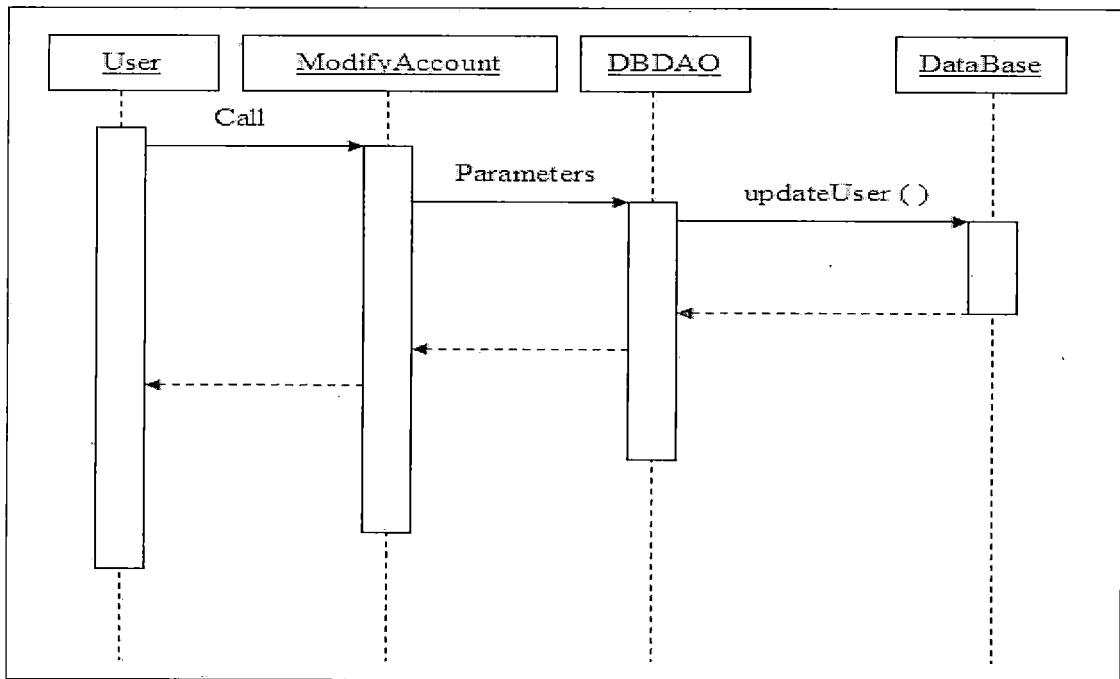


Figure 80. Modify User

Figure 81 represents the process to list users, while the system administrator is trying to list all users, the system will call the function, namedd "Listuser," then the function will transfer some parameters to the DBDAO, the system will call "listall" to list all users.

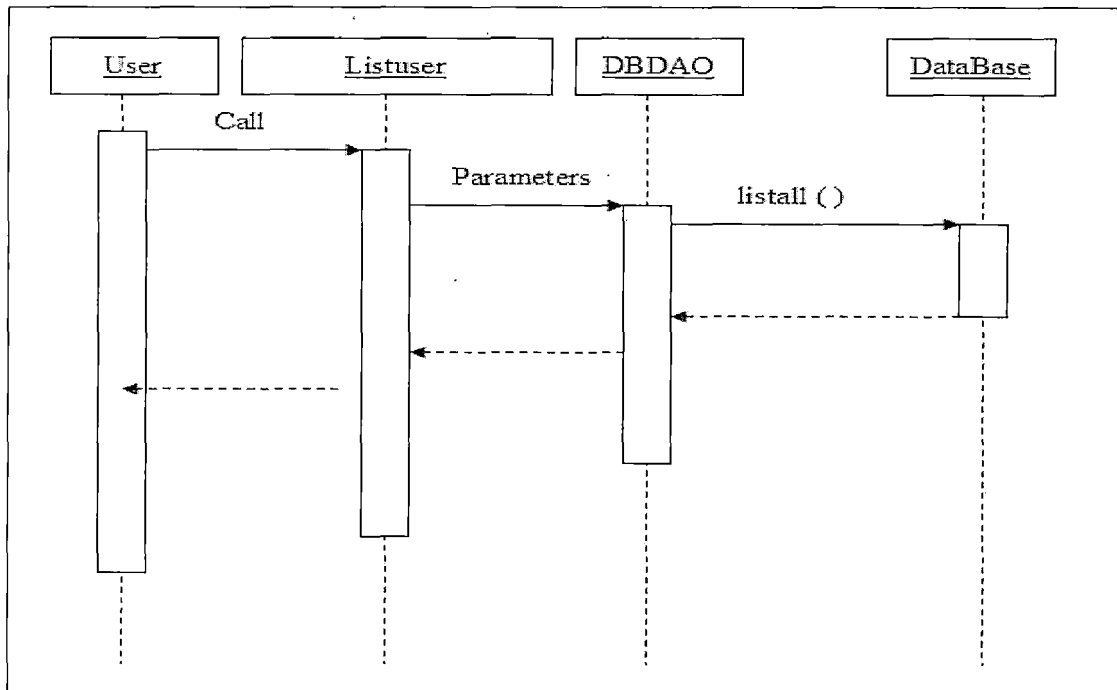


Figure 81. List Users

CHAPTER FOUR
SYSTEM IMPLEMENTATION

In this project, two web applications were built: Vacation System and Employee Records System. The applications will access the database through Web Service.

The code architecture is represented by figure 82. The servlet code in web applications calls DBDAO.servlet, then DBDAO.servlet will connect to Web Service, it connects to AccountDAO.servlet or LDaysDAO.servlet to finish the request.

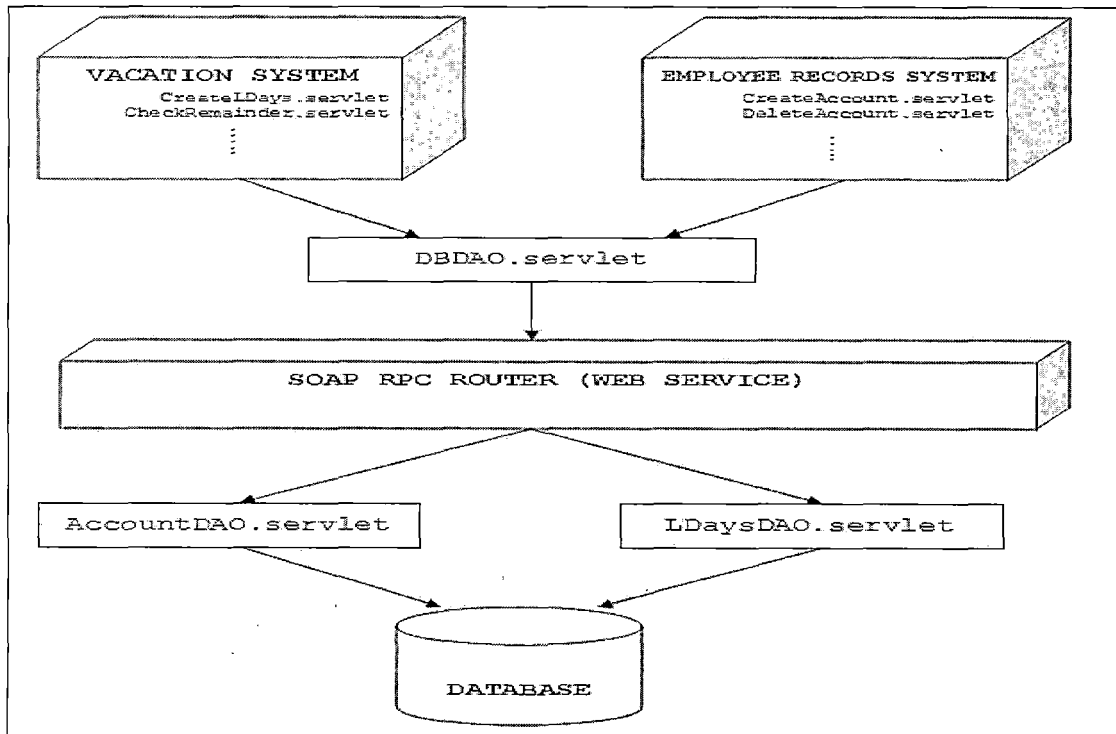


Figure 82. Code Architecture

Now I will use CreateAccount.servlet, DBDAO.servlet and AccountDAO.servlet as examples to explain how the servlet works (see Appendix A).

Appendix B has code snippets of DBDAO.servlet and AccountDAO.servlet for CreateAccount.servlet.

Following are the class diagrams for these two systems.

In Figure 83, DBDAO is the interface to access Web Service, while LDays is the name of the leave application.

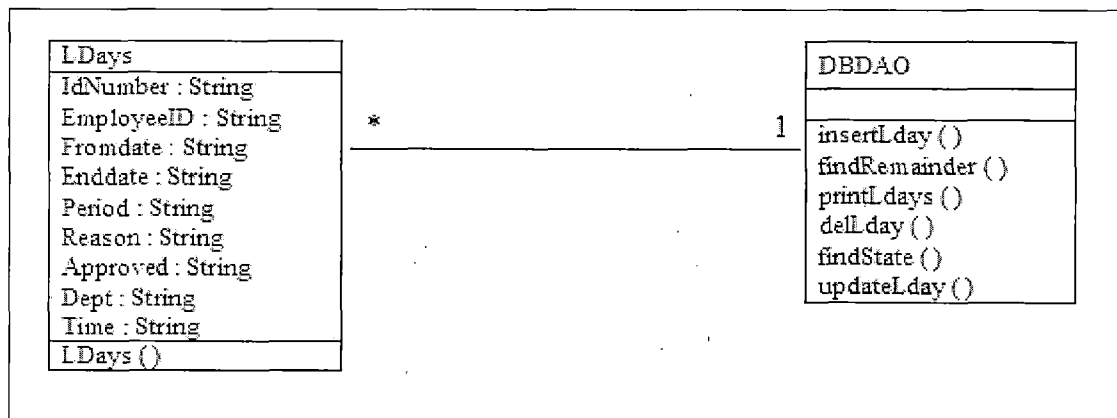


Figure 83. Class Diagram for Vacation System

In Figure 84, DBDAO is the interface to access Web Service, while Account is the name of the user.

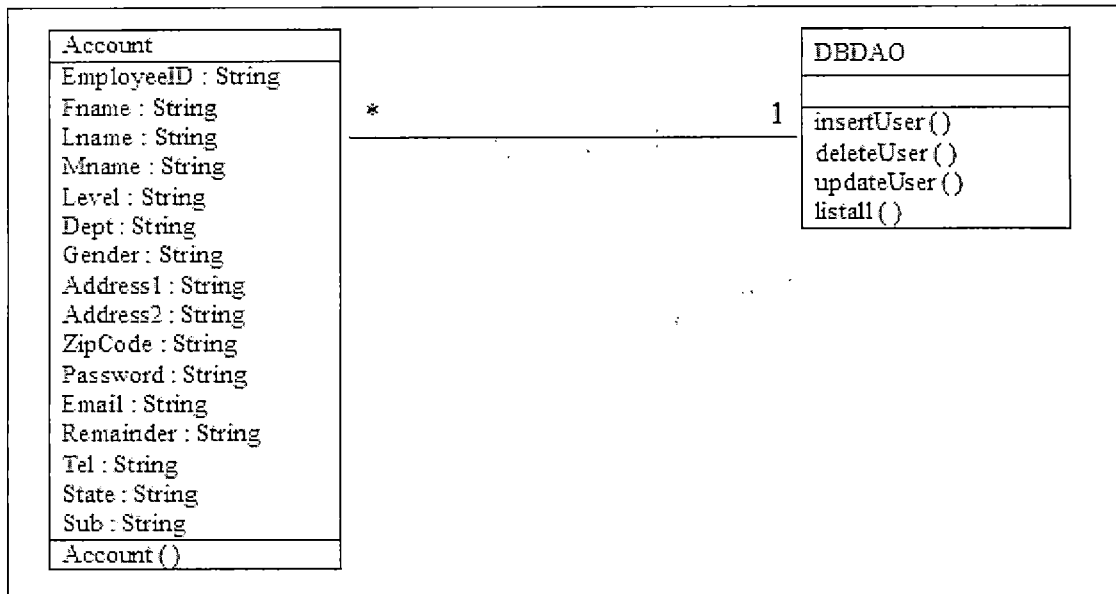


Figure 84. Class Diagram for Employee Records System

CHAPTER FIVE

PROJECT IMPLEMENTATION

5.1 User Interface Design

User interfaces are established on the web and therefore the system can use all user interface features provided by the web browser, including plug-ins and any added functionality that the browser may possess. Any standard web browser, such as Microsoft Internet Explorer or Netscape Navigator/Communicator can provide the user interface to the system. There are two types of user interfaces. 1) Static user interfaces: the interface will be static for all people browsing the site, regardless of the access rights of the person. They will be in the form of HTML forms or pages that have HTML Data and pictures on them. 2) Dynamic user interface: the interfaces will be generated dynamically on the server side using JSP and JavaServlets. These user interfaces will provide information tailored to the user who is logged in.

5.2 Graphical User Interface and Description

5.2.1 Secure Login Page

This page is the starting page for all people (General User, Manager, Senior Manager and System Administrator) who are going to use this software product.

All users must be created by System Administrator and assigned different level. All users can log in by providing a user id and a password on this page. The login servlet will verify the user id and password. If it is correct, it forwards to different page base on the "Level" of each user.

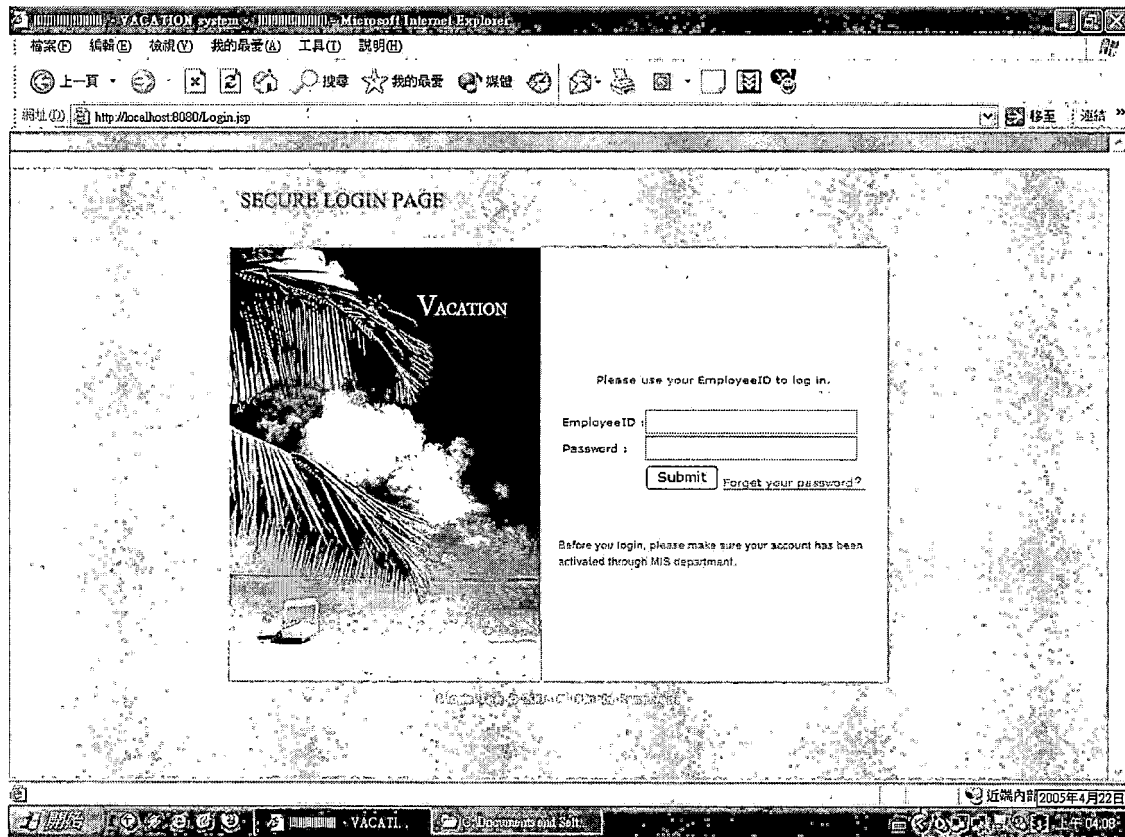


Figure 85. Secure Login Page

5.2.2 Apply Page

This page is essentially supposed to serve as an interface for users to apply vacation. Only the system administrator can apply other user's vacation.

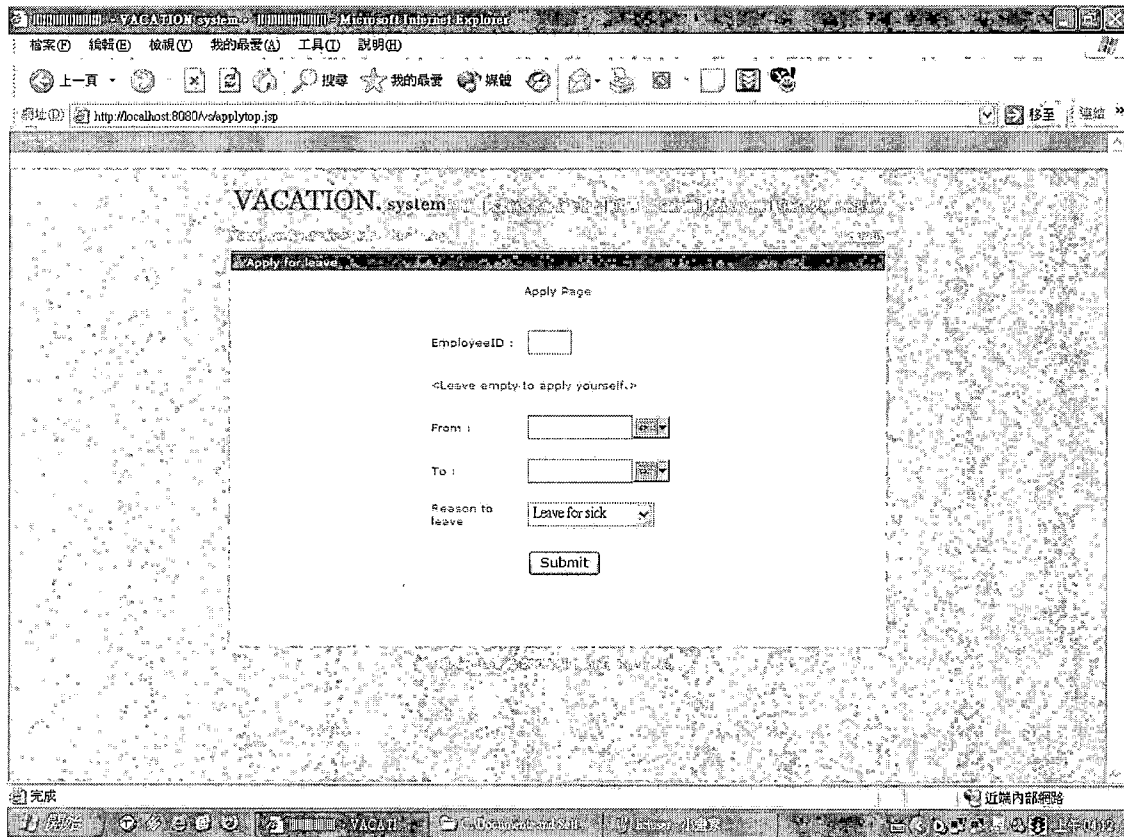


Figure 86. Apply Page

5.2.3 CheckRemainder Page

On this page, users can check how many days are available for leave. After the user clicks on the button, the server will check the remainder then display how many days the user has for leave.

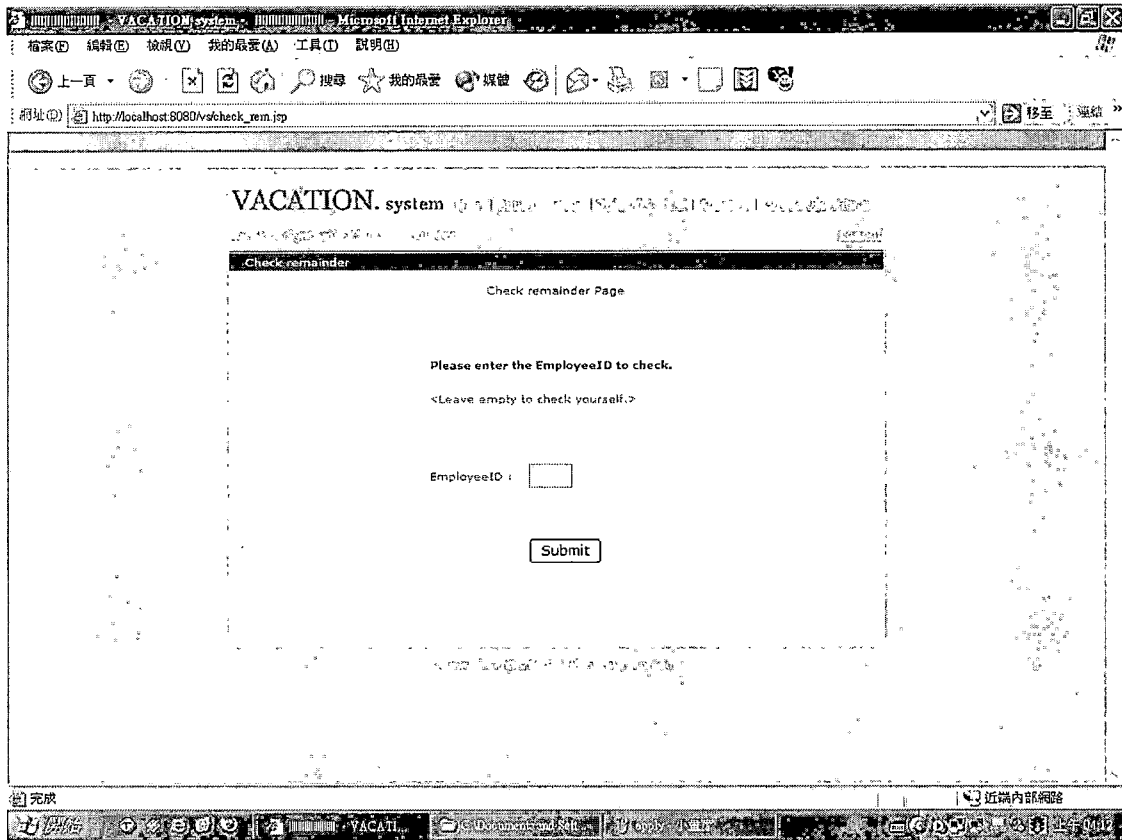


Figure 87. Check Remainder Page

5.2.4 Track/Cancel Page

This page shows the user's application. The page will list the details of the application. A user can delete the application if it has not been approved. Only the system administrator can delete approved applications.

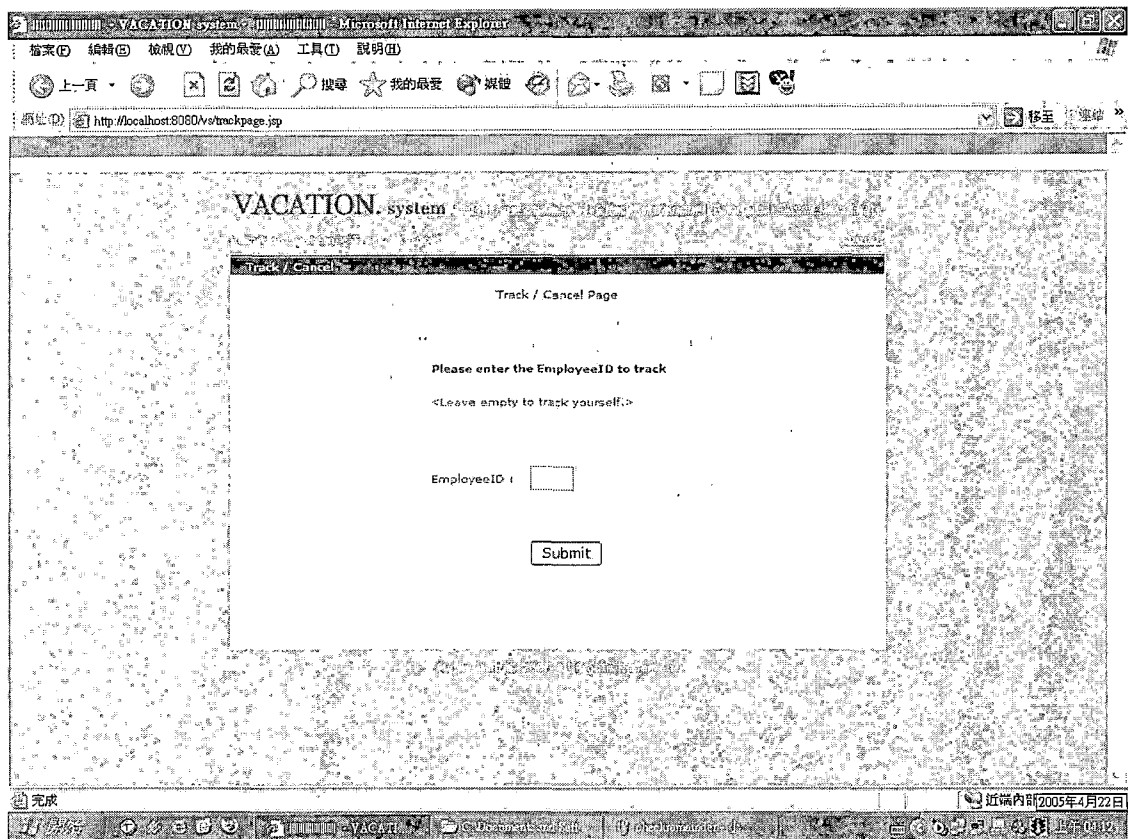


Figure 88. Track/Cancel Page

5.2.5 Approve Page

This page shows the applications which have not been approved by department manager. While the senior manager will only see the application which is over 3 days. The system administrator will see all the applications. User enters the IdNumber to approve the selected application.

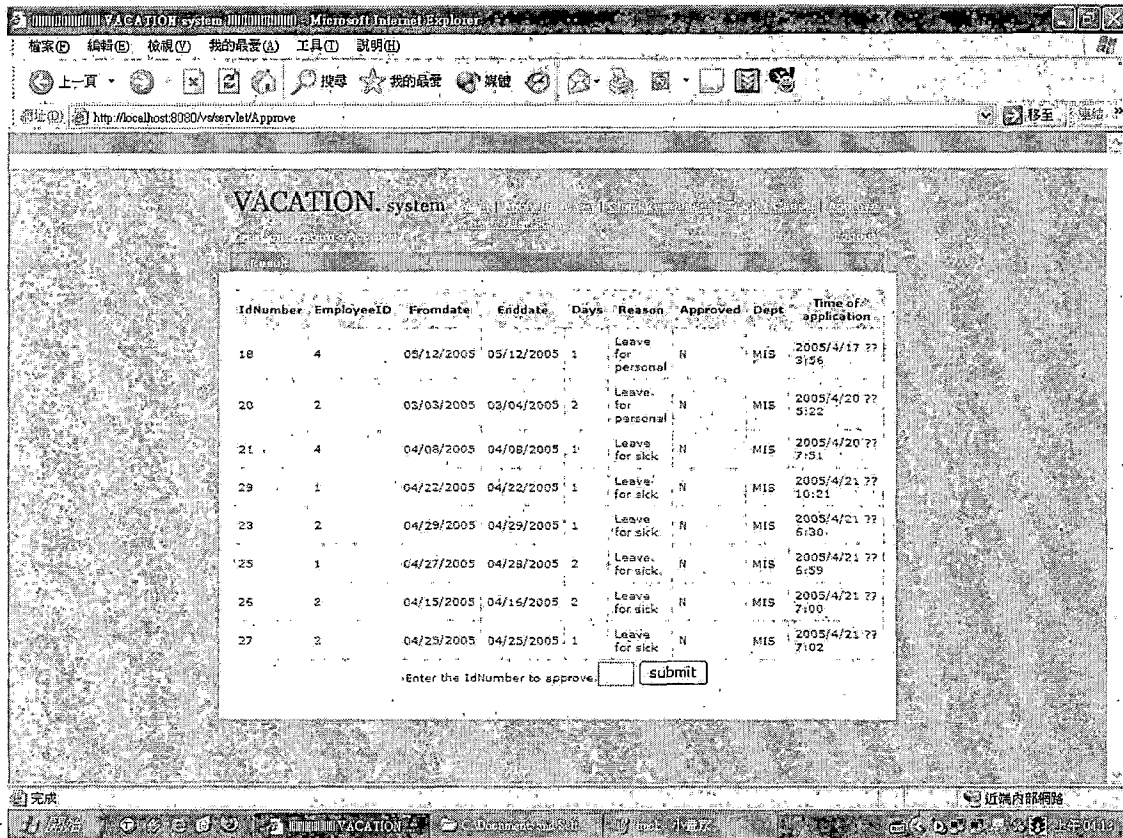


Figure 89. Approve Page

5.2.6 Create New User Page

The system administrator creates new user here. The administrator fills each column of the form, then clicks submit to create a new user.

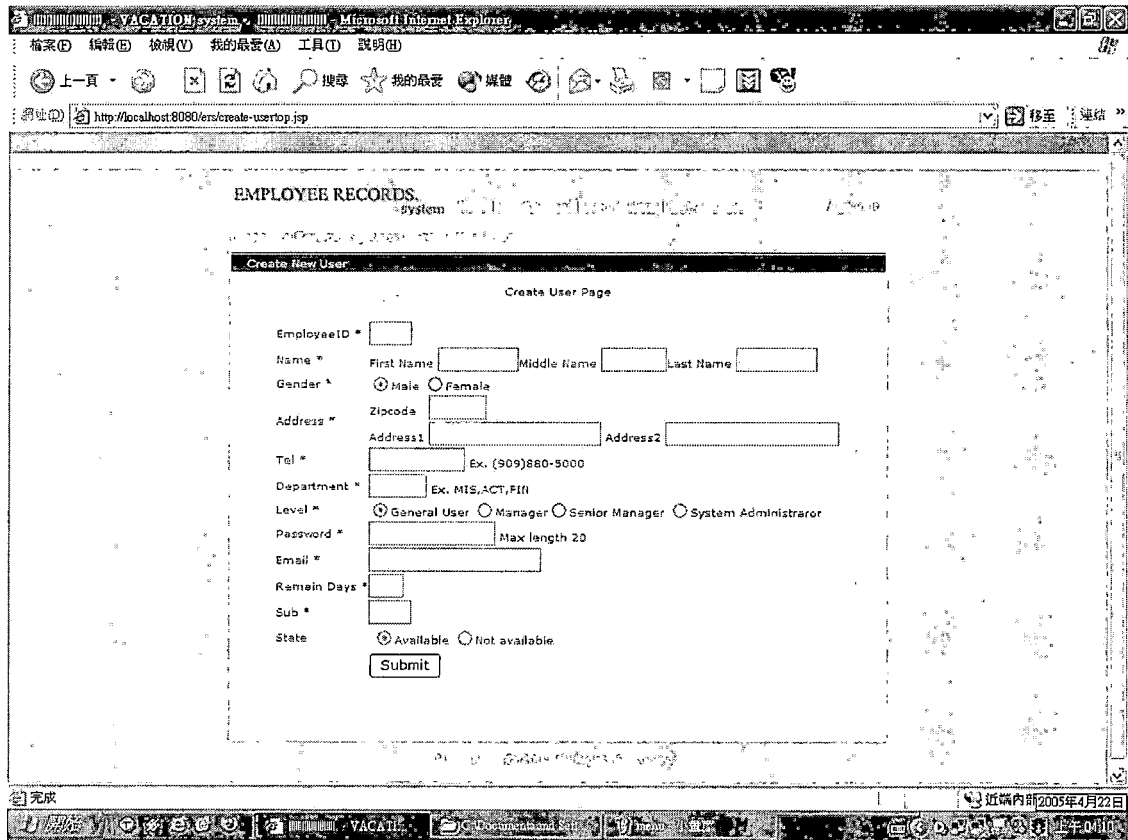


Figure 90. Create New User Page

5.2.7 Delete User Page

The administrator can enter the EmployeeID to delete the user. If the EmployeeID is not correct, the server will generate an error message.

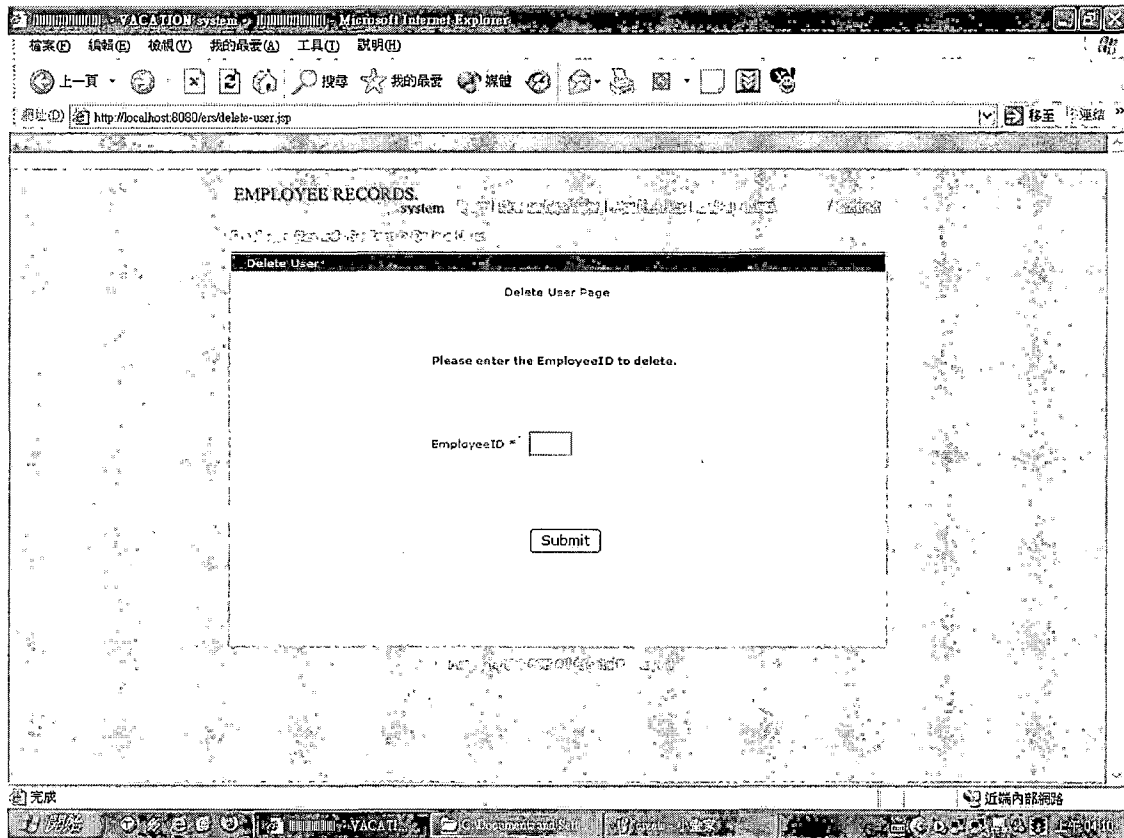


Figure 91. Delete User Page

5.2.8 Modify User Page

User can modify one's profile at this page. The administrator can modify each user's profile through this page.

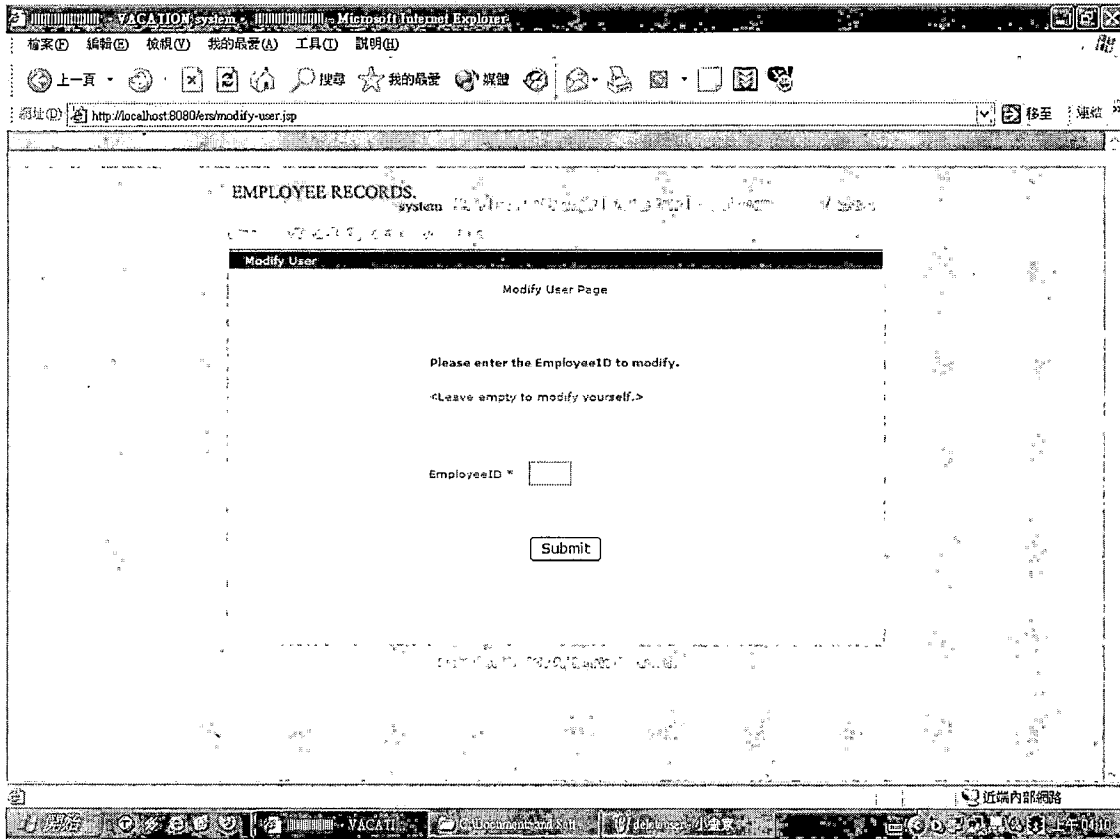


Figure 92. Modify User Page

5.2.9 Check User State Page

On this page, users can check the availability of leave. If the user's substitute has applied for leave on the designated date, the server will generate an error message.

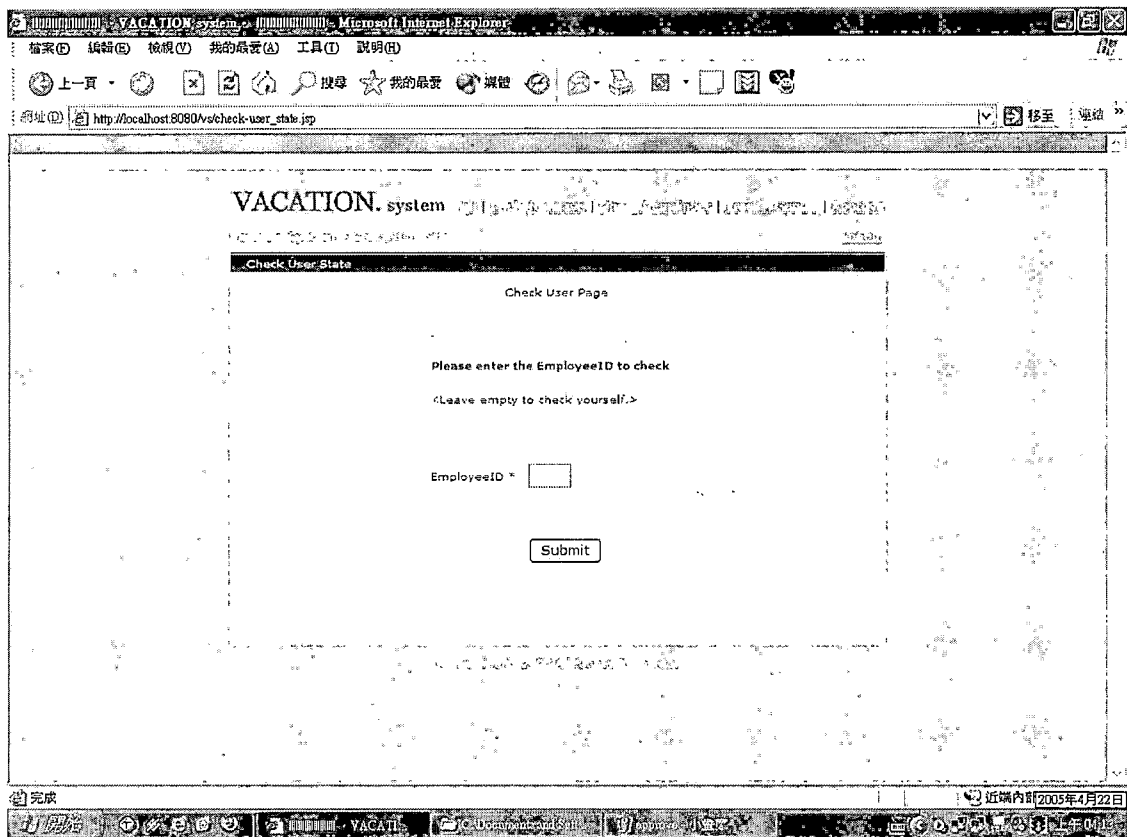


Figure 93. Check User Page

5.2.10 Forget Password Page

On this page, users provide their user ID and email to ask their password. If the user ID and the email are correct, the server will send an email with password to the user.

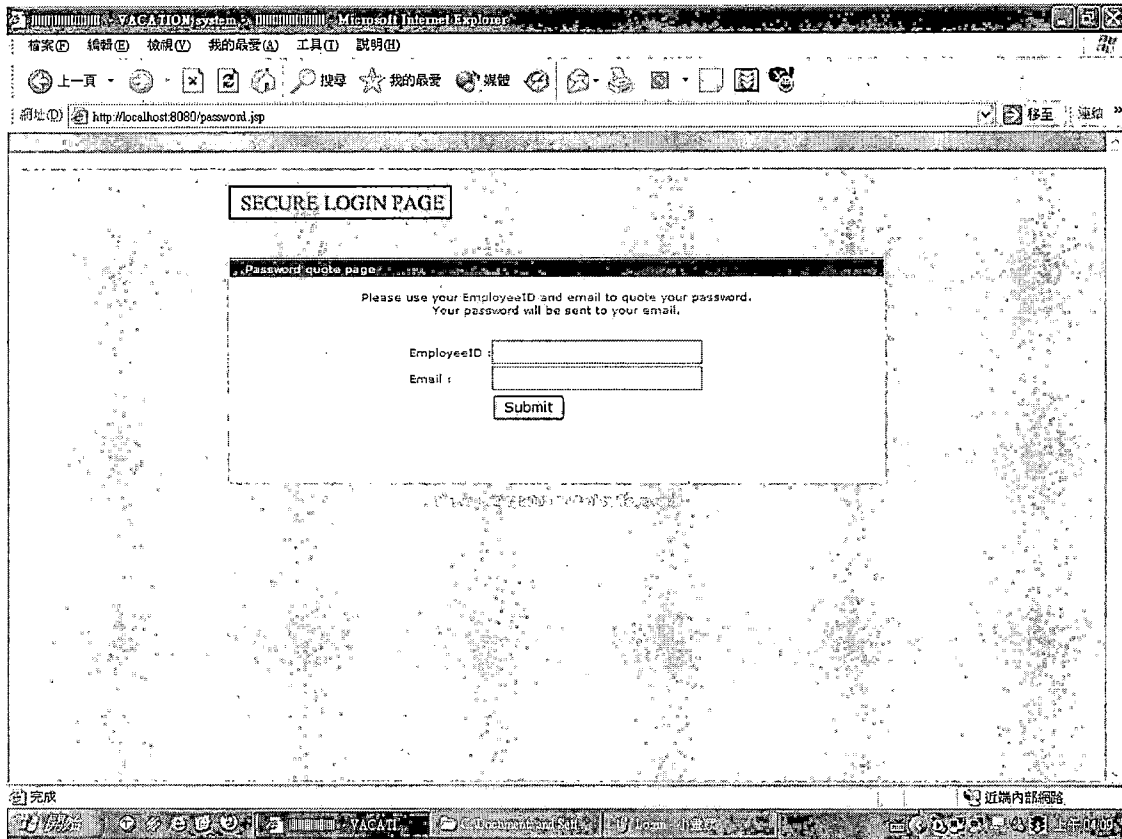


Figure 94. Password Page

5.2.11 List Department Users Page

On this page, the department manager can view all the users profile in the department. The user can enter the employeeID to view the detail profile.

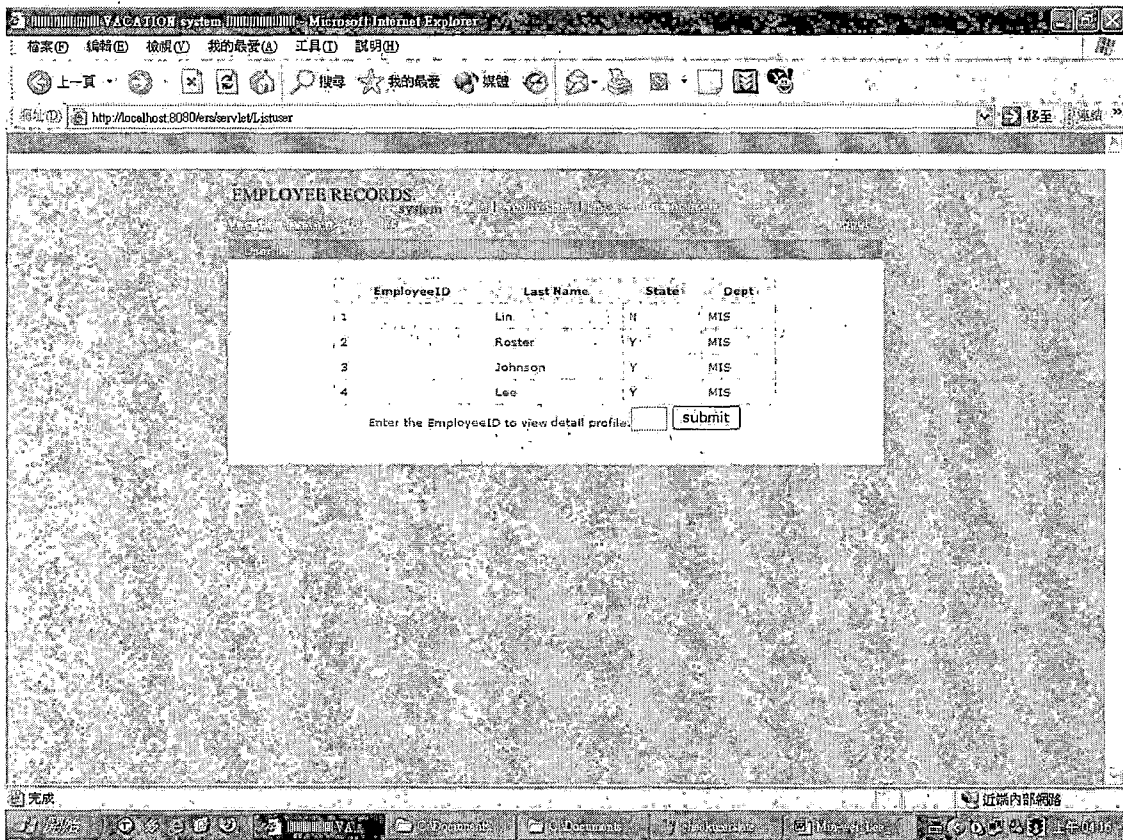


Figure 95. List Department Users Page

5.2.12 List All Users Page

On this page, the system administrator can view all users in this system. The user can enter the employeeID to view the detail profile.

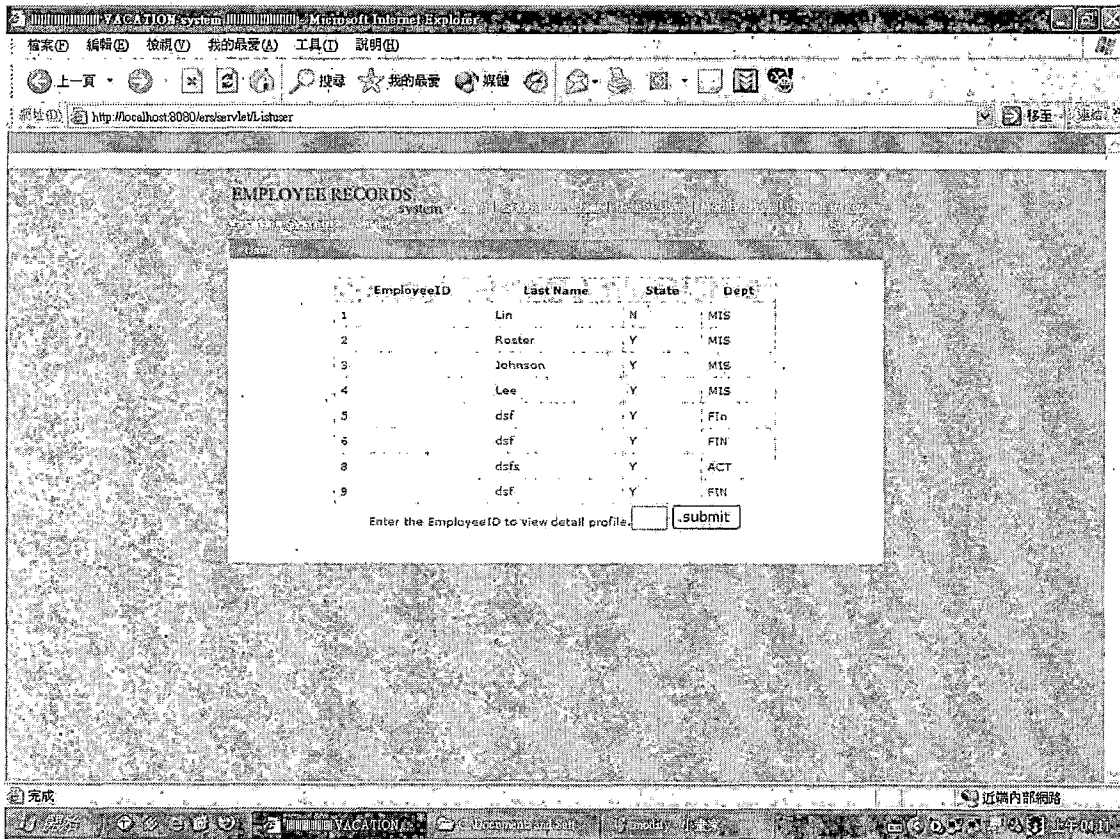


Figure 96. List All Users Page

CHAPTER SIX

SYSTEM VALIDATION

To validate a system, just having an understanding of computers and software systems is not adequate. It is essential to fully understand the process and the equipment that is being validated. Validation testing is a concern that overlaps with integration testing. Ensuring that the application fulfils its specification is a major criterion for the construction of an integration test. Validation testing also overlaps to a large extent with system testing, where the application is tested with respect to its typical working environment.

6.1 Unit Test

The Unit test presents the basic level of testing on all the individual components. The individual components include the object, the class, and the program in the system. The following table shows the results of the unit test for the Vacation System.

Table 3. The Unit Test Results

Unit Test	Tests Performed	Results
Login page	<ul style="list-style-type: none"> • Check the correctness of the displayed data in JSP included pages components. • Verify the login function working properly and get the error message properly. • Test the SSL function working as expected. 	OK
Menu Page	<ul style="list-style-type: none"> • Check the correctness of the displayed data. • Check the buttons, and links working properly. 	OK
Apply page	<ul style="list-style-type: none"> • Check the correctness of the displayed data. • Check the entire selection list, buttons, and links working properly. • Check JavaScript function working properly. • Test the correctness of the validation for all input data. • Verify the page can get the error message and work properly by the message. 	OK
Check Remainder Page	<ul style="list-style-type: none"> • Check the correctness of the displayed data. • Verify all links and buttons working as expected. • Test the correctness of the validation for all input data. • Verify the page can get the error message and work properly by the message. 	OK
Track / Cancel Page	<ul style="list-style-type: none"> • Check the correctness of the displayed data. • Verify all links and buttons working as expected. • Test the correctness of the validation for all input data. • Verify the page can get the error message and work properly by the message. 	OK
Approve Page	<ul style="list-style-type: none"> • Check the correctness of the displayed data. • Verify all links and buttons working as expected. • Test the correctness of the validation for all input data. • Verify the page can get the error message and work properly by the message. 	OK

Unit Test	Tests Performed	Results
Create New User Page	<ul style="list-style-type: none"> • Check the entire buttons and links working properly. • Check the entire selection list, buttons, and links working properly. • Check JavaScript function working properly. • Test the correctness of the validation for all input data. • Verify the page can get the error message and working properly by the message. 	OK
Delete User Page	<ul style="list-style-type: none"> • Check the entire buttons and links working properly. • Verify all links and buttons working as expected. • Test the correctness of the validation for all input data. • Verify the page can get the error message and working properly by the message. 	OK
Modify User page	<ul style="list-style-type: none"> • Check the entire buttons and links working properly. • Verify all links and buttons working as expected. • Test the correctness of the validation for all input data. • Verify the page can get the error message and working properly by the message. 	OK
Check User State Page	<ul style="list-style-type: none"> • Verify the page get the correct user information. • Verify all links and buttons working as expected. • Test the correctness of the validation for all input data • Verify the page can get the error message and working properly by the message. 	OK
List department users Page	<ul style="list-style-type: none"> • Verify the page shows all deparment users in the screen. • Check the entire buttons and links working properly. • Verify all links and buttons working as expected. • Verify the form is generated correctly by the servlet. • Verify the page can get error page when the user enters invalid EmployeeID. 	OK
List all users Page	<ul style="list-style-type: none"> • Verify the page shows all users in the screen. • Check the entire buttons and links working properly. • Verify all links and buttons working as expected. 	OK

Unit Test	Tests Performed	Results
	<ul style="list-style-type: none"> • Verify the form is generated correctly by the servlet. • Verify the page can get error page when the user enters invalid EmployeeID. 	

6.2 Subsystem Testing

Subsystem testing is the next step up in the testing process where all related units from a subsystem does a certain task. Thus, the subsystem test process is useful for detecting interface errors and specific functions.

Table 4 show subsystem test results in detail.

Table 4. Subsystem Test Results

Subsystem	Tests Performed	Results
Apply vacation Subsystem	<ul style="list-style-type: none"> • Make sure the employee has enough days for leave. • Make sure the period is counted correctly. • Make sure the employee's state is OK for leave. • Make sure the subsystem deducts the days from available ones. 	OK
Track / Cancel Subsystem	<ul style="list-style-type: none"> • Make sure the subsystem list all the applications. • Make sure all the columns are printed correctly. • Test if it can get the error message. • Make sure after canceling the application, the available days will restore to correct one. • Make sure user can not cancel the application which has been approved except for system administrator. 	OK
Approve Subsystem	<ul style="list-style-type: none"> • Make sure each column is listed correctly. • Make sure the user will receive the confirmation letter after approved. 	OK
Create User Subsystem	<ul style="list-style-type: none"> • Check if the subsystem can detect the error of creating of the user that exists in the subsystem. • Make sure the subsystem can automatically assign the substitute. 	OK
Modify User Subsystem	<ul style="list-style-type: none"> • Check if the user can update his personal account properly. • Verify if the subsystem shows the page properties is the users are the owner or the administrator. 	OK
Delete User Subsystem	<ul style="list-style-type: none"> • Verify the subsystem can delete a user account properly. • Verify when the user trys to access the servlet through doGet, the servlet will redirect to the Login page. 	
Security	<ul style="list-style-type: none"> • Verify the servlet can get the password to access Web Service from web.xml. • Verify the page will be redirected to Login page when the user is not authenticated. 	OK

6.3 System Test Plan

System test plan is a test process that uses real data, which the system is intended to manipulate, to test the system. First of all the subsystem will be integrated into one system. Then test the system by using a variety of data to see the overall results.

The steps for the system test plan are showed in the following table:

Table 5. System Test Results

System Test	Results
1. Install the Vacation System, Employee Records System, SOAP into server.	OK
2. Start up all services such as JSP engine, MySQL database engine.	OK
3. Running testing by using real data on all forms and reports.	OK

CHAPTER SEVEN

MAINTENANCE MANUAL

The Maintenance Manual provides maintenance personnel with the information necessary to maintain the system effectively. The manual provides the definition of the software support environment, the roles and responsibilities of maintenance personnel, and the regular activities essential to the support and maintenance of program modules, job streams, and database structures. In Vacation System, there are 3 major issues: Software Installation, Variable Installation, and Vacation System Installation.

7.1 Software Installation

In this system, it requires Gentoo, MySQL, JSDK, TOMCAT, and JDBC to run the programs. Following will detail the installation of those five software systems.

7.1.1 Gentoo Installation

Gentoo is a very good distribution for Linux. Following are the steps to install Gentoo Linux onto your machine.

1. Download the latest version of the Gentoo Universal LiveCD operating systems from <http://gentoo.osuosl.org/releases/x86/2004.3/liv>

ecd/ and burn the iso files into CDs.

2. Install the operating system by inserting CD into the CD-ROM and make sure the bios is set to boot from CD.
3. The machine will startup via CD-ROM and start to install Gentoo.
4. Check the date and time, and correct if needed. Set the root password. Configure networking. Start sshd. Exit shell running on local console.
5. Setup the environment and install the components, then Gentoo Linux is installed.

7.1.2 Install Java 2 Platform, Standard Edition

To install J2SDK in your machine under the Gentoo Linux System, you only have to type the command:

```
emerge jdk
```

Then the system will automatically finish the installation.

7.1.3 Install Tomcat

Following are the steps to install Tomcat onto your machine.

1. Run emerge tomcat.
2. Run rc-update add tomcat5 default.
3. Run chmod 755 /opt/tomcat5

4. Restart Tomcat and type `http://localhost:8080` into the browser to test that your system works OK.

7.1.4 MySQL Installation

The database system in Vacation System is MySQL. To install MySQL, follow the following steps:

1. Run `emerge mysql`.
2. Initialize the data directory and create the MySQL grant tables.
`mysql_install_db --user=mysql`
3. Add to default run-level, and start the server.
`rc-update add mysql default`
`/etc/init.d/mysql start`
4. Remove accounts without passwords.
`mysql -u root -p`
`mysql> use mysql;`
`mysql> delete from user where password = ``;`
`mysql> flush privileges;`

7.1.5 Remote Procedure Calls Router Installation

The Web Service in this project is constructed by RPC Router. To install RPC Router, follow the following steps:

1. Download the file, `soap.war` from
`http://www.apache.org/dyn/closer.cgi/ws/soap/`
2. Place `soap.war` in the `$TOMCAT_HOME/webapps`
3. Restart the Tomcat.

7.2 System Variables

In this system, we don't have to change any environment variables in the Linux system and server.xml in Tomcat server configuration directory.

The default configuration for the the MySQL database is sufficient.

7.3 Vacation System Installation/Migration

1. All the JSP programs and HTML programs are stored in
 - /opt/tomcat5/webapps/ROOT
 - /opt/tomcat5/webapps/vs
 - /opt/tomcat5/webapps/ers
2. All the *.java are stored in
 - /opt/tomcat5/webapps/ROOT/WEB-INF/classes
3. All the classes are stored in
 - /opt/tomcat5/webapps/ROOT/WEB-INF/classes
 - /opt/tomcat5/webapps/vs/WEB-INF/classes
 - /opt/tomcat5/webapps/ers/WEB-INF/classes
4. Place the web.xml for Vacation System in
 - /opt/tomcat5/webapps/vs

Place the web.xml for Employee Records System in

 - /opt/tomcat5/webapps/ers

7.4 Backup and Restore

Protecting system information is one of the system administrator's most important tasks. Backups allow the administrator to restore a file system to the condition it was in at the time of the last backup. Backups must be done carefully and on a strict schedule. The backup system and backup media must also be tested regularly to verify that they are working correctly. There are two steps to back up Vacation System (VS). One is to backup the system files. The other step is to backup the database which is used by VS.

7.4.1 System Backup

All the system files are stored in the directory `"/opt/tomcat5/webapps/ROOT," "/opt/tomcat5/webapps/vs," "/opt/tomcat5/webapps/ers," "/opt/tomcat5/webapps/soap"` and the subdirectory of its. Thus, in order to backup the system files, we can compress this directory by using the compress program `"tar"` to backup the system files:

```
tar -cf root.tar /opt/tomcat5/webapps/ROOT
tar -cf vs.tar /opt/tomcat5/webapps/vs
tar -cf ers.tar /opt/tomcat5/webapps/ers
tar -cf soap.tar /opt/tomcat5/webapps/soap
```

7.4.2 Database Backup

To backup the database system, we use `mysqldump` command to backup all the database used by the system. The following command is used to backup the database:

1. Create a subdirectory named `backup` under the directory `/var/lib/mysql/vacation`

2. Using command:

```
mysqldump --tab=/var/lib/mysql/vacation/backup -  
-opt vacation
```

After executing the backup command above, we will have 4

files: `ldays.sql`, `ldays.txt`, `users.sql`, `users.txt`.

7.4.3 System Restore

To restore the system file, simply extract the backup file by using the following command:

```
tar -xzvf root.tar /  
tar -xzvf vs.tar /  
tar -xzvf ers.tar /  
tar -xzvf soap.tar /
```

7.4.4 Database Restore

To restore the database needed for the system, go to the directory `/var/lib/mysql/vacation/backup`, and execute the following commands:

1. `create database vacation;`

2. restore table users from
/var/lib/mysql/vacation/backup
3. restore table ldays from
/var/lib/mysql/vacation/backup

CHAPTER EIGHT

CONCLUSION AND FUTURE DIRECTIONS

8.1 Conclusion

All the functions are described as follows:

- 1) Members can apply for leave in this system.
- 2) The user can apply for leave online without creating paper work.
- 3) The company will not need paper work for leave applications, which is more efficient.
- 4) Users get confirmation letter through the e-mail submitted by server.
- 5) Members, in turn, can log on and check the history of their leave.

The objective of this project is to explore how the Web Service can be used in a company. By using Web Services, the company can save many costs on integrating all systems and producing new systems. In this project, I use the Web Service as a persistence service to access the database. There are two systems in this project, I can save the time and the cost by not to build the part to access the database twice. If each system can be developed as a Web Service, while the programmers are trying to develop a new system, the programmers only have to integrate the finished systems, they don't need to develop

from zero. You can imagine, the more modules were built, the more you can save to build a new system.

The Vacation System makes a company more efficient and reduces the paper work. Taking advantage of this system, for example, a company can significantly save cost of paper, offer a more efficient service of leave and users can easily review the history of their applications.

8.2 Future Directions

The goals for the future are to build more friendly graphical interfaces, to integrate other systems to reduce paper work inside the company, and to improve the performance of the systems.

More importantly, I will try to read more documents about the Web Services, and explore the benefits and more detail information to apply Web Service in a company.

APPENDIX A

CREATEACCOUNT.SERLET, DBDAO.SERVLET AND ACCOUNTDAO.SERVLET

AS EXAMPLES TO EXPLAIN HOW THE SERVLET WORKS

//CreateAccount.servlet

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
import java.util.*;
import java.net.URL;
import org.apache.soap.SOAPException;

public class CreateAccount extends HttpServlet {
    public void doGet( HttpServletRequest request,
                      HttpServletResponse response)
        throws ServletException, IOException
    {
        try{
            doPost(request, response);
        }
        finally {}
    }

    public void doPost( HttpServletRequest request,
                       HttpServletResponse response)
        throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        HttpSession session = request.getSession();
        //Get session from cookie

        ServletContext context = session.getServletContext();
        String chpd = context.getInitParameter("erspassword");
        String ipaddr = context.getInitParameter("ipaddr");
        //Get password and ipaddress from xml file

        URL url = new URL("http://"+ipaddr+":8080/soap/servlet/rpcrouter");
        String id = (String)session.getAttribute("id");
        String user = (String)session.getAttribute("user");
        String Login = "https://"+ipaddr+":8443/Login.jsp";
        //Get information from form field
        String EmployeeID = request.getParameter("EmployeeID");
        String Fname = request.getParameter("Fname");
        String Lname = request.getParameter("Lname");
        String Mname = request.getParameter("Mname");
        String Level = request.getParameter("Level");
        String Dept = request.getParameter("Dept");
        String Gender = request.getParameter("Gender");
        String Address1 = request.getParameter("Address1");
        String Address2 = request.getParameter("Address2");
        String ZipCode = request.getParameter("ZipCode");
        String Password = request.getParameter("Password");
        String Email = request.getParameter("Email");
        String Remainder = request.getParameter("Remainder");
        String Tel = request.getParameter("Tel");
        String State = request.getParameter("State");
        String Sub = request.getParameter("Sub");

        //Save form information in the session
        session.setAttribute("surl", url);
        session.setAttribute("sEmployeeID", EmployeeID);
        session.setAttribute("sFname", Fname);
        session.setAttribute("sLname", Lname);
        session.setAttribute("sMname", Mname);
        session.setAttribute("sLevel", Level);
        session.setAttribute("sDept", Dept);
    }
}
```

```

session.setAttribute("sGender", Gender);
session.setAttribute("sAddress1", Address1);
session.setAttribute("sAddress2", Address2);
session.setAttribute("sZipCode", ZipCode);
session.setAttribute("sPassword", Password);
session.setAttribute("sEmail", Email);
session.setAttribute("sRemainder", Remainder);
session.setAttribute("sTel", Tel);
session.setAttribute("sState", State);
session.setAttribute("sSub", Sub);

```

```

String[] array = new String[3];
String mes = null;

```

```

try {
    String mgr = DBDAO.findMgr(url, Dept, chpd);

    // if id=null means this servlet is not accessed from legal procedure.
    if(id == null)
    {
        response.sendRedirect("http://localhost:8080/Login.jsp");
    }
    //end if

    // To check if this EmployeeID from form field has been occupied.
    else if((DBDAO.ifvaliduser(url, EmployeeID, chpd)).equals("true"))
    {
        mes = "EmployeeID does already exist.";
        session.setAttribute("mes",mes);
        response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers.jsp");
    }
    //end else if

    // To check if this user is system administrator
    else if(user.equals("system_administrator"))
    {
        //If this new user is a system administrator
        if(Level.equals("4"))
        {

            // To insert this new user.
            DBDAO.insertUser(url, EmployeeID, Fname, Lname, Mname, Level, Dept, Gender, Address1, Address2,
                ZipCode, Password, Email, Remainder, Tel, State, Sub, chpd);

            session.setAttribute("mes","Account "+EmployeeID+" created.");

            // Redirect to the message page.
            response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers2.jsp");

            array = AccountDAO.info();

        }
        //end if

        // If this new user is a senior manager
        else if(Level.equals("3"))
        {

            //Check make sure there is only one senior manager in one department.
            if(DBDAO.noMgr(url, Dept, Level, chpd).equals("true"))
            {
                DBDAO.insertUser(url, EmployeeID, Fname, Lname, Mname, Level, Dept, Gender, Address1, Address2,
                    ZipCode, Password, Email, Remainder, Tel, State, Sub, chpd);
                session.setAttribute("mes","Account "+EmployeeID+" created.");
            }
        }
    }
}

```

```
response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers2.jsp");

array = AccountDAO.info();

}
else
{
mes = "There does already exist one senior manager in this department.";
session.setAttribute("mes", mes);
response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers.jsp");

}

} //end else if

//If the new user is a manager
else if (Level.equals("2"))
{

//To make sure there is only one manager in one department.
if (DBDAO.noMgr(url, Dept, Level, chpd).equals("true"))
{
DBDAO.insertUser(url, EmployeeID, Fname, Lname, Mname, Level, Dept, Gender, Address1, Address2,
ZipCode, Password, Email, Remainder, Tel, State, Sub, chpd);
session.setAttribute("mes", "Account " + EmployeeID + " created.");
response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers2.jsp");

array = AccountDAO.info();

}
else
{
mes = "There does already exist one manager in this department.";
session.setAttribute("mes", mes);
response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers.jsp");

}

} //end else if

// The new user is a general user.
else
{

DBDAO.insertUser(url, EmployeeID, Fname, Lname, Mname, Level, Dept, Gender, Address1, Address2,
ZipCode, Password, Email, Remainder, Tel, State, Sub, chpd);
session.setAttribute("mes", "Account " + EmployeeID + " created.");
response.sendRedirect("http://" + ipaddr + ":8080/ers/message_ers2.jsp");

array = AccountDAO.info();

} //end else
session.setAttribute("eid", EmployeeID);

} //end else if

else
{
response.sendRedirect("https://" + ipaddr + ":8443/Login.jsp");
}
```

```
        } //end else

    } //end of try
    catch (SOAPException e)
    {}
    catch (RuntimeException e)
    {
        request.setAttribute("message", array[0]+array[1]+array[2]);
    } //end of catch

} //end of doPost
} //end of class
```

APPENDIX B

CODE SNIPPETS OF DBDAO.SERVLET AND ACCOUNTDAO.SERVLET
FOR CREATEACCOUNT.SERVLET

// DBDAO.servlet

```
import java.net.URL;
import java.util.Iterator;
import java.util.Vector;
import java.util.Hashtable;
import org.apache.soap.Constants;
import org.apache.soap.Fault;
import org.apache.soap.SOAPException;
import org.apache.soap.encoding.SOAPMappingRegistry;
import org.apache.soap.encoding.soapenc.BeanSerializer;
import org.apache.soap.rpc.Call;
import org.apache.soap.rpc.Parameter;
import org.apache.soap.rpc.Response;
import org.apache.soap.util.xml.QName;

public class DBDAO {

    public static void insertUser(URL url, String EmployeeID, String FName, String Lname, String Mname, String Level,
        String Dept, String Gender, String Address1, String Address2, String ZipCode, String Password, String
        Email, String Remainder, String Tel, String State, String Sub, String password)
        throws SOAPException {

        SOAPMappingRegistry registry = new SOAPMappingRegistry();
        BeanSerializer serializer = new BeanSerializer();

        // Build the Call object
        Call call = new Call();
        call.setSOAPMappingRegistry(registry);
        call.setTargetObjectURI("urn:db-service");
        call.setMethodName("DBinsertUser");
        call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC);

        // Set up parameters
        Vector params = new Vector();
        params.addElement(new Parameter("EmployeeID", String.class, EmployeeID, null));
        params.addElement(new Parameter("Fname", String.class, FName, null));
        params.addElement(new Parameter("Lname", String.class, Lname, null));
        params.addElement(new Parameter("Mname", String.class, Mname, null));
        params.addElement(new Parameter("Level", String.class, Level, null));
        params.addElement(new Parameter("Dept", String.class, Dept, null));
        params.addElement(new Parameter("Gender", String.class, Gender, null));
        params.addElement(new Parameter("Address1", String.class, Address1, null));
        params.addElement(new Parameter("Address2", String.class, Address2, null));
        params.addElement(new Parameter("ZipCode", String.class, ZipCode, null));
        params.addElement(new Parameter("Password", String.class, Password, null));
        params.addElement(new Parameter("Email", String.class, Email, null));
        params.addElement(new Parameter("Remainder", String.class, Remainder, null));
        params.addElement(new Parameter("Tel", String.class, Tel, null));
        params.addElement(new Parameter("State", String.class, State, null));
        params.addElement(new Parameter("Sub", String.class, Sub, null));
        params.addElement(new Parameter("password", String.class, password, null));
        call.setParams(params);

        // Invoke the call
        Response response;
        response = call.invoke(url, "");
    } //end of insertUser
} //end of class
```

//AccountDAO.servlet

```
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.Date;
import java.text.SimpleDateFormat;
import javax.mail.*;
import javax.mail.internet.*;
public class AccountDAO extends ParentDAO
{
    public static void insert(String EmployeeID, String FName, String LName, String MName, String Level, String Dept,
        String Gender, String Address1, String Address2, String ZipCode, String Password, String Email, String
        Remainder, String Tel, String State, String Sub)
    {
        //connect to database to make an insertion
        try {
            Connection connection =
                DriverManager.getConnection( dbUrl,
                                            dbUsername,
                                            dbPassword);

            String sql =
                "insert user (EmployeeID, FName, LName, MName, Level, Dept, Gender, Address1, Address2, ZipCode,
                    Password, Email, Remainder, Tel, State, Sub)" +
                " values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
            PreparedStatement ps = connection.prepareStatement(sql);

            // set parameters
            ps.setString(1, EmployeeID);
            ps.setString(2, FName);
            ps.setString(3, LName);
            ps.setString(4, MName);
            ps.setString(5, Level);
            ps.setString(6, Dept);
            ps.setString(7, Gender);
            ps.setString(8, Address1);
            ps.setString(9, Address2);
            ps.setString(10, ZipCode);
            ps.setString(11, Password);
            ps.setString(12, Email);
            ps.setString(13, Remainder);
            ps.setString(14, Tel);
            ps.setString(15, State);
            ps.setString(16, Sub);
            ps.executeUpdate();

        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }
}
//end of insert
//end of class
```

REFERENCES

- [1] "Beginning JSP Web Development." Jayson Falkner. August 2001.
- [2] IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1993).
- [3] "JavaScript: The Definitive Guide 4th edition." David Flanagan. O'Reilly & Associates, December 2001.
- [4] "JavaServer Pages." Larne Pekowsky. April 2000.
- [5] "*Java Servlet Programming Second Edition*." Jason Hunter and William Crawford. O'Reilly and Associates, 2002
- [6] "JavaSpaces Principles, Patterns, and Practice." Eric Freeman. November 1999.
- [7] SSL 3.0 Specification.
<http://wp.netscape.com/eng/ssl3/>, November 1996.
- [8] "The Java Programming Language Second Edition." Ken Arnold and James Gosling. February 2000.
- [9] "UML Distilled, A Brief Guide to Standard Object Modeling Language", Fowler & Scott, 1999.
- [10] "A Web Services Primer." Venu Vasudevan,
<http://webservices.xml.com/pub/a/ws/2001/04/04/webservices/index.html>, April 2001.
- [11] "What is Service-Oriented Architecture?" Hao He,
<http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html>, September 2003.
- [12] "JAVA and XML 2nd Edition" Brett McLaughlin. O'Reilly & Associates, January 2002.
- [13] "XML Web Services Basics" Roger Wolter. Microsoft Corporation, December 2001