**1.Introduction:**

As a team, we have got an opportunity to choose our final semester project as "ONLINE VEHICAL PARKING". Vehicle traffic congestion is a worldwide problem. In recent years, efforts have been made to introduce a method to reduce parking problems such as congestion, accidents and hazards.

The main goal of this project is to build an online system which helps variety of occasions turn up when we visit various public places like Shopping malls, 5-star and 7-star hotels, multiplex cinema halls, etc. The difficulty we encounter at these places is finding the availability of parking space.

Most of the times we need to traverse through multiple parking slots to find a free space for parking. The problem becomes more tedious if the parking are multi- stored. Thus the problem is time-consuming. This situation calls for the need for an automated parking system that not only regulates parking in a given area but also keeps the manual intervention to a minimum.

Our proposed system presents an Autonomous car parking that regulates the number of cars that can be parked in a given space at any given time based on the parking space availability. When a car arrives at the entrance, it will be stopped at the main gate and provide a parking slip. This parking slip provide the full details about the vehicle parking. Like parking slot number, vehicle number, vehicle type, date, In time, amount for parking etc. .This will help the driver for easy parking to the given available space.

**1.2Objective & Scope of the project:**

**Objective:**

The main purpose of the project is to automate the existing system of manually go and check for slot and book it .

**Scope:**

The “Online Vehical Parking” is new type of web application. Which has developed to automate the manual work and maintain the records. It saves the time perform well. This system gives opportunity to the User to check there parking slots and other related details.

**1.3 Problem statement:**

* Parking facilities has become a huge problem.
* There is lack of proper free spaces for parking due to increased unplanned housings in many places of the capital.
* There has been increase in the number of vehicles, but without sufficient parking spaces.
* Such growing number of small vehicles especially motorcycles and micro buses have created mess in the city including the increase in traffic jam.
* Another challenge due to the increased number of vehicles is undisciplined driving, which created obstacle for the traffic management system.
* To improve all these, there is a need to create enough parking spaces.

2. **System Analysis:**

**2.1Existing System**

The existing system does not have any facility to Book the slot and find about the near by service stations. User have to come to the parking area to see any slot is empty or no and even he don’t get the service stations so easy that his problems would be solved soo now days as the modernisation the people need the services too early and also they need the good servies and also it doesn’t consume much time hear so this web application will help everyone.

* + 1. **Problems in Existing System:**
* Existing parking system consist of many disadvantages. The system only provide the Entrance to the parking area not provide the parking slot
* The car owner search the parking slot after entering to the parking area. It will take time to find the slot
* The car owner have no idea about the parking area is fill or not. His responsibility to find the parking slot and park his vehicle safely
* The owner need to remain the parking slot for his vehicle parked.
* Difficult to find his vehicle from a large area of parking after complete his shopping
* Safety for the vehicle is less

**2.2Proposed System**

In the proposed system, The admin side from the parking entrance provide the parking slip to the vehicles owner it provide the full details like Vehicle type, Vehicle number , Date, In Time, Parking Slot..etc. Provide Different Parking areas for different Vehicle type, four wheelers need more area for parking than two wheelers. It will helps to park more vehicles. The admin side know about the parking status. If the parking area is fill admin block the entrance. Admin has to check the available slot in any time. No need to remained the parking slot. It is to be printed in the parking slip. It’s easy to find the vehicle from a large area of parking using the parking slip. Parking will take less time and safety is high.Validate the Admin before entering to the application.Admin Can take the detailed report of parked vehicle.

**2.2.1 Advantages of Proposed System:**

* User friendliness is provided in the application with various controls.
* The system makes the overall project management much easier and flexible.
* It can be accessed over the internet.
* Vast amount of data can be stored.
* There is no risk of data mismanagement at any level while the project development is under process.
* Relationship between the administrator, owner/developer and subcontractor can be maintained very easily.
* It provides high level of security.
* There is no risk of data mismanagement at any level while the project development is under process.
* Relationship between the administrator, owner/developer and subcontractor can be maintained very easily.
* It provides high level of security using different protocols like https etc.

**Single Tier:**

In a single tier the server and client are the same in the sense that a client program that needs information (client) and the source of this type of architecture is also possible in java, in case flat files are used to store the data. However this is useful only in case of small applications. The advantage with this is the simplicity and portability of the application developed

Server and client

DataBase

**Two Tiers (client-server):**

In two tier architecture the database resides in one machine and client in different machine they are connected through the network. In this type of architecture a database management takes control of the database and provides access to clients in a network. This software bundle is also called as the server. Software in different machines, requesting for information are called as the clients.

Server

Client

Client

DataBase

**Three Tier and N-Tier:**

In the three-tier architecture, any number servers can access the database that resides on server. Which in turn serve clients in a network. For example, you want to access the database using java applets, the applet running in some other machine, can send request only to the server from which it is down loaded. For this reason we will need to have a intermediate server which will accept the requests from applets and them to the actual database server. This intermediate server acts as a two-way communication channel also. This is the information or data from the database is passed on to the applet that is requesting it. This can be extended to make n tiers of servers, each server carrying to specific type of request from clients, however in practice only 3 tiers architecture is popular.

Clint

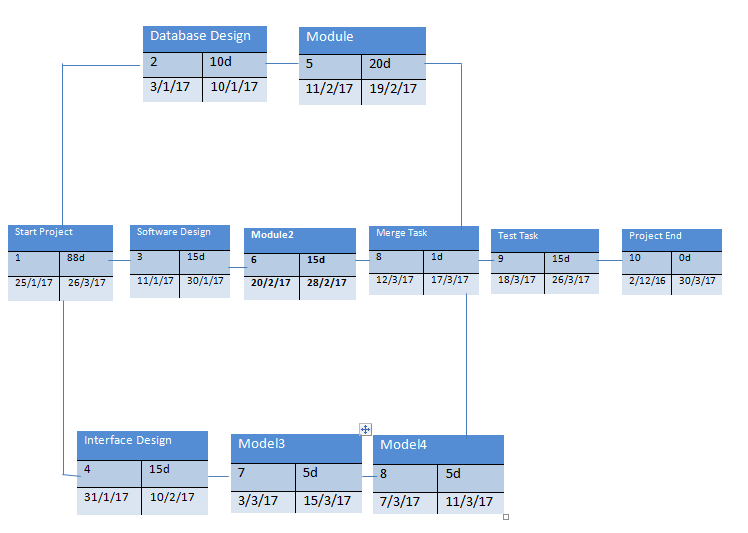
Application

Application Server

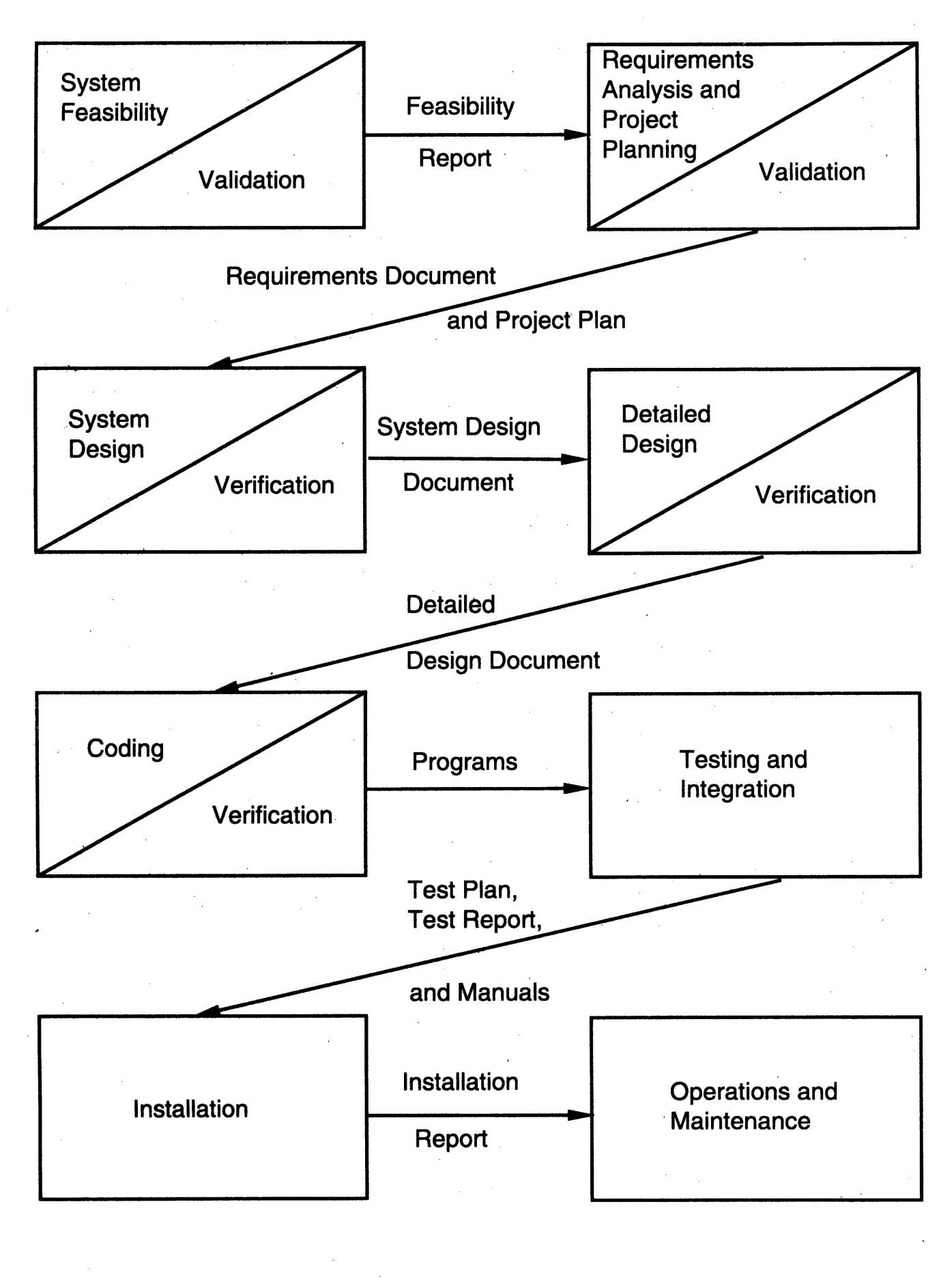
DataBase

**3.User Requirement System Planning(pert chart):**

**Pert: chart**



**WATERFALL MODEL:**



**WATERFALL MODEL:**

The consequence of the need for certification is that each phase must havesome defined output that can be evaluated and certified. That is, when the activities of a phase are completed, there should be some product that is producedby that phase. The outputs of the earlier phases are often called work productsand are usually in the form of documents like the requirements document or16 2. Software Processesdesign document. For the coding phase, the output is the code. Though the setof documents that should be produced in a project is dependent on how theprocess is implemented, the following documents generally form a reasonableset that should be produced in each project:

– Requirements document

– Project plan

– Design documents (architecture, system, detailed)

– Test plan and test reports

– Final code

– Software manuals (e.g., user, installation, etc.)

One of the main advantages of the waterfall model is its simplicity. It is conceptuallystraightforward and divides the large task of building a software systeminto a series of cleanly divided phases, each phase dealing with a separate logicalconcern. It is also easy to administer in a contractual setup—as each phaseis completed and its work product produced, some amount of money is givenby the customer to the developing organization.

**FEASIBILITY STUDY:**

Feasibility study is conducted once the problem is clearly understood. Feasibility study is a high level capsule version of the entire system analysis and design process. The objective is to determine quickly at a minimum expense how to solve a problem. The purpose of feasibility is not to solve the problem but to determine if the problem is worth solving.

The system has been tested for feasibility in the following points.

1. Technical Feasibility

2. Economical Feasibility

3. Operational Feasibility.

**1. Technical Feasibility**

The project entitles "University Notification System” is technically feasibility because of the below mentioned feature. The project was developed in Java which Graphical User Interface.

It provides the high level of reliability, availability and compatibility. All these make Java an appropriate language for this project. Thus the existing software Java is a powerful language.

**2. Economical Feasibility**

The computerized system will help in automate the selection leading the profits and details of the organization. With this software, the machine and manpower utilization are expected to go up by 80-90% approximately. The costs incurred of not creating the system are set to be great, because precious time can be wanted by manually.

**3. Operational Feasibility**

In this project, the management will know the details of each project where he may be presented and the data will be maintained as decentralized and if any inquires for that particular contract can be known as per their requirements and necessaries.

**5.System Implementation:**

The goal of the coding phase is to translate the design into code in the given programing language. The coding steps translate the detailed design of the system into programming language. The translation process continues when complier accepts source code as input and produces machine dependent object code as output. Linking of object files are done to produce the machine code. Internal documentation is another important factor, to facilitate other to understand the code and the logic

**Code review and walk through:**

Both reviews and walk through used to deliver the correct codes. The code review is done as soon as code is ready to be executed, this is to reduce syntax errors and also check the coding standard.

**Module Specification:**

The modules specified in the design are implemented using various “.html”, “.jsp” and “.class” files. These files in the source code shares the common routines and share the data structure, to establish the hierarchical relationship.

**Compilation and Building the executable:**

The source code for the system organizedin various files is compiled using the “java” utility provided in the JAVA. The application is made to run in web browser the address as “http://localhost:8081/staff” present in ROOT directory of Tomcat Server.

**Running the package:**

The following steps are undertaken to execute this application-

* Frist start Tomcat Server.
* Open any Web Browser like Google Chrome, Mozilla Firefox.
* Type the following address in address bar of Web browser.
* http://localhost:8081/notification
* Now the Home page of application is displayed on screen. If new registration is there then register or sing up first and use the application. If already registered then directly logon into the system.

**Java:**

**Java** is a computer programming language that isconcurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to byte code (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is, as of 2014, one of the most popular programming languages in use, particularly for client-server web applications, with a

reported 9 million developers. Java was originally developed by James Gosling at Sun

Microsystems(which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Javacompilers, virtual machines, and class libraries weredeveloped by Sun from 1991 and first released in 1995.As of May 2007, in compliance with the specificationsof the Java Community Process, Sun relicensed most of

its Java technologies under the GNU. Others have alsodeveloped alternative implementations of these Suntechnologies, such as the GNU Compiler for Java (bytecode compiler), GNU Class path (standard libraries),and Iced Tea-Web (browser plugin for applets).

**Advantages of java:**

* Purely Object oriented.
* Platform independent.
* It is dynamic, simple and robust.
* Easy to learn.
* Multithreaded.
* Secure.
* Wide variety of Application Programmer Interfaces (APIs).
* Excellent networking capability.

**The java Platform:**

The Java platform is the name given to the computing platform from Oracle that helps users to run and develop Java applications. The platform does not just enable a user to run and develop Java application, but also features a wide variety of tools that can help developers work efficiently with the Java programming language.

**The platform consists of two essential software’s:**

* The Java Runtime Environment (JRE), which is needed to run Java applications and applets.
* The Java Development Kit (JDK), which is needed to develop those Java applications and applets. If you have installed the JDK, you should know that it comes equipped with a JRE as well. So, for all the purposes of this book, you would only require the JDK.

**Java Components:**

Java has two components those are

1. Java virtual machine (JVM).

2. Java Application Programmers Interface (API).

**JVM-**

A Java virtual machine (JVM) interprets compiled Java binary code (called byte code) for a computer's processor (or "hardware platform") so that it can perform a Java program's instructions. Java was designed to allow application programs to be built that could be run on any platform without having to be rewritten or recompiled by the programmer for each separate platform. A Java virtual machine makes this possible because it is aware of the specific instruction lengths and other particularities of the platform

**API-**

An application programming interface (API) is a library of functions that Java provides for programmers for common tasks like file transfer, networking, and data structures.

|  |
| --- |
| Java Program |
| Java API  Java Virtual Machine |
| Hardware-Based Platform |

**Java in web:**

Java covers the whole application form server to client and back again, it provides many powerful technologies, it can be used to extend the browser, and it provides good security system.

**HTML:**

HTML stands for hypertext markup Language. It is very useful to make web pages and very easy to learn. Hypertext Markup file is a text file containing small markup tags. These marks up tags tell the browser how to display a web page. It has two types of extensions one is .html and second is .html but both are used for html web page For hyper text markup language you can use the simple text editor for example; use notepad for writing your HTML code in the windows. If you are using Mac you can use simple text editor.

HTML uses approach of what you see is what you get. You can also use to write tags other software that is FrontPage and Dreamweaver. In HTML character are surrounded by the tags. HTML tags come in pair. The beauty of this language is that it is not case sensitive. Every web page need HTML with it you cannot make the good web pages. And it is the base for every web page and used to display the text in the web pages there are some other latest version of HTML like DHTML which stands for dynamic html and is used to make the web pages more interactive.

**Features of HTML:**

* It is simple to understand and implement.
* HTML constructs are very easy to comprehend, and can be used effectively by anybody.
* HTML syntax is a worldwide standard.
* The methodology used by HTML to markup information is independent of its representation on a particular hardware or software architecture.
* It is not a programming language.
* And it is also not a description language

**Java Server Pages (JSP):**

Java Server Pages (JSP) lets you separate the dynamic part of your pages from the static HTML. We simply write the regular html in the normal manner, using whatever Web-page-building tools you normally use. We then enclose the code for the c parts in special tags, most of which start with "<%" and end with "%>". We normally give your file a .jsp extension, and typically install it in any ace you could place a normal Web page. Although what you write often looks more a regular html file than a servlet, behind the scenes, the JSP page just gets converted to a normal servlet, with the static html simply being printed to the output stream associated with the servlet's service method.

This is normally done the first time the page is requested, and developers can simply request the page themselves when first installing it if they want to be sure that the first real user doesn't get a momentary’ delay when the JSP page is translated to a servlet and the servlet is compiled and loaded. Many Web servers let you define aliases that so that a URL that appears to reference an html file really points to a servlet or JSP page

**Advantages of jsp:**

* Separation of static from dynamic content.
* Write Once Run Anywhere.
* Recommended Web access layer for n-tier architecture.
* Completely leverages the Servlet API.
* Platform independent.
* Reuse of components and tag libraries.
* Encapsulation of functionality.
* They have a better performance and scalability than ordinary CGI scripts, because they are persistent in memory and multi-threaded.
* They have built in support for HTTP sessions, which makes application Programming possible.
* They have full access to Java Technology-Network awareness, threads and Database connectivity-without the limitations of client side application applets.
* They are automatically recompiled when necessary.
* They exist in the ordinary Web server document space, no special URL mapping is required to address them.

**How jsp works?**

**JSP pages exist in 3 forms or versions-**

* JSP source code consists of text file with an extension of .jsp and contains a mix of HTML template code, Java language statements and JSP directives and actions that describe how to generate a web page to service a particular request.
* Java source code: the jsp container translates the jsp source code into the source code for an equivalent Java Servlet as needed..
* Compiled Java class: Like any other Java class, the generated servlet code is compiled into byte-codes in a .class file ready to be loaded and executed.

**Java Script:**

**What is Java Script?**

* Java Script is embedded into html.
* It is browser dependent.
* JavaScript depends on the web browser to support it. If the browser doesn't support it, JavaScript code will be ignored. Internet Explorer 3.0 and Netscape Navigator 2.0 onwards support JavaScript.
* It is an interpreted language, loosely typed, object based language.
* Java script is not Java

**Data Validation:**

JavaScript provides the means for basic data validation before it is sent to the server. Whether the values entered are correct or not or whether all the fields in a form are filled out or not can be checked before sending data to web server, if JavaScript is not used t hen data is sent to web server, and the web server would response with a message that the data sent to it is incorrect or incomplete Thus JavaScript ensures data validation and also reduces the network traffic. .

**Reasons for Back end:**

As Oracle 8.0 is object oriented and robust RDBMS & a number of tools available to work with it make it quite attractive as a backend.

In a market place populated by computer companies with “proprietary” hardware, “proprietary” operating systems, “proprietary” databases and “proprietary” applications, ORACLE gives business users systems departments’ new control over their lives and futures. They are no longer bound to the database product of a single hardware vendor. ORACLE can run on nearly every kind of computer they can own. This is a basic revolution in the workplace and in application development, with 0consequences that will extend far into the future.

**Java Data Base Connectivity (JDBC):**-

In an enterprise computing which is largely the black art of managing huge databases? People associated with the enterprise need to be able to use and update the data easily, quickly and securely.

The powerful Java Data Base Connectivity (JDBC) suit the java.sql.\* package realizes java’s promise as a serious business programming tools.

Java Data Base Connectivity is a standard SQL database access interface providing uniform access to a wide range of relational databases. It also provides a common base on which higher level tools and inter faces can be built. This comes with an “ODBC Bridge”. That bridge is a library, which implements JDBC in terms of ODBC standard API.

There are many types of drivers used in connecting such as Native API Partly Java driver, a net protocol all java driver. The driver used here is JDBC-ODBC Bridge.

JDBC-ODBC bridge plus ODBC driver –This is the crudest possible solution. Applets access data base using a combination of the JDBC-ODBC Bridge and an ODBC driver. This requires both drivers to be installed on the user’s computer- A very cumbersome solution for both internet and intranet users.

JDBCTM is a Java Tm API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of as standing for "Java Database Connectivity".) It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. In other words, with the JDBC API, it isn't necessary to write one program to access a Sybase database, another program to access an MySQL database, another program to access an Informix database, and so on. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. And, with an application written in the Java programming language, one also doesn't have to worry about writing different applications to run on different platforms. The combination of Java and JDBC lets a programmer write it once and run it anywhere.

Java being robust, secure, easy to use, easy to understand, and automatically downloadable on a network, is an excellent language basis for database applications.

What is needed is a way for Java applications to talk to a variety of different databases. JDBC is the mechanism for doing this. JDBC extends what can be done in Java. For example, with Java and the JDBC API, it is possible to publish a web page containing an applet that uses information obtained from a remote database. Or an enterprise can use JDBC to connect all its employees (even if they are using a conglomeration of Windows, Macintosh, and UNIX machines) to one or more internal databases via an intranet. With more and more programmers using the Java programming language, the need for easy database access from Java is continuing to grow. MIS managers like the combination of Java and JDBC because it makes disseminating information easy and economical. Businesses can continue to use their installed databases and access information easily even if it is stored on different database management systems. Development time for new applications is short. Installation and version control are greatly simplified. A programmer can write an application or an update once, put it on the server, and everybody has access to the latest version. And for businesses selling information services, Java and JDBC offer a better way of getting out information updates to external customers.

**JDBC does the following things:**

* Establish a connection with a database
* Send SQL statements
* Process the results.

**The following code fragment gives a basic example of these three steps:**

Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con=DriverManager.getConnection("Jdbc:

Odbc:dsnname");

Statement stmt=con.createStatement();

**Connection:**-

A connection object represents a connection with a database. A connection session includes the SQL statements that are executed and the results that are returned over the connection. A single application can have one or more connections with a

single database, or it can have connections with many different databases.

**Opening a Connection:**

The standard way to establish a connection with a database is to call the method DriverManager.getConnection. This method takes a string containing a URL. The Driver Manager class, referred to a the JDBC management layer, attempts to locate a driver than can connect to the database represented Driver classes, and when the method get Connection is called, it checks with each driver in the list until it finds one that can connect uses this URL to actually establish the connection.

**Sending Statement:**

Once a connection is established, it is used to pass SQL statements to its underlying database. JDBC does not put any restrictions on the kinds of SQL statements that can be sent; this provides a great deal of flexibility, allowing the use of database-specific

statements or even Non-SQL statements. It requires, however, that the user be responsible for making sure that the underlying database can process the SQL statements being sent and suffer the consequences if it cannot.

**Driver Manager:**

The Driver Manager class is the management layer of JDBC, working between the user and the drivers. It keeps track of the drivers that are available and handles establishing a connection between a database and the appropriate driver. It addition, the

driver manager class attends to things like driver login time limits and the printing of log and tracing messages. The only method in this class that a general programmer needs to use directly is DriverManager.getConnection. As its name implies, this method establishes a connection to a database.

**Why we need JDBC?**

* ODBC is not appropriate for direct use from Java because it uses a C interfaces.
* ODBC is hard to learn. It mixes simple and advanced features together, and it has
* Complex options even for simple queries.
* A Java API like JDBC is needed in order to enable a “Pure Java “solution.
* When ODBC is used, the ODBC driver manager and drivers must be manually
* Installed on every client machine.

**SESSION:**

This is the ***HttpSession***object associated with the request. Recall that sessions are created automatically, so this variable is bound even if there was no incoming session reference. The one exception is if you use the ***session*** attribute of the page

directive to turn sessions off, in which case attempts to reference the session variable cause errors at the time the JSP page is translated into a servlet.

**Apache Tomcat:**

Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Tomcat is a web server that supports servlets and JSPs.

Tomcat comes with the Jasper compiler that compiles JSPs into servlets. Tomcat is available for commercial use under the ASF license from the Apache web site in both binary and source versions.

Apache Tomcat is developed in an open and participatory environment and released under the Apache Software License. Apache Tomcat powers numerous large scale, mission-critical web applications across a diverse range of industries and organizations. Different versions of Apache Tomcat are available for different versions of the Servlet and JSP specifications.

The Tomcat servlet engine is often used in combination with an Apache web server or other web servers. Tomcat can also function as an independent web server. Earlier in its development, the perception existed that standalone Tomcat was only suitable for development environments and other environments with minimal requirements for speed and transaction handling. However, that perception no longer exists; Tomcat is increasingly used as a standalone web server in high-traffic, high availability environments. Since its developers wrote Tomcat in Java, it runs on any operating system that has a JVM

**Usage:**

Java Web Applications, Java Mobile Applications using J2ME

**Access:**

Create a Java Web Application or Mobile Application using Net Beans Build and run project, this will automatically launch the Apache Tomcat as default.

**Viewing Web Applications:**

http://localhost: 8080/WebpageName.jsp

**Directory Structure:**

The typical and default directory hierarchy of a Tomcat installation comprises the following:

* bin - startup, shutdown and other scripts and executable.
* common - common classes that Catalina and web applications can use.
* conf - XML files and related DTDs to configure Tomcat.
* logs - Catalina and application logs.
* server - classes used only by Catalina.
* shared - classes shared by all web applications.
* webapps - directory containing the web applications.
* work - temporary storage for files and directories.

**A web application is basically a web site that:**

* "Knows who you are"--it doesn't just give you static pages, it interacts with you.
* Can permanently change data (such as in a database).
* A web application can consist of multiple pieces.
* Static web pages (possibly containing forms).
* Servlets.
* JSP.

Tomcat organizes all these parts into a single directory structure for each web application.

**The flow that takes place is:**

* The user submits an HTML form.
* Tomcat finds the servlet based on the URL and the deployment descriptor.
* (web.xml) and passes the request to the servlet
* The servlet computes a response.
* The servlet writes an HTML page containing the response.
* The servlet forwards the response to the JSP.
* The JSP embeds the response in an HTML page
* Tomcat returns the HTML page to the user.

**Status:**

Tomcat is available at the Jakarta binary downloads page. The Tomcat server is a Java based Web Application container that was created to run Servlets and Java Server Pages (JSP) in Web applications. As part of Apache's open source Jakarta project, it has nearly become the industry accepted standard reference.

Implementation for both the Servlets and JSP API. Written by expert Servlets and JSP software architect and author James Goodwill, this column will feature introductory Web application development issues, Tomcat installation and configuration, deploying Web applications onto Tomcat, Struts and much more.

Apache Tomcat (formerly under the Apache Jakarta Project; Tomcat is now a top level project) is a web container developed at the Apache Software Foundation. Tomcat implements the servlet and the Java Server Pages (JSP) specifications from Sun Microsystems, providing an environment f or Java code to run in cooperation with a web server. It adds tools for configuration and management but can also be configured by editing configuration files that are normally XML-formatted. Because Tomcat includes its own HTTP server internally, it is also considered a standalone web server.

Tomcat is a web server that supports servlets and JSPs. Tomcat comes with the Jasper compiler that compiles JSPs into servlets. The Tomcat servlet engine is often used in combination with an Apache webserver or other web servers. Tomcat can also function as an independent web server

Earlier in its development, the perception existed that standalone Tomcat was only suitable for development environments and other environments with minimal requirements for speed and transaction handling. However, that perception no longer exists;

Tomcat is increasingly used as a standalone web server in high-traffic, high availability environments. Since its developers wrote Tomcat in Java, it runs on any operating system that has a JVM

**Beans:**

It’s used to achieve reusability in java it is introduced to overcome the drawbacks of the traditional inheritance and object relations in JavaBean. The reusability is achieved using bean class. Bean is a component equivalent to ActiveX component of visual

basic. To create beans we must follow design pattern rules

* The class must be placed inside the package
* Class must be defined as public
* Variables in the class are defined as private to avoid direct accessibility of variables
* Variables must be defined in lower case
* A bean class can have public variable also, for every private variable there must be Set and Get methods. Set takes parameter which is used to assign the values for the variable. Get is used to retrieve the values. Every bean class should have implicit constructor (default constructor).

**FUNCTIONAL REQUIRMENT:**

**Admin:**

* Admin maintains the entire application
* Admin can add Parking area and service stations as well as block them

**User:**.

* user can use the thinks which he needs.
* user can also view the parking area and near by service station.

**Parking Owner:**

* This module is designed for the Adding Parking area .
* In this module we give facility to parking owner to enter his area.
* Parking owner can add Rates, facilitys or any other Details etc.

**Service Station:**

* Service station owner can add his area where he is.
* He can mention the services which all he provide.

**5.Details Of Software & Hardware:**

**Software Requirements:**

Operating System : Windows XP/Windows 7/windows8

IDE : Micro media Dreamweaver

Designing Technology : HTML

Server Side Technology : JSP

Server : Tomcat 5.5

Back End : My SQL

Web Browser : Google Chrome, MozillaFirefox ,InternetExplorer

**Hardware Requirements:**

Processor : Pentium 4 or Above

RAM : Minimum 256 MB

HDD : Minimum 512 MB

**6.System Maintenance:**

The term “software maintenance” is used to describe the software engineering activities that occur following delivery of a software product to the customer. The maintenance phase of the software life cycle is the time period in which a software product performs useful work. Maintenance activities involve making enhancement to software products, adapting products to new environments and correcting problems. Software product enhancement may involve providing new functional capabilities, improving user display and modes of interaction, and upgrading external documents. Adaptation of software to a new environment may involve moving the software to a different machine. Problem correction involves modification and revalidation of software to correct errors. The enhancement of this project can be accomplished easily. That is, any new functional capabilities can be added to the project by simply including the new module in the homepage and giving a hyperlink to that module. Adaptation of this project to a new environment is also performed easily.

Even with the best quality assurance activities, it is likely that the customer will uncover defects in the software. Corrective maintenance changes the software to correct defects.

**Cost–benefit analysis** (**CBA**), sometimes called **benefit–cost analysis** (**BCA**), is a systematic approach to estimating the strengths and weaknesses of alternatives that satisfy transactions, activities or functional requirements for a business. It is a technique that is used to determine options that provide the best approach for the adoption and practice in terms of benefits in labor, time and cost savings etc.The CBA is also defined as a systematic process for calculating and comparing benefits and [costs](http://en.wikipedia.org/wiki/Cost) of a project, decision [government policy](http://en.wikipedia.org/wiki/Government_policy) (hereafter, "project").

Broadly, CBA has two purposes:

1. To determine if it is a sound investment/decision (justification/feasibility),
2. To provide a basis for comparing projects. It involves comparing the total expected cost of each option against the total expected benefits, to see whether the benefits outweigh the costs, and by how much.

CBA is related to, but distinct from cost-effectiveness analysis. In CBA, benefits and costs are expressed in monetary terms, and are adjusted for the time value of money, so that all flows of benefits and flows of project costs over time (which tend to occur at different points in time) are expressed on a common basis in terms of their "net present value."

Closely related, but slightly different, formal techniques include cost-effectiveness analysis, cost–utility analysis, risk–benefit analysis, economic impact analysis, fiscal impact analysis, and Social return on investment (SROI) analysis.

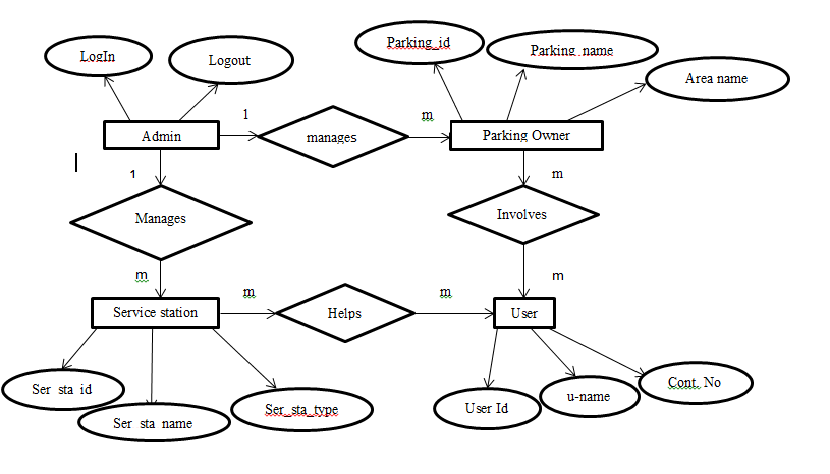
**7.Cost Estimation Of The Project:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Cost Head** | **Unit Rates(rs/month)** | **Estimated usage(person/month)** | **Cost(rs)** |
| **Manpower** | **1500** | **2** | **3000** |
| **h/w cost work station server** | **5000** | **1** | **5000** |
| **Software cost** | **2500** | **6 months** | **2500** |
| **Travel cost** | **500** | **3 months** | **1500** |
| **Training cost** | **10000** | **-------** | **10000** |
| **Total** |  |  | **22000** |

**8.Detailed Life Cycle Of The Project :**

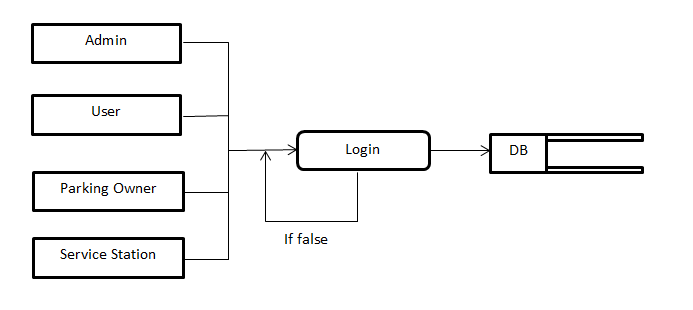
**8.1 High Level Design:**

**ER Diagram:**

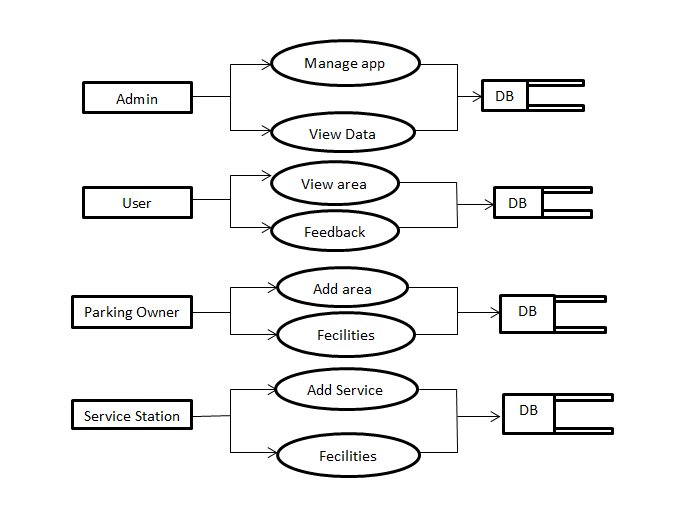


**8.2 Data Flow Diagram:**

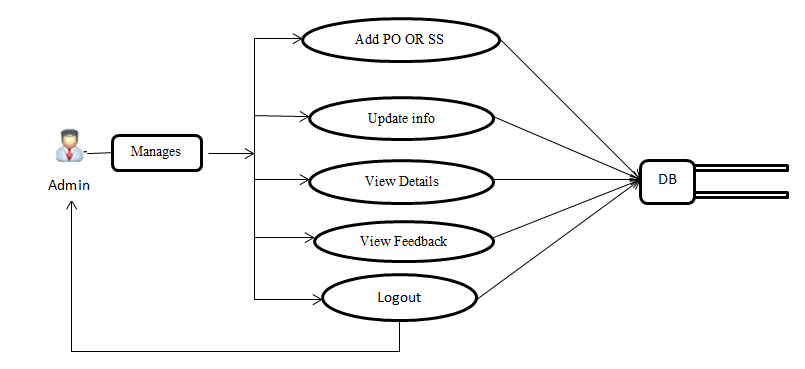
**0 Level Diagram:**

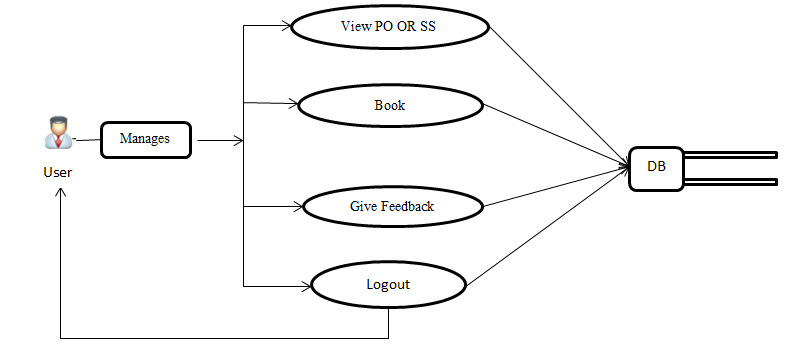


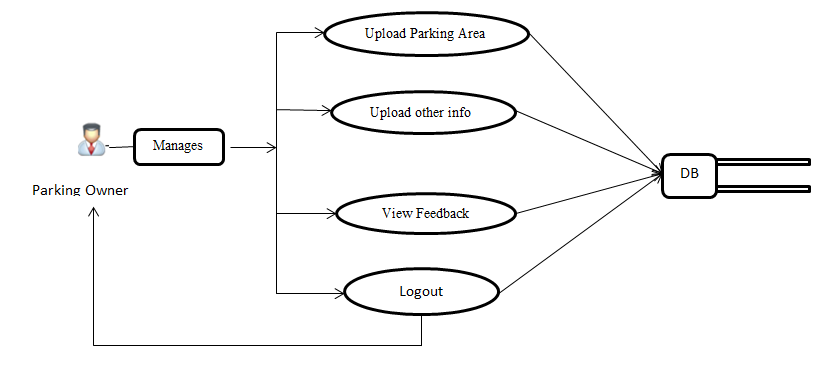
**1st Level Design:**

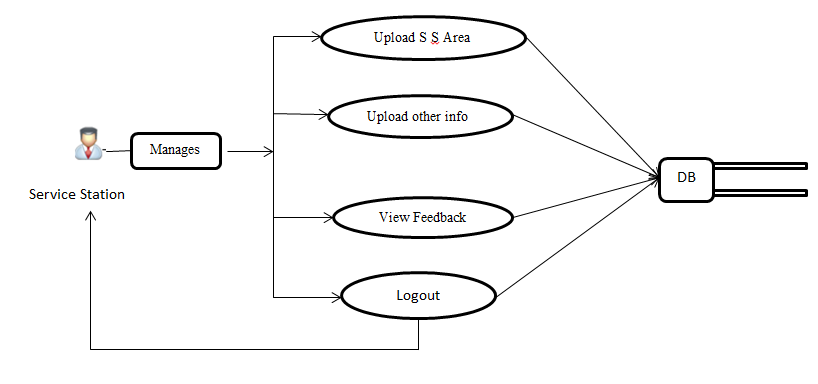
****

**2nd Level Design:**

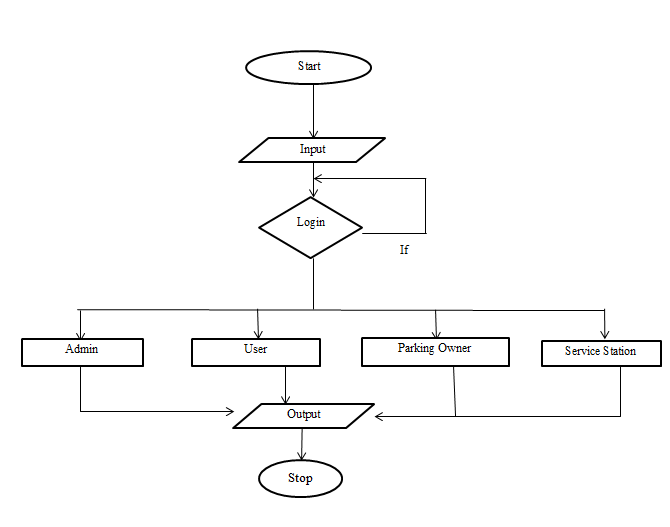








**Low Level Design:**



False

**8.3 INPUT AND OUTPUT SCREEN DESIGN:**

**Inputs:**

**Sample Code:**

**Bean Program:**

packagepltfrm;

import java.sql.\*;

public class db\_connect

{

private Connection con;

public Statement stmt;

public String getConn()

{

try

{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

con=DriverManager.getConnection("Jdbc:Odbc:PF");

stmt=con.createStatement();

}

catch(Exception ex)

{

System.out.println(ex);

}

return"";

}

}

**User Details:**

**Form-**

<jsp:include page="metatags.jsp"></jsp:include>

<jsp:include page="header.jsp"></jsp:include>

<jsp:include page="sidebar.jsp"></jsp:include>

<!-- Page Content -->

<div id="page-content">

<!-- END Breadcrumb -->

<!-- General Forms Block -->

<div class="block block-themed block-last">

<!-- General Forms Title -->

<div class="block-title">

<h4>User Registration</h4>

</div>

<!-- END General Forms Title -->

<!-- General Forms Content -->

<div class="block-content">

<jsp:include page="val.jsp"></jsp:include>

<form name="form1" method="post" action="User\_Registration\_insert.jsp" id="formID">

<div align="center">

<table width="366">

<tr>

<td colspan="2"><div align="center">User Registration </div></td>

</tr>

<tr>

<td width="115">Name</td>

<td width="235"><input name="name" type="text" id="name" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>Address</td>

<td><input name="address" type="text" id="address" class="validate[required]"></td>

</tr>

<tr>

<td>Contact No </td>

<td><input name="contact\_no" type="text" id="contact\_no" class="validate[required,custom[onlyNumber]]"></td>

</tr>

<tr>

<td>City</td>

<td><input name="city" type="text" id="city" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>State</td>

<td><input name="state" type="text" id="state" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>User Type </td>

<td><input name="user\_type" type="text" id="user\_type" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td height="43" colspan="2"><div align="center">

<input type="submit" name="Submit" value="Submit" class="btn btn-primary">

<input type="reset" name="Reset" value="Reset" class="btn btn-danger">

</div></td>

</tr>

</table>

</div>

</form>

</div>

<!-- END General Forms Content -->

</div>

<!-- END General Forms Block -->

</div>

<!-- END Page Content -->

<jsp:include page="footer.jsp"></jsp:include>

**Insert-**

<%@page import="java.sql.\*"%>

<%@page import="ips.dbconnect"%>

<jsp:useBean id="d" class="ips.dbconnect"/>

<jsp:getProperty name="d" property="conn"/>

<%

String S1,S2,S3,S4,S5,S6;

S1=request.getParameter("name");

S2=request.getParameter("address");

S3=request.getParameter("contact\_no");

S4=request.getParameter("city");

S5=request.getParameter("state");

S6=request.getParameter("user\_type");

int k=d.stmt.executeUpdate("insert into user\_registration values(null,'"+S1+"','"+S2+"','"+S3+"','"+S4+"','"+S5+"','"+S6+"')");

%>

<script>

alert("values are inserted.....");

document.location="User\_Registration\_view.jsp";

</script>

**Delete:**

<%@page import="java.sql.\*"%>

<%@page import="ips.dbconnect"%>

<jsp:useBean id="d" class="ips.dbconnect"/>

<jsp:getProperty name="d" property="conn"/>

<%

String id=request.getParameter("delete");

int z=d.stmt.executeUpdate("delete from user\_registration where user\_id="+id+" ");

%>

<script>

alert("values are deleted....");

document.location="User\_Registration\_view.jsp";

</script>

**View:**

<jsp:include page="metatags.jsp"></jsp:include>

<jsp:include page="header.jsp"></jsp:include>

<jsp:include page="sidebar.jsp"></jsp:include>

<div id="page-content">

<!-- With Borders Style -->

<h4 class="page-header">User Registration</h4>

<div class="block-section">

<!-- With Borders Section -->

<table class="table table-bordered">

<thead>

<tr>

<th>User ID </th>

<th>Name</th>

<th>Address</th>

<th>Contact No </th>

<th>City</th>

<th>State</th>

<th>User\_type</th>

<th>Delete</th>

<th>Edit</th>

</tr>

</thead>

<%@page import="java.sql.\*"%>

<%@page import="ips.dbconnect" %>

<jsp:useBean id="d" class="ips.dbconnect"/>

<jsp:getProperty name="d" property="conn"/>

<%

ResultSet rs=d.stmt.executeQuery("select \* from user\_registration");

while(rs.next())

{

%>

<thead>

<tr>

<td><%=rs.getInt("user\_id")%></td>

<td><%=rs.getString("name")%></td>

<td><%=rs.getString("address")%></td>

<td><%=rs.getString("contact\_no")%></td>

<td><%=rs.getString("city")%></td>

<td><%=rs.getString("state")%></td>

<td><%=rs.getString("user\_type")%></td>

<td><a onClick="return confirm('Do You Really Want To Delete....?')" href="User\_Registration\_delete.jsp?delete=<%=rs.getInt("user\_id")%>"><img height="30px" width="30px" src="delete.png"/></td>

<td><a href="User\_Registration\_edit.jsp?edit=<%=rs.getInt("user\_id")%>" onClick="return confirm('Do You Really Want To Edit....?')"><img height="30px" width="30px" src="photo.jpg"/></a></td>

</tr>

<%

}

%>

</thead>

</table>

<!-- END With Borders Section -->

<!-- END With Borders Style -->

</div>

<!-- END Page Content -->

<jsp:include page="footer.jsp"></jsp:include>

**Update:**

**Update1:**

<jsp:include page="metatags.jsp"></jsp:include>

<jsp:include page="header.jsp"></jsp:include>

<jsp:include page="sidebar.jsp"></jsp:include>

<!-- Page Content -->

<div id="page-content">

<!-- END Breadcrumb -->

<!-- General Forms Block -->

<div class="block block-themed block-last">

<!-- General Forms Title -->

<div class="block-title">

<h4>User Registration </h4>

</div>

<!-- END General Forms Title -->

<!-- General Forms Content -->

<div class="block-content">

<jsp:include page="val.jsp"></jsp:include>

<jsp:include page="val.jsp"></jsp:include>

<%@page import="java.sql.\*"%>

<%@page import="ips.dbconnect"%>

<jsp:useBean id="d" class="ips.dbconnect"/>

<jsp:getProperty name="d" property="conn"/>

<div align="center">

<%

String id=request.getParameter("edit");

ResultSet rs=d.stmt.executeQuery("select \*from user\_registration where user\_id="+id+"");

rs.next();

%>

<form name="form1" method="post" action="User\_Registration\_update.jsp" id="formID">

<table width="317" height="250" border="0">

<input name="user\_id" type="hidden" id="user\_id" value="<%=rs.getInt("user\_id")%>">

<tr>

<td width="115">Name</td>

<td width="186"><input name="name" type="text" id="name" value="<%=rs.getString("name")%>" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>Address</td>

<td><input name="address" type="text" id="address" value="<%=rs.getString("address")%>" class="validate[required]"></td>

</tr>

<tr>

<td>Contact No </td>

<td><input name="contact\_no" type="text" id="contact\_no" value="<%=rs.getString("contact\_no")%>" class="validate[required,custom[onlyNumber]]"></td>

</tr>

<tr>

<td>City</td>

<td><input name="city" type="text" id="city" value="<%=rs.getString("city")%>" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>State</td>

<td><input name="state" type="text" id="state" value="<%=rs.getString("state")%>" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td>User Type </td>

<td><input name="user\_type" type="text" id="user\_type" value="<%=rs.getString("user\_type")%>" class="validate[required,custom[onlyLetter]]"></td>

</tr>

<tr>

<td colspan="2"><input type="submit" name="Submit" value="Submit" class="btn btn-primary">

<input type="reset" name="Reset" value="Reset" class="btn btn-danger"></td>

</tr>

</table>

</form>

</div>

<!-- END General Forms Content -->

</div>

<!-- END General Forms Block -->

</div>

<!-- END Page Content -->

<jsp:include page="footer.jsp"></jsp:include>

**Update 2:**

<%@page import="java.sql.\*"%>

<%@page import="ips.dbconnect"%>

<jsp:useBean id="d" class="ips.dbconnect"/>

<jsp:getProperty name="d" property="conn"/>

<%

String S1,S2,S3,S4,S5,S6,S7;

S7=request.getParameter("user\_id");

S1=request.getParameter("name");

S2=request.getParameter("address");

S3=request.getParameter("contact\_no");

S4=request.getParameter("city");

S5=request.getParameter("state");

S6=request.getParameter("user\_type");

int k=d.stmt.executeUpdate("update user\_registration set name='"+S1+"',address='"+S2+"',contact\_no='"+S3+"',city='"+S4+"',state='"+S5+"',user\_type='"+S6+"' where user\_id="+S7+"");

%>

<script>

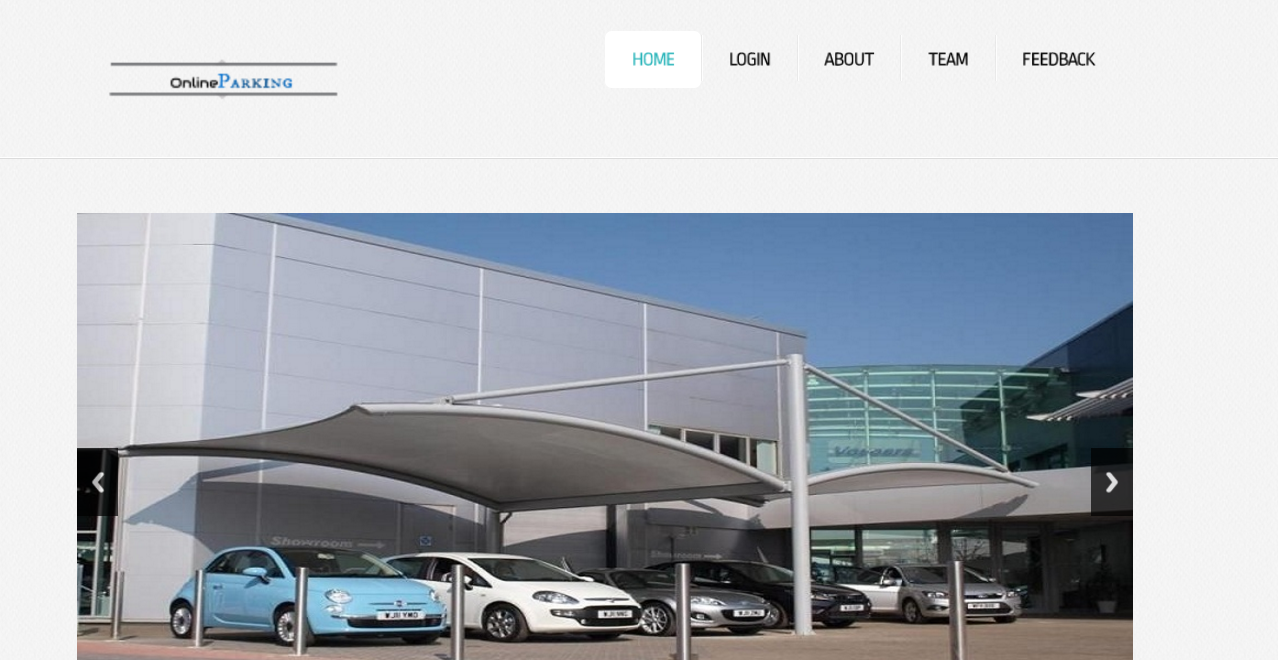
alert("values are Updated.....");

document.location="User\_Registration\_view.jsp";

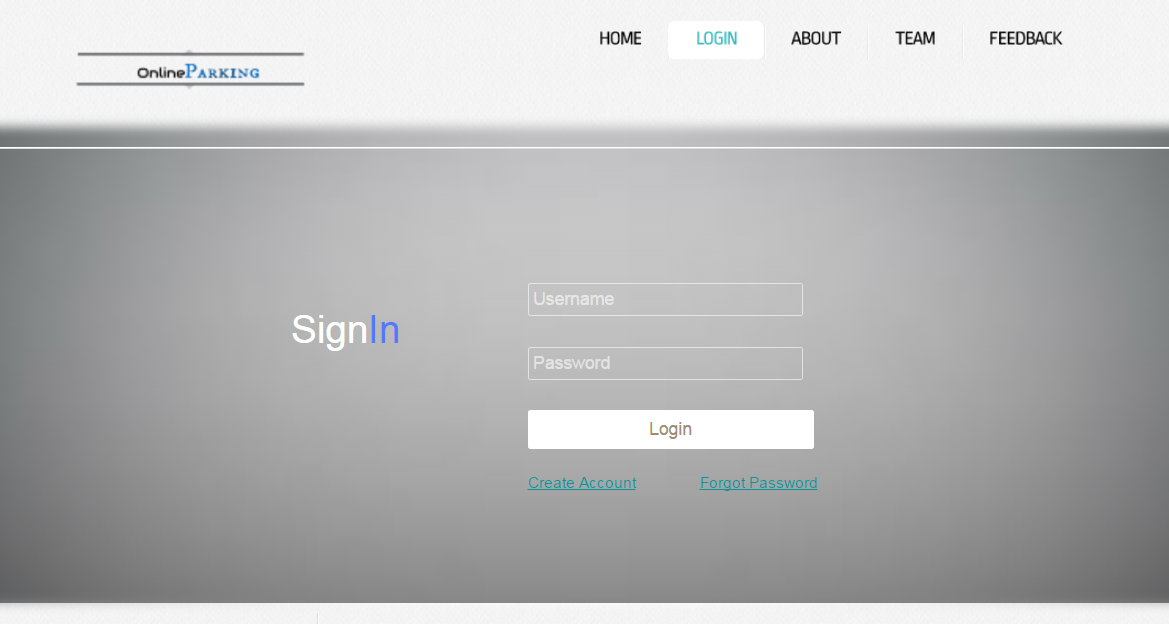
</script>

**8.3.1Screen Design:**

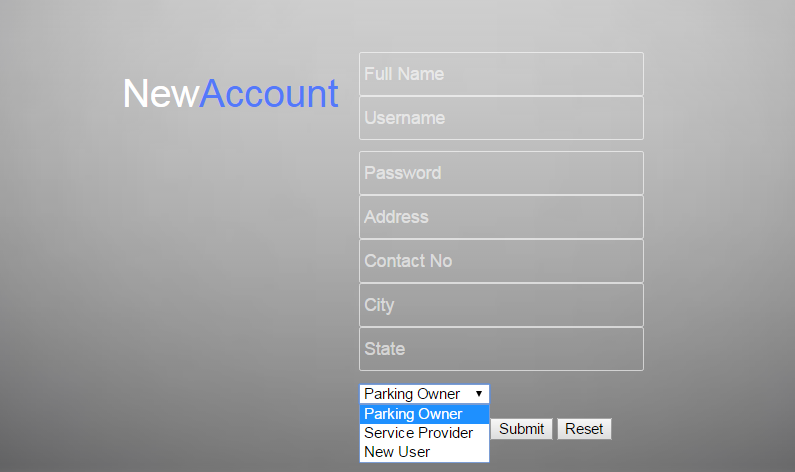
**Home page:**



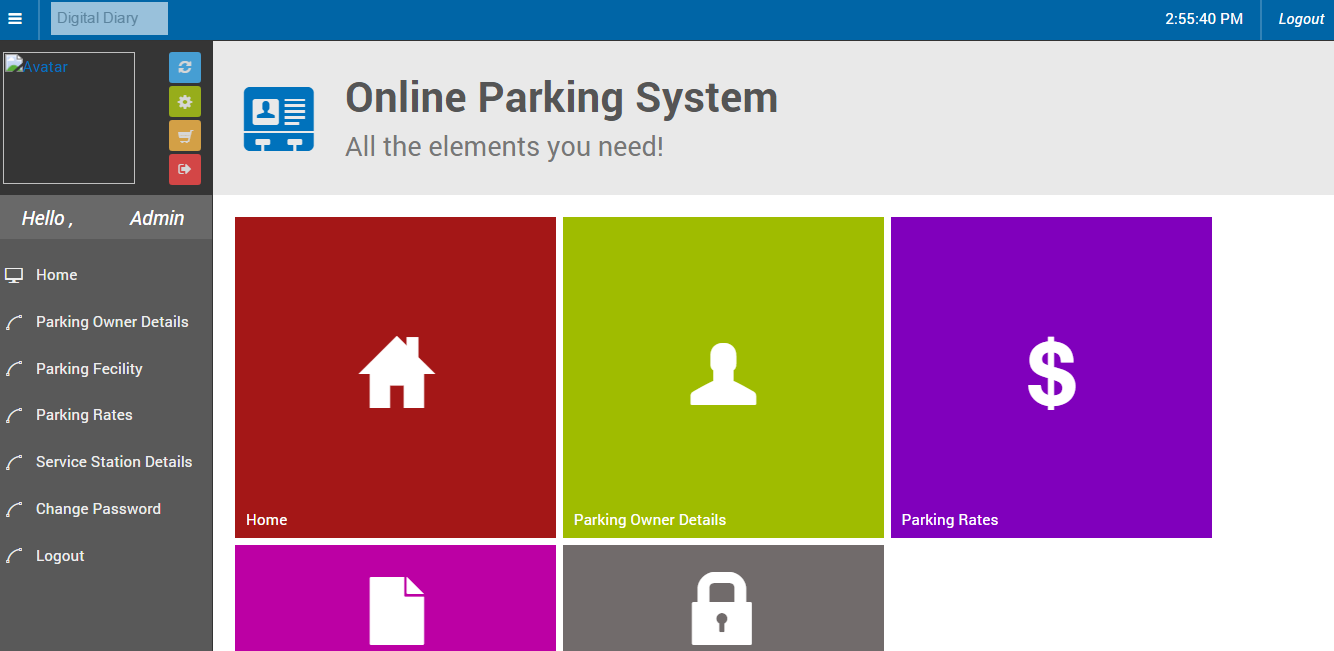
**Login form:**



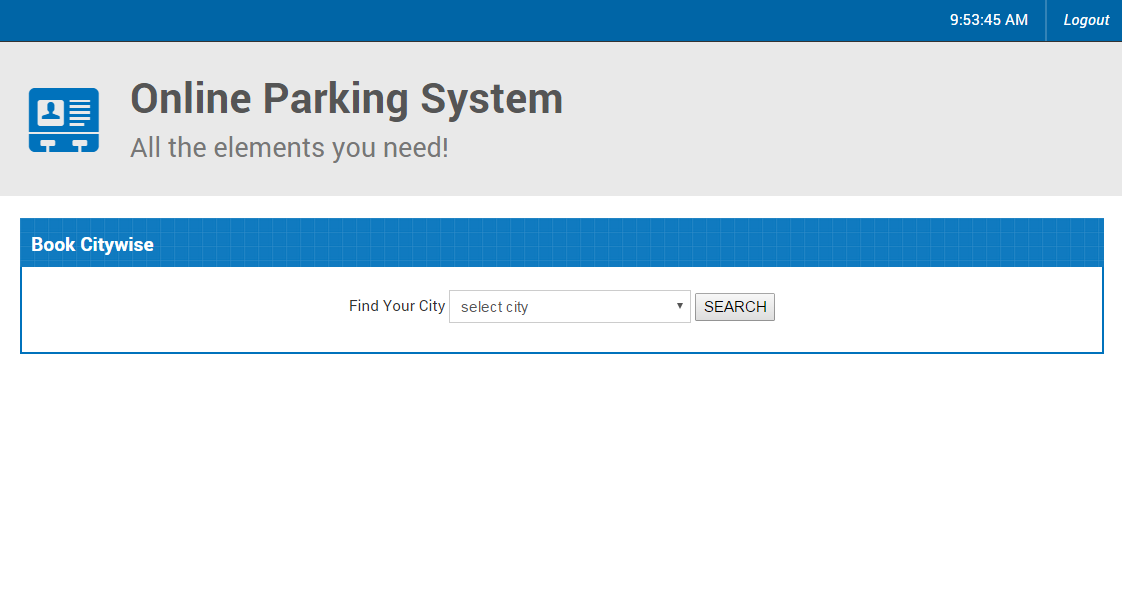
**Registration:**

****

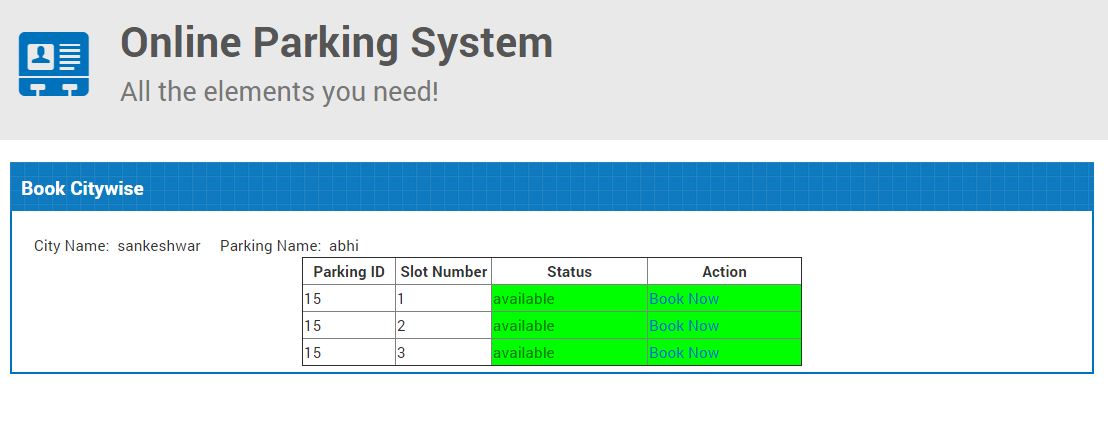
**Admin Panel:**



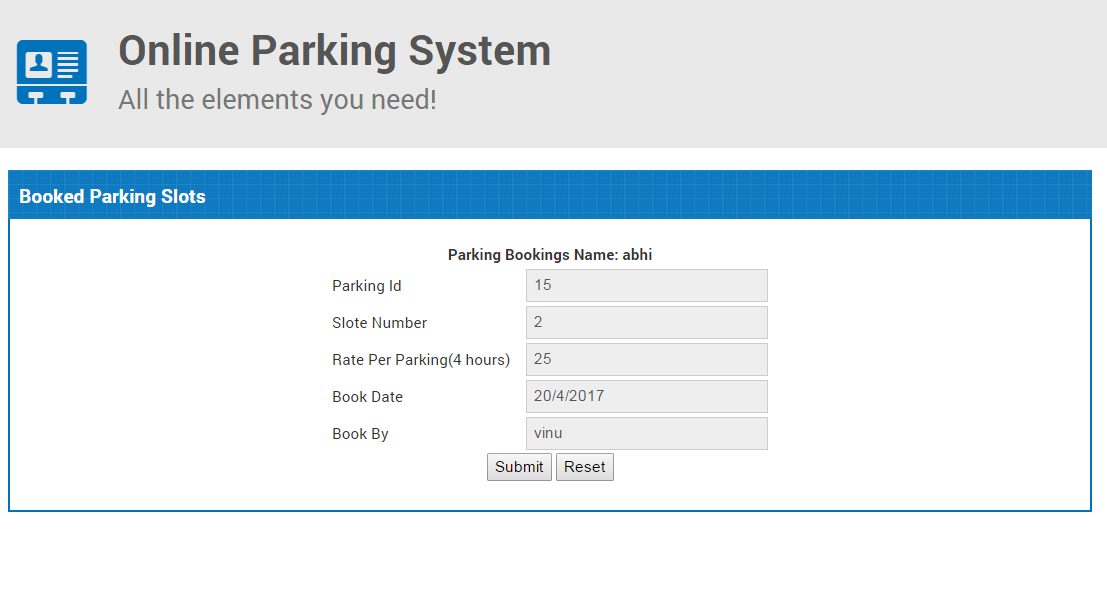
**Booking slot:**



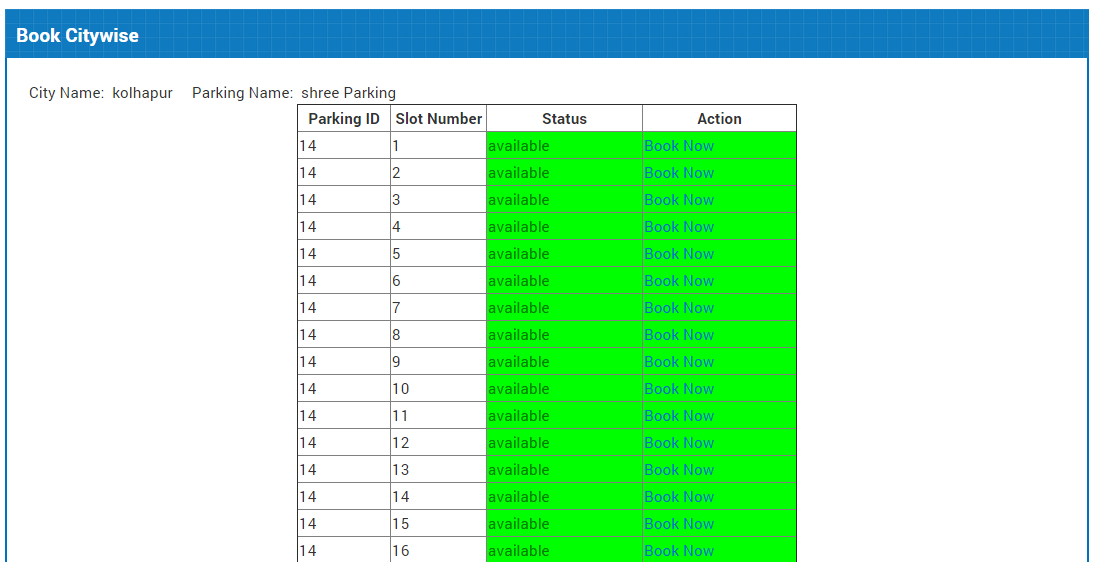
**After Selecting the city :**



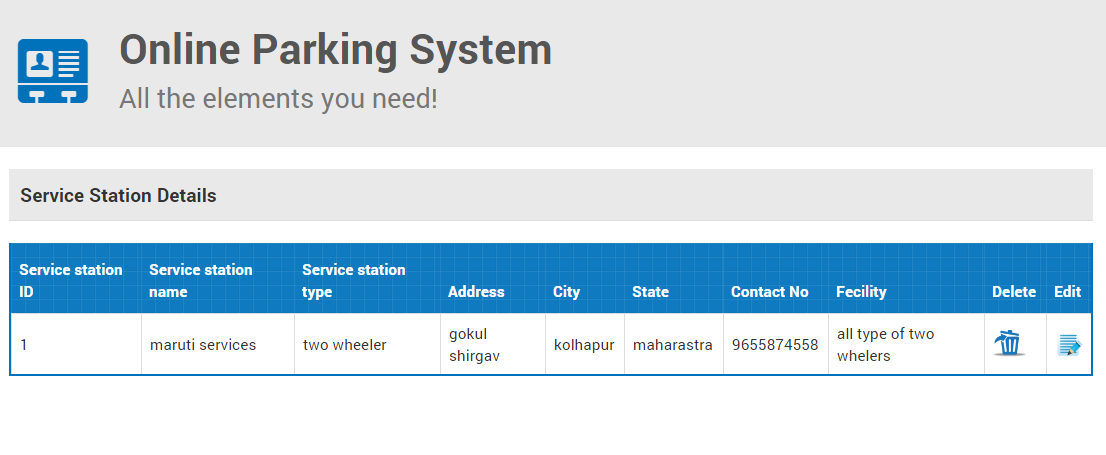
**Slots confiramation :**



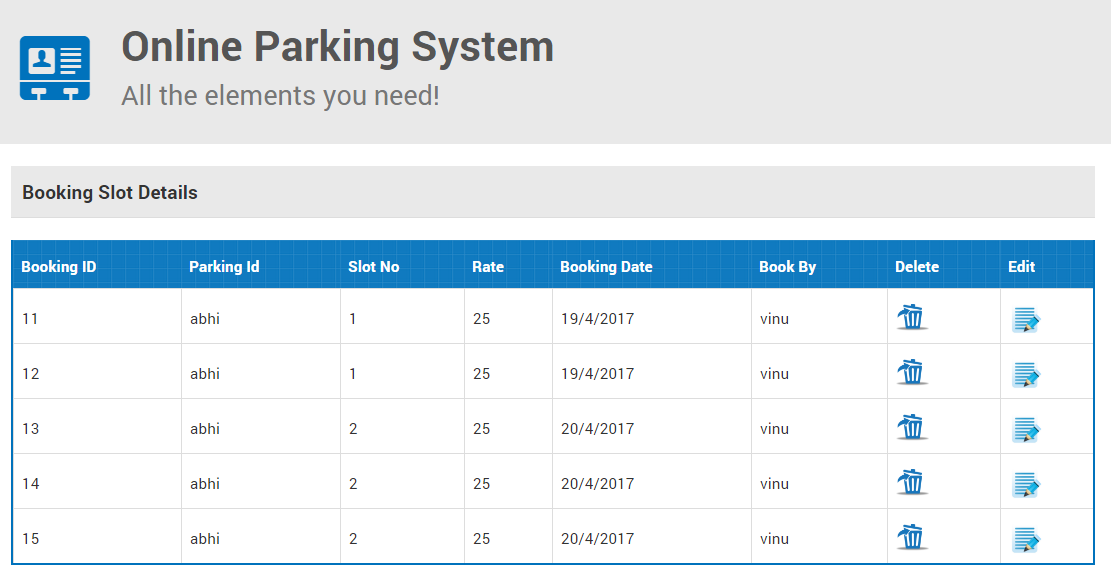
**Slots for booking:**



**Service Station :**

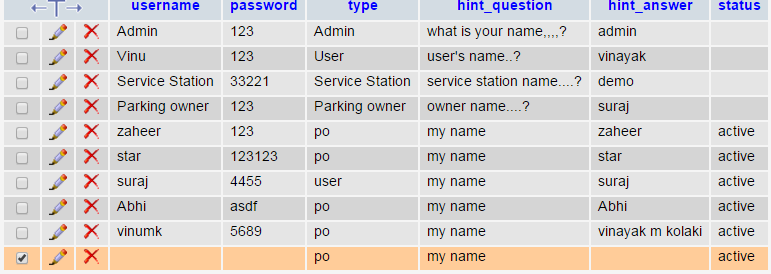


**Booked Parking details:**

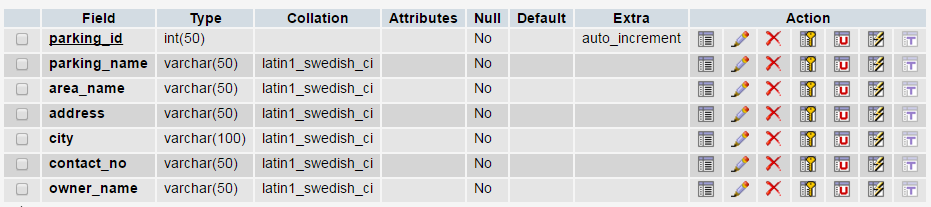


**Data Dictionary:**

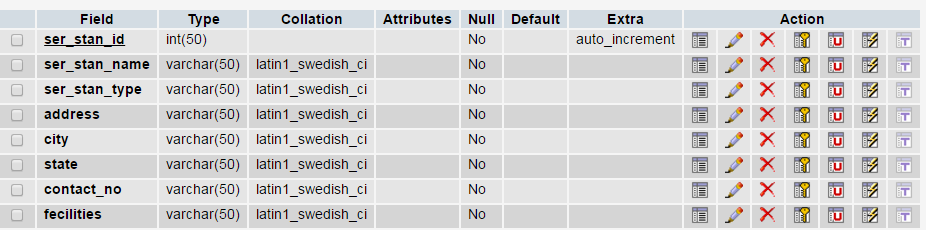
**Login**



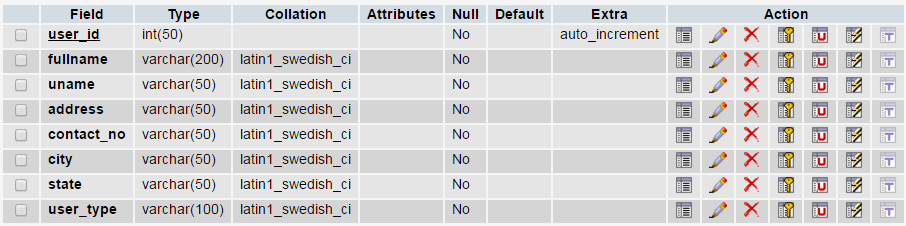
**Parking Details**



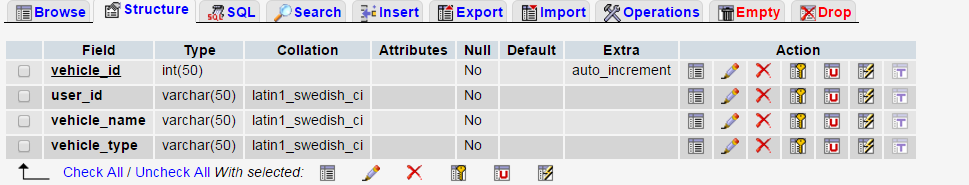
**Ser ivce station Details**



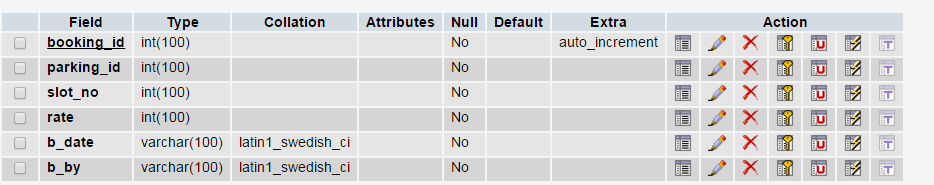
**New registration**



**Vehical details**



**Booked Parking Slot**



**8.4 Methodology Used For Testing:**

**Testing:**

Testing is a process of executing a program with the indent of finding an error. Testing is a crucial element of software quality assurance and presents ultimate review of specification, design and coding.

System Testing is an important phase. Testing represents an interesting anomaly for the software. Thus a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

A good test case is one that has a high probability of finding an as undiscovered error. A successful test is one that uncovers an as undiscovered error.

**Testing Objectives:**

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

**Testing Principles:**

* All tests should be traceable to end user requirements
* Tests should be planned long before testing begins
* Testing should begin on a small scale and progress towards testing in large
* Exhaustive testing is not possible
* To be most effective testing should be conducted by a independent third party.

The primary objective for test case design is to derive a set of tests that has the highest livelihood for uncovering defects in software. To accomplish this objective two different categories of test case design techniques are used. They are

* White box testing.
* Black box testing.

# **White-box testing:**

White box testing focus on the program control structure. Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

# **Block-box testing:**

Black box testing is designed to validate functional requirements without regard to the internal workings of a program. Black box testing mainly focuses on the information domain of the software, deriving test cases by partitioning input and output in a manner that provides through test coverage. Incorrect and missing functions, interface errors, errors in data structures, error in functional logic are the errors falling in this category.

**Testing strategies:**

A strategy for software testing must accommodate low-level tests that are necessary to verify that all small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

**Testing Information flow:**

Information flow for testing flows the pattern. Two class of input provided to test the process. The software configuration includes a software requirements specification, a design specification and source code.

**Unit testing:**

Unit testing is essential for the verification of the code produced during the coding phase and hence the goal is to test the internal logic of the modules. Using the detailed design description as a guide, important paths are tested to uncover errors with in the boundary of the modules. These tests were carried out during the programming stage itself. All units of Vienna SQL were successfully tested..

**Integration testing:**

Integration testing focuses on unit tested modules and build the program structure that is dictated by the design phase.

**System testing:**

System testing tests the integration of each module in the system. It also tests to find discrepancies between the system and it’s original objective, current specification and system documentation. The primary concern is the compatibility of individual modules. Entire system is working properly or not will be tested here, and specified path ODBC connection will correct or not, and giving output or not are tested here these verifications and validations are done by giving input values to the system and by comparing with expected output. Top-down testing implementing here.

## Acceptance Testing:

This testing is done to verify the readiness of the system for the implementation. Acceptance testing begins when the system is complete. Its purpose is to provide the end user with the confidence that the system is ready for use. It involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements.

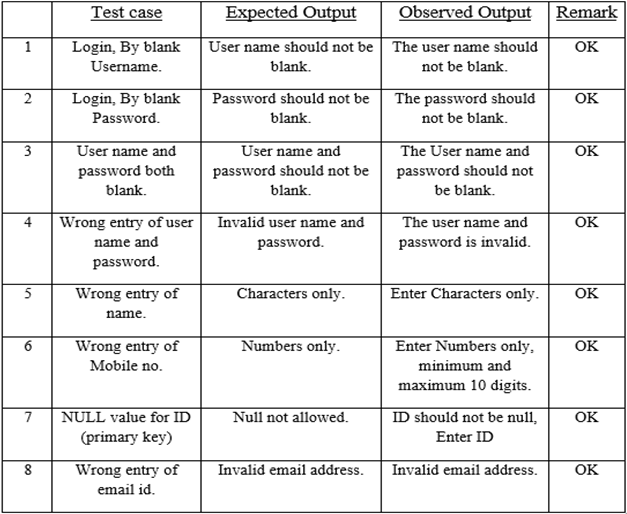
**Tools to special importance during acceptance testing include:**

Test coverage Analyzer – records the control paths followed for each test case.

eTiming Analyzer – also called a profiler, reports the time spent in various regions of the code are areas to concentrate on to improve system performance.

Coding standards – static analyzers and standard checkers are used to inspect code for deviations from standards and guidelines.

**8.5 Test Report:**



**9. Future Enhancement:**

* Navigation to the parking area.
* The location of the user from where he is booking for the security
* Message alert

**9.1Conclusion:**

This project is actually designed to automate the work of maintaining records and to help the people in booking of the area and also to chack the near by service station to the use to help him easyly.Admin can add new members to the system in three type as user , parking owner and service stationsas well as the parking owner , user, service station can add them self to our system.Admin can see all the details of the parking owner and service station and users.parking owner and service stations can update there facilities. And even user can give the feedback which can be seen to the Admin as well as the parkin owner or the services station to whom they have give the feedback.

**REFERENCES:**

**Text Books:**

* The Complete Reference (Helbertschildtpublished fifth edition)
* Server side programming with (Aptech J2EE book JSP & Servlet)

**Web sites:**

* Google.
* Encyclopedia.
* Wikipedia.
* http://Java.sun.com/j2ee/faq/html
* http://Java.sun.com/products/jdbc/reference/faqs/index.html
* http://www.apt.jhu.edu/~hall/java/servlet-Tutorial
* [**http://www.projectsof8051.com/automatic-car-parking-indicator-system/**](http://www.projectsof8051.com/automatic-car-parking-indicator-system/)
* [**http://www.projecttopics.info/**](http://www.projecttopics.info/)
* [**http://www.sciencedirect.com/science/article/pii/S1665642313715803**](http://www.sciencedirect.com/science/article/pii/S1665642313715803)
* [**http://www.docfoc.com/proposal-report-vehicle-parking-management-system-USZE**](http://www.docfoc.com/proposal-report-vehicle-parking-management-system-USZE)
* **<http://www.slideshare.net/IJMER/ijmer-42027277>**