**SUMMARY**

This project has been developed using **Servlets** framework, **NetBeans** platform, **Glassfish** server and **Oracle** database.

**Servlets** is an application development framework that is designed for and used with the popular J2EE (Java 2, Enterprise Edition) platform. It cuts time out of the development process and makes developers more productive by providing them a series of tools and components to build applications with. It is non-proprietary and works with virtually any J2EE-compliant application server. Servlets with an Open Source license (meaning it has no cost and its users have free access to all its internal source code). Servlets also helps make your end work products better. So many people are using and developing Servlets that bugs are found and get fixed quickly. Servlets is based on the time-proven **Model-View-Controller (MVC)** design pattern. The MVC pattern is widely recognized as being among the most well-developed and mature design patterns in use. By using the MVC design pattern, processing is broken into three distinct sections aptly named the Model, the View, and the Controller.

The **Netbeans** Platform is designed for building integrated development environments (IDEs). It can be used to create diverse end-to-end computing solutions for multiple execution environments. Netbeans Project is an open source framework that provides many of the underlying services software developers need. This would be a "toolkit for designing toolkits." The Netbeans Platform is the foundation for constructing and running integrated end-to-end software development tools. The platform consists of open source software components that tool vendors use to construct solutions that plug in to integrated software workbenches.

In this project, a user must be registered with the system to order the items he wish to buy from the company. If he is not, he must sign up first. Each employee has his/her own username and password. The employee logs in into the system. He can perform the following applications:

* Order Item he/she wishes to buy
* View Previous Orders Booked
* Check the status of the orders
* View and edit his personal information.
* Update his password.

Business Rules includes all the rules for filling the compensation sheet.

Employee Compensation Sheet allows the employee to fill in his compensation structure and submit it for approval.

With check status the employee can view the status of all the compensation sheets submitted by him.

There is only 1 manager for this system. A separate link is provided to login as manager. All the sheets that are submitted by the various employees and are in pending status are known to the manager and can be accepted/rejected by him.

When the employee logins in back, he can view the status of his pending sheet.

Oracle 10g has been used to create a database having following tables:

1. registration: Stores the details of all the registered employees.
2. login: Stores the username, password.
3. items: Stores the items supplied by the company.
4. booking

Further tables are added by the programming logic.

**INTRODUCTION TO THE DEVELOPMENT ENVIRONMENT**

***HTTP (Hypertext Transfer Protocol)*** is similar to many other Internet protocols, like FTP (File Transfer Protocol) and SMTP (Simple Mail Transfer Protocol), because HTTP is a protocol to transfer data from the server to the client. One fundamental difference between most other Internet protocols and HTTP is that the typical Internet protocol supports several requests per connection, HTTP supports only one request per connection. This means that with HTTP the client connects to the server to retrieve one file and then disconnects. This mechanism allows more users to connect to a given server over a period of time. Therefore HTTP is called the stateless protocol, that is the server need not to hold the connection and hence no need to manage the connection’s state. The stateless nature of HTTP is fundamental to the efficiency and scalability of what is called the World Wide Web (www).

HTTP was created in conjunction with the related HTML (Hypertext Mark-up Language) standard. HTML is a document display language that lets users link from one document to another. HTML also permits images and other media objects to be embedded in an HTML document. The media objects are stored in files on a server. HTTP also retrieves these files. The combination of the HTTP protocol and the HTML page description language constitutes the foundational technologies of the World Wide Web.

**HTTP CLIENT-SERVER ARCHITECTURE**

The web browser sends a single request to the server. The web server determines which file is being requested and sends the data in that file back as the response. The browser interprets the response and represents the content on the screen.

The request information consists of the location of the file, or other resource that the user wants, and information about the browser. The response information contains the requested resource and other information. The request is particularly in plain text. The response can be plain text or part plain text, part binary data.

***Servlets*** was developed by Sun Microsystems as an advance over traditional Common Gateway Interface (CGI) technology. A Java servlet is a Java technology program that, similar to a CGI program, runs on the server. The types of tasks that can be run are similar to those as of CGI, however underlying architecture is different.

As with CGI scripts, servlets can recognize HTTP requests, generate the response dynamically, and then send a response containing an HTML page or document to the browser.

The basic processing steps for java servlets are quite similar to the steps for CGI. However, the servlet runs as a thread in the web container instead of in a separate OS (operating System) process. The web container itself is an OS process, but it runs as a service and is available continuously. This is opposed to a CGI script in which a new OS process and shell is created for each request.

When the number of requests for a servlet rises, no additional instances of the servlet or OS processes are created. Each request is processed concurrently using one Java thread per request.

Servlets run within the Java Enterprise Edition (J2EE) component container architecture. This container is called the web container. The web container is a java Virtual Machine (JVM) tool interface that supplies an implementation of the servlet API. Servlet instances are components that are managed by the web container or respond to HTTP requests.

**THREE-TIER ARCHITECTURE**

The three-tier architecture has a user-interface client that runs on a user’s machine that communicates with business-logic software on a server machine, which in turn communicates with software responsible for the long-term storage of data on database machines. By acknowledging the three tiers, web application development can be conceptualized as modules with simple, well defined interfaces.

The user interface resides in presentation (or client) tier. The business-logic software resides in business-logic (middle ware) tier.

This architecture is also referred as multi-tier or *n-tier* architecture.

In a two-tiered architecture the client communicates directly with the data tier. A three-tier architecture separates the components of a two-tier architecture by adding a business logic tier between the data and client tiers. The problem with two-tiered architecture is the client and the data are inflexibly bound to each other. If the data structures change, a two-tierded web application might not work until the client tier is updated to reflect the changes. When the client must be updated on many user systems, the process of changing an application can be quite vexing.

By introducing a middle ware layer, changes in the data layer can be masked by updating the middle ware to account for the changes. New applications can be created by applying new business-logic to the existing data structures. Changes to the data layer should be avoided because data is the most important for modern business applications.

**Model-View-Controller (MVC)**

Servlet is based on the time-proven Model-View-Controller (MVC) design pattern. The MVC pattern is widely recognized as being among the most well-developed and mature design patterns in use. By using the MVC design pattern, processing is broken into three distinct sections aptly named the Model, the View, and the Controller. These are described in the following subsections:

#### Model Components

Model components provide a "model" of the business logic or data behind a Servlets program. For example, in a Servlets application that manages customer information, it may be appropriate to have a "Customer" Model component that provides program access to information about customers.

It's very common for Model components to provide interfaces to databases or back-end systems. For example, if a Servlets application needs to access employee information that is kept in an enterprise HR information system, it might be appropriate to design an "Employee" Model component that acts as an interface between the Servlets application and the HR information system.

Model components are generally standard Java classes. There is no specifically required format for a Model component, so it may be possible to reuse Java code written for other projects.

#### View Components

View components are those pieces of an application that present information to users and accept input. In Servlets applications, these correspond to Web pages.

View components are used to display the information provided by Model components. For example, the "Customer" Model component discussed above would need a View component to display its information. Usually, there will one or more View components for each Web page in a Servlets application.

View components are generally built using JavaServer Page (JSP) files. Servlets provides a large number of "JSP Custom Tags" (sometimes referred to as Servlets Tags) which extend the normal capabilities of JSP and simplify the development of View components.

#### Controller Components

Controller components coordinate activities in the application. This may mean taking data from the user and updating a database through a Model component, or it may mean detecting an error condition with a back-end system and directing the user through special error processing. Controller components accept data from the users, decide which Model components need to be updated, and then decide which View component needs to be called to display the results.

One of the major contributions of Controller components is that they allow the developer to remove much of the error handling logic from the JSP pages in their application. (After all, if errors in processing occur, the Controller component forwards to an error-processing View component, not the primary results View component.) This can significantly simplify the logic in the pages and make them easier to develop and maintain.

Controller components in Servlets are Java classes and must be built using specific rules. They are usually referred to as "Action classes."

The advantage of the MVC paradigm is an application architecture that separates the three concerns into separate software modules that communicate with each other using a relatively simple interface. The advantage of separating these concerns has been code that is much easier to debug because analysis can focus on the interface between the modules and the source of problems can be more readily isolated.

#### ADVANTAGES OF USING SERVLETS

* Each request is run in a separate thread, so servlet request processing is significantly faster than traditional CGI processing.
* Servlets are scalable. Many more requests can be executed because the web container uses a thread rather than an OS process, which is a limited system resource.
* Servlets are robust and object oriented as servlets can only be written in Java programming language.
* Therefore servlets are platform independent.
* Servlets have access to login capabilities, most CGI programs do not.
* The web container provides additional services to the servlets, such as error handling and security.

**DISADVANTAGES OF USING SERVLETS**

* Servlets often contain both business logic and presentation logic.

Presentation logic is anything that controls how the application presents information to the user. Generating the HTML response within the servlet code is presentation logic.

Business logic is anything that manipulates data to accomplish something, such as storing data.

Mixing presentation and business logic means that whenever a webpage changes the servlets must be rewritten, recompiled, and redeployed.

This disadvantage led to the development of template pages, including JavaServer Pages (JSP) technology

* There are concurrency issues.

**JavaServer Pages (JSP) Technology**

All template page technologies have the same fundamental structure: an HTML page that a web designer can easily create, with special tags that indicate to the web server that code needs to be executed at request time.

JSP pages are opposite of servlets. Instead of Java technology code that contains HTML, template pages are HTML that contains Java technology code. JSP pages are converted by the web container into a servlet instance. That servlet then processes each request to that JSP page.

The JSP page runs as a servlet. Everything that can be done in a servlet, can be dine in JSp page.

**ADVANTAGES OF USING JSP PAGES**

* Web applications using JSP pages have high performance and scalability because they use threads rather than the OS’s shells or processes.
* JSP technology is built on Java technology, so it is platform independent.
* JSP scripting can be written in Java language so that JSP pages can take advantage of the object oriented language and its API’s.

**DISADVANTAGES OF USING JSP PAGES**

* If JSP pages are used in isolation, then the scripting code that performs business and control logic can become cumbersome in the JSP pages.
* JSP pages are difficult to debug.
* There is separation of concerns into business logic and presentation logic.
* There are concurrency issues.

**JDBC**

JDBC (Java Database Connectivity) is an application program interface (API) specification for connecting programs written in Java to the data in popular database. It is provided by Sun Microsystems, the application program interface lets you encode access request statements in structured query language (SQL) that are then passed to the program that manages the database. It returns the results through a similar interface.   
From the user’s point of view, Java application looks something like this:

**Features of JDBC API:**

Java Database Connectivity (JDBC) provides a database programming API for Java programs. Some of the features of JDBC API are as follows:

• Contains a set of classes and interfaces that are used to connect to a database built using any DBMS/RDBMS, submit SQL queries to a database, and retrieve and process the results of SQL queries.

• Is a low-level interface in which SQL select and update statements are called directly from within Java programs.

• Can be used with both two-tier and three-tier database architectures. In two-tier architecture, a Java program invokes the methods of JDBC API, which in turn communicates with the database server. In three-tier architecture, a Java applet or an HTML form submits SQL queries to a middle-tier server. Middle-tier server in turn uses JDBC API to communicate with the database server.

**JDBC Architecture:**  
The JDBC architecture is based on a collection of Java interfaces and classes that together enables you to connect to data sourced, to create and execute SQL statements, and to retrieve and modify sata in a database. These operations are illustrated in the figure below:  
Each of the boxes in the illustration represents a JDBC class or interface that has fundamental role in accessing a relational database.

**JDBC Drivers:**  
JDBC API takes care of converting Java commands to generic SQL statements. However, to address specific database issues, each database vendor provides a driver along with the database. Java Applications invoke the methods JDBC API. JDBC API in turn uses a driver to communicate with a specific database.  
JDBC API submits queries to the JDBC driver. The JDBC driver converts queries to a form that a particular DBMS/RDBMS can understand. The JDBC driver also retrieves the results of SQL queries, converts it into equivalent JDBC API classes and objects that can be used by the application. Since the JDBC Driver only takes care of the interactions with database, any change made to the database does not affect the application. 

**JDBC Architecture**  
There are several categories of JDBC Drivers provided by different database vendors. They are:

**JDBC-ODBC Bridge Driver:** The first category of JDBC drivers provides a bridge between the JDBC API and the ODBC API. There are several DBMS/RDBMS, such as MS Access and SQL Server that contain the ODBC Driver embedded into them. Since the ODBC API is written in the C language and makes use of pointers and other constructs that Java does not support, a Java program cannot directly communicate with an ODBC Driver. The bridge translates the standard JDBC calls to corresponding ODBC calls, and sends them to ODBC data source via ODBC libraries. 

**Native API Partly Java Driver:** These drivers use a mixture of Java implementation and vendor specific native APIs to provide data access. JDBC database calls are translated into vendor specific API calls. The database will process the request and sends the result back through the API, which will in turn forward them back to the JDBC driver. The JDBC driver will translate the result to the JDBC standard and return them to the Java application. There is one layer fewer to go through than for a type 1 driver and so in general a type 2 driver will be faster than a type 1 driver.  
Some DBMS/RDBMS such as DB2 and Informix contain a JDBC driver supplied by the database vendor.

**Intermediate Database Access Server**: Type 3 drivers use an intermediate database server that has the ability to connect multiple Java clients to multiple database servers. Clients connect to database server via an intermediate server component that acts as a gateway for multiple database servers. The java client application sends a JDBC call through a JDBC driver to the intermediate data access server, which completes the request to the data sourcing using another driver ( for example, a type 2 driver).

BEA WebLogic includes a type 3 driver. One of the benefit of using a type 3 driver is that it allows flexibility on the architecture of the application, as the intermediate server can can abstract details of connection to database servers.  
Native Protocol Pure Java Driver/JDBC-Net Pure Java Driver: These drivers convert the JDBC API calls to direct network calls using vendor specific networking protocol. They do this by making direct socket connections with the database. Type 4 drivers offer better performance than others.

***The Netbeans IDE*** is an integrated development environment for creating, debugging, and deploying Java applications.

**NetBeans IDE 7.0** introduces language support for development to the Java SE 7 specification with the JDK 7 language features, support for GlassFish 3.1, Oracle WebLogic, Oracle Databse, Maven 3, HTML 5 and more.

A free, open-source Integrated Development Environment for software developers. All the tools needed to create professional desktop, enterprise, web, and mobile applications with the Java platform, as well as with C/C++, PHP, JavaScript and Groovy.

**Modularity**: All the functions of the IDE are provided by modules. Each module provides a well defined function, such as support for the Java language, editing, etc. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules

The **NetBeans Platform** is a reusable [framework](http://en.wikipedia.org/wiki/Software_framework) for simplifying the development of [Java Swing](http://en.wikipedia.org/wiki/Java_Swing) desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download [digitally-signed](http://en.wikipedia.org/wiki/Digital_signature) upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

* User interface management (e.g. menus and toolbars)
* User settings management
* Storage management (saving and loading any kind of data)
* Window management
* Wizard framework (supports step-by-step dialogs)
* NetBeans Visual Library
* Integrated Development Tools

**Compatibility**

The Netbeans Platform has been released for and tested on Windows NT, Windows XP, Windows 2000, Windows 98, Windows ME, and Red Hat Linux® V7.1

NetBeans IDE runs on the Java SE Development Kit (JDK) which consists of the Java Runtime Environment plus developer tools for compiling, debugging, and running applications written in the Java language.

**Advantages of Netbeans :**

Open source is the only way to deliver an open platform for tool integration. But open source also has several other advantages, including:

1. Reuse — Reuse is the highest form of compliment. Why rebuild something when it exists in a working format already? By using the open Netbeans Platform, tool builders can focus on domain specific expertise and functions (thus focusing on what makes them successful), while providing the tooling infrastructure for building IDEs. But reusing someone else's code takes trust.
2. Confidence — The Netbeans Platform builds confidence and trust by providing the source code for the platform. Open source provides all of the APIs, without internal, proprietary or hidden interfaces. Developers, whose trust is earned slowly, can look at the source and learn. Trust the source, then launch innovation.
3. Quality — Open source consortiums also can produce a higher quality of code. When code review is collaborative, people put extra effort into it. The source that they contribute becomes a reflection of the work that they do, establishing both individual and corporate reputation. Trust the source, then launch market position.
4. Clarity — Open source based upon clear specifications can deliver code that is easier to understand. An Interface describes (in black-box terms) the promise of component behavior. By directly inspecting the source, you can examine, line by line, how the code works. It is hard to trust someone else's interface. Trust the source, then inspect technology.
5. Simplicity — Open source can be easier to debug. Late at night, when encountering a bug, source code can speed identification of the root cause. It could be your fault, or the fault of the platform and environment. With access to the source, guesswork is eliminated. With access to a collaborative discussion forum, it's even possible to compare notes with someone else familiar with the environment or problem. If the problem appears to be in shared open source, it's easy to patch it and attempt a workaround. Trust the source, then verify the base. It's difficult to work alone on complex technology. Few tool builders want to take on the risk of pioneering new tooling technology. Trust the source, then share the risk.
6. Longevity — Tool vendors come, and tool vendors go. Corporate developers want to know that a platform can provide long-term support. With the source code, corporations can start quickly in the short term and sustain business for the long term. Trust the source, then build a business.
7. Flexibility — Flexibility is a fundamental value of Netbeans. With the open Netbeans Platform, an unsatisfactory component can be modified to suit your needs. For example, if the editor is inadequate, create your own, or plug in a popular one created in the open component market established by the Netbeans Platform. Want to tie a new deployment platform (for example, a set-top box) into existing end-to-end support? Trust the source, then create a plug-in.

**INTRODUCTION TO THE PROJECT**

***‘blueberry’ ONLINE SHOPPING CART***

This is designed to provide the facility to the customers of booking the order online. Once the order is placed the store person would assemble the order in one place. The customer can visit the store, pay for the order and take away his parcel in no time. It is aimed at customers who find it difficult to visit a store to buy grocery, or breads, or any other items by reducing customer’s time. Further Online payment and home delivery of the products can be added.

**Advantages:**

* Provides online interface to employees for applying leaves
* Would keep the track of all kind of leaves
* Will help in implementing leave policy efficiently
* Will provide a central repository of records, which can be churned out for n number of purposes.
* Automation makes the entire system more efficient and consistent
* Time Saving(otherwise spent in manual application and approvals)
* Cost Reduction, in terms of man power and documentation
* Manual labour saved
* Transparency of the system making employees and managers accountable
* Dynamic report generation
* Latent info like the which managers/employee are regular, who submit applications in time, efficiency of approvals, strictness of management etc. can be inferred by the system generated reports

ROLES

* Customer
  + Order Online
  + Check Order Status
  + View Previous Orders
  + View & Edit Personal Details
  + Change Password
* Employee
  + View Customer’s Orders
  + Assemble Order and Update Order Status
* Store Manager
  + Update Items In Database
  + Add Items In Database
  + Update Item Quantity, Price In Database
  + View Registered Users, Orders Booked, Order Status

DATA REPOSITORY

* login Table – stores user ids and password of all employees and whether they are managers or not
* registration Table – stores details of all the customers and employees and managers
* items Table – stores details of all the items
* booking\_history – stores all the bookings done by customers with code
* bh<code> - stores particular order

*Preliminary Investigation*

**INTRODUCTION**

The objective of this project is to develop a general purpose e-commerce store where any product (such as books, CDs, computers, mobile phones, electronic items, and home appliances) can be bought from the comfort of home through the Internet.

An online store is a virtual store on the Internet where customers can browse the catalog and select products of interest. The selected items may be collected in a shopping cart. At checkout time, the items in the shopping cart will be presented as an order.  
  
This document explores the basic features of the proposed system and its various functionalities.

**SCOPE**

* The project is designed to encompass three roles – Customer, Employee and Store Manager
* It shall be a password protected system
* Three-tier architecture will be followed: Presentation layer, Business logic and Database Storage
* Booked Orders can be viewed
* Booking History, Customer Information can be viewed.

**EXCLUSIONS**

* Online Payment Of order through certain Bank.
* Shipment of Order.
* Cancellation of Order.
* Dynamic report generation
* Simultaneous Multi user logins

**SOFTWARE & HARDWARE SPECIFICATIONS**

* J2EE architecture shall be used
* Front-end will be made using Servlets and JSP’s.
* Application will use java classes and other components
* Database will be made using Oracle 10g Server
* The system will be platform-independent and require a web browser.

**FUNCTIONALITY**

* Store Manager
  + Update Items In Database
  + Add Items In Database
  + Update Item Quantity, Price In Database
  + View Registered Users, Orders Booked, Order Status
* Employee
  + View Customer’s Orders
  + Assemble Order and Update Order Status
* Customer
  + Order Online
  + Check Order Status
  + View Previous Orders
  + View & Edit Personal Details
  + Change Password

**DATA REPOSITORY**

As a part of the preliminary study it has determined that the following tables will be required

* Bookings Date and Codes
* Every Booking Details
* Login Names and Password
* Registered Users Details
* Store Stock Item Details

**ADDITIONAL FEATURES**

The following features will be explored if time/cost permits

* Multiple logins
* Administration responsibilities
* Integration of authentication and authorization- making it a single sign-in system
* Online Payment Of Order, thus introducing e-commerce on this website.
* Shipping Details and Sur Charge Calculations

**FEASIBILITY STUDY**

**OPERATIONAL FEASIBILITY**

A development of any infringement, violation or liability that could result from development of the system makes it operationally infeasible. There is no law or rule, which will be violated by the development of this software.

The proposed system meets the requirements of the hour and provides for future updations. The functionality is basic and easily understood. Its user-friendly and easily maintained.

**TECHNICAL FEASIBILITY**

It’s the study of function, performance, and constraints that may affect the ability to achieve an acceptable system. It’s the most difficult area to assess at this stage of the product engineering process. It is essential that the process of analysis and definition be conducted in parallel with an assessment of technical feasibility.

The considerations normally involved are:-

**Development Risk: -** Can the system element be designed so that necessary function and performance are achieved within the constraints uncovered during analysis?

* Our project performs all the needed performance and functions in accordance with the analysis of our project. No such new materials, methods, algorithms or processes were required whose development risk is high.

**Resource Availability: -** Are skilled staff available to develop the system element in question? Are the hardware and software resources required available to build the system?

* Our project involves java technology, servlets, and JSP, SQL, Oracle 10g which are well within the scope of the developer. The hardware and software resources required by the project are quite easily available and no special hardware or software resource was required by the application which could not be made available.

**ECONOMIC FEASIBILITY**

It’s an evaluation of development cost weighed against the ultimate income or benefit derived from the developed system or product. It includes a broad range of concerns that include cost-benefit analysis, cost of resources needed for growth etc.

Our project has a positive cost benefit analysis i.e. cost of project is much less than the benefits achieved by the system. The project does not require any special resources for its development and functioning, neither some special hardware resources nor any costly software. It is fully functional using a system having windows environment. The cost of our project is the total cost of all the peripherals like monitor, keyboard, mouse, processor, memory. The software’s used are easily available and don’t cost a thing. .

Thus, from the economy point of view our project does not require a lot of monetary inputs.

***THE PROJECT IS DETERMINED TO BE FEASIBLE AND ADVANTAGEOUS***

*Requirement Analysis*

**INTRODUCTION**

The projects main aim is to provide e-commerce facility to customers. With growing size and complexity of businesses, a common man becomes to occupied by the work load and finds it difficult to take out time to even buy grocery. Our project solves customer problems by providing them the facility to order the items required just by a single click. Thus online shopping helps customer to save time.

**SCOPE**

The project offers a great scope in this e-commerce world where customers have become a key resource or most valued resource for a company. Our project can be used in any company offering the facility of e-commerce or online shopping.

* The project is designed to encompass three roles – Customer, Employee and Store Manager
* It shall be a password protected system
* Three-tier architecture will be followed: Presentation layer, Business logic and Database Storage
* Booked Orders can be viewed
* Booking History, Customer Information can be viewed.

#### Overview of the functional requirements

* The customer, employees and store managers must be able to connect to the server through the authorization process i.e. by using their id and password.
* The customer, employees and store managers should be able to login at any time through internet.
* The customer should be registered before using the facilities of the system.
* The customer must be able to select and prepare a shopping cart.
* Customer can view and update his personal details stored in database.
* Customer should be able to view his previous bookings.
* There should be a record of all bookings. The code and date of booking should be saved.
* The employee should be able to view customers cart and after packing the order in a bag change the order status.
* Store Manager should be able to update the items in database.

**Overview of the data requirements**

* The customer, employees and store manager need to enter their login id and password before coming into action.
* After the customer is logged into the system he chooses one of the menu option available.

**User view of the product**

There are three views of our product one each for customer, employee and store manager, which is distinguished on the basis of the login information provided in the login page. The customer can login by typing in his id and password in his login box or he can register if he is a new user. On clicking on the new user or register link, a page will open containing fields to be filled by the customer. If he is a old user and has logged in to the site a page will appear which provides links for the customer to go to view items and build a shopping cart, to view or update his personal details, view booking history.

On the employee side, when he logs in he is presented with a page which shows him the orders with their status. He can choose pending order and after update it as confirmed order.

On the store manager side, when he logs in to the system he is presented with a page with links for updating or inserting new items, or running SQL commands.

All users are given an option of logging out and closing their session.

**PLATFORM SPECIFICATION**

**Hardware Requirements:**

* Pentium II or higher processor. (Or equivalent).
* 3 GB or more hard disk space.
* 128 Mb RAM

**Software Requirements:**

* Windows–9X or Windows-NT or any advanced windows operating system.
* Application tool – Java Development Kit.
* Internet Explorer

**TECHNOLOGIES TO BE USED:**

1. Java
   * Java is a platform independent, object-oriented, robust, secure and innovative programming language.
   * It consist of Two Parts
     + JVM (Java Virtual Machine), which is software component that is used to execute java programs.
     + Java API (Application Programming Interface) that consist of inbuilt classes that are used in different programs.
2. Database Programming
   * JDBC is Java Database Connectivity that is used for communication between database server and a java application. JDBC is a specification and API that is in programming database applications in java.
3. Web Page Programming
   * Servlet is based on the time-proven Model-View-Controller design pattern which is widely recognized as being among the most well-developed and mature design patterns in use. By using the MVC design pattern, processing is broken into three distinct sections aptly named the Model, the View, and the Controller.

**MODULES:**

Following modules are to be developed:

1. Login
2. User Home Screen
3. Shopping Item View
4. Saving Cart
5. Booking History
6. View And Edit Personal Details
7. Change Password
8. Employee Login
9. Employee to view Bookings and change status
10. Administrator/Store Manager Login
11. Updating/Inserting Items
12. SQL Execution

**BUSINESS OVERVIEW AND BENEFITS**

**Overview:**

The Leave Management (LM) is a bolt-on solution for a company managing a large staff and hosted on its intranet under HR applications. All employees on the permanent rolls of the company and its subsidiary companies can access the solution through the “Employee Self-Service”. The said system provides flexibility to an employee in applying for leave, cancelling a leave and viewing records related to his/ her own leave. The system will also enable managers to sanction leave as well as view leave records of those reporting to them, online.

**Benefits of application:**

* Provides online interface to employees for applying leaves
* Would keep the track of all kind of leaves
* Will help in implementing leave policy efficiently
* Will provide a central repository of records, which can be churned out for n number of purposes.
* Automation makes the entire system more efficient and consistent
* Time Saving(otherwise spent in manual application and approvals)
* Cost Reduction, in terms of man power and documentation
* Manual labour saved
* Transparency of the system making employees and managers accountable

**BUSINESS FUNCTIONAL REQUIREMENT**

|  |  |  |
| --- | --- | --- |
| Business Functional Requirement ID | Requirement | Priority  E-Expected ; O-Optional |
| Customer Role | | |
| BRF1 | Registration   * New customers can register on the website | E |
| BFR2 | Personal Information   * All the details of the employee captured from DB will be displayed | E |
| BRF3 | Booking an order   * An order form will be displayed. * All the items in database will be displayed and user can choose the items he wishes to buy * Details of the booked items are displayed * Customer can delete items he ordered before saving the cart if he wishes to * Customer can Save the cart | E |
| BRF4 | Changing Password   * Customer is given an option to change his password by providing correct old and new matching passwords | E |
| BRF5 | Editing Personal Details   * Customer has an option to update his personal details stored in database | O |
|  |  |  |
| Employee Role | | |
| BRF6 | Viewing Booked Order   * All the pending requests and their details of employees under that manager will be displayed | E |
| BRF7 | Booking Status   * The Booking status of the orders can be updated by an employee from ‘pending’ to ‘confirmed’ | E |
| BRF8 | Report   * A day to day report of the no. of orders taken by employee | O |
|  |  |  |
| Manager Role | | |
| BRF9 | Inserting new Items   * Insert new items and its corresponding details in the database | E |
| BRF10 | Updating Items   * Updating existing items details in the database | E |
| BRF11 | Running Database Commands   * Running SQL Queries by an administrator | O |
| BRF12 | Report   * A monthly report of the no. of orders booked by customers | O |

**ACCEPTENCE CRITERIA TABLE**

|  |  |  |
| --- | --- | --- |
| Business Functional Requirement ID | Description | Acceptance Criteria |
| Customer Role | | |
| BRF1 | Registration   * New customers can register on the website | Web page conating a registration form is displayed. Valid Information results in creation of a new user account and insertion of information in database |
| BFR2 | Personal Information   * All the details of the employee captured from DB will be displayed | Web Page containing all personal information of the customer fetched from database is displayed |
| BRF3 | Booking an order   * An order form will be displayed. * All the items in database will be displayed and user can choose the items he wishes to buy * Details of the booked items are displayed * Customer can delete items he ordered before saving the cart if he wishes to * Customer can Save the cart | Web Page with all the items available in database is shown along with the input of quantity for each item displayed. Customer fills the form and the form is processed and his booked items are entered in a cart and displayed.  He can delete certain items if he wishes to or proceed to save his cart.  The status of the cart is set as ‘pending’ |
| BRF4 | Changing Password   * Customer is given an option to change his password by providing correct old and new matching passwords | Web page asking for old password, new password, Confirm New Password is displayed.  All correct entries results in updation of password |
| BRF5 | Editing Personal Details   * Customer has an option to update his personal details stored in database | Web page containing a form similar to registration is displayed.  Valid entry results in updation of information |
|  |  |  |
| Employee Role | | |
| BRF6 | Viewing Booked Order   * All the pending requests and their details of employees under that manager will be displayed | Order booked by customer is displayed for assembling and packing |
| BRF7 | Booking Status   * The Booking status of the orders can be updated by an employee from ‘pending’ to ‘confirmed’ | Once order is packed, the status of the order is changed from ‘pending’ to ‘confirmed’ |
| BRF8 | Report   * A day to day report of the no. of orders taken by employee | Report generation of all the day to day transactions |

|  |  |  |
| --- | --- | --- |
| Manager Role | | |
| BRF9 | Inserting new Items   * Insert new items and its corresponding details in the database | Form for entering new items details is shown. Correct entries results in successful insertion of data in database |
| BRF10 | Updating Items   * Updating existing items details in the database | Form for updating the item details is shown. Correct entries result in successfully updating the data |
| BRF11 | Running Database Commands   * Running SQL Queries by an administrator | Web Page for executing SQL queries |
| BRF12 | Report   * A monthly report of the no. of orders booked by customers | Monthly statement generation |

**LIMITATIONS**

* Online Payment is not provided at this stage.
* Shipment of order is not provided at this stage.
* Cancellations are not provided at this stage.
* Change of Password in case Customer forgets it is not included at this stage.

*Design Phase*

**VIEW DESIGNS**

Personal Details

Booking History

Change Password

Personal Details

Login Page

**Employee Pages**

Homepage

Back to Homepage

User Details are displayed

Edit Details

Shopping

SUBMIT

RESET



Password

Edit Details

Form to update personal details

Back to Homepage

Shopping

Form for shopping items

Back to Homepage

Cart

Shopping Cart Is Displayed

Back to Homepage

Booking History

Previous Bookings Are Displayed

Back to Homepage

Employee

All Bookings Are Displayed

Back to Homepage

Employee Updation of Bookings

Booking Details Are Displayed

Back to Homepage

Change Status Option

Manager Login

Insert/Update

SQL Queries

Back to Homepage

SQL Queries

Text Box to run SQL Queries and display result below the textbox

Back to Homepage

Insert/Update

Form to insert/update items

Back to Homepage

**DATABASE RELATION DIAGRAM**

**‘login’** table consists of following structure:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Default** | **Primary Key** |
| USERNAME | VARCHAR2(20) | No | - | - |
| PASSWORD | VARCHAR2(20) | No | - | - |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 -2 |  |  | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Constraint** | **Type** | | **Table** | **Search Condition** | **Delete Rule** | **Status** | **Last Change** | **Index** | **Invalid** | | LOGIN\_FK | | R | LOGIN | - | CASCADE | ENABLED | 09-SEP-11 | - | - | | SYS\_C004318 | | C | LOGIN | "USERNAME" IS NOT NULL | - | ENABLED | 09-SEP-11 | - | - | | SYS\_C004319 | | C | LOGIN | "PASSWORD" IS NOT NULL | - | ENABLED | 09-SEP-11 | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 3 |  |  | | | | | | | | | | |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |

**‘registration’** table consists of following structure:

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Column Name** | **Data Type** | **Nullable** | **Default** | **Primary Key** | | USERNAME | VARCHAR2(20) | No | - | 1 | | NAME | VARCHAR2(20) | No | - | - | | DOB | DATE | No | - | - | | EMAIL | VARCHAR2(40) | No | - | - | | PHONE | NUMBER | No | - | - | | BOOKINGS | NUMBER | No | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 6 |  |  | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Constraint** | **Type** | **Table** | **Search Condition** | **Delete Rule** | **Status** | **Last Change** | **Index** | **Invalid** | | REGISTRATION\_PK | P | REGISTRATION | - | - | ENABLED | 09-SEP-11 | REGISTRATION\_PK | - | | SYS\_C004125 | C | REGISTRATION | "BOOKINGS" IS NOT NULL | - | ENABLED | 06-AUG-11 | - | - | | SYS\_C004324 | C | REGISTRATION | "NAME" IS NOT NULL | - | ENABLED | 09-SEP-11 | - | - | | SYS\_C004325 | C | REGISTRATION | "EMAIL" IS NOT NULL | - | ENABLED | 09-SEP-11 | - | - | | SYS\_C004326 | C | REGISTRATION | "PHONE" IS NOT NULL | - | ENABLED | 09-SEP-11 | - | - | | SYS\_C004327 | C | REGISTRATION | "DOB" IS NOT NULL | - | ENABLED | 09-SEP-11 | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 6 |  |  | | | | | | | | | |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |
| |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |

‘**items’** table consists of following structure:

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Column Name** | **Data Type** | **Nullable** | **Default** | **Primary Key** | | ITEMCODE | VARCHAR2(20) | No | - | 1 | | NAME | VARCHAR2(20) | Yes | - | - | | QTY | VARCHAR2(20) | Yes | - | - | | ITEMPRICE | VARCHAR2(20) | Yes | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 4 |  |  | | | | | |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Constraint** | **Type** | **Table** | **Search Condition** | **Delete Rule** | **Status** | **Last Change** | **Index** |  | | ITEMS\_PK | P | ITEMS | - | - | ENABLED | 09-SEP-11 | ITEMS\_PK | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 1 |  |  | | | | | | | | | | |  |
|  |  |

‘**bh’** table consists of following structure:

|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Column Name** | **Data Type** | **Nullable** | **Default** | **Primary Key** | | ITEMCODE | VARCHAR2(20) | Yes | - | - | | NAME | VARCHAR2(20) | Yes | - | - | | QTY | VARCHAR2(20) | Yes | - | - | | ITEMPRICE | VARCHAR2(20) | Yes | - | - | | TOTAL | VARCHAR2(20) | Yes | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 5 |  |  | | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Constraint** | **Type** | **Table** | **Search Condition** | **Delete Rule** | **Status** | **Last Change** | **Index** | | **Invalid** | | | SYS\_C004339 | C | BH\_SHOBHIT\_1 | "TOTAL" IS NOT NULL | - | ENABLED | 09-SEP-11 | | - | | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 1 |  |  | | | | | | | | | | | |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |

**‘booking\_history’** table consists of following structure:

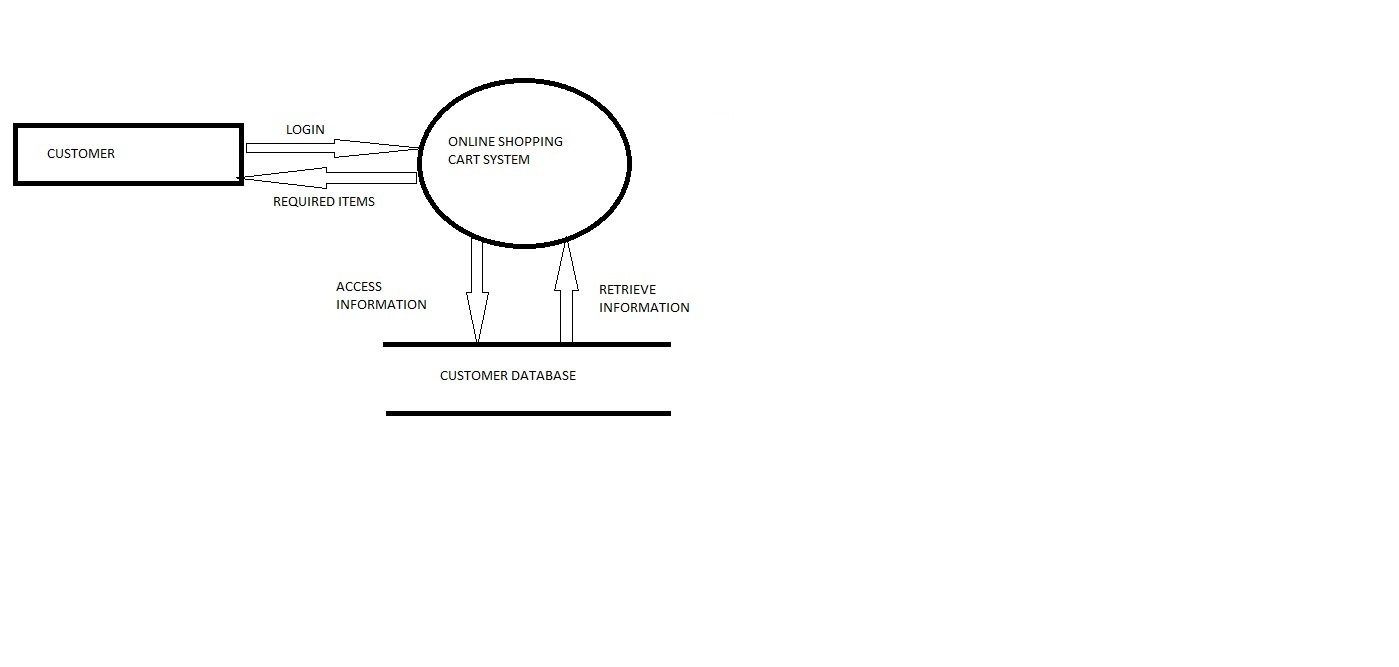
|  |
| --- |
|  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Column Name** | **Data Type** | **Nullable** | **Default** | **Primary Key** | | BOOKING\_CODE | VARCHAR2(20) | No | - | - | | USERNAME | VARCHAR2(20) | No | - | - | | BOOKING\_DATE | DATE | No | - | - | | STATUS | VARCHAR2(10) | Yes | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 4 |  |  | | | | | |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |

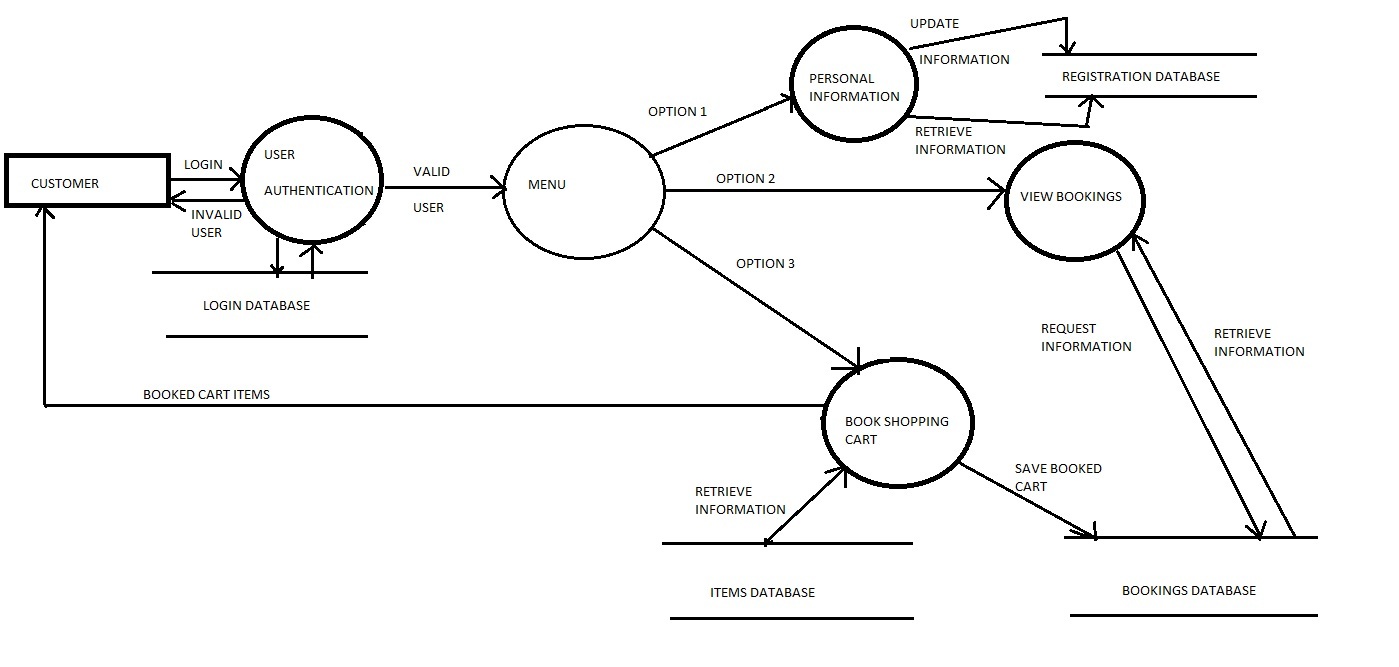
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Constraint** | **Type** | **Table** | **Search Condition** | **Delete Rule** | **Status** | **Last Change** | **Index** | **Invalid** | | SYS\_C004107 | C | BOOKING\_HISTORY | "BOOKING\_CODE" IS NOT NULL | - | ENABLED | 03-AUG-11 | - | - | | SYS\_C004108 | C | BOOKING\_HISTORY | "USERNAME" IS NOT NULL | - | ENABLED | 03-AUG-11 | - | - | | SYS\_C004109 | C | BOOKING\_HISTORY | "BOOKING\_DATE" IS NOT NULL | - | ENABLED | 03-AUG-11 | - | - | | BOOKING\_HISTORY\_CHECK | C | BOOKING\_HISTORY | "STATUS" IN ('Pending','Confirmed') | - | ENABLED | 09-SEP-11 | - | - | | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  |  | 1 - 4 |  |  | | | | | | | | | |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |

**DATA FLOW DIAGRAM**

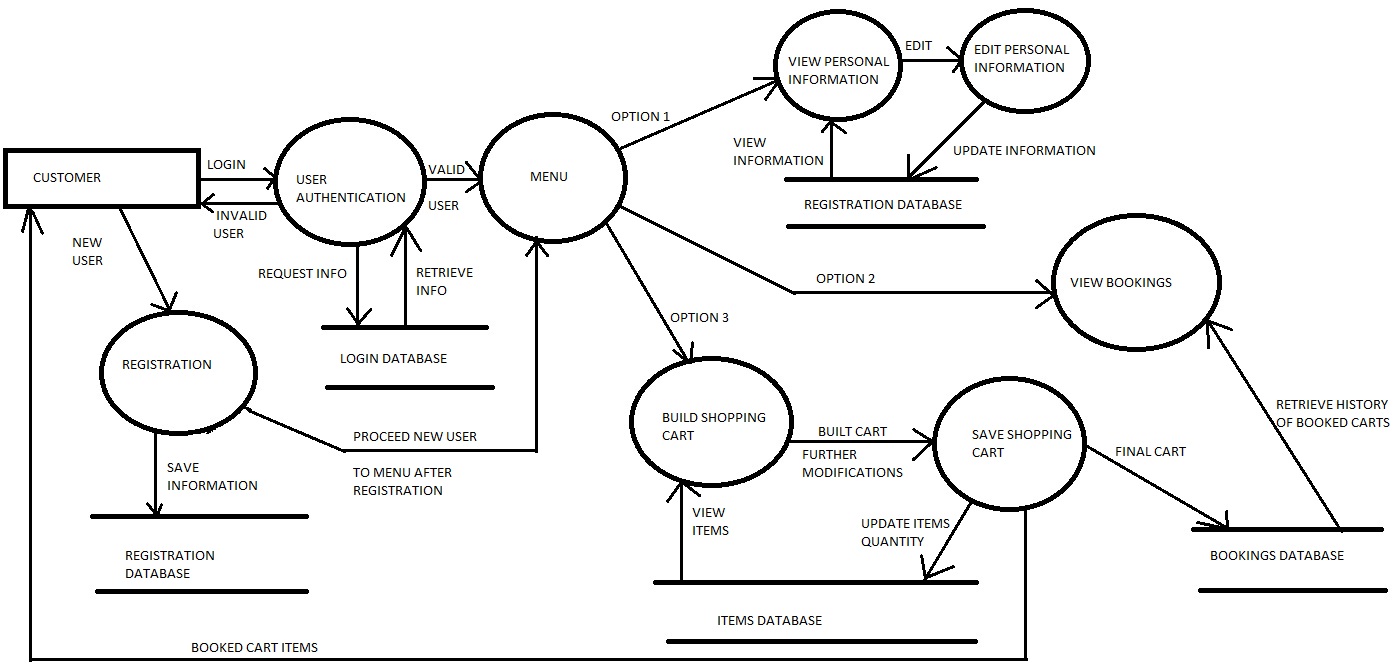
**0 Level**

****

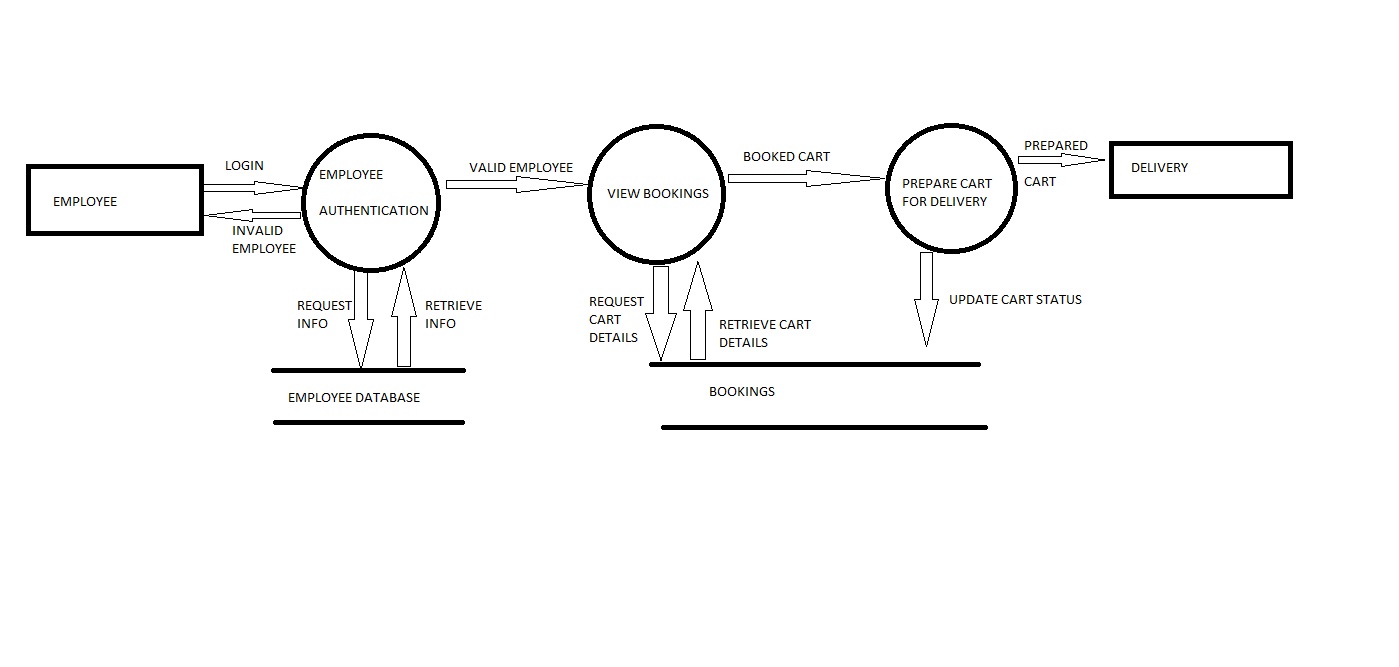
**1 Level**

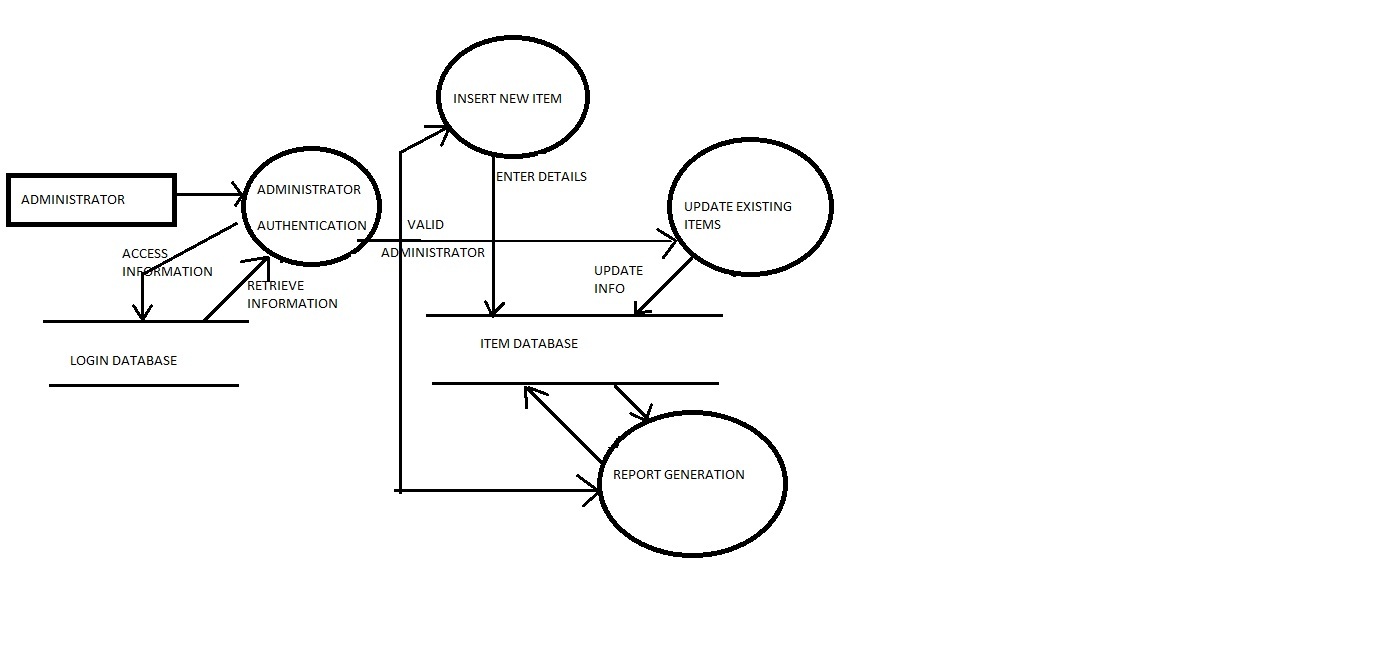


**2 Level**



**EMPLOYEE**



**ADMINISTRATOR**

*Development Phase*

**PROJECT SCREENSHOTS WITH TEST CASES**

**Login Page**



This is the opening page of the system. The user needs to enter the id and password

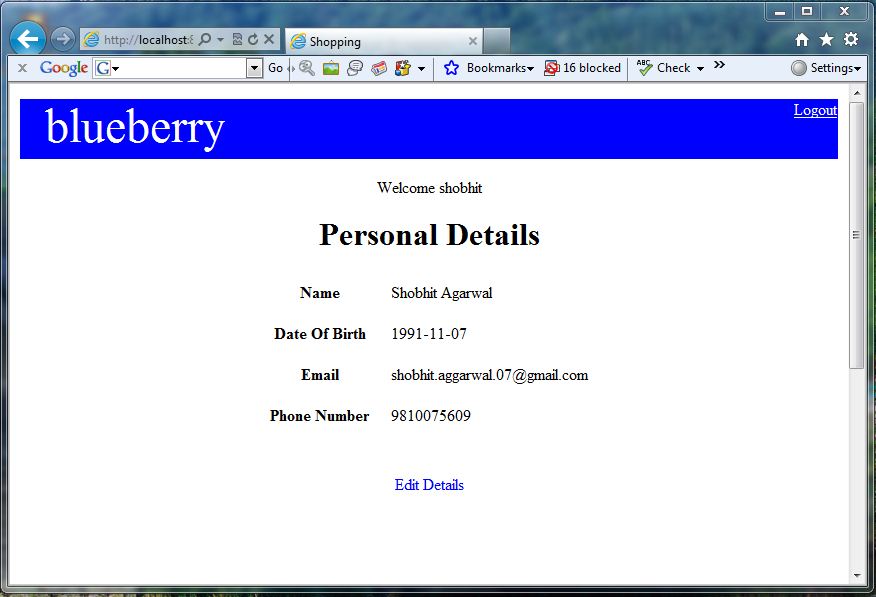
If user enters invalid username or password

**Succcessful Login and the HomePage**

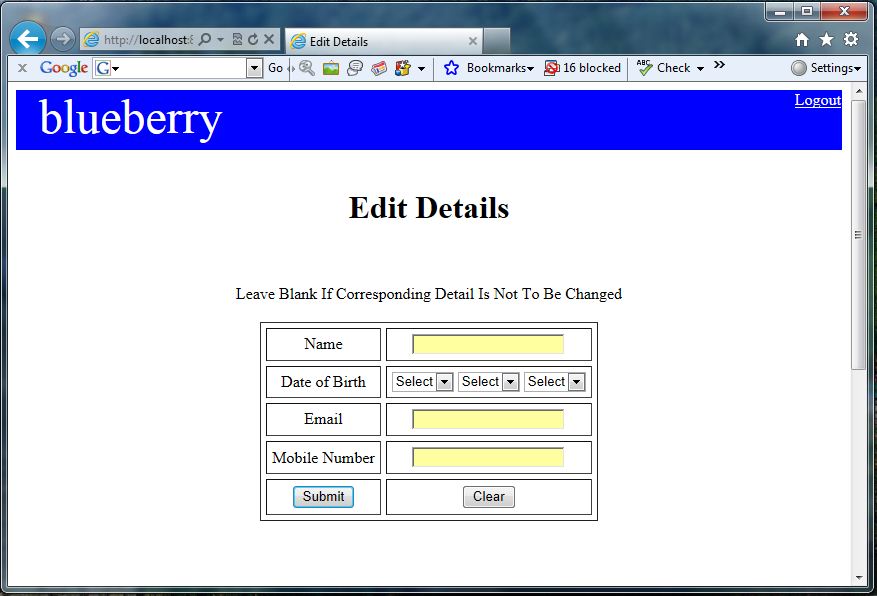


This is the page the that is displayed when the user successfully logins in

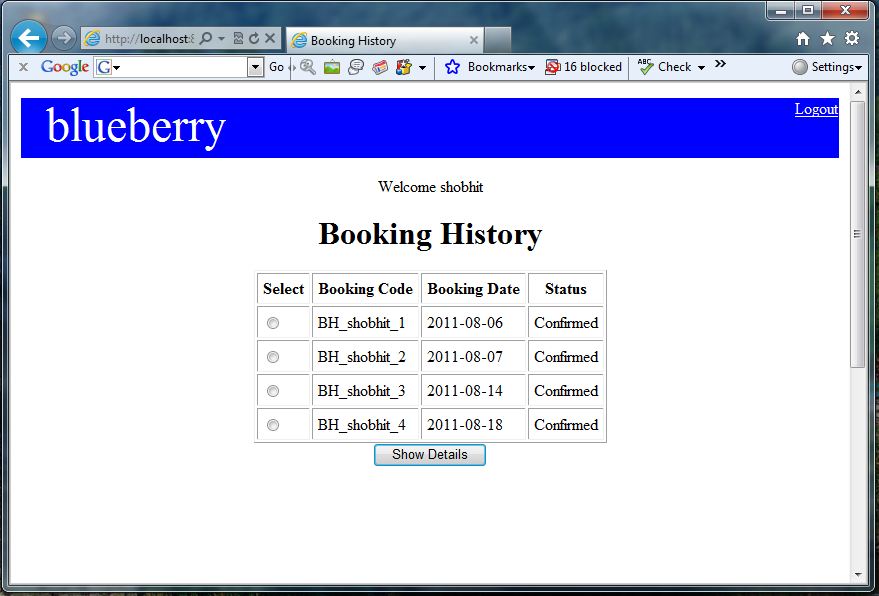
**Personal details page**



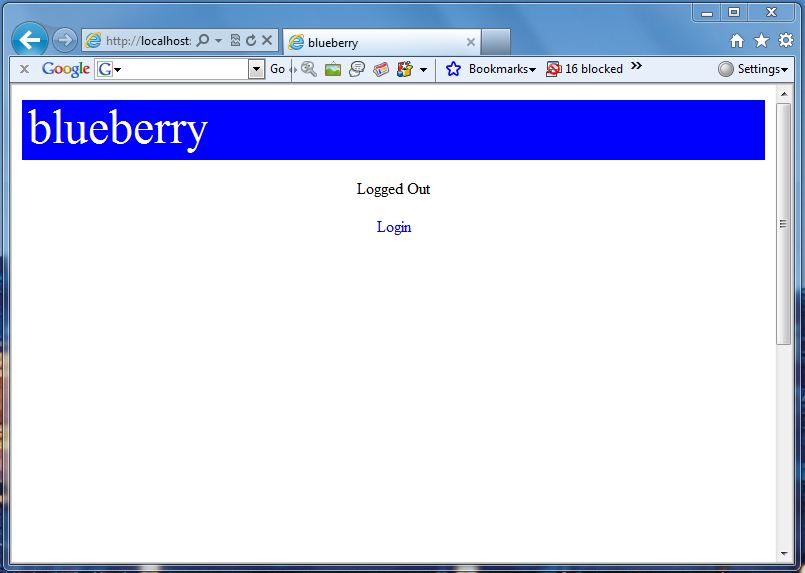
**Edit Details Page**



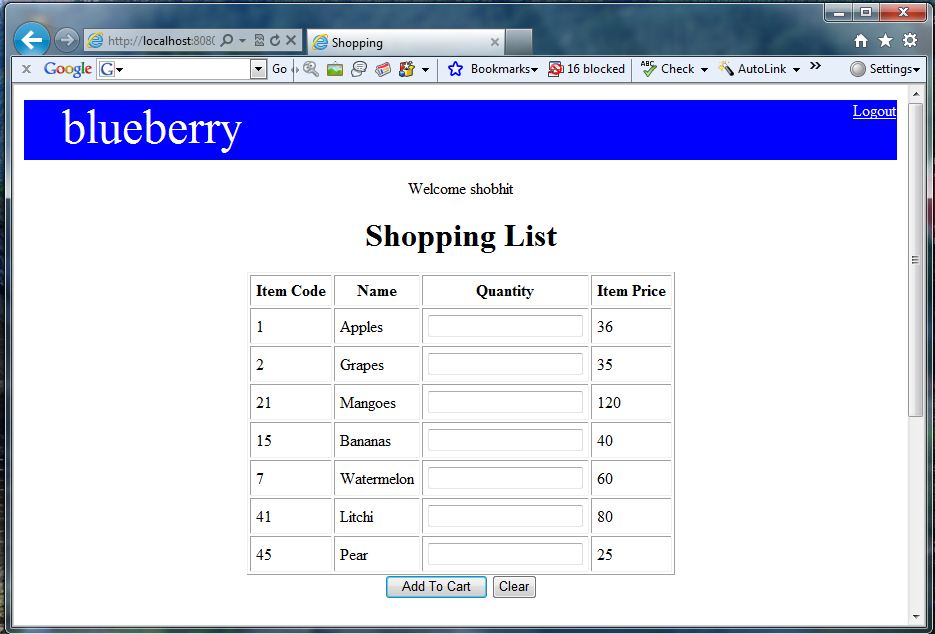
**Booking History Page**



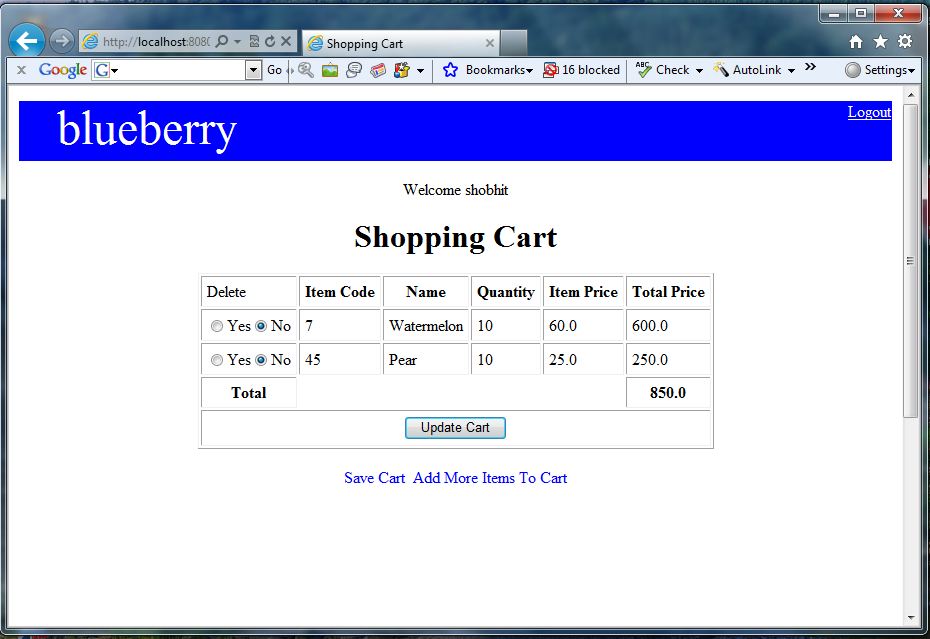
Logout Page



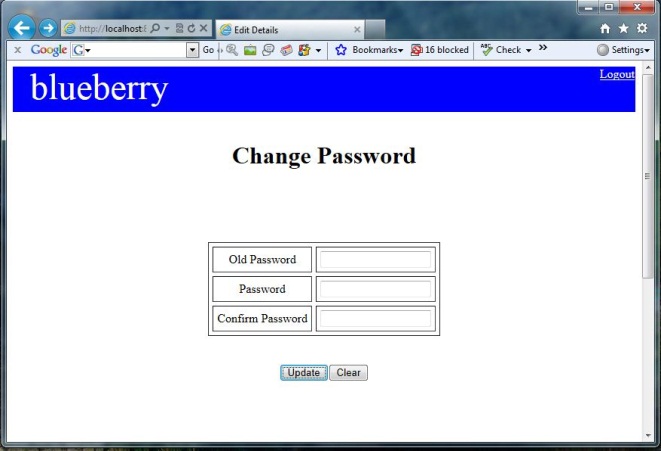
**Shopping Cart First Page**

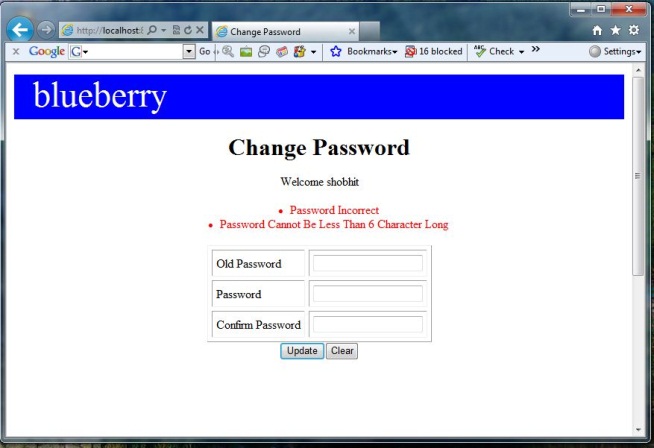


**Shopping Cart Final Page**



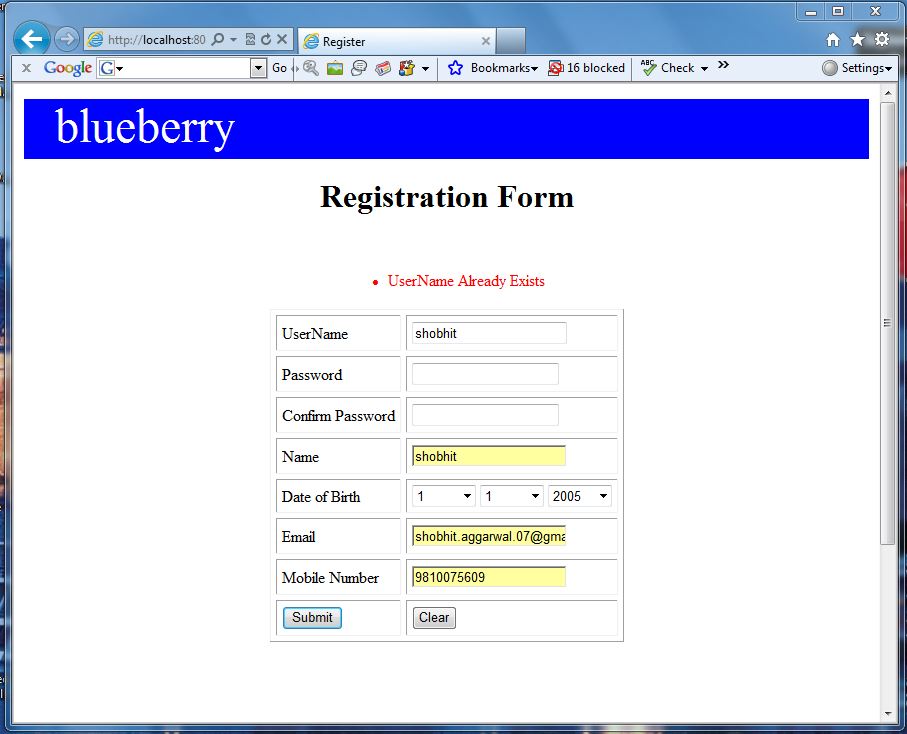
**Change Password Page**

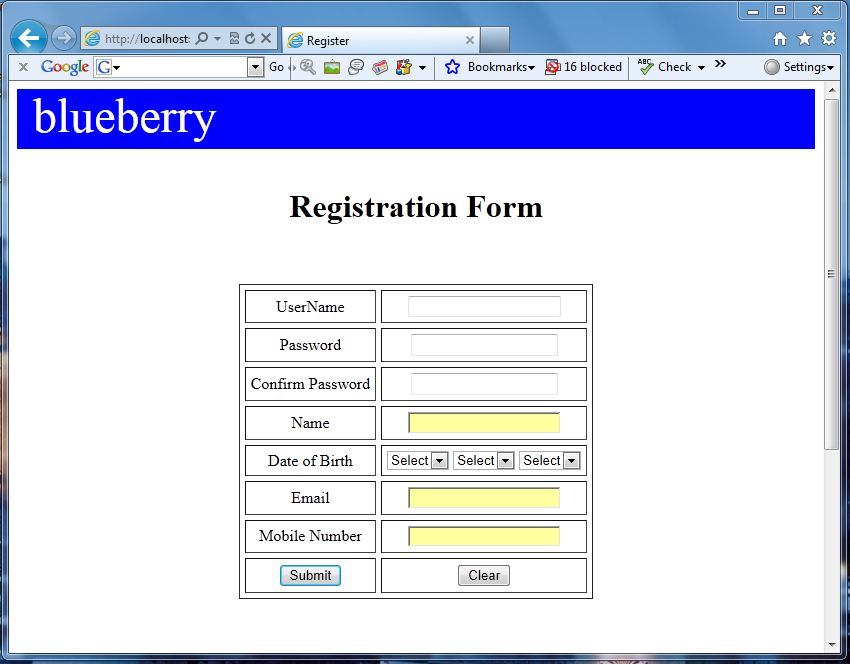




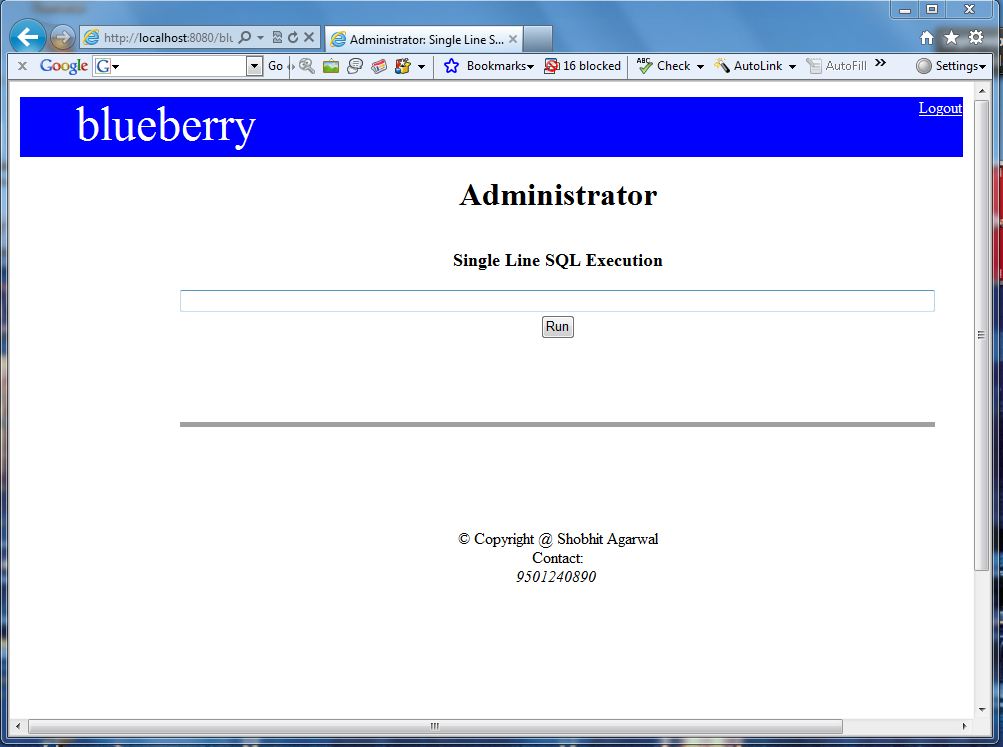
Any error in changing password shows the second page

**Registration Page**

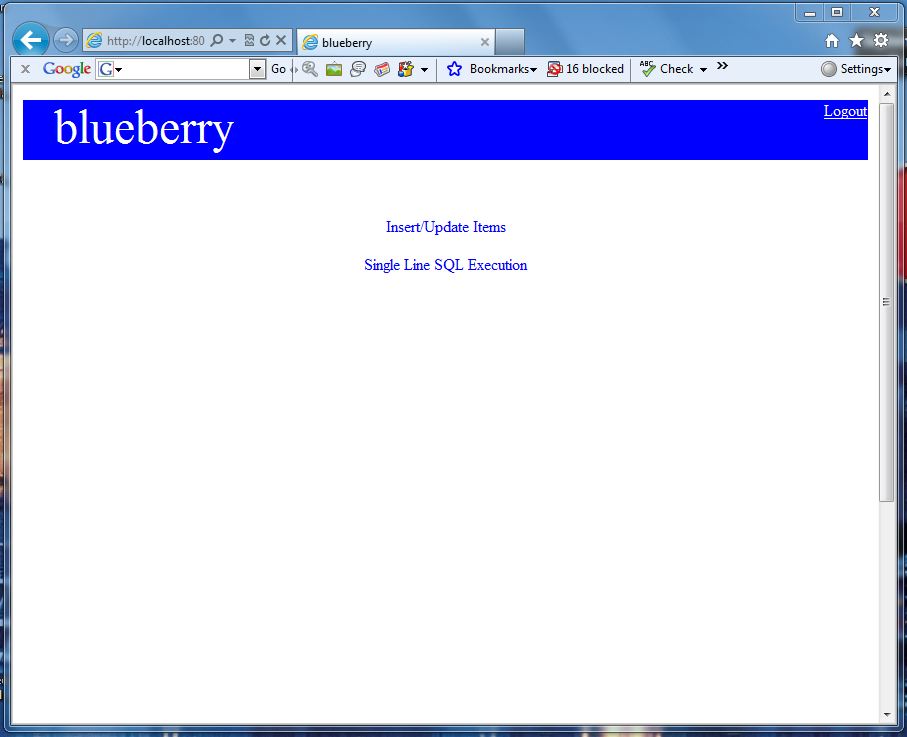




**Administrator Home Page**



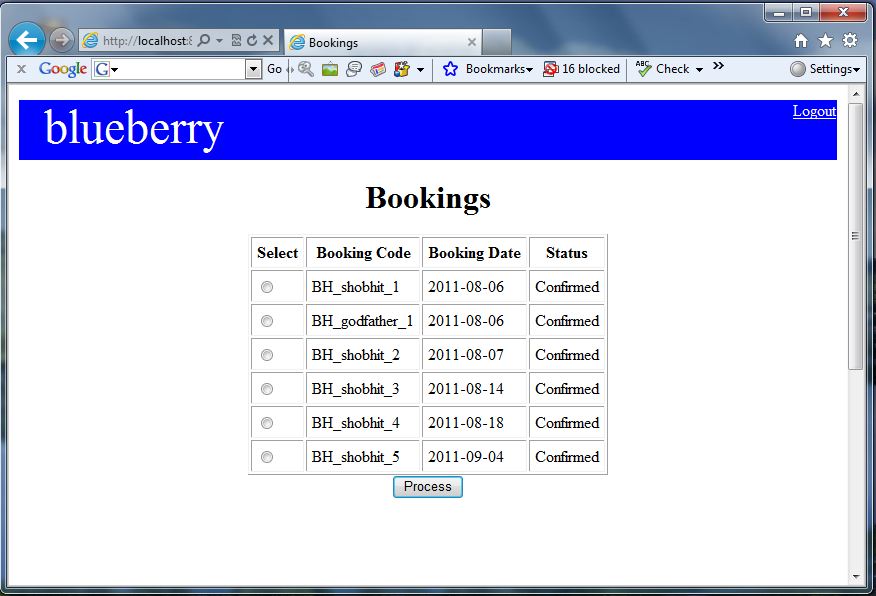
**Administrator Options**



**Insert Or Update Items Page**



**Employee Home Page**



*Testing Phase*

**Testing**

Software testing is a critical element of software quality assurance and the ultimate review of specification, design and code generation .Testing of the software leads to uncovering of errors in the software and reveal that whether software functional and performance requirement are met. Testing also provides a good indication of software reliability as software quality as a whole. The result of different phases are evaluated and then compared with the expected results. If the errors are uncovered they are debugged and corrected. A strategy approach to software testing has the generic characteristics:

1. Testing begins at the module level and works outwards towards the integration of the entire computer based system
2. Different testing technique are appropriate at different point of time
3. Testing and debugging are different activities, but debugging must be accommodating in the testing strategy.

A strategy for the software testing must be accommodate low level tests that are necessary to verify that a small source code segment is performing correctly according to the customers requirement and that of developers expectations .

**Testing objective**

1. Testing is a process of executing a program with the intent of finding an error
2. A good test case is one which has a high probability of finding an as yet undiscovered error
3. A successful test is one that uncovers an as yet undiscovered error
4. Our objective is to design tests that systematically uncovers different classes of errors and to do so with minimum amount of time and effort

**TESTING PRINCIPLES**

Before applying methods to design effective test case, we must understand the basic principles that guide software testing, following are some principles that must be followed.

* All tests should be traceable to customer requirements.

As we have seen, the objective of software testing is to uncover errors. It follows that the most severe defects are those that cause the program to fail to meet its requirement.

* Tests should be planned long before testing begins.

Testing planning begins as soon as the requirement model is complete. Detail definition of the test case should begin as soon as the design model is solidified

* Exhaustive testing is not possible.

The number of path permutations for even a moderately sized program is exceptionally large. For this reason it is impossible to execute every combination of paths during the testing phase

* To be most effective testing should be conducted by independent third party. The person who created the system is not the best person to test the system

**Testing method used**

Any engineered product can be tested in one of two ways:

1. ***Black-Box Testing*** – Knowing that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operational, at the same time searching for errors in each function.
2. ***White-Box Testing*** - Knowing the internal workings of a product, tests can be conducted to ensure that the internal operation performs according to specification and all internal components have been adequately exercised.

For testing our project, we have used the Black-Box testing method, and a short description of this testing method follows

**TEST CASES**

**Test Case:01**

Input:- Enter User ID and Password

Expected OutPut:- If User ID and password is correct then home page displayed.

Else if one of these is invalid then re login page opened and a message is displayed that invalid username or password.

Actual Output:- Same as expected

**Test Case:02**

Input:- Enter User ID and Password

Expected OutPut:- If user id or password not entered and submit button is pressed then An alert message is displayed invalid username or password.

.Actual Output:- Same as expected

**Test Case:03**

Input: - Enter Employee ID and Password

Expected OutPut : - If Employee ID and password is correct then home page displayed.

Else if one of these is invalid then re login page opened and a message is displayed that invalid username or password.

Actual Output: - Same as expected

**Test Case:04**

Input:- Enter Employee ID and Password

Expected OutPut:- If Employee id or password not entered and submit button is pressed then An alert message is displayed invalid username or password.

Actual Output:- Same as expected

**Test Case:05**

Input:- Enter Administrator ID and Password

Expected OutPut:- If administrator ID and password is correct then home page displayed, Else if one of these is invalid then re login page opened and a message is displayed that invalid username or password.

Actual Output:- Same as expected

**Test Case:06**

Input:- Fill Various Entry for new User Registration

Expected OutPut:- If user fills all entries correctly then his/her account is made.

Else if fills wrong entries or leaving blank then alert message is displayed. If user id chosen by user is already in database then an error message is displayed.

Actual Output:- Same as expected

**Test Case:-07**

Input:- customer wants to change his password

Expected OutPut:- customer password is changed if he provides correct old password, and valid and matching new password in 2 fields, appropriate error message is displayed.

Actual Output:- Same as expected

**Test Case:-08**

Input:- customer want to place an order

Expected OutPut:- all the items available are displayed, customer selects the items he wishes to buy, their price, and see his cart.

Actual Output:- Same as expected

**Test Case:-09**

Input:- customer wants to delete items from an unsaved cart

Expected OutPut:- he checks item to delete and new updated cart is displayed

Actual Output:- Same as expected

**Test Case:-10**

Input:- customer wish to save his cart

Expected OutPut:- he clicks on save cart, cart is saved in database and the employee can view the cart and do the needful

Actual Output:- Same as expected

**Test Case:-11**

Input:- Customer clicks on personal details

Expected OutPut:- Customer wants to see his details or edit them

Actual Output:- Same as expected

**Test Case:-12**

Input:- administrator want to run a SQL query

Expected OutPut:- the result of the query

Actual Output:- Same as expected

**Test Case:-13**

Input:- administrator wants to insert/update items in database

Expected OutPut:- Items are inserted/updated

Actual Output:- Same as expected

**Test Case:-14**

Input:- customer wants to view booking history

Expected OutPut:- customer is able to see all his previous bookings

Actual Output:- Same as expected

**Test Case:-15**

Input:- employee wants to see customer bookings

Expected OutPut:- employee successfully sees full customer order

Actual Output:- Same as expected

**Test Case:-16**

Input:- employee wants to change status of a booking

Expected OutPut:- employee successfully updates the status

Actual Output:- Same as expected

**Project Evaluation**

|  |  |  |
| --- | --- | --- |
| Business Functional Requirement ID | Description | Acceptance Criteria |
| Customer Role | | |
| BRF1 | Registration   * New customers can register on the website |  |
| BFR2 | Personal Information   * All the details of the employee captured from DB will be displayed |  |
| BRF3 | Booking an order   * An order form will be displayed. * All the items in database will be displayed and user can choose the items he wishes to buy * Details of the booked items are displayed * Customer can delete items he ordered before saving the cart if he wishes to * Customer can Save the cart |  |
| BRF4 | Changing Password   * Customer is given an option to change his password by providing correct old and new matching passwords |  |
| BRF5 | Editing Personal Details   * Customer has an option to update his personal details stored in database |  |
|  |  |  |
| Employee Role | | |
| BRF6 | Viewing Booked Order   * All the pending requests and their details of employees under that manager will be displayed |  |
| BRF7 | Booking Status   * The Booking status of the orders can be updated by an employee from ‘pending’ to ‘confirmed’ |  |

|  |  |  |
| --- | --- | --- |
| Manager Role | | |
| BRF9 | Inserting new Items   * Insert new items and its corresponding details in the database |  |
| BRF10 | Updating Items   * Updating existing items details in the database |  |
| BRF11 | Running Database Commands   * Running SQL Queries by an administrator |  |

*Installation and Maintenance*

**Installation and Maintenance Report**

The ***software*** has been developed on Netbeans 7.0 IDE. Hence can be easily transferred by making a .war file of the project can be made and uploaded on a different computer. This way the code cannot be accessed by the second computer. Only an executable module is installed.

The software can be uploaded on internet and any user with internet access can access the website.

The ***database*** needs to be replicated on the server. Oracle 10g is preferred.

The ***server*** used here is Glassfish 3.0.

The same integrated environment can be recreated on the secondary node

For a more open compatibility changes can be made in the primary software regarding the database to be accessed and the address of the server as local host.

The ***maintenance*** of the system will require a small team of trained professional.

For further modifications and upgradations the development team can be called in.

**RESULTS**

Some people develop web applications with Perl or PHP and implement their SQL Queries and the business logics directly in the HTML document. This can be used when we are using java servlets or JSP. It is convenient in small projects. But imagine you have 70 dialogs, many database queries in this dialogs and you want to define a field status, setting if a book is deleted or not.

**Advantage of separation**

• Changes on the functionality, without working in the dialogs.

• Better overview, functionality not mixed with the dialogs.

• Easy maintain a application

• Different dialogs, but the same functionality

Servlets is very competitive technically, but to my mind the biggest advantages Servlets has over competing technologies are practical. These features are:

* ongoing development by a large number of committed users/developers
* knowledgeable and responsive project leadership
* access to source code
* strong connection to and commitment to future integration with forward-looking technologies like JSP
* increasing mind share that we expect will make it easier to hire new developers already familiar with the framework

To summarize, Servlets is an application "Framework" for building Web-based applications in Java using the J2EE platform. Servlets makes developers more productive by giving them pre-built components to assemble applications from. Servlets was built using industry best practices including the MVC design pattern and it can be deployed in a wide range of environments.

**CONCLUSION**

So, the question arises, should you consider adopting Servlets? Of course, your answer depends on your particular circumstances and environment, but here are some criteria to consider:

1. Are you using the J2EE platform (that is, developing applications using J2EE-compliant servers such as Weblogic Server, Glassfish,)? If the answer to this is yes, Servlets is likely worth considering.
2. Do your developers have Java expertise? Although this isn't a "make or break" criterion, it helps. If your developers don't have Java experience but you are dedicated to moving to Java anyway, Servlets may actually make the transition easier.
3. Are you building applications that need to work in a Web browser? This is the niche that Servlets fills. If the answer is no, Servlets isn't for you. Unless you have requirements for browser-based application delivery, Servlets won't add much value.
4. Are you building applications now that use Java Server Pages (JSP)? If so, you should definitely be looking at Servlets. It may increase your developer productivity significantly.
5. Is your development team considering building a "custom framework" for building Web-based applications? If so, ask them to justify *not* using Servlets. Anything they would build on their own would likely not undergo near the amount of testing and development that Servlets has. While they may have good reasons for not using it, Servlets should definitely be on their radar.

**FUTURE SCOPE**

The following points elaborate the future scope of the ‘Online Shopping Cart’:

* Date validations
* Change password in case user forgets it.
* Online Payment .
* Shipment of order.
* Cancellations of orders.
* Report generation of day to day transactions, monthly statements.