

ONLINE BUS PASS SYSTEM

*A project work submitted to
Jawaharlal Nehru Technological University, Kakinada
in partial fulfillment of the requirements for the award of the degree of*

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

CH. RAJESWARARAO

19851A0503

SK. AHAMAD JANI

19851A0522

Under the Guidance of

Dr. B. SRIDHARA RAO, M. Tech., Ph.D.,

Asst. Professor



Department of Computer Science and Engineering

MALINENI LAKSHMAIAH ENGINEERING COLLEGE

(Approved by AICTE, Affiliated to JNTUK, Kakinada)

KANUMALLA (Vill), SINGARAYAKONDA -523101, Prakasam (Dist.), A.P.

2022-2023

MALINENI LAKSHMAIAH ENGINEERING COLLEGE

Department of Computer Science and Engineering

(Approved by AICTE, Affiliated to JNTUK, Kakinada)

KANUMALLA (Vill), SINGARAYAKONDA -523101, Prakasam (Dist.), A.P.



CERTIFICATE

This is to certify that it is a bonafide record of the industry project work entitled
“ONLINE BUS PASS SYSTEM” done by

CH. RAJESWARARAO

19851A0503

SK. AHAMAD JANI

19851A0522

the students of **B .Tech.,** Department of *Computer Science and Engineering,*
M.L.E.C., Singarayakonda during **2019-2023** in partial fulfillment of the requirements
for the award of Degree of **B .Tech** in *Computer Science and Engineering.* This work
is not submitted to any other University or Institute for the award of any Degree.

Dr. B. SRIDHARA RAO, M. Tech., Ph.D.,

Supervisor

Dr. B. SRIDHARA RAO, M. Tech., Ph.D.,

Head of the Department

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

We would like to express our deep sense of gratitude to our guide and Head of the Department, **Dr. B. SRIDHARA RAO** for his technical support and guidance. His critical evaluation of our work and suggestions have been great help to us.

We convey our sincere thanks to our principal **Dr. A.G.K. MOORTHY** for his valuable cooperation throughout the project duration.

We take immense pleasure in conveying our sincere and warm thanks to **Dr. M. PERUMALLU**, Secretary & Correspondent of our college whose unfailing enthusiasm and inspiring guidance motivated us to complete this project.

We would also like to extend our thanks to the **Teaching and Non-teaching staff members** whose help cannot be ignored in completing this project in time.

Special thanks to our **Friends** for their cooperation during my course of study. Last but not the least, we wish to thank our **Parents** and **Family members** without whom it would have been impossible for us to be in this level.

CH. RAJESWARARAO

19851A0503

SK. AHAMAD JANI

19851A0522

ABSTRACT

ABSTRACT

E-Bus Pass System Project is a real time project which is useful for the students who Are facing problems with the current manual work of bus pass Registration and renewal. It also increases the validity period, frequently Warns to the student before completion of his validity period by sending sms or mails. His / Her Renewal or Registration can be done using a voucher or even by a credit card. This online bus pass registration application will help students save their time and renewal bus passes without standing in a line for hours near counters. Initially students need to register with the application by submitting details of photo address proof, and required details and submit through online. They will verify your details and if they are satisfied they will approve bus pass. You can even renewal using credit card or other wire transfer methods..

INDEX

CHAPTER	CONTENTS	PAGE NO.
1	INTRODUCTION	1-3
2	SYSTEM ANALYSIS	4-6
	2.1. EXISTING SYSTEM	
	2.1.1. DISADVANTAGES OF EXISTING SYSTEM	
	2.2. PROPOSED SYSTEM	
	2.2.1. ADVANTAGES OF PROPOSED SYSTEM	
	2.3. SYSTEM STUDY	
	2.3.1. TYPES OF FEASIBILITY	
3	SYSTEM REQUIREMENTS	7
	3.1. SYSTEM REQUIREMENTS SPECIFICATIONS	
4	SOFTWARE ENVIRONMENT	8-31
5	SYSTEM DESIGN & DEVELOPMENT	32-53
	5.1. UML DIAGRAMS	
	5.1.1. USE CASE DIAGRAM	
	5.1.2. SEQUENCE DIAGRAM	
	5.1.3. COLLOBARATION DIAGRAM	
	5.1.4. STATE CHART DIAGRAM	
	5.1.5. CLASS DIAGRAM	
	5.1.6. ACTIVITY DIAGRAM	
	5.1.7. COMPONENT DIAGRAM	
	5.1.8. DEPLOYMENT DIAGRAM	
	5.1.9. DATA BASE DESIGN	
6.	SAMPLE SOURCE CODE	54-74

INDEX

CHAPTER	CONTENTS	PAGE NO.
7	SYSTEM TESTING	75-83
	7.1. INTRODUCTION	
	7.2. DESIGN OF TEST CASES AND SCENARIOS	
	7.3. TEST CASES AND SCENARIOS	
	7.4. TEST CASE DESCRIPTION	
	7.5. TEST CASES	
8	RESULTS	84-104
	SCREEN SHORTS	
9	CONCLUSION & FUTURE ENCHANCEMENT	105
	9.1. CONCLUSION	
	8.2. FURTHER ENHANCEMENT	
	BIBLIOGRAPHY	106

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO.
5.1.1.	USE CASE DIAGRAM FOR ADMIN AND USER	40
5.1.2 .	SEQUENCE DIAGRAM	42
5.1.3.	COLLOBARATION DIAGRAM	44
5.1.4 .	STATE CHART DIAGRAM OF ADMIN AND USER	45
5.1.5.	CLASS DIAGRAM	46
5.1.6.	ACTIVITY DIAGRAM	48
5.1.7.	COMPONENT DIAGRAM	49
5.1.8.	DEPLOYMENT DIAGRAM	49

Chapter - 1

INTRODUCTION

Problem Definition

The project will be used by the students to apply for bus pass and also to get the bus pass renewal in online .If we consider the current system then the students has to wait for long time standing in queue to apply for bus pass near the counters. Even if they want to get their bus pass get renewal then they have follow the same procedure of waiting for long hours near the counters. The current procedure is a time consuming procedure.

The current system leads to the inconvenience and dissatisfactory of the users. People have to travel to the nearby and respective Bus Stops and to apply for bus pass and also for the bus pass renewal. It requires a lot of man power.

Hence to overcome the above problems a system is being developed. This system reduced the time in issue/renew of bus-pass considerably, thus leading to convenience & satisfaction to the commuters. It has reduced the time taken to issue/renew bus-passes from 25 – 30 minutes to 3 – 5 minutes. Now, there are almost no queues on bus-pass issue centers. With computerization, possibility of fake/duplicate passes has reduced considerably. It has led to commuter satisfaction by reduction in time and fatigue in getting the pass issued. This online bus pass registration application will help students save their time and renewal bus passes without standing in a line for hours near counters. Initially students need to register with the application by submitting details of photo address proof, and required details and submit through online. They will verify your details and if they are satisfied they will approve bus pass. You can even renewal using credit card or other wire transfer methods.

Project Analysis:

Online Bus Pass System Project is a real time project which is useful for the students who Are facing problems with the current manual work of bus pass Registration and renewal. It also increases the validity period, frequently Warns to the student before completion of his validity period by sending sms or mails. His / Her Renewal or Registration can be done using a voucher or even by a credit card.

This online bus pass registration application will help students save their time and renewal bus passes without standing in a line for hours near counters. Initially students need to register with the application by submitting details of photo address proof, and required details and submit through online. They will verify your details and if they are satisfied they will approve bus pass. You can even renewal using credit card or other wire transfer methods.

This application consists of following modules

1. Online Bus-Pass Registration
2. Online Bus-Pass Renewal
3. Online Pass Schemes
4. Status for Registrations & Renewals
5. Search Routes & Fares
6. Reports

Module I: Online Bus-Pass Registration:

Different types of bus pass registration for the people of the twin cities. In this module initially students need to register with the application by submitting details of photo, address proof and required details and submit through online. They will verify your details and if they are satisfied they will approve bus pass.

Module II: Online Bus-Pass Renewal:

Different types of bus pass renewals for the people of the twin cities. In this module students have to Login to the website and apply for Renewal and they have to check their status which shows whether their bus pass is renewal or not. You can even renew using credit card or other wire transfer methods. Once the bus pass gets renewal Amount is deducted automatically.

Module III: Online Pass Schemes:

Different types of bus pass schemes for the people of the twin cities. In this module User can view different schemes which are available

Module IV: Status for Registrations & Renewals:

Checking the status of registrations and renewal of passes. This module is very useful as students will get to know what is their status is. In this module Admin will verify the details of user and generates status either as Accepted or rejected. Based on this status information further actions will be carried out such as deduction of bus pass amount from the user account.

Module V: Search Routes & Fares:

To know the fares and routes for different bus pass Registrations. In this module different routes and fares are provided for the user so that he can easily come to know the routes and their fares.

Module VI: Reports:

The user can give suggestions regarding our website or can give any complaints.

Chapter - 2

STUDY OF THE SYSTEM

2.1. EXISTING SYSTEM

In existing system the students has to wait for long time standing in queue to apply for bus pass near the counters. Even if they want to get their bus pass get renewal then they have follow the same procedure of waiting for long hours near the counters. The current procedure is a time consuming procedure. The current system leads to the inconvenience and dissatisfactory of the users. People have to travel to the nearby and respective Bus Stops and to apply for bus pass and also for the bus pass renewal. It requires a lot of man power.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

- Takes lot of time in issue/renew of bus-pass considerably.
- Leads to the inconvenience and dissatisfactory of the users.
- The time taken in the issue/renew of bus-pass considerably is 25 – 30 minutes.
- Students have to wait for long time standing in queue near counters.

2.2. PROPOSED SYSTEM

The Proposed system will help students save their time and renewal bus passes without standing in a line for hours near counters. Initially students need to register with the application by submitting details of photo, address proof, and required details and submit through online. They will verify your details and if they are satisfied they will approve bus pass. You can even renewal using credit card or other wire transfer methods.

2.2.1. ADVANTAGES OF PROPOSED SYSTEM

- Has reduced the time in issue/renew of bus-pass considerably.
- Leads to the convenience and satisfactory of the users.
- The time taken in the issue/renew of bus-pass considerably is 3-5 minutes.
- Now, there are almost no queues on bus-pass issue centers.
- With computerization, possibility of fake/duplicate passes has reduced considerably.

2.3. FEASIBILITY STUDY

A feasibility study is an evaluation of the proposal is to determine the difficulty in carry out a task. Generally a feasibility study precedes technical development and project implementation in other words, a feasibility study is a evaluation (or) analysis of the potential impact of the proposed project.

2.3.1. TYPES OF FEASIBILITY

- **TECHNOLOGY FEASIBILITY:** The assessment is based the outline design of system required in terms of input processes, output, fields, programs and procedures. This can be quantified in terms of volumes of data trends, frequency of updating etc in order to estimate whether the new system will perform adequately or not. Technology feasibility is carried to determine whether the company has capability in terms of software, hardware, personal and expertise the completion of project.
- **ECONOMIC FEASIBILITY:** Economic feasibility analysis is the most frequently used method for evaluating the effectiveness of new system, commonly cost of benefit analysis. The procedure is to determine the benefits outweigh cost then the decision is made to design and implemented the system entrepreneur must accurately weigh the cost isa benefit before taking an action.

- **LEGAL FEASIBILITY:** Determines whether the proposed system conflicts with legal requirements eg-a data processing system comply with a local protection acts.
- **OPERATING FEASIBILITY:** It is a measure how we are proposed system solves the problems and takes advantages of the opportunity identify during scope definition and how it satisfies the requirements identified the requirements analysis phase of system development.
- **SCHEDULE FEASIBILITY:** A project will fail if it takes too long to be completion before it is used. Typically this means estimating how long the system will take to develop and if it can be completed in a given time period using so many methods like payback period. Schedule feasibility is a measure how the responsible the project timetable is.

Chapter – 3

SYSTEM SPECIFICATION

3.1. SYSTEM REQUIREMENTS SPECIFICATIONS

SOFTWARE REQUIREMENTS

Operating System	:	Windows 7
Technology	:	Java/J2EE (Servlets, JSP, JDBC)Web Technologies : Html, JavaScript, CSS
Web Server	:	Tomcat 7.0
Database	:	Oracle 10g Express Edition
Software's	:	JDK 1.6

HARDWARE REQUIREMENTS

Hardware	:	Pentium based systems with a minimum of P4
RAM	:	1GB (minimum)

Chapter – 4

SOFTWARE ENVIRONMENT

Java Technology

Java technology is both a programming language and a platform.

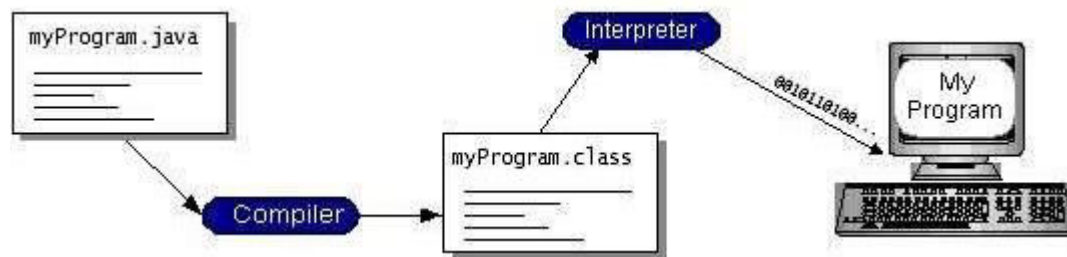
The Java Programming Language

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

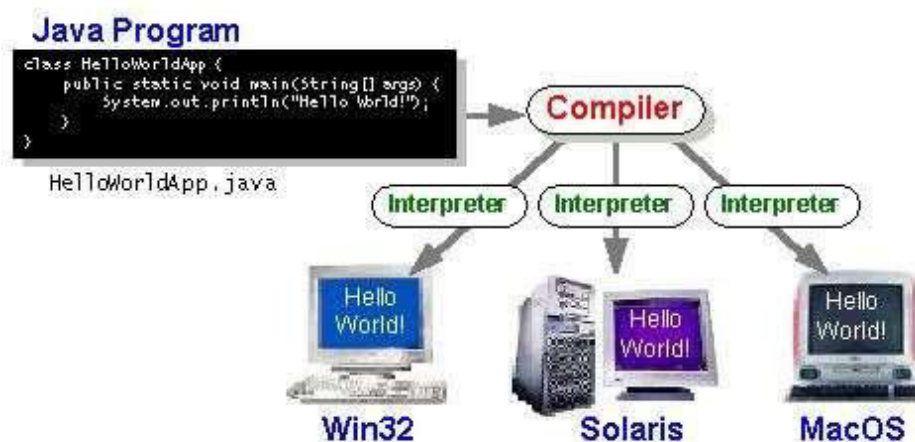
- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make "write once, run anywhere" possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

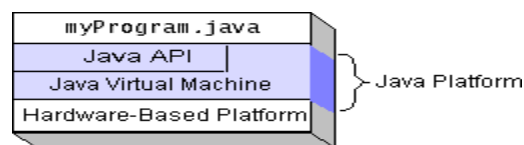
- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

Native code is code that after you compile it, the compiled code runs on a specific hardware



platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser. However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality.

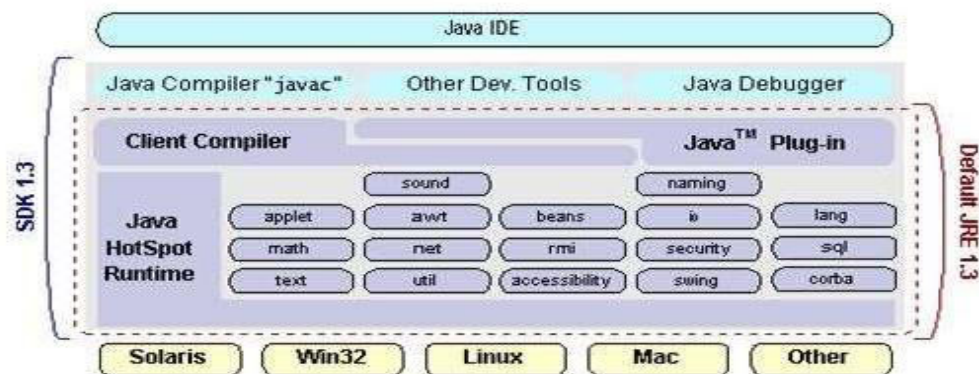
Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Datagram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users

worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.

- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components:** Known as JavaBeans™, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™):** Provides uniform access to a wide range of relational databases. The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more.

The following figure depicts what is included in the Java 2 SDK.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages.

We believe that Java technology will help you do the following:

- **Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
- **Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java TM Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere:** Because 100% Pure Java programs are compiled to machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded "on the fly," without recompiling the entire program.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers.

Before ODBC became a *de facto* standard for Windows programs to interface with database

systems, programmers had to use proprietary languages for each database they wanted to connect to.

Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database.

For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE.

There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources. From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor.

The source code of the application doesn't change whether it talks to Oracle or SQL Server.

We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or *drivers*.

If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on. To gain a wider acceptance of JDBC, Sun

based JDBC's framework on ODBC.

As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after. The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java. The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC's complexities from the end user.

2. SQL Conformance

SQL syntax varies as you move from database vendor to database vendor. In an effort to

support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non- standard functionality in a manner that is suitable for its users

3. JDBC must be implemental on top of common database interfaces The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. Provide a Java interface that is consistent with the rest of the Java system

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. Keep it simple

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. Use strong, static typing wherever possible

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. Keep the common cases simple

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible. Java has two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple Architecture-neutral

Object-oriented Portable

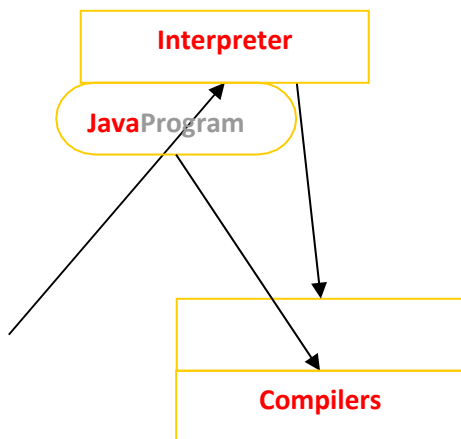
Distributed High-performance

Interpreted multithreaded

Robust Dynamic

Secure

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM.

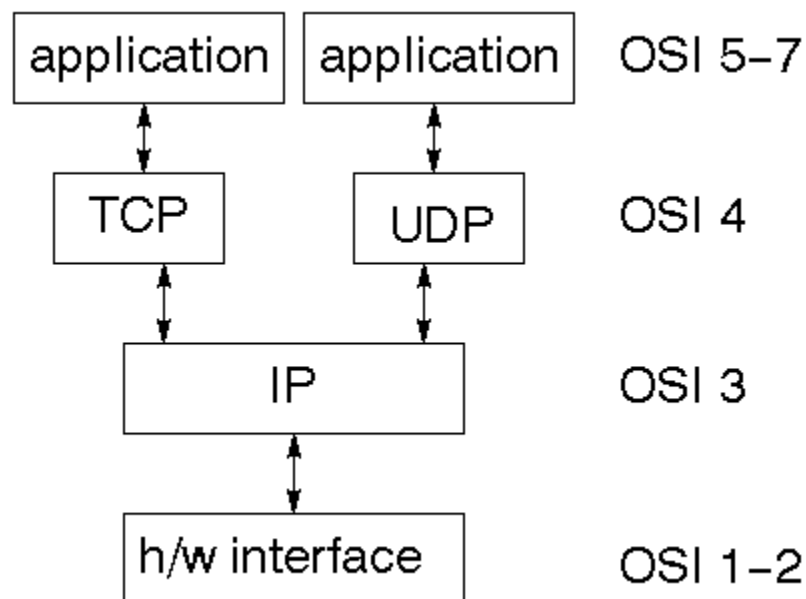
The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then

be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers.

The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and

reassembling them at the other end.

UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model

- see later.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

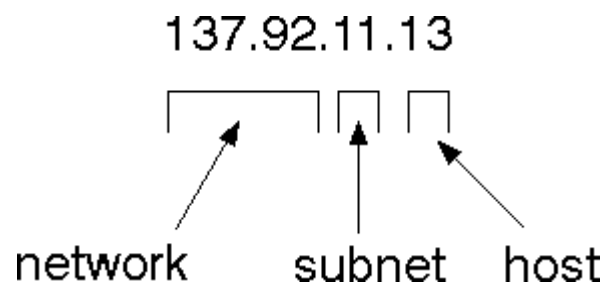
Subnet address

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

Total address



The 32 bit address is usually written as 4 integers separated by dots.

Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

```
#include <sys/types> #include <sys/socket>
```

```
in socket(in family, in type, in protocol);
```

Here "family" will be `AF_INET` for IP communications, protocol will be zero, and type will

depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe
- but the actual pipe does not yet exist.

JFree Chart

JFree Chart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFree Chart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFree Chart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include:

(a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include: Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFree Chart;

Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFree Chart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

4. Property Editors

The property editor mechanism in JFree Chart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

What is a Java Web Application?

A Java web application generates interactive web pages containing various types of markup language (HTML, XML, and so on) and dynamic content. It is typically comprised of web components such as Java Server Pages (JSP), servlets and JavaBeans to modify and temporarily store data, interact with databases and web services, and render content in response to client requests.

Because many of the tasks involved in web application development can be repetitive or require a surplus of boilerplate code, web frameworks can be applied to alleviate the overhead associated with common activities.

For example, many frameworks, such as Java Server Faces, provide libraries for templating pages and session management, and often promote code reuse.

What is Java EE?

Java EE (Enterprise Edition) is a widely used platform containing a set of coordinated

technologies that significantly reduce the cost and complexity of developing, deploying, and managing multi-tier, server-centric applications.

Java EE builds upon the Java SE platform and provides a set of APIs (application programming interfaces) for developing and running portable, robust, scalable, reliable and secure server-side applications.

Some of the fundamental components of Java EE include:

- Enterprise JavaBeans (EJB): a managed, server-side component architecture used to encapsulate the business logic of an application. EJB technology enables rapid and simplified development of distributed, transactional, secure and portable applications based on Java technology.

JavaScript and Ajax Development

JavaScript is an object-oriented scripting language primarily used in client-side interfaces for web applications. Ajax (Asynchronous JavaScript and XML) is a Web 2.0 technique that allows changes to occur in a web page without the need to perform a page refresh. JavaScript toolkits can be leveraged to implement Ajax-enabled components and functionality in web pages.

Web Server and Client

Web Server is a software that can process the client request and send the response back to the client.

For example, Apache is one of the most widely used web server. Web Server runs on some physical machine and listens to client request on specific port.

A web client is a software that helps in communicating with the server. Some of the most widely used web clients are Firefox, Google Chrome, Safari etc. When we request something from server (through URL), web client takes care of creating a request and sending

it to server and then parsing the server response and present it to the user.

HTML and HTTP

Web Server and Web Client are two separate software's, so there should be some common language for communication. HTML is the common language between server and client and stands for **Hyper Text Markup Language**.

Web server and client needs a common communication protocol, HTTP (**H**yper **T**ext **T**ransfer **P**rotocol) is the communication protocol between server and client. HTTP runs on top of TCP/IP communication protocol.

Some of the important parts of HTTP Request are:

- **HTTP Method** – action to be performed, usually GET, POST, PUT etc.
- **URL** – Page to access
- **Form Parameters** – similar to arguments in a java method, for example user , password details from login page.

Sample HTTP Request:

```
1GET /FirstServletProject/jsp/hello.jsp HTTP/1.1Host: localhost:8080
```

```
3Cache-Control: no-cache
```

Some of the important parts of HTTP Response are:

- **Status Code** – an integer to indicate whether the request was success or not.

Some of the well known status codes are 200 for success, 404 for Not Found and 403 for Access Forbidden.

- **Content Type** – text, html, image, pdf etc. Also known as MIME type
- **Content** – actual data that is rendered by client and shown to user.

MIME Type or Content Type: If you see above sample HTTP response header, it

contains tag “Content-Type”. It’s also called MIME type and server sends it to client to let them know the kind of data it’s sending. It helps client in rendering the data for user. Some of the mostly used mime types are text/html, text/xml, application/xml etc.

Understanding URL

URL is acronym of Universal Resource Locator and it’s used to locate the server and resource. Every resource on the web has it’s own unique address. Let’s see parts of URL with an example. **http://localhost:8080/FirstServletProject/jsp/hello.jsp**

http:// – This is the first part of URL and provides the communication protocol to be used in server-client communication.

localhost – The unique address of the server, most of the times it’s the hostname of the server that maps to unique IP address. Sometimes multiple hostnames point to same IP addresses and web server virtual host takes care of sending request to the particular server instance.

8080 – This is the port on which server is listening, it’s optional and if we don’t provide it in URL then request goes to the default port of the protocol. Port numbers 0 to 1023 are reserved ports for well known services, for example 80 for HTTP, 443 for HTTPS, 21 for FTP etc.

FirstServletProject/jsp/hello.jsp – Resource requested from server. It can be static html, pdf, JSP, servlets, PHP etc.

Why we need Servlet and JSPs?

Web servers are good for static contents HTML pages but they don’t know how to generate dynamic content or how to save data into databases, so we need another tool that we can use to generate dynamic content. There are several programming languages for dynamic content like PHP, Python, Ruby on Rails, Java Servlets and JSPs.

Java Servlet and JSPs are server side technologies to extend the capability of web servers by providing support for dynamic response and data persistence.

Web Container

Tomcat is a web container, when a request is made from Client to web server, it passes the request to web container and it's web container job to find the correct resource to handle the request (servlet or JSP) and then use the response from the resource to generate the response and provide it to web server. Then web server sends the response back to the client.

When web container gets the request and if it's for servlet then container creates two Objects `HttpServletRequest` and `HttpServletResponse`. Then it finds the correct servlet based on the URL and creates a thread for the request. Then it invokes the servlet `service()` method and based on the HTTP method `service()` method invokes `doGet()` or `doPost()` methods. Servlet methods generate the dynamic page and write it to response. Once servlet thread is complete, container converts the response to HTTP response and send it back to client.

Some of the important work done by web container are:

- **Communication Support** – Container provides easy way of communication between web server and the servlets and JSPs. Because of container, we don't need to build a server socket to listen for any request from web server, parse the request and generate response. All these important and complex tasks are done by container and all we need to focus is on our business logic for our applications.
- **Lifecycle and Resource Management** – Container takes care of managing the life cycle of servlet. Container takes care of loading the servlets into memory, initializing servlets, invoking servlet methods and destroying them. Container also provides utility like JNDI for resource pooling and management.
- **Multithreading Support** – Container creates new thread for every request to the servlet and when it's processed the thread dies. So servlets are not initialized for each

request and saves time and memory.

- **JSP Support** – JSPs doesn't look like normal java classes and web container provides support for JSP. Every JSP in the application is compiled by container and converted to Servlet and then container manages them like other servlets.
- **Miscellaneous Task** – Web container manages the resource pool, does memory optimizations, run garbage collector, provides security configurations, support for multiple applications, hot deployment and several other tasks behind the scene that makes our life easier.

Web Application Directory Structure

Java Web Applications are packaged as Web Archive (WAR) and it has a defined structure. You can export above dynamic web project as WAR file and unzip it to check the hierarchy. It will be something like below image.



Deployment Descriptor

web.xml file is the deployment descriptor of the web application and contains mapping for servlets (prior to 3.0), welcome pages, security configurations, session timeout settings etc. That's all for the java web application startup tutorial, we will explore Servlets and JSPs more in future posts.

MySQL:

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site (<http://www.mysql.com/>) provides the latest information about MySQL software.

- **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables.

The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and

several versions exist.

In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), <http://www.fsf.org/licenses/>, to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information (<http://www.mysql.com/company/legal/licensing/>).

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together. You can find a performance comparison of MySQL Server with other database managers on our

benchmark page. MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different back ends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

- **A large amount of contributed MySQL software is available.**

MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

CHAPTER - 5

SYSTEM DESIGN & DEVELOPMENT

5.1. UML DIAGRAMS

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic, semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

i. User model view

- This view represents the system from the users perspective.
- The analysis representation describes a usage scenario from the end-users perspective.

ii. Structural model view

- In this model the data and functionality are arrived from inside the system.
- This model view models the static structures.

iii. Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

iv. Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be built.

v. Environmental Model View

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the system.
- UML design modeling, which focuses on the behavioral modeling,

- implementation modeling and environmental model views.
- Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.
- Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer ...etc., or another system like central database.

The UML Is a Language for Documenting:

A healthy software organization produces all sorts of artifacts in addition to raw executable code. These artifacts include (but are not limited to)

- ☐ Requirements
- ☐ Architecture
- ☐ Design
- ☐ Source code
- ☐ Project plans
- ☐ Tests
- ☐ Prototypes
- ☐ Releases

Depending on the development culture, some of these artifacts are treated more or less formally than others. Such artifacts are not only the deliverables of a project, they are also critical in controlling, measuring, and communicating about a system during its development and after its deployment. The UML addresses the documentation of a system's architecture and all of its details

. The UML also provides a language for expressing requirements and for tests. Finally, the UML provides a language for modeling the activities of project planning and release

management.

Applications:

The UML is intended primarily for software-intensive systems. It has been used effectively for such domains as

- ☐ Enterprise information systems
- ☐ Banking and financial services
- ☐ Telecommunications
- ☐ Transportation
- ☐ Defense/aerospace
- ☐ Retail
- ☐ Medical electronics
- ☐ Scientific
- ☐ Distributed Web-based services

The UML is not limited to modeling software. In fact, it is expressive enough to model no software systems, such as workflow in the legal system, the structure and behavior of a patient healthcare system, and the design of hardware.

☐ **A Conceptual Model of the UML:**

To understand the UML, you need to form a conceptual model of the language, and this requires learning three major elements: the UML's basic building blocks, the rules that dictate how those building blocks may be put together, and some common mechanisms. Those apply throughout the UML. Once you have grasped these ideas, you will be able to read UML models and create some basic ones. As you gain more experience in applying the UML, you can build on this conceptual model, using more advanced features of the language.

□ **Building Blocks of the UML:**

The vocabulary of the UML encompasses three kinds of building blocks:

- Things
- Relationship
- digarams

Things are the abstractions that are first-class citizens in a model; relationshipstie these things together; diagrams group interesting collections of things.

□ **Things in the UML:**

There are three kinds of things in the UML:

- Structural things
- Behavioral things
- Grouping things

□ **Relationships in the UML:**

There are four kinds of relationships in the UML:

- Dependency
- Association
- Generalization
- Realization

These relationships are the basic relational building blocks of the UML.

1. First, a dependency is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing. Graphically, a dependency is rendered as a dashed line, possibly directed, and occasionally includinga label.

2. Second, an association is a structural relationship that describes a set of links, a link

being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts. Graphically, an association is rendered as a solid line, possibly directed, occasionally including a label, and often containing other adornments.

3. Third, a generalization is a specialization/generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent). Graphically, a generalization relationship is rendered as a solid line with a hollow arrowhead pointing to the parent.

4. Fourth, a realization is a semantic relationship between classifiers, wherein one classifier specifies a contract that another classifier guarantees to carry out. Graphically, a realization relationship is rendered as a cross between a generalization and a dependency relationship.

□ **Diagrams in the UML**

A **diagram** is the graphical presentation of a set of elements, most often rendered as a connected graph of vertices (things) and arcs (relationships). You draw diagrams to visualize a system from different perspectives, so a diagram is a projection into a system. For all but the most trivial systems, a diagram represents an elided view of the elements that make up a system. The same element may appear in all diagrams, only a few diagrams (the most common case), or in no diagrams at all (a very rare case). In theory, a diagram may contain any combination of things and relationships. In practice, however, a small number of common combinations arise,

which are consistent with the five most useful views that comprise the architecture of a software-intensive system. For this reason, the UML includes nine such diagrams:

- Class diagram
- Object diagram

- Use case diagram
- Sequence diagram
- Collaboration diagram
- State chart diagram
- Activity diagram
- Component diagram
- Deployment diagram

A class diagram shows a set of classes, interfaces, and collaborations and their relationships. These diagrams are the most common diagram found in modeling object-oriented systems. Class diagrams address the static design view of a system. An object diagram shows a set of objects and their relationships. Object diagrams represent static snapshots of instances of the things found in class diagrams.

These diagrams address the static design view or static process view of a system as do class diagrams, but from the perspective of real or prototypical cases.

A sequence diagram is an interaction diagram that emphasizes the time- ordering of messages; a collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.

Both sequence diagrams and collaboration diagrams are kinds of interaction diagrams. A collaboration diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. Interaction diagrams address the dynamic view of a system.

A use case diagram shows a set of use cases and actors (a special kind of class) and their relationships. Use case diagrams address the static use case view of a system.

These diagrams are especially important in organizing and modeling the behaviors of a system

A statechart diagram shows a state machine, consisting of states, transitions, events, and activities. Statechart diagrams address the dynamic view of a system. They are especially important in modeling the behavior of an interface, class, or collaboration and emphasize the event-ordered behavior of an object, which is especially useful in modeling reactive systems. An activity diagram is a special kind of a state chart diagram that shows the flow from activity to activity within a system. Activity diagrams address the dynamic view of a system. They are especially important in modeling the function of a system and emphasize the flow of control among objects.

A component diagram shows the organizations and dependencies among a set of components. Component diagrams address the static implementation view of a system. They are related to class diagrams in that a component typically maps to one or more classes, interfaces, or collaborations.

A deployment diagram shows the configuration of run-time processing nodes and the components that live on them. Deployment diagrams address the static deployment view of architecture. They are related to component diagrams in that a node typically encloses one or more components

5.1.1. USE CASE DIAGRAM

A use case diagram shows a set of use cases and actors (a special kind of class) and their relationships. Use case diagrams address the static use case view of a system. These diagrams are especially important in organizing and modeling the behaviors of a system

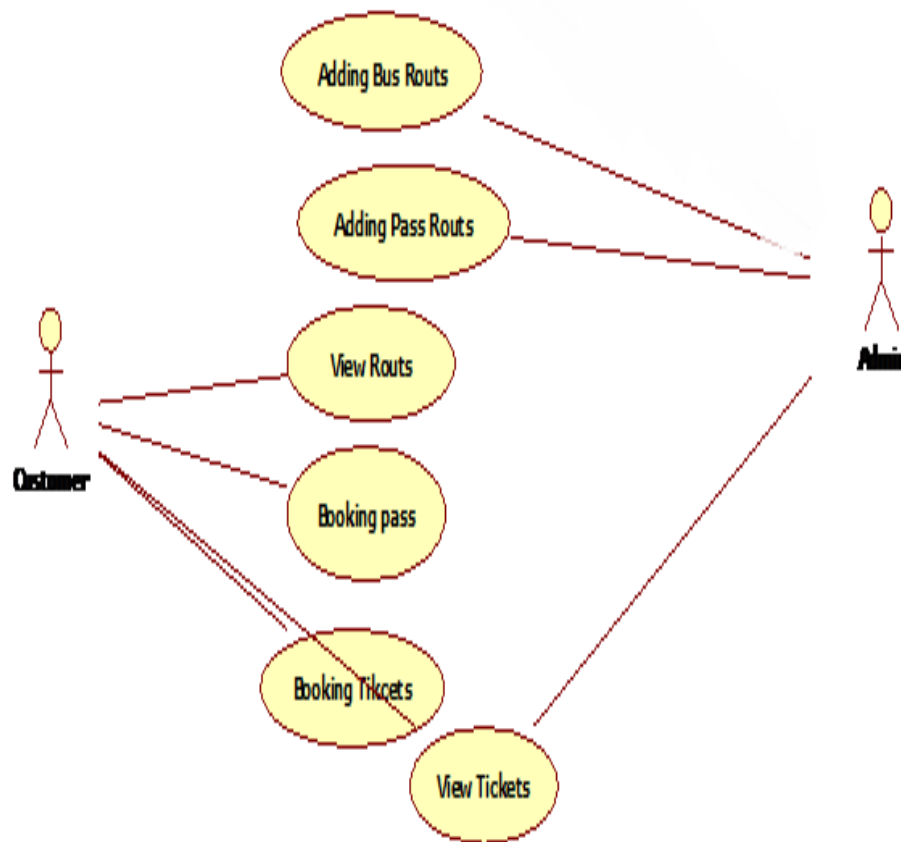
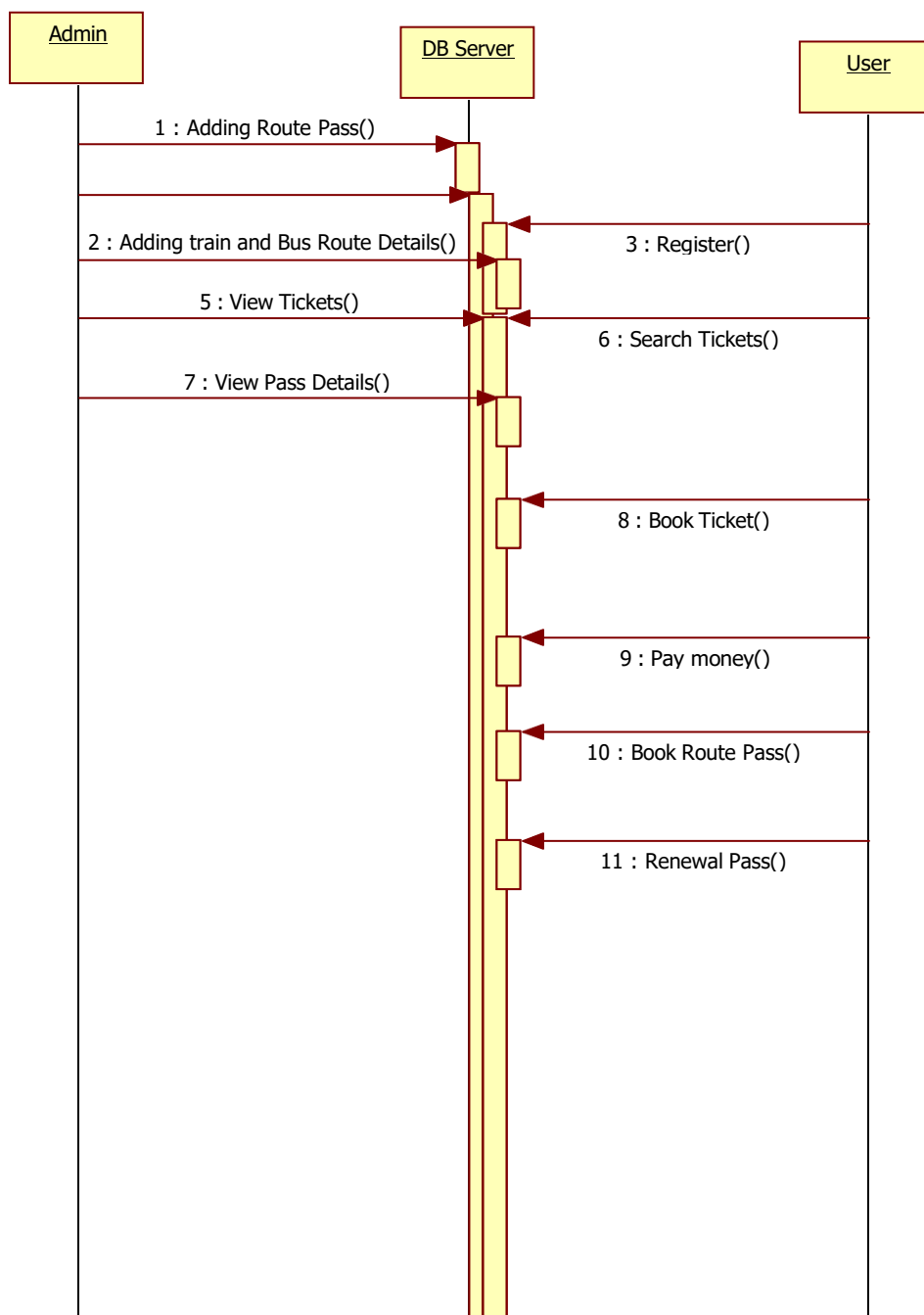


Fig : 5.1.1 :USE CASE DIAGRAM FOR ADMIN AND USER

5.1.2. SEQUENCE DIAGRAM

A sequence diagram is an interaction diagram that emphasizes the time- ordering of messages; a collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.

SEQUENCE DIAGRAM**Fig 5.1.2 :SEQUENCE DIAGRAM**

5.1.3. COLLOBARATION DIAGRAM

A collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.

Both sequence diagrams and collaboration diagrams are kinds of interaction diagrams. A collaboration diagram shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. Interaction diagrams address the dynamic view of a system.

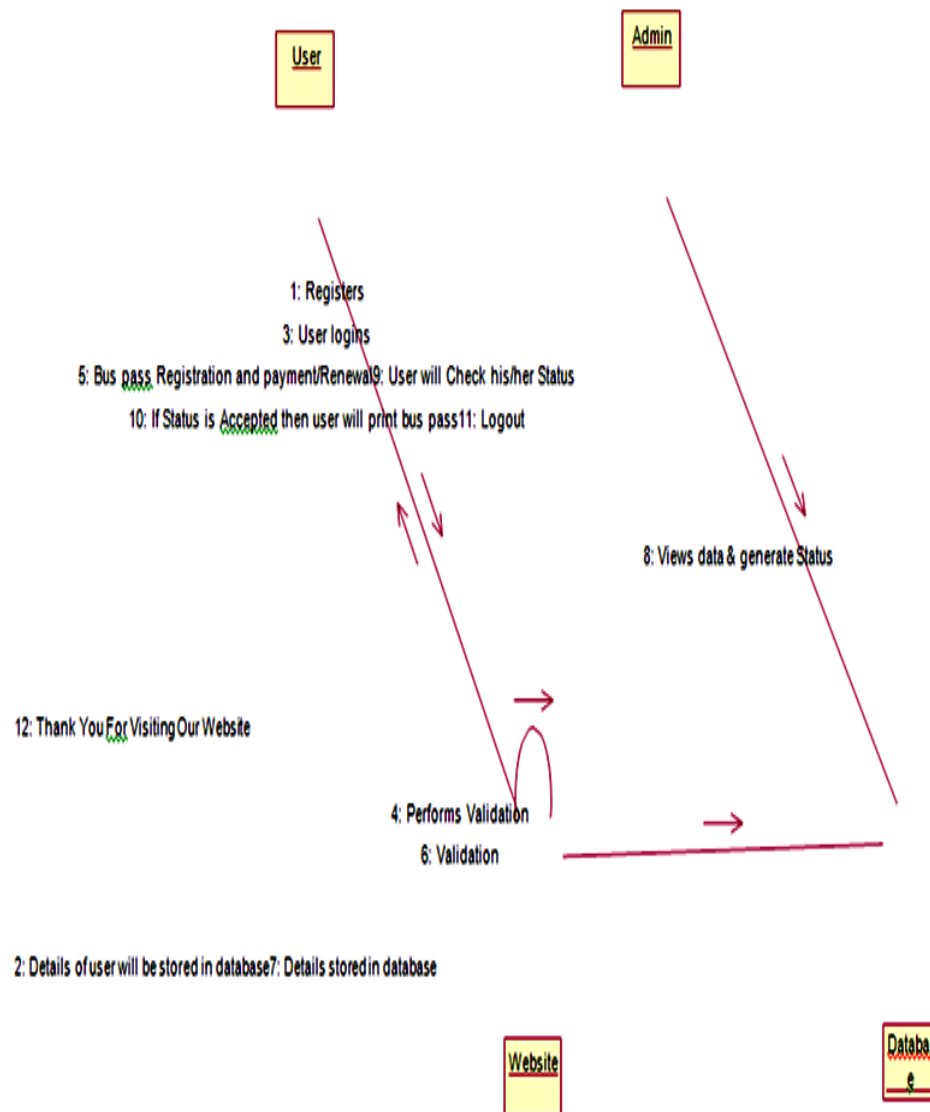


Fig 5.1.3 : COLLABORATION DIAGRAM

5.1.4. STATE CHART DIAGRAM

A statechart diagram shows a state machine, consisting of states, transitions, events, and activities. Statechart diagrams address the dynamic view of a system. They are especially important in modeling the behavior of an interface, class, or collaboration and emphasize the event-ordered behavior of an object, which is especially useful in modeling reactive systems.

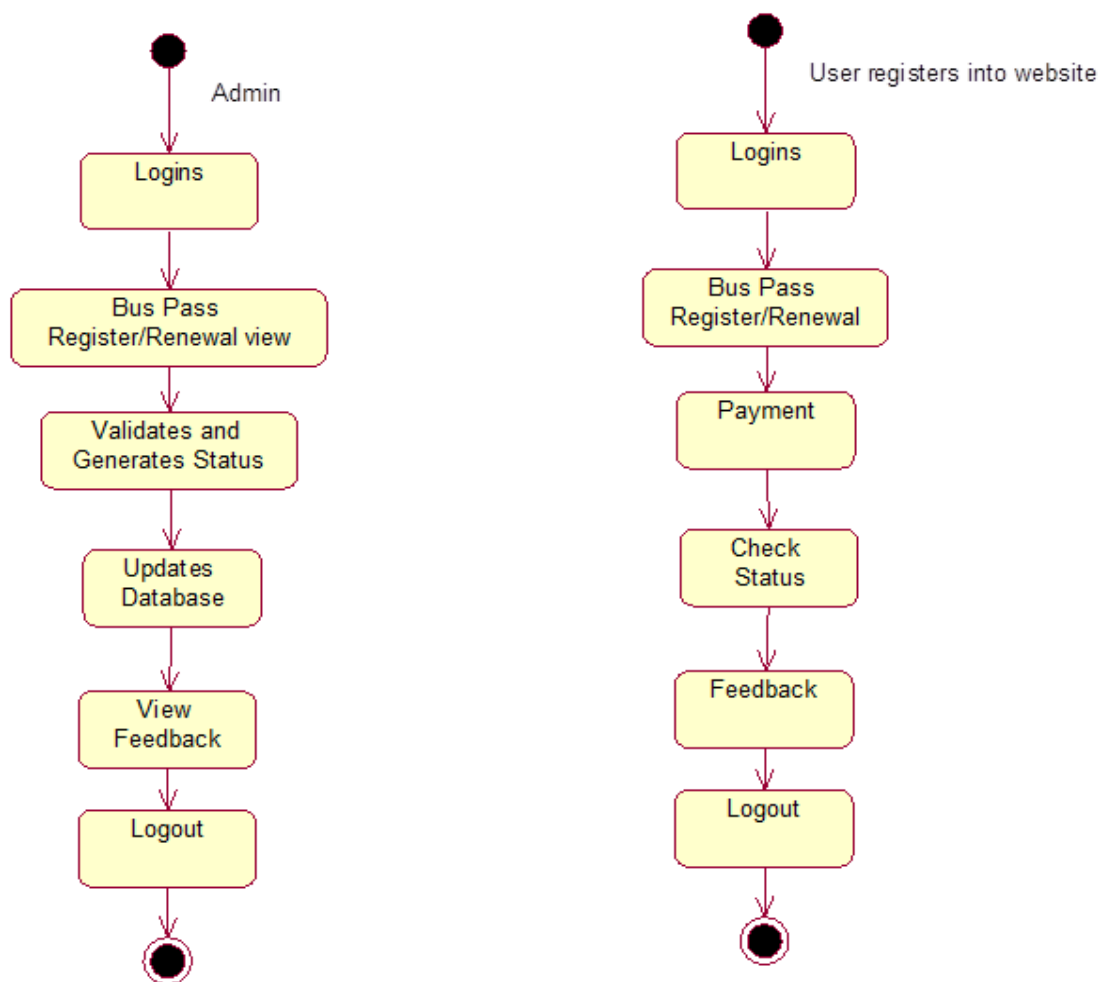


Fig 5.1.4 : STATE CHART DIAGRAM OF ADMIN AND USER

51.5. CLASS DIAGRAM

A class diagram shows a set of classes, interfaces, and collaborations and their relationships. These diagrams are the most common diagram found in modeling object-oriented systems. Class diagrams address the static design view of a system.

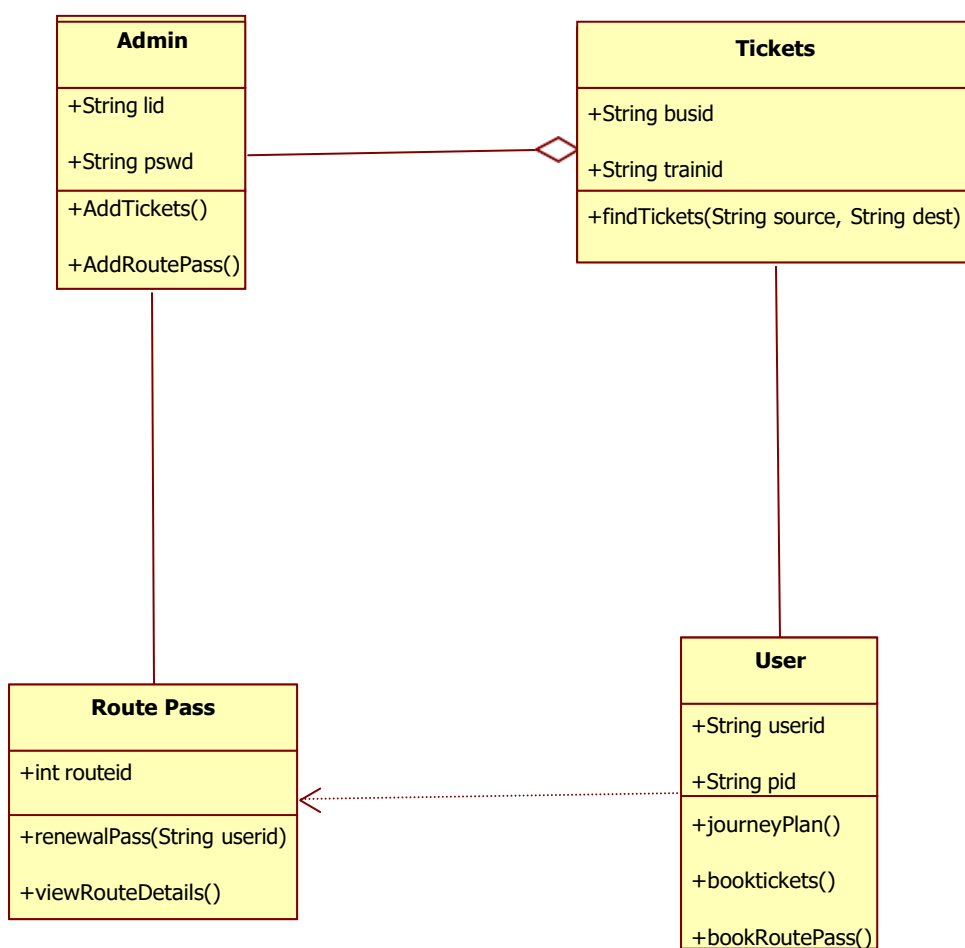
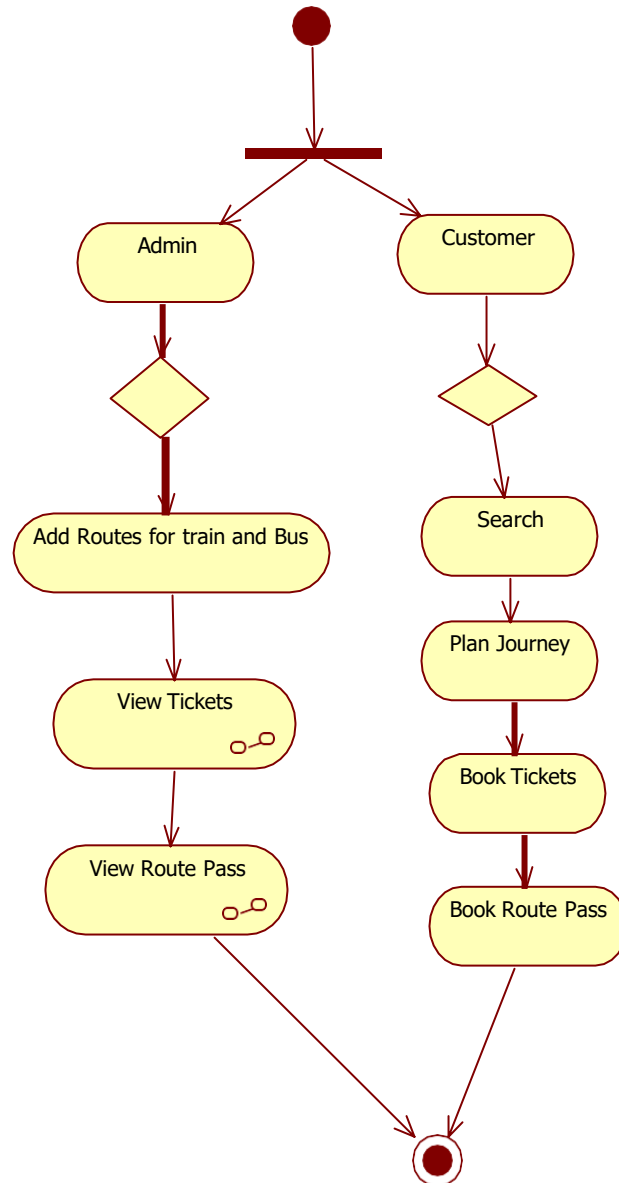


FIG 5.1.5 :CLASS DIAGRAM

5.1.6. ACTIVITY DIAGRAM

An activity diagram is a special kind of a statechart diagram that shows the flow from activity to activity within a system. Activity diagrams address the dynamic view of a system. They are especially important in modeling the function of a system and emphasize the flow of control among objects.

ACTIVITY DIAGRAM**FIG; 5.1.6. ACTIVITY DIAGRAM**

5.1.7. COMPONENT DIAGRAM

A deployment diagram shows the configuration of run-time processing nodes and the components that live on them. Deployment diagrams address the static deployment view of architecture. They are related to component diagrams in that a node typically encloses one or more components

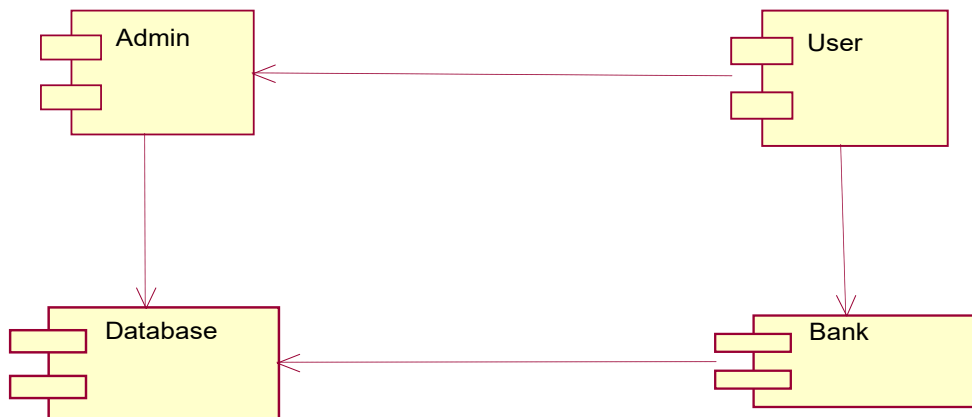


FIG 5.1.7 :COMPONENT DIAGRAM

5.1.8. DEPLOYMENT DIAGRAM:

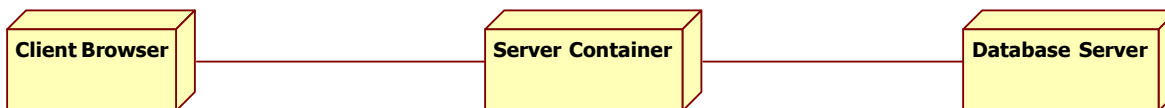


FIG 5.1.8 :DEPLOYMENT DIAGRAM

5.1.9. DATA BASE DESIGN

➤ **Data Base Tables:**

Column Name	Data Type	<u>Nullabls</u>	Default	Primary Key
USERNAME	VARCHAR2(50)	No	-	1
FIRSTNAME	CHAR(50)	No	-	-
MIDDLENAME	CHAR(50)	Yes	-	-
LASTNAME	CHAR(50)	No	-	-
DOB	VARCHAR2(12)	No	-	-
SEX	VARCHAR2(12)	No	-	-
MAILID	VARCHAR2(50)	No	-	-
PASSWORD	VARCHAR2(50)	No	-	-
REENTER_PASSWORD	VARCHAR2(50)	No	-	-
ADDRESS	VARCHAR2(1000)	No	-	-
PHONENO	NUMBER(38,10)	No	-	-
				1 - 11

TABLE FOR WEBSITE REGISTRATION

Column Name	Data Type	Nullable	Default	Primary Key
EID	VARCHAR2(1000)	No	-	1
FIRST_NAME	CHAR(50)	No	-	-
MIDDLE_NAME	CHAR(50)	Yes	-	-
SUR_NAME	CHAR(50)	No	-	-
FATHERS_OR_MOTHERS_NAME	CHAR(50)	No	-	-
FATHERS_OR_MOTHERS_SUR_NAME	CHAR(50)	No	-	-
D_O_B	VARCHAR2(1000)	No	-	-
SEX	VARCHAR2(20)	No	-	-
TYPE_OF_PASS	CHAR(20)	No	-	-
BUS_PASS_CODE	VARCHAR2(20)	No	-	-
INSTITUTION_NAME	CHAR(50)	No	-	-
INSTITUTION_CODE	VARCHAR2(20)	No	-	-
COURSE_NAME	CHAR(20)	No	-	-
COURSE_CODE	VARCHAR2(20)	No	-	-
ADMISSION_NUMBER	VARCHAR2(40)	No	-	-
ROUTE_FROM	CHAR(20)	No	-	-
ROUTE_VIA	CHAR(20)	No	-	-
ROUTE_TO	CHAR(20)	No	-	-
ADDRESS	VARCHAR2(1000)	No	-	-
PHONE_NUMBER	NUMBER(10,0)	No	-	-
IMAGE	BLOB	No	-	-
DECLARATION	VARCHAR2(30)	No	-	-
STATUS	VARCHAR2(500)	Yes	-	-
TYPE_OF_BUS	VARCHAR2(500)	No	-	-
VALID_FROM	VARCHAR2(500)	Yes	-	-
VALID_TO	VARCHAR2(500)	Yes	-	-

1 - 26

TABLE FOR BUS PASS REGISTRATION

Column Name	Data Type	Nullable	Default	Primary Key
BID	VARCHAR2(500)	Yes	-	-
AMOUNT	VARCHAR2(500)	Yes	-	-
ACCNUM	VARCHAR2(500)	Yes	-	-
STATUS	VARCHAR2(500)	Yes	-	-
OLD_VF	VARCHAR2(500)	Yes	-	-
OLD_VT	VARCHAR2(500)	Yes	-	-
NEW_VF	VARCHAR2(500)	Yes	-	-
NEW_VT	VARCHAR2(500)	Yes	-	-
COUNT	NUMBER	Yes	-	-
				1 - 9

TABLE FOR BUS PASS RENEWAL

Column Name	Data Type	Nullable	Default	Primary Key
ACCOUNT_NUMBER	VARCHAR2(500)	No	-	1
FIRST_NAME	VARCHAR2(500)	No	-	-
LAST_NAME	VARCHAR2(500)	No	-	-
D_O_B	VARCHAR2(20)	No	-	-
ACCOUNT_TYPE	VARCHAR2(500)	No	-	-
BRANCH	VARCHAR2(500)	No	-	-
ADDRESS	VARCHAR2(4000)	No	-	-
PHONE_NUM	NUMBER	No	-	-
BALANCE	NUMBER	No	-	-
PIN	NUMBER	No	-	-
				1 - 10

TABLE FOR BANK

Chapter - 6

SAMPLE SOURCE CODE

DATABASE CONNECTION:

```
package com.ebuss.db;

import java.sql.Connection; import java.sql.DriverManager;
import java.sql.PreparedStatement;import java.sql.SQLException;

public class DBConnection { private static Connection con=null;
private static PreparedStatement ps =null;

public static PreparedStatement getDBConnection(String query){ try{
DriverManager.registerDriver(new com.mysql.jdbc.Driver());

con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/ebussticket","root","root");
//con = DriverManager.getConnection("jdbc:mysql://node103575-env-
2386197.j.layershift.co.uk/ebussticket","root","BPGpgz11774");
ps = con.prepareStatement(query);if(con!=null){
return ps;

}
} catch(SQLException sqle){
System.out.println(sqle.getMessage());
} catch(Exception e){ System.out.println(e.getMessage());
}
}
```

```
return ps;
```

```
}
```

```
}
```

ADMIN.JAVA:

```
/*
```

```
*      To change this template, choose Tools | Templates
```

```
*      and open the template in the editor.
```

```
*/
```

```
package com.ebuss.dao;
```

```
import com.ebuss.beans.AddTrainAndBusPass;import com.ebuss.beans.AddingBussRoots;
```

```
import com.ebuss.beans.AddingTrainRoots; import com.ebuss.beans.AdminLoginBeans;
```

```
import com.ebuss.db.DBConnection;
```

```
import com.utils.InsertTrainRoots;import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet; import java.sql.SQLException; import java.text.DateFormat;
import java.text.SimpleDateFormat; import java.util.ArrayList;
import java.util.Date; import java.util.Iterator;
```

```
/**
```

```
*
```

```
*      @author Tanish
```

```
*/
```

```
public class AdminDAO {
```

```
public String loginCheck(AdminLoginBeans alb) { String msg=null;
String query="select count(*) from adminlogin where username=? and pwd=?";
PreparedStatement ps=null;
ResultSet rs = null;int count=0;
try{ ps=DBConnection.getDBConnection(query);ps.setString(1, alb.getUserName());
ps.setString(2, alb.getPwd());
rs = ps.executeQuery();

if(rs.next()){

count = rs.getInt(1);

}

}
```



```
if(count>0){ msg ="valid";return msg;
}else{ msg="invalid";return msg;
}
```

```
}catch(Exception e){ System.out.println(e.getMessage());
                        }finally{ try{ ps.close();
                        }catch(Exception e){ }
}
```

```
return msg;
```

```
}
```

```
public String addBussRoot(AddingBussRoots abr){ String result=null;
String query="insert into
bussroot(sourcecity,sourcestate,distance,destinationcity,desinationstate,cost,serviceno,bus
name,busstatus) values(?,?,?,?,?,?,?,?) ";
PreparedStatement ps=null;
```

```
//ResultSet rs = null;int count=0;
```

```
try{
```

```
ps = DBConnection.getDBConnection(query);ps.setString(1, abr.getSourcecity());
ps.setString(2,abr.getSourcestate()); ps.setFloat(3, abr.getDistance()); ps.setString(4,
abr.getDestinationcity()); ps.setString(5, abr.getDestinationstate()); ps.setFloat(6,
abr.getCost());
ps.setString(7, abr.getServiceNo());ps.setString(8, abr.getBusname());
```

```
ps.setString(9,"no");
count = ps.executeUpdate();if(count>0){
result ="success";return result;
}else{ result="faild";return result;
}

}

} catch(SQLException sql){ System.out.println(sql.getMessage());
} catch(Exception e){ System.out.println(e.getMessage());

} catch(Exception e){ }

}

return result;

}
```

AddingBusRoutes.java

```
/*
 *      To change this template, choose Tools | Templates
 *      and open the template in the editor.
 */
package com.ebuss.beans;

/**
 *
 *      @author Tanish
```

```
*/  
  
public class AddingBussRoots {private String sourcecity; private String sourcestate; private  
float distance;  
private String destinationcity; private String destinationstate;private float cost;  
  
        }finally{ try{ ps.close();  
  
private String serviceno;private String busname;  
public String getServiceno() {return serviceno;  
}  
public void setServiceno(String serviceno) {this.serviceno = serviceno;  
}  
public String getBusname() {return busname;  
}  
public void setBusname(String busname) {this.busname = busname;  
}  
public float getCost() {return cost;  
}  
public void setCost(float cost) {this.cost = cost;  
}  
public String getDestinationcity() {return destinationcity;  
}  
  
public void setDestinationcity(String destinationcity) {this.destinationcity = destinationcity;  
}  
public String getDestinationstate() {
```

```
return destinationstate;
}

public void setDestinationstate(String destinationstate) {this.destinationstate =
destinationstate;
}
public float getDistance() {return distance;
}
public void setDistance(float distance) {this.distance = distance;
}
public String getSourcecity() {return sourcecity;
}
public void setSourcecity(String sourcecity) {this.sourcecity = sourcecity;
}
public String getSourcestate() {return sourcestate;
}
public void setSourcestate(String sourcestate) {this.sourcestate = sourcestate;
}
}
```

AdminLoginBeans.java

```
/*
 *      To change this template, choose Tools | Templates
 *      and open the template in the editor.
 */
package com.ebuss.beans;
/**
 *
```

```
*      @author Tanish
*/

public class AdminLoginBeans {private String userName;
private String pwd;
public String getPwd() {return pwd;
}
public void setPwd(String pwd) {this.pwd = pwd;
}
public String getUsername() {return userName;
}
public void setUsername(String userName) {this.userName = userName;
}
}
```

LoginCheck.java

```
import com.ebuss.db.DBConnection;import java.sql.PreparedStatement; import
java.sql.ResultSet;
```

```
/**
 *
 *      @author Tanish
 */

public class LoginCheck {
public String loginCheck(String loginID,String pwd,String role){String uid = null;
PreparedStatement ps =null;ResultSet rs = null;
try{
String query = "select sid,status from logintable where loginname=? and pwd=? androle=?";

ps = DBConnection.getDBConnection(query);ps.setString(1, loginID);
```

```
ps.setString(2, pwd);ps.setString(3, role);
rs = ps.executeQuery();if(rs.next()){
uid = rs.getString(1); System.out.println("User ID:"+uid);
}
if(uid!=null){
package com.ebuss.common;
return uid;
}
else{return "error";}
}catch(Exception e){
System.out.println("Error at:"+e.getMessage());
}finally{try{
rs.close();
ps.close();
}catch(Exception e){}
}

return uid;
}

public String getRegisterStatus(String loginID){ String status = null;
PreparedStatement ps =null;ResultSet rs = null;
try{
String query = "select status from logintable where sid=?";ps =
DBConnection.getDBConnection(query); ps.setString(1, loginID);
rs = ps.executeQuery();if(rs.next()){
status = rs.getString(1); System.out.println("User ID:"+status);
}

}catch(Exception e){
```

```
System.out.println("Error at:"+e.getMessage());
```

```
}finally{try{
```

```
rs.close();
```

```
ps.close();
```

```
}catch(Exception e){}
```

```
}
```

```
return status;
```

```
}
```

```
}
```

```
RegisterDAO.java
```

```
/*
```

```
*      To change this template, choose Tools | Templates
```

```
*      and open the template in the editor.
```

```
*/
```

```
package com.ebuss.common; import com.ebuss.db.DBConnection;import
```

```
java.sql.PreparedStatement;
```

```
/**
```

```
*
```

```
*      @author Tanish
```

```
*/
```

```
public class RegisterDAO {
```

```
public String studentRegister(StudentRegisterBeans srb){
```

```
String msg =null; PreparedStatement ps1=null; PreparedStatement ps2 = null;
```

```
String query1 = "insert into studentregister values(?,?,?,?,?,?,?,?,?,?,?,?,?)"; String
```

```
query2 = "insert into logintable(sid,loginname,pwd,role) values(?,?,?,?);int count=0;
try{
ps1 = DBConnection.getConnection(query1);ps1.setString(1, srb.getSid());
ps1.setString(2, srb.getName()); ps1.setString(3, srb.getLoginid()); ps1.setString(4,
srb.getPwd()); ps1.setString(5, "Infy College"); ps1.setString(6, srb.getAddress());
ps1.setString(7, srb.getUniversity());ps1.setString(8, srb.getCity()); ps1.setString(9,
srb.getState()); ps1.setString(10, srb.getZipcode()); ps1.setString(11, srb.getHallticket());
ps1.setString(12, srb.getSunique()); ps1.setString(13, srb.getEmail()); ps1.setString(14,
srb.getMobile()); ps1.setDate(15, srb.getDate()); ps1.setString(16, srb.getRole()); count
=ps1.executeUpdate(); if(count>0){
ps2 = DBConnection.getConnection(query2);
```

```
String msg =null; PreparedStatement ps1=null; PreparedStatement ps2 = null;
```

```
String query1 = "insert into studentregister values(?,?,?,?,?,?,?,?,?,?,?,?,?)"; String
query2 = "insert into logintable(sid,loginname,pwd,role) values(?,?,?,?);int count=0;
try{
ps1 = DBConnection.getConnection(query1);ps1.setString(1, srb.getSid());
ps1.setString(2, srb.getName()); ps1.setString(3, srb.getLoginid()); ps1.setString(4,
srb.getPwd()); ps1.setString(5, "Infy College"); ps1.setString(6, srb.getAddress());
ps1.setString(7, srb.getUniversity());ps1.setString(8, srb.getCity()); ps1.setString(9,
srb.getState()); ps1.setString(10, srb.getZipcode()); ps1.setString(11, srb.getHallticket());
ps1.setString(12, srb.getSunique()); ps1.setString(13, srb.getEmail()); ps1.setString(14,
srb.getMobile()); ps1.setDate(15, srb.getDate()); ps1.setString(16, srb.getRole()); count
=ps1.executeUpdate(); if(count>0){
ps2 = DBConnection.getConnection(); ps2.setString(4, srb.getRole()); int no = ps2.executeUpdate();
if(no>0){

msg="success";return msg;
```



```
}else{ return "faild";}
}
else{
return "faild";

}
}catch(Exception e){ System.out.println(e.getMessage());e.printStackTrace();
}finally{ try{ ps2.close();
ps1.close();
}catch(Exception ee){ }
}
return msg;

}
}
StudentRegisterBeans.java
Connection(query2);

ps2.setString(1, srb.getSid()); ps2.setString(2, srb.getLoginid());ps2.setString(3, srb.getPwd

private java.sql.Date date;private String role;
public String getSid() {return sid;
}

public void setSid(String sid) {this.sid = sid;
}

public String getRole() {return role;
```

```
}

public void setRole(String role) {this.role = role;
}

public String getAddress() {return address;
}

public void setAddress(String address) {this.address = address;
}

public String getCity() {return city;
}

public void setCity(String city) {

this.city = city;

}

public String getCollegeName() {return collegeName;
}

public void setCollegeName(String collegeName) {this.collegeName = collegeName;
}

public Date getDate() {return date;
```

```
}

public void setDate(Date date) { this.date = date;
}

public String getEmail() {return email;
}

public void setEmail(String email) {this.email = email;
}

public String getHallticket() {return hallticket;
}

public void setHallticket(String hallticket) {this.hallticket = hallticket
}

public String getLoginid() {return loginid;
}

public void setLoginid(String loginid) {this.loginid = loginid;
}

public String getMobile() {return mobile;
}

public void setMobile(String mobile) {this.mobile = mobile;
}

public String getName() {return name;
```

```
}

public void setName(String name) {this.name = name;
}

public String getPwd() {return pwd;
}

public void setPwd(String pwd) {this.pwd = pwd;
}

public String getState() {return state;
}

public void setState(String state) {this.state = state;
}

public String getSunique() {return sunique;
}

public void setSunique(String sunique) {this.sunique = sunique;
}

public String getUniversity() {return university;
}

public void setUniversity(String university) {this.university = university;
}

public String getZipcode() {return zipcode;
```

```
}
```

```
public void setZipcode(String zipcode) {this.zipcode = zipcode;
```

```
}
```

```
}
```

```
AdminModel.java package com.ebuss.model;
```

```
import com.ebuss.beans.AddTrainAndBusPass;import
```

```
com.ebuss.beans.AddingBussRoots; import com.ebuss.beans.AddingTrainRoots; import
```

```
com.ebuss.beans.AdminLoginBeans; import com.ebuss.dao.AdminDAO;
```

```
import java.sql.SQLException;public class AdminModel {
```

```
AdminDAO aDAO = new AdminDAO();
```

```
public String adminLoginCheck(AdminLoginBeans alb) {String msg =
```

```
aDAO.loginCheck(alb);
```

```
return msg;
```

```
}
```

```
public String adminAddBussRoots(AddingBussRoots abr) {String result =
```

```
aDAO.addBussRoot(abr);
```

```
return result;
```

```
}
```

```
public String adminAddTrainRoots(AddingTrainRoots atr){String result =
```

```
aDAO.addTrainRoot(atr);
```

```
return result;
```

```
}
```

```
public String adminAddBusPass(AddTrainAndBusPass atabp){String result =
```

```
aDAO.addBusAndTrainRoots
```

```
(atabp);
```

```
return result;
```

```
}
```

```
}
```

```
StudentModel.java package com.ebuss.model;
```

```
import com.ebuss.dao.StudentDAO;
```

```
/**
```

```
*
```

```
*      @author Tanish
```

```
*/
```

```
public class StudentModel {
```

```
    StudentDAO sDAO = new StudentDAO();
```

```
    public String loginCheck(String lid,String pwd,String role){ String uid = null;
```

```
    uid = sDAO.loginCheckStudent(lid, pwd, role);return uid;
```

```
}
```

```
}
```

```
StudentRegisterController.java
```

```
package com.ebuss.common;

import java.io.IOException;import java.io.PrintWriter;

import javax.servlet.ServletException;import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest; import
javax.servlet.http.HttpServletResponse;import java.text.DateFormat;
import java.text.SimpleDateFormat;

/**
 *
 *      @author Tanish
 */

public class StudentRegisterController extends HttpServlet {

/**
 *      Processes requests for both HTTP <code>GET</code> and <code>POST</code>
methods.
 *
 *      @param request servlet request
 *      @param response servlet response
 *      @throws ServletException if a servlet-specific error occurs
 *      @throws IOException if an I/O error occurs

protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException { response.setContentType("text/html;charset=UTF-
8");PrintWriter out = response.getWriter();

StudentRegisterBeans srb = new StudentRegisterBeans();
srb.setSid(request.getParameter("sid")); srb.setName(request.getParameter("sname"));
srb.setLoginid(request.getParameter("loginid"));
srb.setPwd(request.getParameter("password"));
srb.setCollegeName(request.getParameter("collegename"));
srb.setAddress(request.getParameter("address"));
```

```
srb.setUniversity(request.getParameter("university"));
srb.setCity(request.getParameter("city")); srb.setState(request.getParameter("state"));
srb.setZipcode(request.getParameter("zip"));
srb.setHallticket(request.getParameter("hallticket"));
srb.setSunique(request.getParameter("sunique"));
srb.setEmail(request.getParameter("email"));
srb.setMobile(request.getParameter("mobile"));
String rdate = request.getParameter("rdate");
```

```
SimpleDateFormat dateFormat = new SimpleDateFormat("yy-MM-dd");
java.sql.Date currentDate=null;try{
currentDate =new java.sql.Date(dateFormat.parse(rdate).getTime());

}catch(Exception e){ } srb.setDate(currentDate);
```

```
srb.setRole(request.getParameter("role"));
RegisterDAO rdao = new RegisterDAO();String msg = rdao.studentRegister(srb);
response.sendRedirect("CustomerHome.jsp?msg="+msg);
```

```
}
// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the +sign on
the left to edit the code.">
/**
```

```
*      Handles the HTTP <code>GET</code> method.
*      @param request servlet request
*      @param response servlet response
*      @throws ServletException if a servlet-specific error occurs
*      @throws IOException if an I/O error occurs
```


`*/ @Override`

`protected void doGet(HttpServletRequest request, HttpServletResponse response) throws`

`ServletException, IOException {`

`processRequest(request, response);`

`}`

`/**`

`* Handles the HTTP <code>POST</code> method.`

`* @param request servlet request`

`* @param response servlet response`

`* @throws ServletException if a servlet-specific error occurs`

`* @throws IOException if an I/O error occurs`

`*/ @Override`

`protected void doPost(HttpServletRequest request, HttpServletResponse response) throws`

`ServletException, IOException {`

`processRequest(request, response);`

`}`

`/**`

`* Returns a short description of the servlet.`

`* @return a String containing servlet description`

`*/ @Override`

`public String getServletInfo() { return "Short description";`

`// </editor-fold>`

`();`

Chapter - 7

SYSTEM TESTING

7.1. INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that results in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted simultaneously. The underlying motivation of program testing to affirm software quality with methods that can economically and effectively be applied to both strategic to both large and small-scale systems.

The following are the testing objectives:

- Testing is a process executing a program with the intent of finding an error.
- A good test has a high probability of finding an as yet undiscovered error.
- A successful test is one that uncovers an as yet undiscovered error.

7.2. DESIGN OF TEST CASES AND SCENARIOS

The objective is to learn tests that systematically uncover different classes of errors and do so with a minimum amount of time and effort. Testing cannot show the absence of defects, it can only show that software defects are present.

Unit Testing:

- Interface
- Number of input characters should be equal to number of arguments.
- Parameters and arguments attributes must match.
- Parameters passed should be in correct order.

- Global variable definitions consistent across module.
- If module does I/O.
- File attribute should be correct.
- Open/Close statements must be correct.
- Format specifications should match I/O statements.
- Buffer size should match record size.
- Files should be opened before use.
- End of file condition should be handled.
- I/O errors should be handled.
- Any textual errors in output information must be checked.
- Local Data Structures (common source of errors).
- Improper or inconsistent typing.
- Erroneous initializing or default values.
- Incorrect variable names.
- Inconsistent data types.
- Overflow, underflow, address exception.
- Boundary conditions and independent paths.
- Error handling.
- Error description unintelligible.
- Error noted does not correspond to error encountered.
- Exception condition processing incorrect.

Integration Testing:

Module integrated by moving down the program design hierarchy. Can use depth first or breadth first top down integration verifies major control and decision points early in design process. Top-level structure tested most.

Depth first implementation allows a complete function to be implemented, tested and demonstrated and does depth first implementation of critical function early. Top down integration forced (to some extent) by some development tools in program with graphical user interfaces. Begin construction and testing with atomic modules (lowest level modules). Bottom up integration testing as its name implies being construction and testing with atomic modules. Because modules are integrated from the bottom up, processing required for module subordinate to a given level is always available and the need for stubs is eliminated.

Top-Down Integration: Top-Down Integration testing is an incremental approach to construction of program structure. Modules are integrated by moving downward through the computer hierarchy, beginning with the main control module.

The Top-Down integration process is performed in the following five steps:

- The main control module is used as a test driver and subs are substituted for all the components directly subordinate to the main control module.
- Depending on the integration approach selected, subordinate stubs are replaced one at a time with actual components.
- Tests are conducted as each component is integrated.
- On completion of each of test, another stub is replaced with the real components.
- Regression testing may be conducted to ensure that new errors have not been introduced.

Bottom-Up Integration: Bottom-Up Integration testing as its name implies, being construction as testing with atomic modules because components are integrated from the bottom up, processing required for components subordinate to a given level is always available and for stubs is eliminated.

- The Bottom-Up Integration is performed in the following four steps:
- Low-Level components are combined into clusters that perform a specific software stub function.
- Driver is written to coordinate test case input and output.
- The cluster is tested.

7.3. TEST CASES AND SCENARIOS

A TEST Plan is a systematic approach to test a system as a machine or software. The plane typically contains a detailed understanding of what the eventual work flow will be UNT LEVEL plan for JOB ad.

Test Report and Results:

TEST REPORT NO	1
PROJECT NAME	Integrated Ticket system
MODULE NAME	STUDENT REGISTRATION
FORM NAME	REGISTRATION
UNIT NAME	USER NAME & DETAILS OF STUDENTS
TEST RESULTS	ON CLICKING SUBMIT BUTTON AFTER PROVIDING YOUR PARTICULAR ACCOUNT WILL BE CREATED AND YOU CAN LOGIN TO THE SITE.

Test Plan 1:

Project Name: Integrated Ticket system Module Name: STUDENT Module.

Unit Name: User Name.

Test Result: The User Name Textbox is tested and verified. Test Plan (Unit Module/Test Integration)

Test plan Integrated Ticket system Unit ID: LOGIN.

Test Case ID: Login Page. Test Type: Unit Case.

Form Name: LOGIN. Base Table: Registration. **Purpose:**

Registration table is used for store the details of registered members details and along with their Username and Password. By using these details the administrator can perform the operations.

7.4. TEST CASE DESCRIPTION

- Test Data**

SNO	INPUT SPECIFICATION	EXPECTED RESULT/OUTPUT
1.	Column Name: User Id Valid Input: If the User id valid along with password then the form will be navigated to allotted page.	Valid Output: If the User Name and Password are correct then form navigation to home page. Invalid Output: If the User Name is incorrect error message is displayed as "User
	Invalid Input: If the User Name should be reentered.	Invalid" and it will ask for User ID & Password.

TEST COMPLETION CRITERIA

When expected results match the actual results performing the test, the test is considered to be completed.

Validation Testing: Validation succeeds when system functions in a manner that can be reasonably by the end-user. This is achieved through a series of black-test that demonstrate with requirements.

There are two tests for system conduction for the system validation:

- **Alpha Testing:** A customer conducts it at the developer's site. The software is used in a nature setting with the developer "looking over the shoulder" of the user and recording errors and usage problems.
- **Beta Testing:** This test is conducted at one or more users sited by the end user of the software. Here the developer generally not presents. Therefore, the beta test in a "line" application of the software in an environment that can't be controlled by the developer.
- **System Testing:** Once the software product is developed, it is thoroughly tested and it is delivered to the users. Now, it has to be tested by developing it on the system i.e., to what the given software is comfortable to the environment. The software engineer should consider these issues during early stages of software development to release himself from the problems which are encountered after completion of the software. Hence, the tests conducted to ensure that the software is comfortable with the system, where it is deployed is referred as "System Testing".

➤ **Recovery Testing :** It is often a nature fact that certain errors may corrupt the system or may make the system not to function properly to a stipulated period of time. Hence, recovery testing is the process which given software id exposed to failures and it is tested to see its recovery capabilities.

Usually the recovery can be of two types:

- ✓ Automatic through human intervention.
- ✓ Recovery through human intervention.

During automatic recovery the software itself recovered. Sometimes requires certain addition support like system restart, reutilization, data recovery, etc., for tis normal execution when the system requires human intervention in order to recover from such recovery is referred as recovery through human intervention. Here, mean-time-to- repair is a value which is calculated to ensure that the software gets recovered within acceptable span of time.

➤ **Security testing:** Security plays a major role especially in that software. Which are made to deal with highly confidence data. For these systems, often several hackers try their to break the security of the system and acquire the confidential data for their foolish requirements. Hence, for these systems, security measures should be given vital importance. For this purpose, the testers themselves disguise into hackers and perform series of attempts to breaks the security of the given software can be truly judged.

➤ **Stress Testing:** Stress testing is usually performance to check the limits of the system i.e., to what extend the system can resists the abnormal conditions. Hence, the system is tested by providing abnormal resources in different proportions. During stress testinga system can be.

- ❖ Providing the excess values of data in different proportions to check its memory management capabilities i.e., how efficiently the system manages the data which is more than its capability.
 - ❖ Exposed to certain programs demanding large memory and resources not available with the current system.
 - ❖ Providing too many interrupts during a specific period of time.
 - ❖ Providing with too many inputs through it can survive only few input
- **Performance testing:** Performance testing is essential to ensure the given software performance to the execution when it is implemented on the system. Hence, in this case it is only the software considered but also the hardware in which it is deployed. Here, the performance testing is combined with the stress testing cases to check the internal aspects such as resources utilization and various other instances.

7.5. TEST CASES

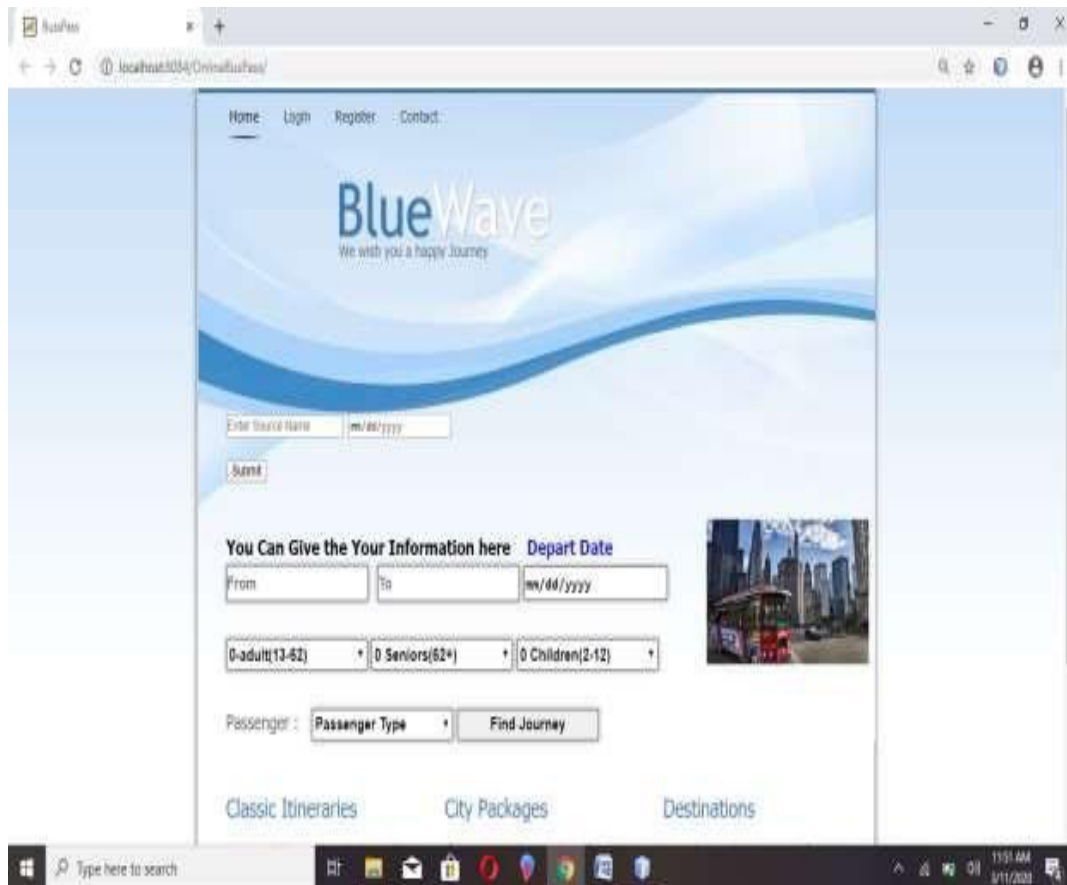
Test Case Number and Test ExecutionSteps Name	Expected Result	Test Result	Type of Test
Check Username And Password	Run application and click Login button If the User Name and Password are correct then navigate user from home page.	PASS	Functional Testing
Check Payment is done or not	Run application and click submit button in Bus Pass Registration Page Your payment is done successfully	PASS	Functional testing
Check Renewal is done or not	Run application And check status Your Status is shown	PASS	Functional testing
Check Feedback is submitted or not	Run application Click on Submit button Your feedback is submitted successfully	PASS	Functional testing
Check whether the values are updated or not	The Admin will view by logging into his/her website Updated successfully	PASS	Functional testing

Chapter - 8

RESULTS

SCREEN SHOTS

Homepage of the Bluewave



View The bus and fare

The screenshot displays the BlueWave online bus pass system interface. The header features the BlueWave logo and the tagline "We wish you a happy Journey". Below the header, there are navigation links: "Back", "Redefine", and "Contact". The main content area is titled "Basic Route" and contains a table with the following data:

S.No	Bus No	Bus Name	From	Departure	Distance	Cost	Available
1	CAUSR2394	GMVLT	Guntur	Vijayawada	40.0	50.0	10
2	CAUSR2845	GMTCR	Guntur	Chirala	40.0	50.0	30
3	CAUSR3888	GMTCR	Guntur	Chiluvula	40.0	50.0	30

Below the table, there are three sections: "Classic Itineraries", "City Packages", and "Destinations". Each section contains a brief description of the service.

Classic Itineraries
Private guided tours meticulously designed by the largest online Blue Wave travel agency and our expert operators to cater to your budget and specific preferences.

City Packages
If you are on a tight travel plan and worried with what to see, you can follow these carefully designed single city packages to make the best use of your time to visit the major highlights.

Destinations
Comprehensive Blue Wave travel information on destinations, activities, transportation, weather, maps to help you achieve an unforgettable trip.

Planning Journey without Login



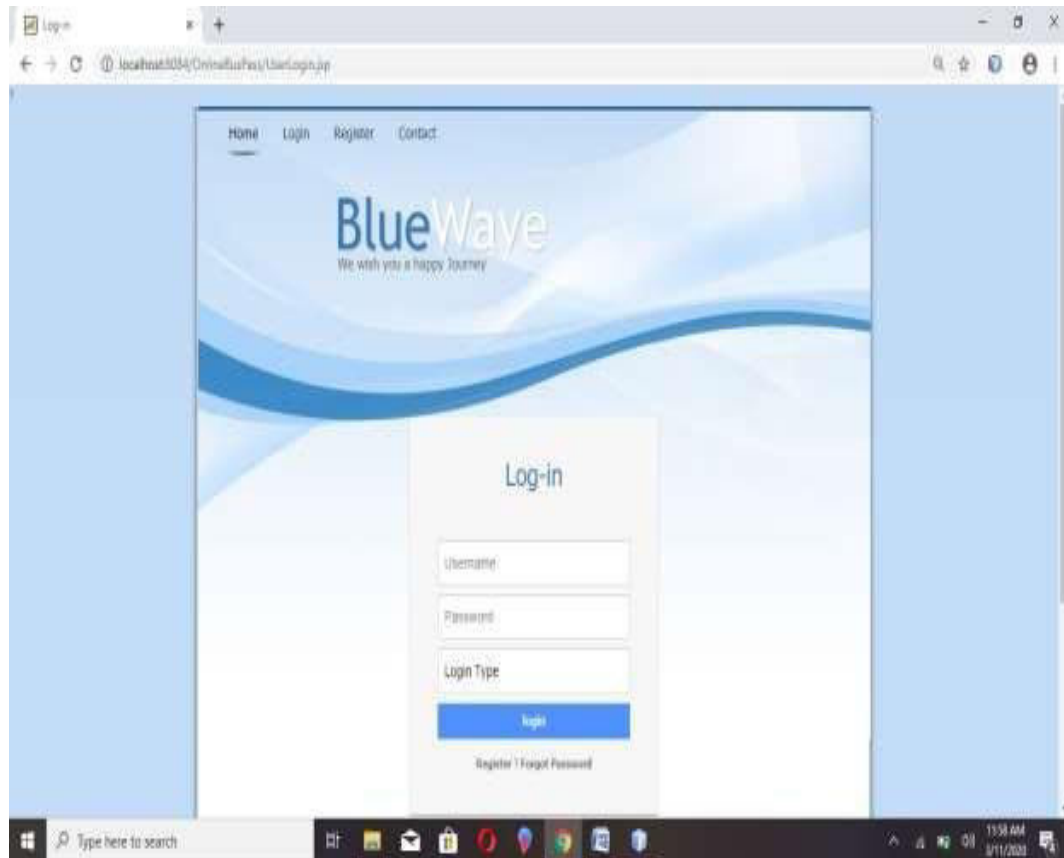
Showing available Data

The screenshot shows the Blue Wave Customer Register Form. The form is titled "Blue Wave Customer Register Form". It contains several input fields for customer information:

- CUSTOMER ID: 100000000
- WRITE HERE YOUR NAME: Customer Name
- LOGIN ID: This is your Login ID
- PASSWORD: Login Password
- ADDRESS: Address
- INSTITUTION: Studied Community
- LIVING CITY: Currently Living City
- STATE: Currently Living State
- ZIP CODE: Zip Code
- HOME CONTACT NUMBER: Contact Number
- YOUR NUMBER: Guest (Unique Number)
- EMAIL ID: Valid Email
- MOBILE NUMBER: Mobile Number
- REGISTER DATE: 2020-03-11
- ENTER FOR ID: Login Type

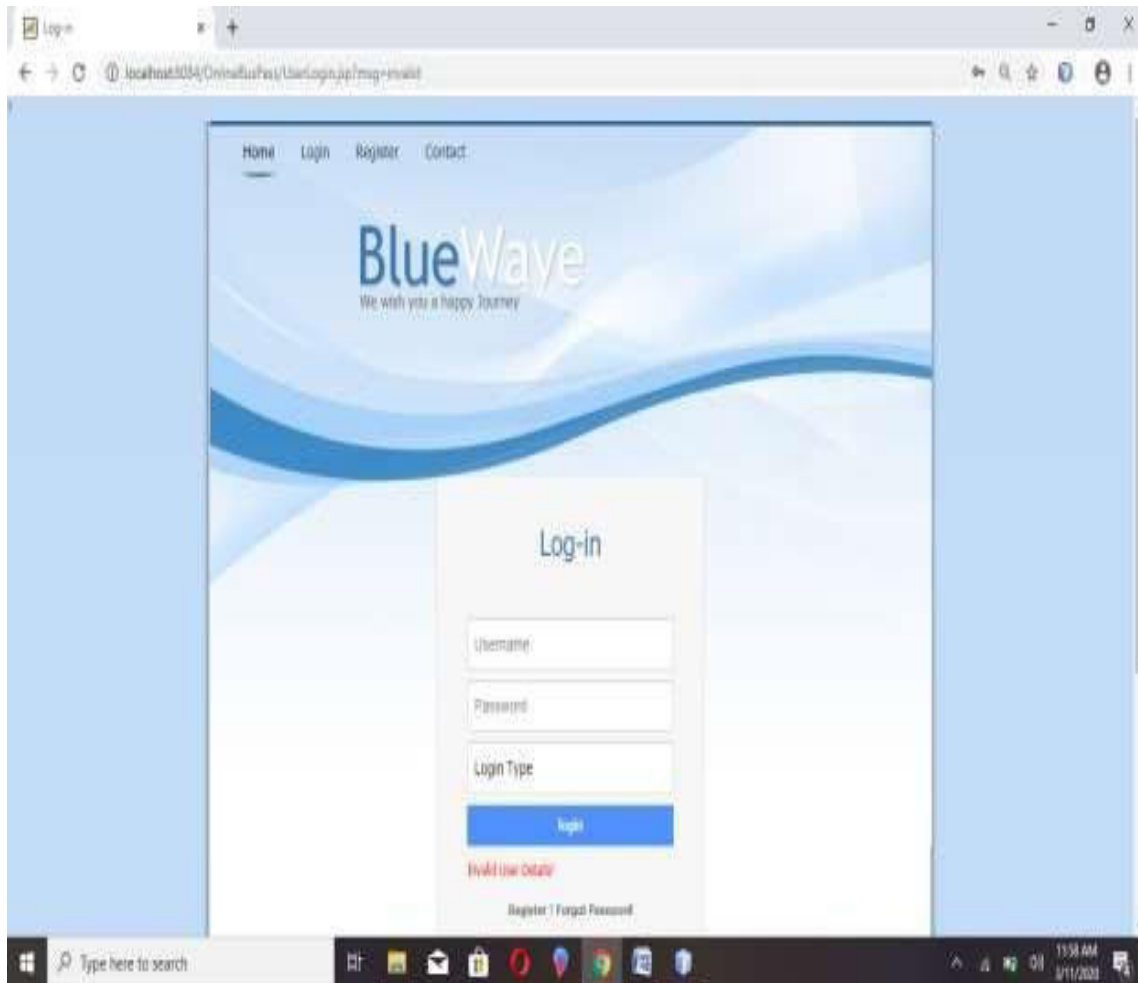
At the bottom of the form, there is a green button labeled "Submit Form".

Registration Form of the Customer

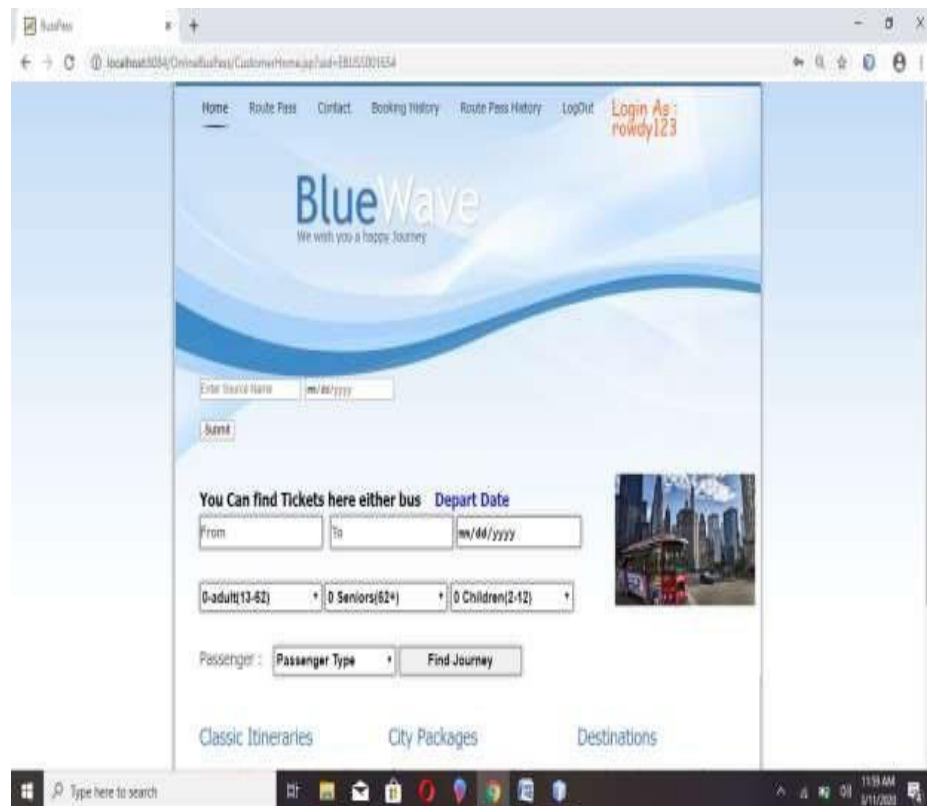


The screenshot shows a web browser window displaying the login page of the BlueWave online bus pass system. The browser's address bar shows the URL `localhost:8084/OnlineBusPass/UserLogin.jsp`. The page features a blue and white wavy background. At the top, there is a navigation menu with links for [Home](#), [Login](#), [Register](#), and [Contact](#). The main heading is "BlueWave" with the tagline "We wish you a happy Journey". Below this, there is a "Log-in" section with three input fields: "Username", "Password", and "Login Type". A blue "Login" button is positioned below these fields. At the bottom of the login section, there are links for "Register" and "Forgot Password". The Windows taskbar at the bottom shows the search bar, task view button, and several application icons, with the system clock indicating 11:58 AM on 9/11/2021.

Customer Login page



Login failed Customer Home Page



Logged Customer Book

The screenshot displays a web browser window with the URL `localhost:8084/OnlineBusPass/FindRoute.jsp`. The page features a blue-themed header with the "BlueWave" logo and the tagline "We wish you a happy Journey". Navigation links for "Back", "Redefine", and "Contact" are visible. Below the header, a table titled "Bus Route" lists available routes. The table has columns for S.No, Bus No, Bus Name, From, Departure, Distance, Cost, Available, and Book Now. A single route is listed: S.No 1, Bus No 123456789, Bus Name RAJESH, From Guntur, Departure 10:30am, Distance 40.0, Cost 10.0, and Available 10. A "Book Now" button is present in the last column. Below the table, there are three sections: "Classic Itineraries" (describing guided tours), "City Packages" (describing single city packages), and "Destinations" (describing comprehensive travel information). The footer includes a copyright notice "Copyright © 2020 My Company" and a Windows taskbar at the bottom showing the time as 12:01 PM on 8/11/2020.

S.No	Bus No	Bus Name	From	Departure	Distance	Cost	Available	Book Now
1	123456789	RAJESH	Guntur	10:30am	40.0	10.0	10	Book Now

Filling the Traveling Persons Names

The screenshot shows a web browser window with the URL `localhost:8084/OnlineBusPass/BusSeatBook.jsp?busServiceNo=CAUS02390&busname=GUINVI&busSource=Guinva&busDestination=Vijayawada&busDistance=40...`. The page features a blue header with the 'Blue Wave' logo and the tagline 'we wish you a happy journey'. The main content area is a form for entering passenger details. It includes a 'Login User: rowdy123' and 'Customer ID: EBUSS001654' at the top. Below this, there's a section for 'Adults Data Write Here' with a 'Ticket Fare Cost: 45.00'. The form has fields for 'Name', 'Age', 'Male' (a dropdown menu), and 'Passport Number'. There's also a section for 'Children Data Write Here' with a 'Ticket Fare: 22.50' and a 'Child Name' field. At the bottom, there are fields for 'Enter Mobile Number' and 'Email', and a 'Submit' button. The Windows taskbar is visible at the bottom of the browser window.

Blue Wave
we wish you a happy journey

Login User: rowdy123 Customer ID: EBUSS001654

Adults Data Write Here Adult Fare Cost: 45.00

1. Name Age Male Passport Number

Children Data Write Here Ticket Fare: 22.50

1. Child Name

Total Fare for Account is = 67.50

Enter Mobile Number Email

Submit

Payment Gateway

The screenshot shows a web browser window displaying the BlueWave Online Payment Gateway. The browser's address bar shows a local URL. The page has a blue header with the 'BlueWave' logo and the tagline 'We wish you a happy Journey'. Navigation links for 'Home', 'Contact', 'LogOut', and 'Login As: rowdy123' are visible. The main content area is titled 'Online Payment Details' and contains a form with the following fields: 'Credit Card Number' (with a 'CW' label), 'Expiry Date' (with 'mm' and 'yyyy' sub-fields), and 'Email ID'. A 'Make Payment' button is located at the bottom right of the form. Below the form, there are links for 'Classic Itineraries', 'City Packages', and 'Destinations'. The Windows taskbar at the bottom shows the search bar and several application icons, with the system clock indicating 12:03 PM on 8/11/2021.

Home Contact LogOut Login As: rowdy123

BlueWave
We wish you a happy Journey

Online Payment Details

Credit Card Number: CW

Expiry Date: mm yyyy

Email ID:

Make Payment

[Classic Itineraries](#) [City Packages](#) [Destinations](#)

Final Ticket For The Customer

Blue Wave Journey Ticket Details Onward

Ticket Number	2661976	Service Code	CAU502390
Service	GUNVI	Date Of Journey	2020-03-11
From	Guntur	TO	Vijayawada
No Of Seats	2	Depart On	2020-03-11
Booking Date	2020-03-11	Total Seats	2
Email	pavani@gmail.com	Mobile	8888888888

Passenger Names

Names	Pavani
Age	10
Gender	Female
Seat Number	01

No Seniors Child Names

Names	Pavani
Seat Number	02

Total Fare: 67.5

ID Proof Note : During bus journey, one of the passenger on an e-ticket appears should carry the original identity card such as: Driving License, Election Card, Ration Card, Photo ID card issued by

Finding Route Pass

The screenshot shows a web browser window with the URL `localhost:8084/OnlineBusPass/RoutePass.jsp`. The page has a navigation bar with links: Home, Route Pass, Contact, Booking History, Route Pass History, and Logout. A user is logged in as 'rowdy123'. The main header features the 'BlueWave' logo with the tagline 'We wish you a happy Journey'. Below this, a section titled 'You Can Give the Your Information here' contains a form with fields for 'From', 'To', 'Depart Date' (with a date picker set to 'mm/dd/yyyy'), 'Select Pass Type--' (a dropdown menu), and 'Student' (a text input). A 'Find Route' button is located to the right of the 'Student' field. To the right of the form is a small image of a bus. Below the form, there are three columns of text: 'Classic Itineraries' (describing private guided tours), 'City Packages' (describing light travel plans and city packages), and 'Destinations' (describing comprehensive travel information). The Windows taskbar at the bottom shows the search bar and various application icons.

Routes pass for the custome

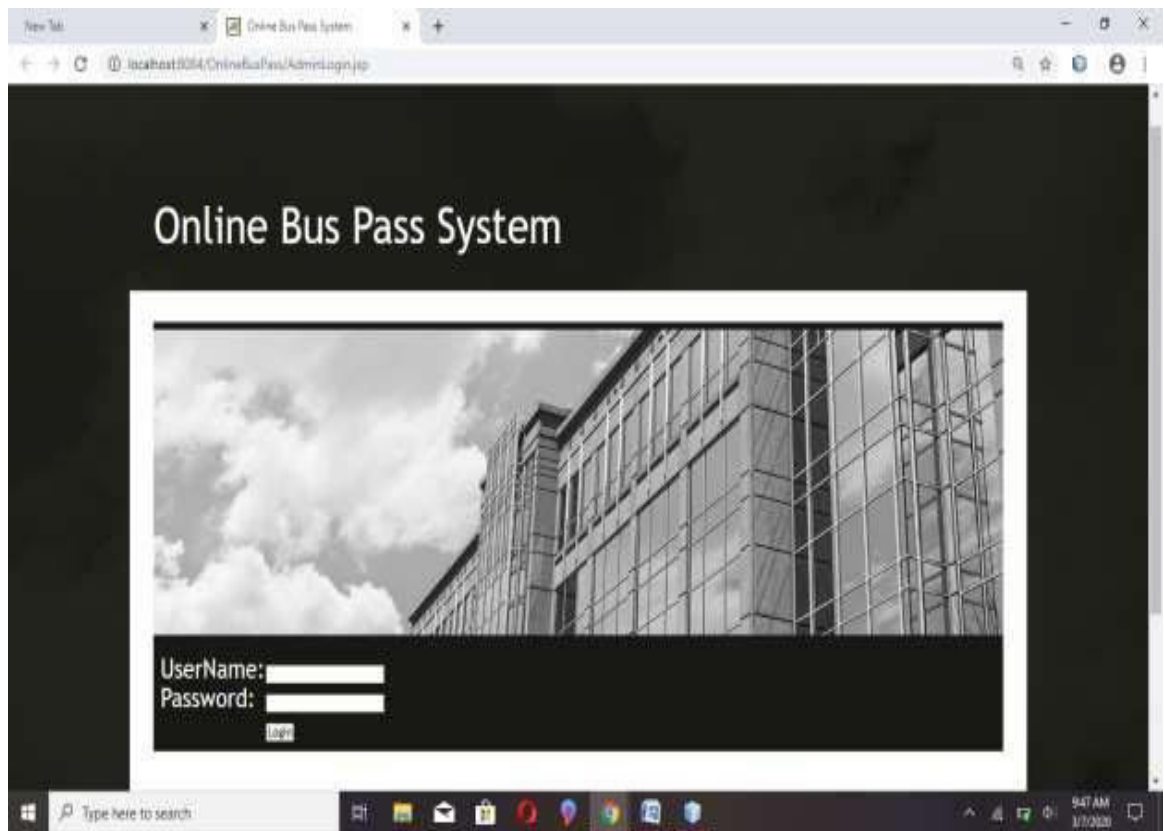
Route Pass Details

Route Pass ID	001538002102
User Name	sandy123
Starting From	Guntur
Destination	Kannam
Pass Start Date	2020-03-11
Type Of Route Pass	Monthly
Total Cost	\$:3600.0
Date Of Booking	2020-03-11
Get Down AT	Vijayawada
Walk Distance	30
Catch The Bus	GNT-VJA
Get Down AT	Kannam
Walk Distance	75
Bus Number	28247

Submit

Classic Itineraries City Packages Destinations

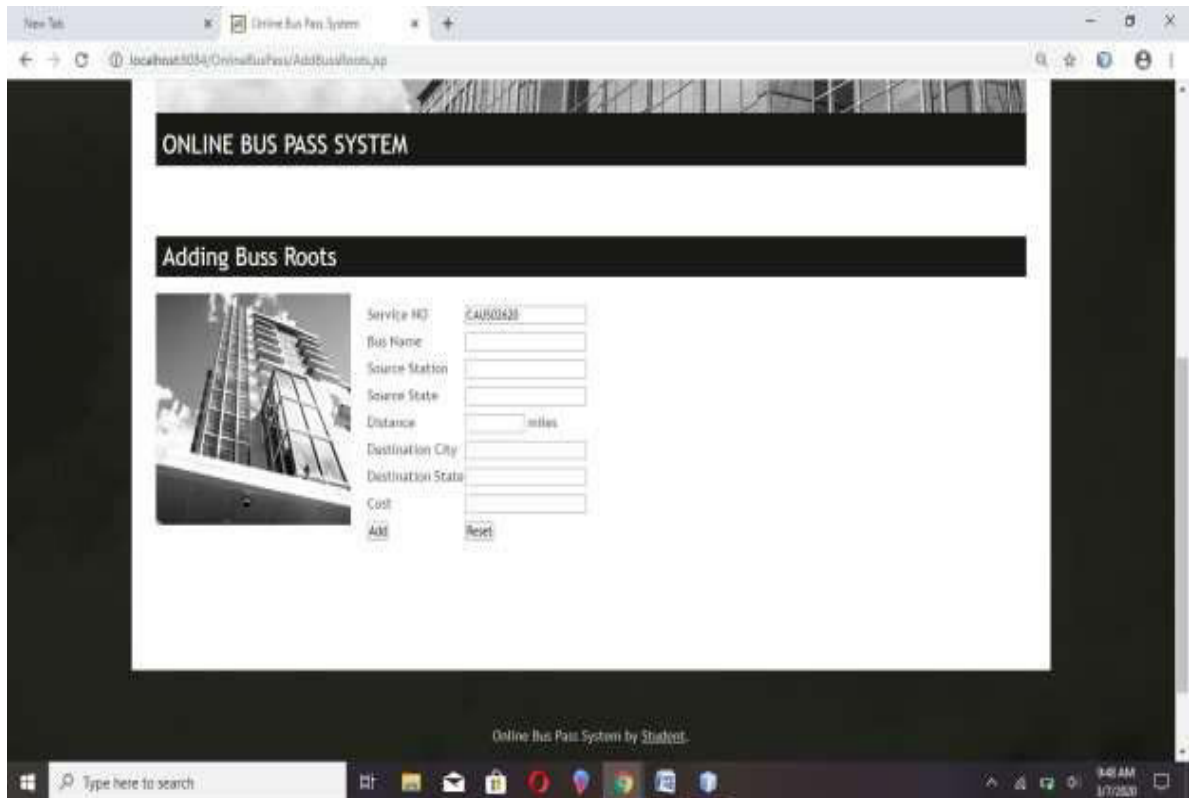
Admin Login page



Admin Homepage



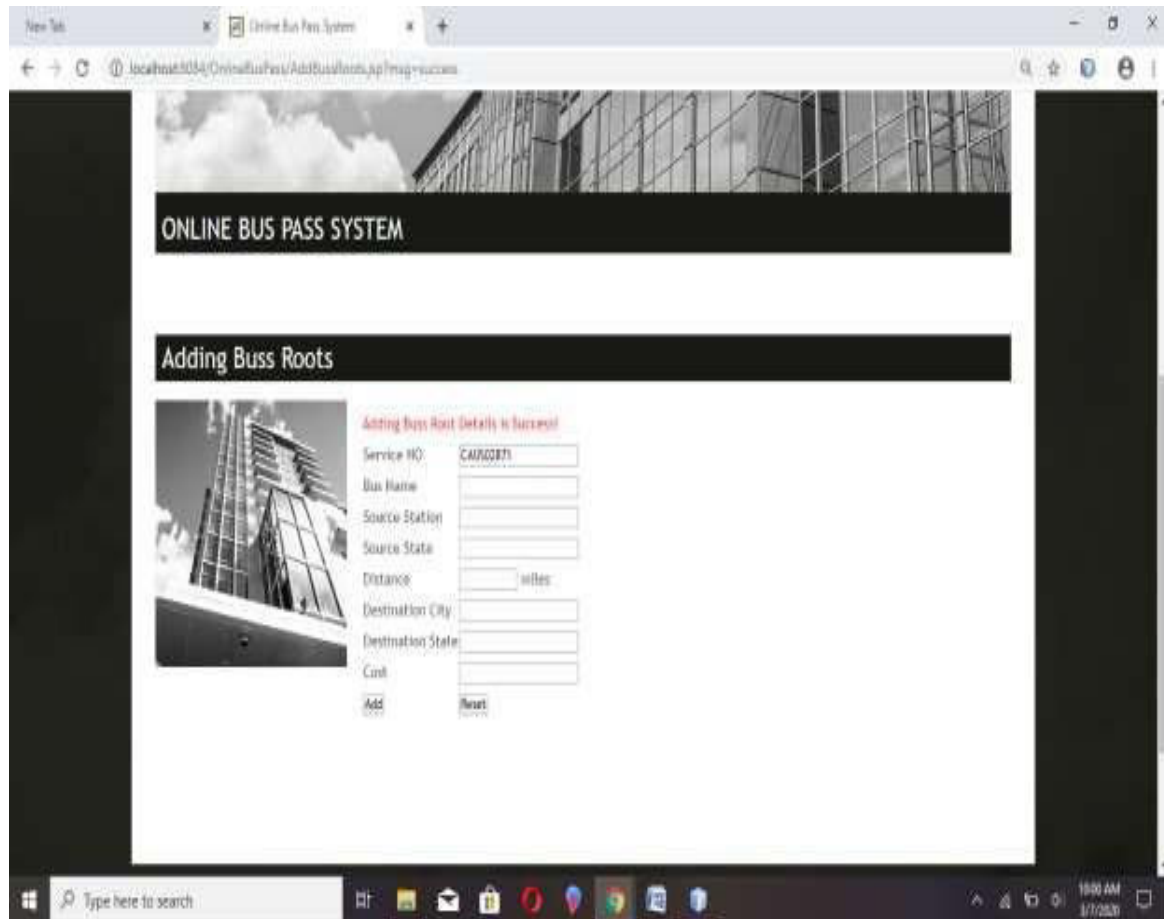
Adding Bus route by Admin



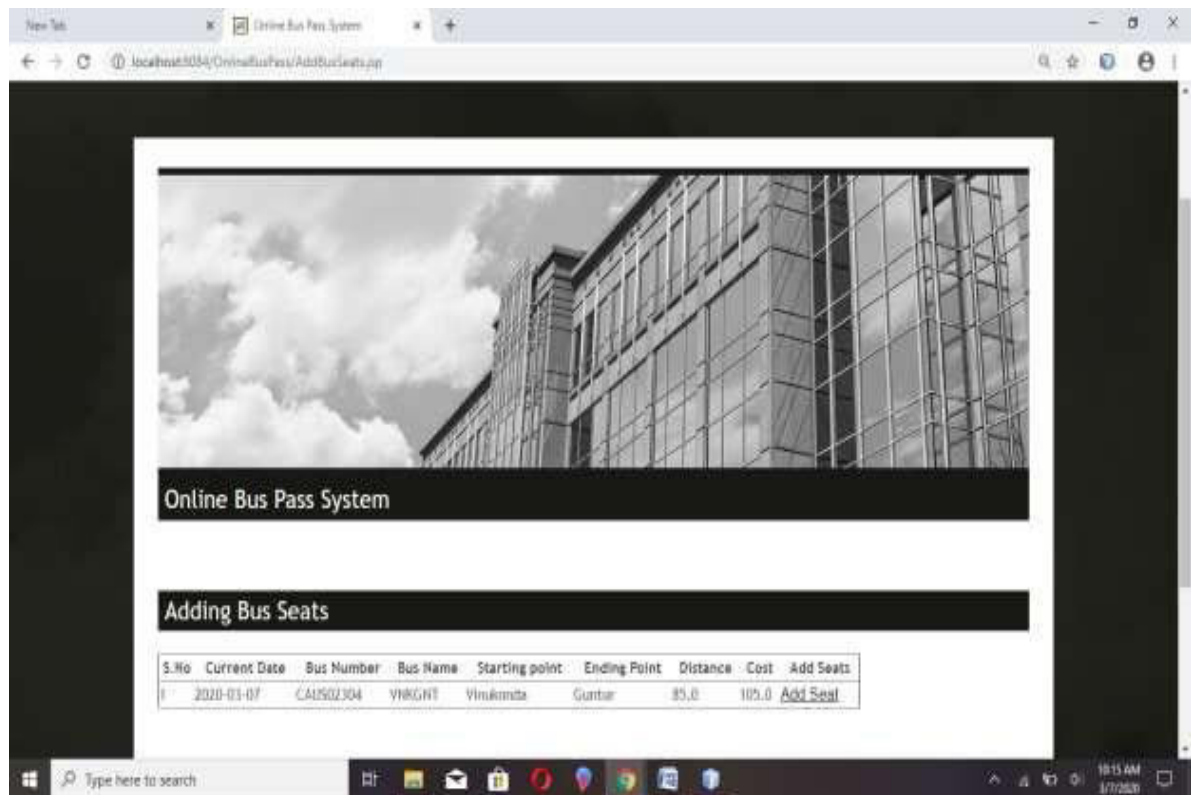
The screenshot shows a web browser window with the title 'Online Bus Pass System'. The address bar shows the URL 'localhost:8084/OnlineBusPass/AddBusRoute.jsp'. The page has a header 'ONLINE BUS PASS SYSTEM' and a sub-header 'Adding Buss Roots'. Below the sub-header is a form with a small image of a bus on the left. The form fields are: Service No (with value 'CAUR00430'), Bus Name, Source Station, Source State, Distance (with unit 'miles'), Destination City, Destination State, and Cost. At the bottom of the form are 'Add' and 'Reset' buttons. The footer of the page says 'Online Bus Pass System by Student'. The Windows taskbar at the bottom shows the search bar and several application icons, with the system clock displaying '9:48 AM 1/7/2020'.

Service No	CAUR00430
Bus Name	
Source Station	
Source State	
Distance	miles
Destination City	
Destination State	
Cost	

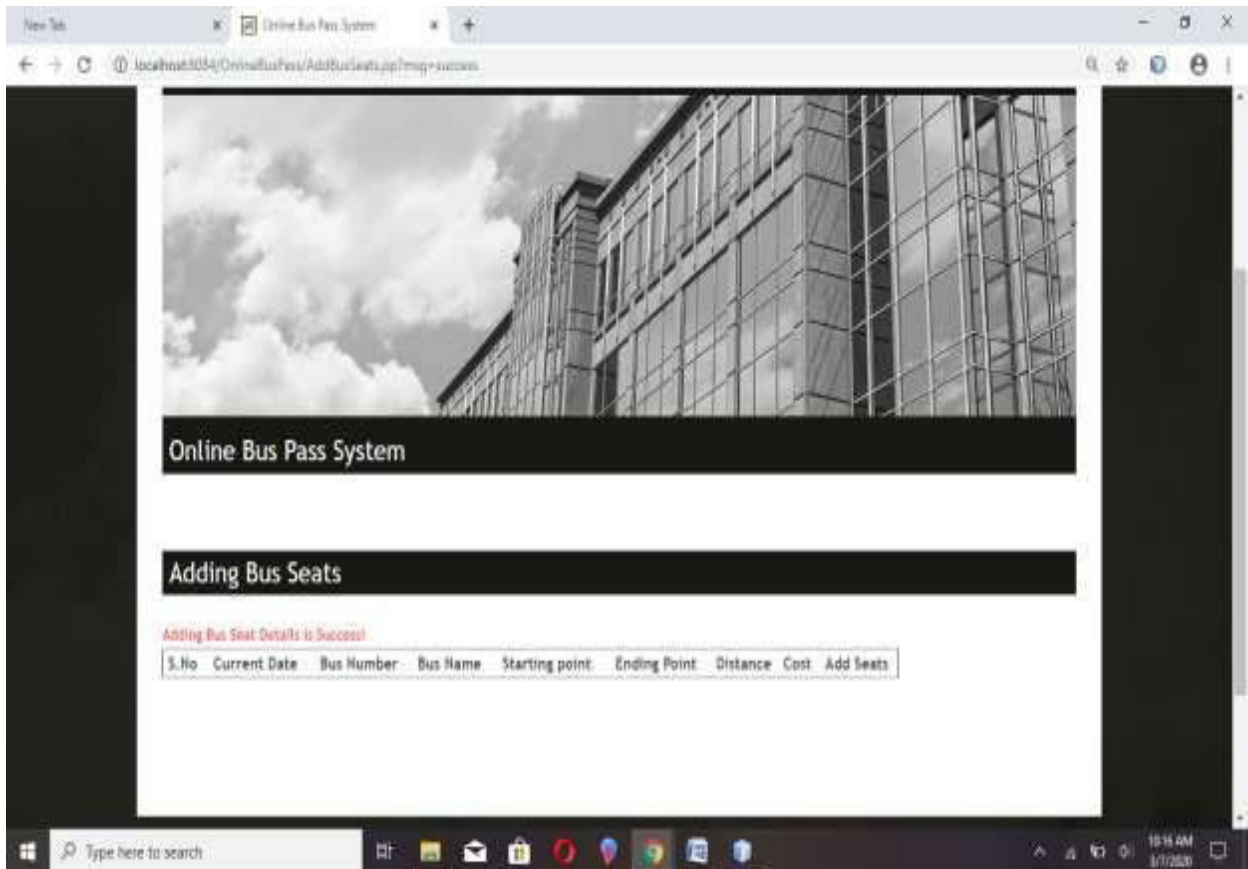
Buss Route Added By Admin Success



Add for Seats in bus here Admin can click the add seats to the particular bus routin that buss initially the 38 seats will be added for next 60 days.



Added Bus Seat Success

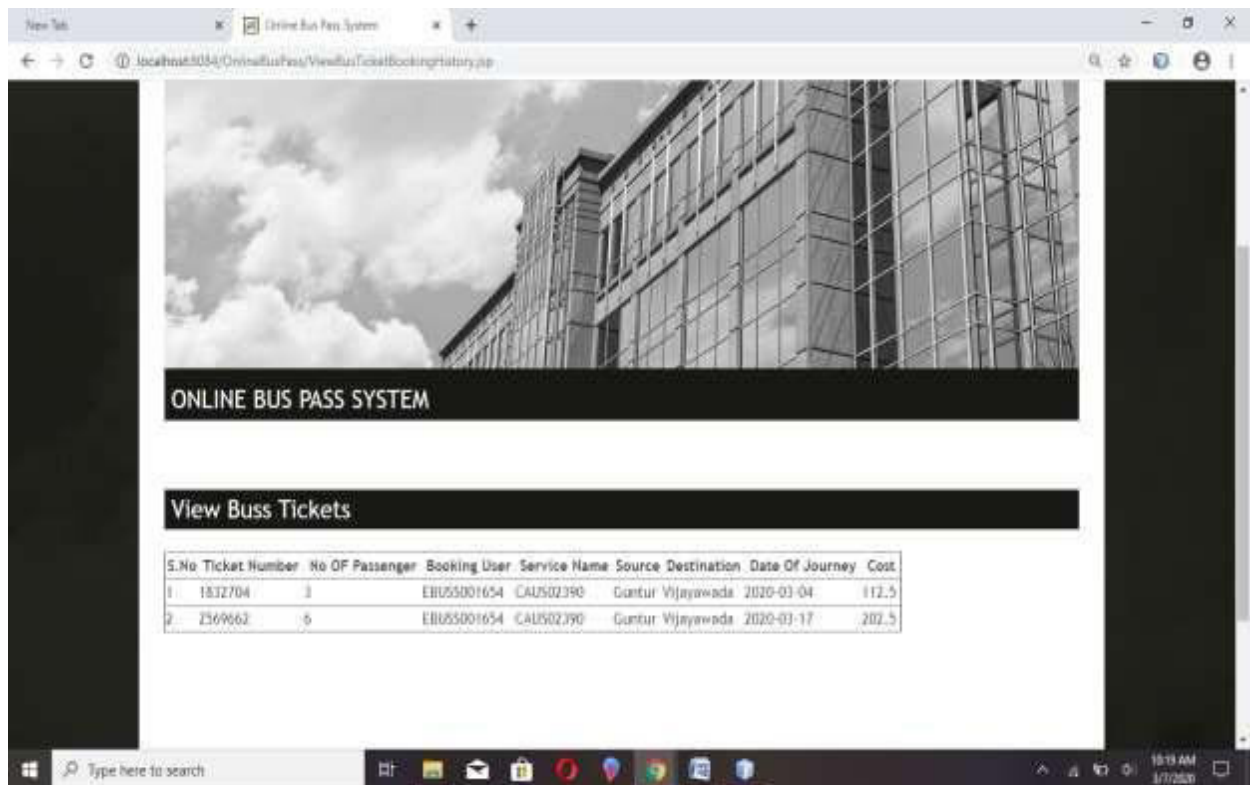


Adding Route Pass data

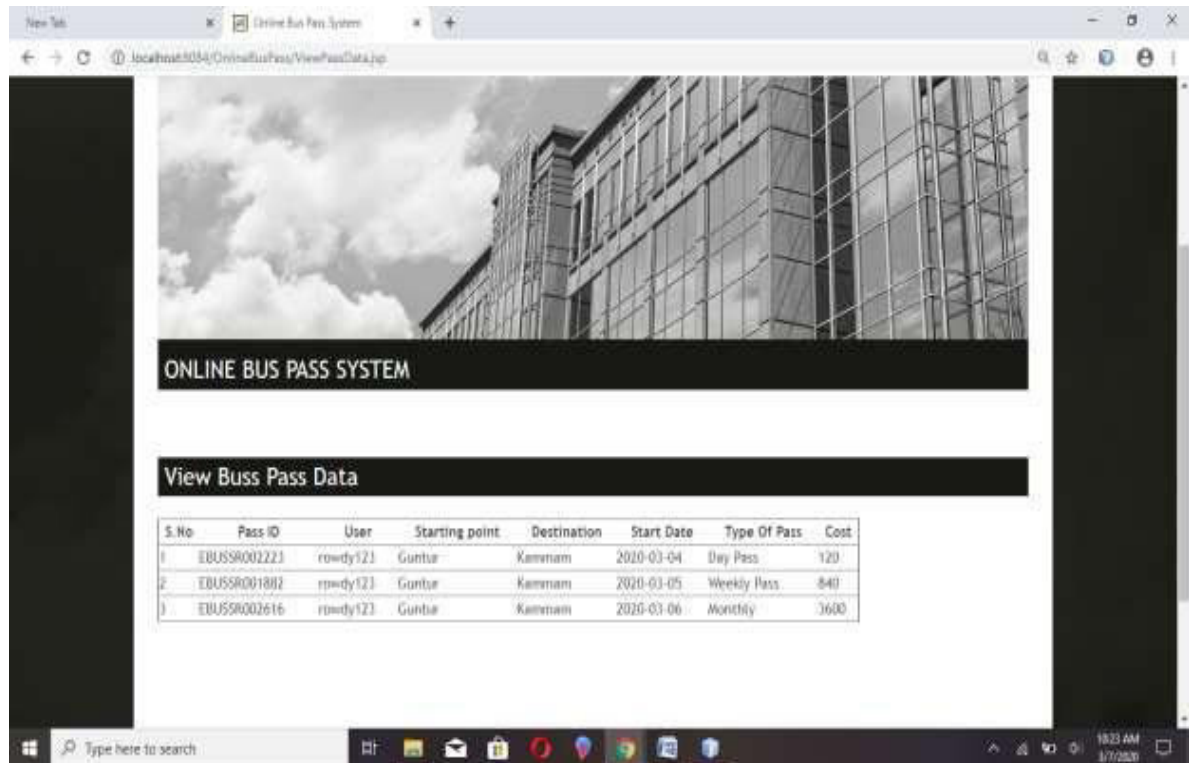
The screenshot shows a web browser window with the title 'Online Bus Pass System'. The address bar shows the URL 'localhost:5034/OnlineBusPass/AddBusAndTrainPass.jsp'. The page content is titled 'Adding Multiple Bus Pass Details'. Below the title is a form with the following fields:

Source Stop Name	<input type="text"/>
Bus Stop	<input type="text"/>
Get In Bus Number	<input type="text"/>
First Cost	<input type="text" value="\$"/>
Get Down at Busstop Name	<input type="text"/>
Walk Distance	<input type="text"/>
Second Bus Station Name	<input type="text"/>
Bus Name	<input type="text"/>
Second Cost	<input type="text" value="\$"/>
Get Down Bus Stop Name	<input type="text"/>
Walk Distance	<input type="text"/>
Get In Bus Number	<input type="text"/>
Third Cost	<input type="text" value="\$"/>
Get Down Destination	<input type="text"/>
Total Cost	<input type="text" value="\$"/>
<input type="button" value="Add"/>	<input type="button" value="Remove"/>

.View Bus Tickets by admin



View Route Pass by Admin



ONLINE BUS PASS SYSTEM

View Buss Pass Data

S.No	Pass ID	User	Starting point	Destination	Start Date	Type Of Pass	Cost
1	EBUSSR002223	rowdy123	Guntur	Kannam	2020-03-04	Day Pass	120
2	EBUSSR001892	rowdy123	Guntur	Kannam	2020-03-05	Weekly Pass	840
3	EBUSSR003616	rowdy123	Guntur	Kannam	2020-03-06	Monthly	3600

Chapter - 9

CONCLUSION AND FUTURE ENHANCEMENT

9.1. CONCLUSION

The package was designed in such a way that future modifications can be done easily. The following conclusions can be deduced from the development of the project.

- Automation of the entire system improves the efficiency.
- It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorized users depending on their permissions.
- It effectively overcomes the delay in communications.
- Updating of information becomes so easier.
- System security, data security and reliability are the striking features.
- The system has adequate scope for modifications in future if it is necessary.

This application avoids the manual work and the problems concern with it. It is an easy and fast way to access the updates information.

9.2. FUTURE ENHANCEMENT

In future we will include various passes like Employees general passes, bus pass for specially challenged people and so on. We will also include bus timing along with routes and fares. We could be making an Android and iPhone application of this application so that the users can download this application and use this easily

BIBLIOGRAPHY

REFERENCES

- The Complete Reference Java

- Herbert Schildt

- www.java.com
- www.oracle10g.com
- www.google.com