Chapter 1
Introduction to Project

1.1Overview

A modeling agency is a company that represents fashion models, to work for the fashion industry. These agencies earn their income via commission, usually from the deal they make with the model and or the head agency.

The top agencies work with big-budget advertising agencies and fashion designers. They invest money into developing their talent so they can increase their status within the industry. These top agencies will help train models, get test shoots, layout portfolios, and put together comp cards (composition photo cards) and other printed materials models need.

The agencies find work for models by presenting them to designers, photographers, and ad agencies. The agencies are also responsible for booking the jobs, billing for the jobs, and eventually paying the models for their time. By handling the details, an agency allows a model to focus on modeling and not on the business end.

Because modeling is a very competitive, fast moving business that extends beyond the traditional 9 AM to 5 PM business hours, an agency generally conducts business 24 hours a day, to handle emergencies such as cancellations or rush jobs. Most agencies have a service or an operator to handle emergency issues after hours.

1.2PROJECT CATEGORY: Relational Database Management System (RDBMS)

The project is entitled "Fashion Model Management System", category "RDBMS". Hence before discussing anything about the project Fashion Model Management System, a brief discussion of related basic concept is necessary.
As a software developer or as a programmer, we are expected to design and develop any program that works correctly, efficiently and the time is easy to be used by every person, who may or may not be well versed with computer and its capabilities.
The Project is based on the Fashion Model Management System, Being the Information System it requires extensive use of some Data base Management System to store, manipulate and handle the huge and complex record, In RDBMS we can act various attributes with the database like editing the records, Modifications Deletions of the records, View the records in various formats, listing the database etc. Project can be categorized by their functioning and relation with their database and other tools can categorize project. Since this project has been developed based on the Relation Data Base Management System So Proposed system comes under RDBMS (Relational Database Management System) category, as there is need to store and manipulate a huge amount of data related to model as per various queries.

1.3Objectives of the Project

- To handle the manual handling and even the workload of the employees of the company by introducing a change in the existing manual system to a single software implementation.

- To save time and cost of the employees and company by providing information very efficiently and easily.
- To have a proper record of the employees and having a list and record of models both old and new.
- To provide a system with proper recovery of data, in case lost, through a proper backup provided.

1.4Identification of Need

Modeling agencies have the appeal of glamour – it's easy to picture flashing cameras, elegant clothing and beautiful models prowling a sleek catwalk. But modeling agencies help satisfy the needs of many different kinds of clients, including print, fashion shows, television and live demonstrations. Modeling agencies scout for new talent, counsel models to increase their marketability and promote models to clients, according to the Bureau of Labor Statistics. At 2,200 jobs, models held a relatively small number of jobs within the United States in 2008, according to the bureau's report titled "Models." The market was projected to grow by 16 percent between 2008 and 2016.

Types

To start a modeling agency, you'll need to evaluate the different types of modeling agencies to determine which is best suited to your strengths as a business owner. High- fashion modeling can be lucrative, but it's highly competitive and most entrepreneurs will need excellent industry contacts and previous experience placing talented models in these contracts, according to E-Model.net, an online resource for the modeling industry, in the article titled "Types of Modeling." Catalog modeling is a more realistic demographic for new modeling agencies because you'll be able to network with local companies preparing their upcoming season catalog. Other agencies may specialize in niche models, such as full-size models.

Legal Requirements

You will need to satisfy numerous legal requirements when starting your modeling agency. As with any new business, you'll need to apply for a business license, tax identification number and liability insurance. Establishing the modeling agency's legal structure, whether a corporation or sole proprietorship, is also necessary. Be prepared to offer legal contracts to models outlining their responsibilities, pay scales and access to your agency's resources, as described by Modeling Advice.com in the online article titled "The Agency."

Relationships

Professional relationships are an integral part of starting a modeling agency, as described by Entrepreneur.com in the online article titled "Modeling Agency." You will need to develop contacts in different parts of the modeling industry, including fashion show bookers, photographers, fashion designers, marketing experts for top fashion companies and the models themselves. International modeling agencies will need contacts with overseas clients, as described by Job Monkey.com in the online article titled “How to Open Your Own Modeling Agency.” Because the modeling industry is highly competitive, these relationships will be crucial in landing gigs for your models.

Marketing

Modeling agencies must be prepared to market and promote their stable of models by assisting them in developing portfolios, websites and print composite cards, as described by the Bureau of Labor Statistics. Maintain a professional storefront location for receiving potential clients, as recommended by Job Monkey.com. List the business with online modeling directories, conduct casting calls and hire additional modeling agents as necessary.

Considerations

You will need to make considerable efforts to ensure that your modeling agency maintains a professional reputation, because there are plenty of businesses running less-than-legitimate operations appealing to starry-eyed model hopefuls, according to “The Agency.” Always adhere to contract stipulations and don’t promise what you can’t deliver when negotiating with models or clients. Regularly check with the Better Business Bureau to determine if there are complaints about your services and be prepared to quickly address these should they arise

1.6 Existing System

- The existing system only provides text-based interface, which is not as user-friendly as Graphical user Interface.
- Since the system is implemented in Manual, so the response is very slow.The transactions are executed in off-line mode, hence on-line data capture and modification is not possible.
- Off-line reports cannot be generated due to batch mode execution.

1.7 Proposed System

Today, advertisements has deeply penetrated in society and has immense impact on personal life of individuals and majority of them are influenced by advertising. So, it requires for the models that can do the work and model agency provide models in a easier and convenient manner for the companies. The companies would not have to put their efforts and visit a particular model or visit physically a respective service provider to get his need fulfilled. The Website would do it all automatically, hence saving a lot of time, money, aggravation and energy of the person. This will save a lot of trouble for the user as he/she need not have to go to a particular place to request service, rather he can apply it from anywhere at any time and the data will automatically processed, reducing the workload Anywhere, anytime availability of application will help an enterprise to advertise in a timely and effective manner. In addition, they are easily accessible and offer seamless user experience It can serve as an excellent customer support and service vehicle. It provides users with enhanced experience. This is because they allow users to post queries and feedback anywhere, anytime.

The detailed requirement analysis of system ,suggests that there is the need of the a website application for rendering disruptive services, which will be able to serve people in more efficient manner as this type of website will save their lot of tremendous time and energy.
This application that is meant to:

1. Provide Models

2. Clear terms and conditions

3. Joining of new models

4. Integrated platform for various services.

5. Real time User Authentication

1.8 Unique Features of the System

Enhancement:
The main objective of Fashion Model Management System is to enhance and upgrade the existing system by increasing its efficiency and effectiveness. The software improves the working methods by replacing the existing manual system with the computer-based system.
Automation:
The Fashion Model Management System automates each and every activity of the manual system and increases its throughput. Thus the response time of the system is very less and it works very fast.
Accuracy:
The Fashion Model Management System provides the uses a quick response with very accurate information regarding the users etc. Any details or system in an accurate manner, as and when required.
User-Friendly:
The software Fashion Model Management System has a very user-friendly interface. Thus the users will feel very easy to work on it. The software provides accuracy along with a pleasant interface .Make the present manual system more interactive, speedy and user friendly.
Availability:
The transaction reports of the system can be retried as and when required. Thus, there is no delay in the availability of any information, whatever needed, can be captured very quickly and easily.
Maintenance Cost:
Reduce the cost of maintenance.

Chapter 2

Requirement Analysis and System Specification

2.1 Feasibility Study

The very first step of developing any system is to study the whole existing system this is called the initial study, analysis and feasibility of the project is being done. Analysis is the detailed study of the various operations performed by the system and their relationship within and outside the system. In context to this project the records are maintained on the daily basis and the same procedure is repeated every day. Hence the need for the computerization and portability is necessary to make the work easier and more comfortable.

Economic Feasibility: The developed system is cost effective in terms of the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. The benefits that the system provides are totally cost effective. As it is developed using Java an SQL which are open sources and are freely available . A Netbeans IDE toolkit was required for the successful completion of the project . There could be various types of intangible benefits on account of automation. These could include increased customer satisfaction, improvement in product quality, better decision making, timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, better employee morale.

Operational Feasibility: Proposed project is beneficial as it can be turned into information systems that will meet the organizations operating requirements that comprises of Linux ,Ubuntu and Windows as Java is Platform independent. So, the system is intended to work efficiently when it is developed and installed. Help has been provided in each and every way possible by the employees and the owner of the company. Since, the present system is too time consuming and cumbersome and due to the absence of any automated system in the company, we hope that the employees welcome a change that will bring about a more operational and useful system. Since the proposed system is aimed to help reduce the hardships encountered. In comparison to the existing manual system, the new system can be considered to be operationally feasible.

Technical Feasibility: Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because,at this point in time, detailed design of the system is not available, making it difficult to access issues like performance, costs on (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis.

In order to understand the different technologies involved in the proposed system before commencing the project we have to be very clear about what are the technologies that are to be required for the development of the new system.

The feasibility study is carried out to select the best process that meets the performance requirements. This entails identification, description and evaluation of the candidate process and the selection of the best candidate process for the job. Now according to this project the analysis is to have the complete knowledge about how the software that is going to develop is going to function. From where is the input going to come and what will be the output vs. what is the required output. There are two major activities in this phase - problem understanding or analysis and requirement specification in problem analysis; the analyst has to understand the problem and its context.

After the detailed feasibility study and analysis, a real time system is proposed. The proposed system is an user friendly Powerful and Interactive real time application, that is handy i.e. it is available on an android device which can be a tablet or a smartphone, and directly links to a database where the request records are updated automatically and the data is readily available anytime it is needed. This application would be accessible by a worldwide audience

The existing system provides the information of models but will not provide updated model details time to time to the users for example if a model leaves a agency and companies want to re-hire the model the companies will waste a lot of time and resources.

2.2 Software Requirement Specfication

Data Requirement

Master Data – It is the key to every operation in the business environment. It includes data about model, photographers, admin details While it is non-transactional in nature, it is not limited to non-transactional data, and supports transactional processes and operations.

Model Details – It contains a record of all the types of models which are currently operating.

Login Details – It implies having a list of all the admin which are working.

Users– It contain list of users which wish to know about the company services are contained in the separate database

Whole of these modules are available to the administrator only.

Functional Requirement:

The functional requirements part discusses the functionalities required from the system. The system is considered to perform a set of high-level functions {*fi*}. The functional view of the system is shown. Each function fi of the system can be considered as a transformation of a set of input data (ii) to the corresponding set of output data (*oi*). The user can get some meaningful piece of work done using a high-level function.

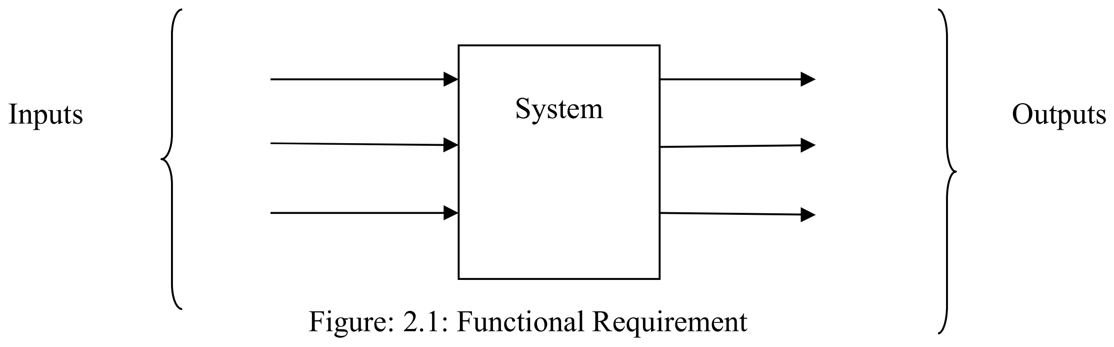


Figure: 2.1: Functional Requirement

Non Functional /Performance Requirements

Table 2.6Hardware Requirements

Device Capabilities	Windows operating system or any other operating system
RAM	1GB & above
Processor	Intel Dual Core & above

Table 2.7: Software Requirements

IDE	Netbeans Ide
SDK	JDK
Operating System	Windows, Linux or Mac OS X System
Programming Language	Java(advance)
Database	MySql
Scripting Language	JSP,HTML/CSS

Performance
The system must be interactive and the delays involved must be less .So in every action-response of the system, there are no immediate delays. In case of opening windows forms, of popping error messages and saving the settings or sessions there is delay much below 2 seconds, In case of opening databases, sorting questions and evaluation there are no delays and the operation is performed in less than 2 seconds for opening ,sorting, computing, posting > 95% of the files. Also when connecting to the server the delay is based editing on the distance of the 2 systems and the configuration between them so there is high probability that there will be or not a successful connection in less than 20 seconds for sake of good communication.
Safety
Information transmission should be securely transmitted to server without any changes in information
Reliability
As the system provide the right tools for discussion, problem solving it must be made sure that the system is reliable in its operations and for securing the sensitive details.

Security Requirements
Security:
This is very important aspect of the design and should cover areas of hardware reliability, fall back procedures, physical security of data and provision for detection of fraud and abuse.

System design involves first logical design and then physical construction of the system. The logical design describes the structure and characteristics of features, like the outputs, inputs, files, database and procedures. The physical construction, which follows the logical design, produces actual program software, files and a working system.

Input Design Guidelines

The design of input play very significant role in getting the correct output. It covers al phases of input from creation of initial data (original recording) to actual entering the data to the system for processing. The input design is the link that ties the information system into the world of its users. Some features of design may vary depending on whether the system is batch-oriented or on-line. Here, we will discuss the various objectives of input design. They focus on:

- Controlling amount of input
- Avoiding delay
- Avoiding errors in data
- Avoiding extra steps
- Keeping the process simple

Chapter 3

Requirement Analysis and System Specification

3.1 User Requirement

The existing system provides the information of models but will not provide updated model details time to time to the users for example if a model leaves a agency and companies want to re-hire the model the companies will waste a lot of time and resources.

Developing a portal makes the users to easily find, navigate and manage information in a consolidated view. The developer has to make sure that the portals are developed according to the need of Fashion Models and current customers. It allows to finding, consolidating, managing, analyzing and distribute information across and outside of an enterprise. Scope: Fashion Studios Model Management gives the opportunity to develop the organization information in a consolidated view. It provides the web based interface that gives customers and Models quick access to mission critical business information and reports. Portal allows enabling more people in your organization to access the organization data through a web browser gives the opportunity to the users to enter into the Fashion Industry.

This Project Consists of three modules:-

- Customers
- Models
- Admin

3.2 Data Flow Diagram:

A data flow diagram(DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated.DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

We usually begin withdrawing a context diagram, a simple representation of the whole system. To elaborate further from that, we drill down to a level 1 diagram with additional information about the major functions of the system. This could continue to evolve to become a level 2 diagram when further analysis is required. Progression to level 3, 4 and so on is possible but anything beyond level 3 is not very common. Please bear in mind that the level of detail asked for depends on your process change plan. Following is the DFD of given application :

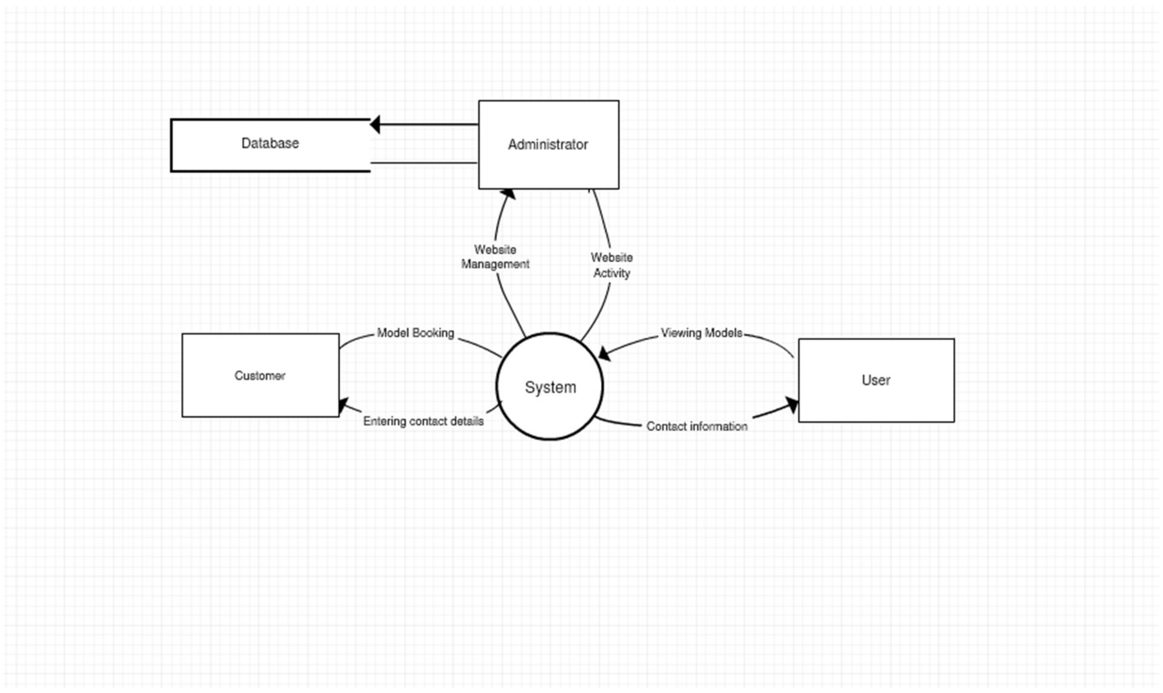


Figure 3.1: Level 0 Project Data Flow Diagram

3.3 Database Design

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching, and replicating the data it holds. Other kinds of data stores can be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems. So now a days we use relational database management systems (RDBMS) to store and manager huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys.

A Relational Database Management System (RDBMS) is a software that:

- Enables you to implement a database with tables, columns, and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables.

RDBMS Terminology:

Before I proceed to explain MySQL Database system, lets revise few definitions related to database.

- **Database -** A database is a collection of tables, with related data.
- **Table -** A table is a matrix with data. A table in a database looks like a simple spreadsheet.
- **Column -** One column (data element) contains data of one and the same kind, for example the column postcode.
- **Row -** A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.
- **Redundancy** - Storing data twice, redundantly to make the system faster.
- **Primary Key -** A primary key is unique. A key value can not occur twice in one table. With a key you can find at most one row.
- **Foreign Key -** A foreign key is the linking pin between two tables.

3.5 ER Diagram:

ER diagram is a graphical modelling to standardize ER modelling. Few terminologies related to ER diagram are as follows

ENTITY: An entity in an real life object that exists and is distinguish from other objects. Example Name, Department etc.

ATTRIBUTES: Attributes are properties of entity types. Examples address, phone no.

RELATIONIP: A relationship is an association between entities types. Examples Teaches is relationship type between teacher and student.

The E-R diagram is used to represent database schema. In E-R diagram

- A rectangle represents an entity set.
- An ellipse represents an attribute
- A diamond represents a relationship
- Lines represent the linking of attributes to entity sets and entity sets to relationship sets.

The ER Model has the power of expressing database entities in a conceptual hierarchical manner. As the hierarchy goes up, it generalizes the view of entities, and as we go deep in the hierarchy, it gives us the detail of every entity included.

In ER model is typically implemented as a database. In the case of a relational database, which stores data in tables, every row of each table represents one instance of an entity. Some data fields in these tables point to indexes in other tables; such pointers are the physical implementation of the relationships. The three schema approach to software engineering uses three levels of ER models that may be developed.

ER Diagram of the Application: It represents 1:M relationship i.e a single customer can request for more than one service at a time.

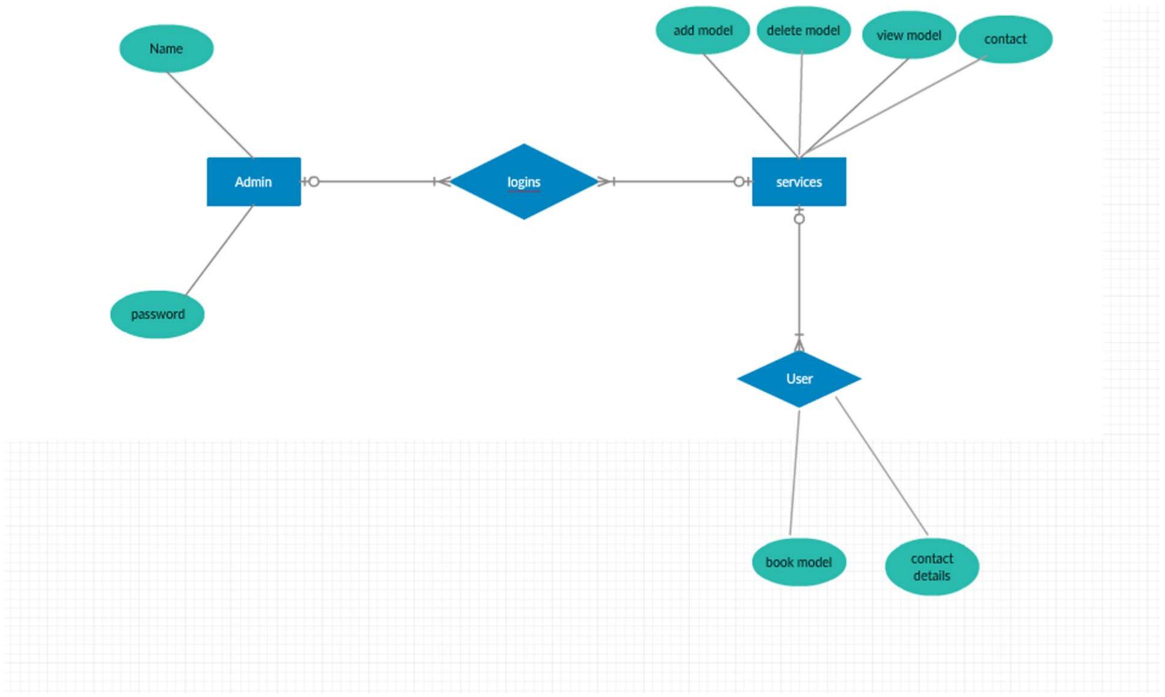


Fig 3.4: ER Diagram

3.6ASSUMPTIONS AND DEPENDENCIES

In this project, the assumption is that each and every model has a particular id. This id is unique for each of them. Even, request submitted and feedbacks also have a unique id.

Dependencies arise when a particular instruction is linked to another instruction and cannot be implemented until the execution of the previous instruction is completed. The dependency of the whole project is on master data. Until and unless the master data is not filled no transactions can be done and previews can also not be seen.

3.7METHODOLOGY

The application will be made using Netbeans Ide ,MySql and an Operating System. Java is used as the programming language . This project used three major steps to implement project starting from planning, implementing and testing.

Step 1: Researching the need of a websitw application and finding out the facilities required to

develop project.

Step 2: Documenting the needs and then preparing the layout for the project, and deciding the various modules to be included in the application.

Step 3: DFDs will be prepared showing various interactions between users and the system.

Step4: Selecting the technology for developing the project and installing the required tools for developing the project. We will install Netbeans Ide and My SQL for back-end and front-end respectively.

Step5: Developing the front end- All the forms required to get users information, create profile etc. will be developed using html, css and webapplication.

Step6: Developing the back end- The tables necessary to store the data relating to customers and tables containing the data of the services will be developed in MySQL. Tables can be updated from time to time. Efforts will be made to ensure no redundancy.

Step7: Testing the system by running it and by entering data and retrieving it.

The following figure shows the layout of the project. First the user goes through registration and then logs in the application. The main home activity is displayed after logging in where the user can take select the type of service he/she needs and further apply for it choosing service type from database Further, the user can view any previously requested service. And the vendor can trak and manage the requested service using his main activity as shown below:

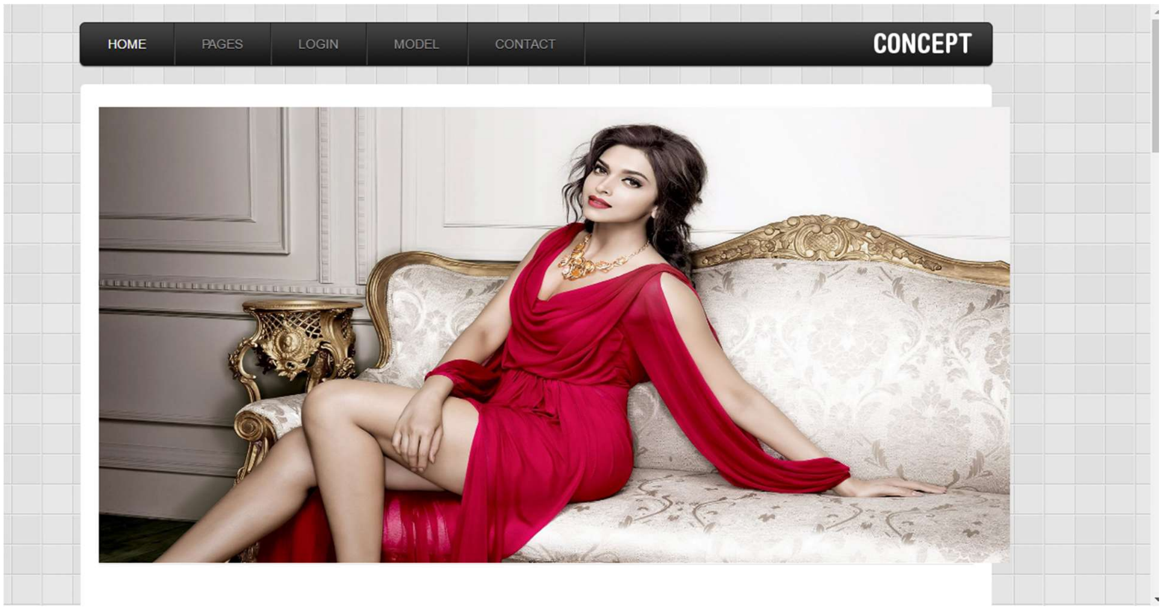


Fig3.3 main page

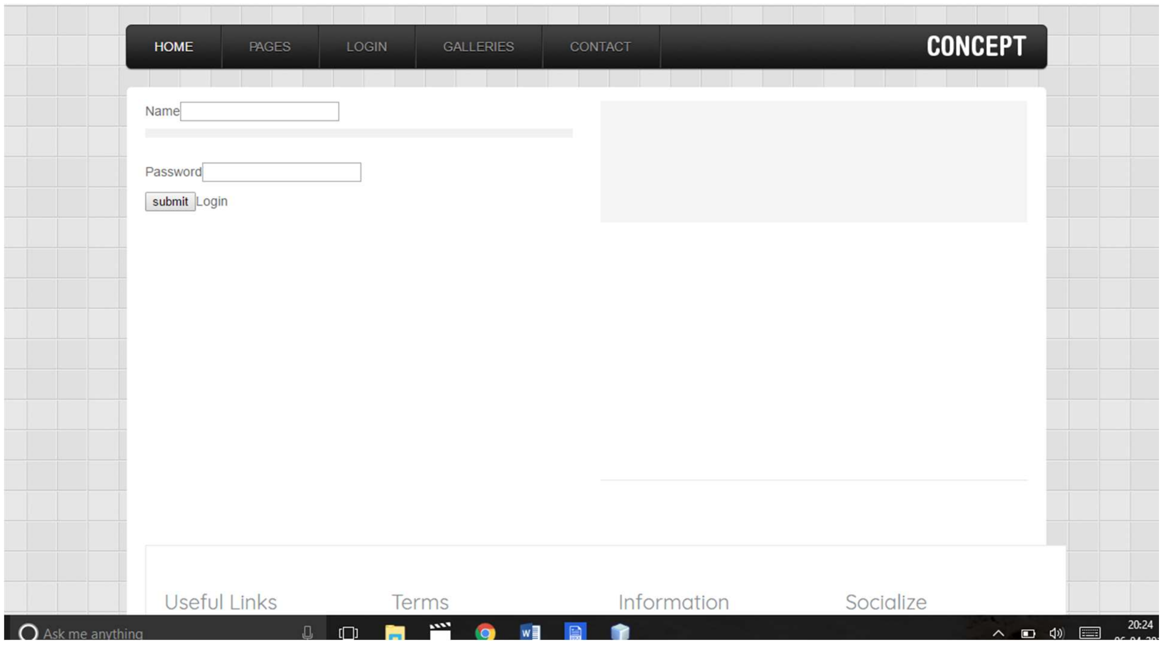


Fig3.4Admin Page

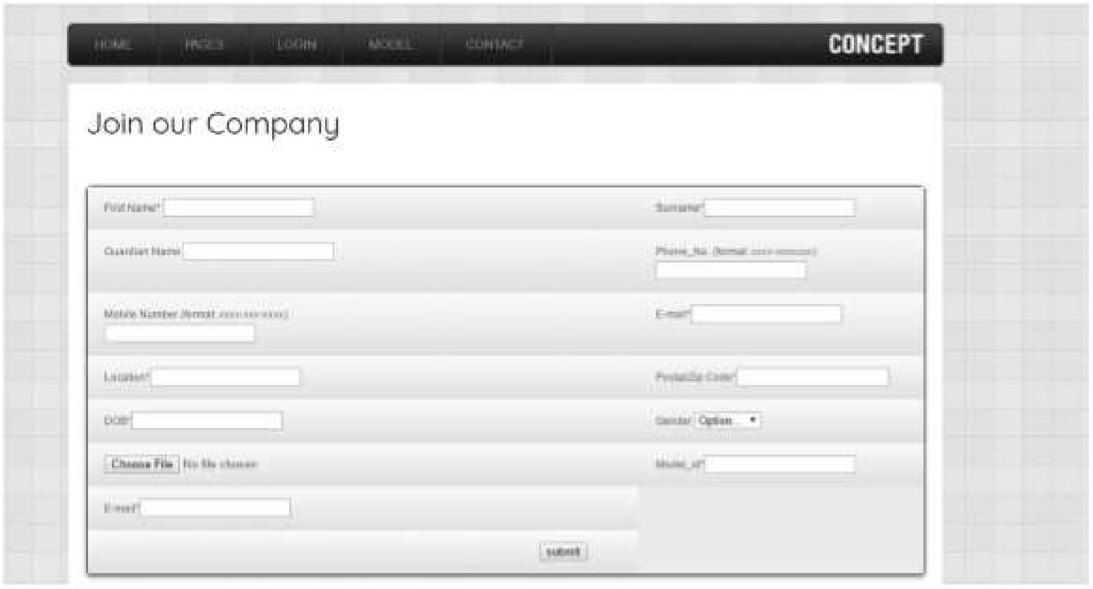


Fig3.5 Join our Company

Chapter 4
Implementation,Testing and Maintenance

4.1 Introduction to Languages,IDE's,Tools and Technologies used for Implementation

Java is a programming language 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. Java applications are typically compiled to bytecode (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is a general-purpose, concurrent, class-based, object-oriented language that is 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 is, as of 2012, one of the most popular programming languages in use, particularly for client-server web applications.

Principles:

There were five primary goals in the creation of the Java language:

It should be "simple, object-oriented and familiar"

It should be "robust and secure"

It should be "architecture-neutral and portable"

It should execute with "high performance"

It should be "interpreted, threaded, and dynamic"

Portability: It means that computer programs written in the Java language must run similarly on any hardware/operating-system platform. This is achieved by compiling the Java language code to an intermediate representation called Java bytecode, instead of directly to platform-specific machine code. Java bytecode instructions are analogous to machine code, but they are intended to be interpreted by a virtual machine (VM) written specifically for the host hardware. End-users commonly use a Java Runtime Environment (JRE) installed on their own machine for standalone Java applications, or in a Web browser Java applets. Standardized libraries provide a generic way to access host-specific features such as graphics, threading, and networking. A major benefit of using bytecode is porting. However, the overhead of interpretation means that interpreted programs almost always run more slowly than programs compiled to native executables would.

Simple: Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object-Oriented: Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust: The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time. Java virtually eliminates the problems of memory management and de-allocation, which is completely automatic. In a well-written Java program, all run time errors can and should be managed by your program.

Java uses an automatic garbage collector to manage memory in the object lifecycle. The programmer determines when objects are created, and the Java runtime is responsible for recovering the memory once objects are no longer in use. Once no references to an object remain, the unreachable memory becomes eligible to be freed automatically by the garbage collector. Something similar to a memory leak may still occur if a programmer's code holds a reference to an object that is no longer needed, typically when objects that are no longer needed are stored in containers that are still in use. If methods for a nonexistent object are called, a "null pointer exception" is thrown.

4.1.2NETBEANS IDE FOR ADVANCE JAVA

.J2EEShort for Java 2 Platform Enterprise Edition, J2EE is a platform-independent ,Java-centricenvironment from Sun for developing, building and deploying Web-based enterprise applicationsonline. The J2EE platform consists of a set of services, APIs, and protocols that provide thefunctionality for developing multitier, Web-based applications.Some of the key features and services of J2EE :

- At the client tier, J2EE supports pure HTML ,as well as Java applets or applications. It relies on Java Server Pages and servlet code to create HTML or other formatted data for the client

- Enterprise JavaBeans(EJBs) provide another layer where the platforms logic is stored. An EJB server provides functions such as threading, concurrency ,security and memory management. These services are transparent to the author.

- Java Database Connectivity(JDBC), which is the Java equivalent to ODBC,is the standard interface for Java databases.

- The Java servlet API enhances consistency for developers without requiring a graphical user interface.MVC ArchitectureModel–View–Controller (MVC) is an architecture that separates the representation of informationfrom the users interaction with it. The model consists of application data and business rules, and the controller mediates input, converting it to commands for the model or view.A view can be anyoutput representation of data, such as a chart or a diagram. Multiple views of the same data arepossible, such as a pie chart for management and a tabular view for accountants. The central ideabehind MVC is code reusability and separation of concerns.

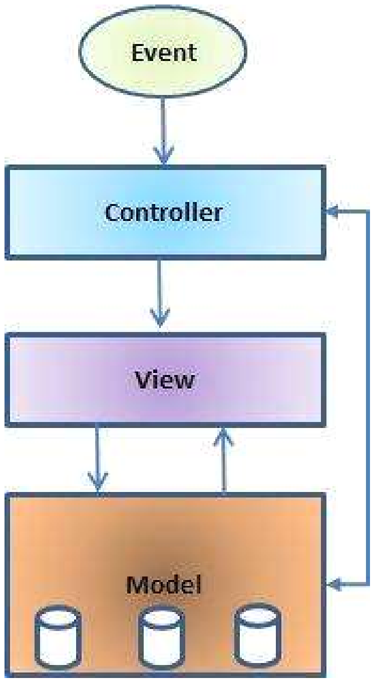


Figure 4.1 **MVC Architecture**

JAVA BEANS:-JavaBeans are reusable software components forJava that can be manipulated visually in a buildertool. Practically, they are classes written in the Java programming language conforming to aparticular convention. They are used to encapsulate many objects into a single object (the bean), sothat they can be passed around as a single bean object instead of as multiple individual objects. AJava Bean is a Java Object that is serializable, has a nullary constructor, and allows access toproperties using getter and setter methods.

BASIC BEAN CONCEPTS:-Individual Java Beans will vary in functionality, but most share certain common defining features.

- Support for introspection allowing a builder tool to analyze how a bean works.
- Support for customization allowing a user to alter the appearance and behavior of a bean.
- Support for events allowing beans to fire events, and informing builder tools about both the events they can fire and the events they can handle.
- Support for properties allowing beans to be manipulated programmatically, as well as

to support the customization mentioned above

A JavaBean is a specially constructed Java class written in the Java and coded according to the JavaBeans API specifications.

Following are the unique characteristics that distinguish a JavaBean from other Java classes:

- It provides a default, no-argument constructor.
- It should be serializable and implement the **Serializable** interface.
- It may have a number of properties which can be read or written.
- It may have a number of "getter" and "setter" methods for the properties.

JavaBeans Properties:

A JavaBean property is a named attribute that can be accessed by the user of the object. The attribute can be of any Java data type, including classes that you define.

A JavaBean property may be read, write, read only, or write only. JavaBean properties are accessed through two methods in the JavaBean's implementation class:

Method	**Description**
get**PropertyName**()	For example, if property name is *firstName*, your method name would be getFirstName() to read that property. This method is called accessor.
set**PropertyName**()	For example, if property name is *firstName*, your method name would be setFirstName() to write that property. This method is called mutator.

A read-only attribute will have only a get**PropertyName**() method, and a write-only attribute will have only a set**PropertyName**() method.

JavaBeans Example:

Consider a student class with few properties:

```
package com.tutorialspoint;


public class StudentsBean implements java.io.Serializable
{
   private String firstName = null;
   private String lastName = null;
```

```
  private int age = 0;

  public StudentsBean() {
  }
  public String getFirstName(){
    return firstName;
  }
  public String getLastName(){
    return lastName;
  }
  public int getAge(){
    return age;
  }
  public void setFirstName(String firstName){
    this.firstName = firstName;
  }
  public void setLastName(String lastName){
    this.lastName = lastName;
  }
  public void setAge(Integer age){
    this.age = age;
  }
}
```

Accessing JavaBeans:

The **useBean** action declares a JavaBean for use in a JSP. Once declared, the bean becomes a scripting variable that can be accessed by both scripting elements and other custom tags used in the JSP. The full syntax for the useBean tag is as follows:

```
<jsp:useBean id="bean's name" scope="bean's scope" typeSpec/>
```

Here values for the scope attribute could be page, request, session or application based on your requirement. The value of the **id** attribute may be any value as a long as it is a unique name among other useBean declarations in the same JSP.

Following example shows its simple usage:

```
<html>
<head>
```

```
<title>useBean Example</title>
</head>
<body>

<jsp:useBean id="date" class="java.util.Date" />
<p>The date/time is <%= date %>

</body>
</html>
```

This would produce following result:

```
The date/time is Thu Sep 30 11:18:11 GST 2010
```

Accessing JavaBeans Properties:

Along with <jsp:useBean...>, you can use <jsp:getProperty/> action to access get methods and <jsp:setProperty/> action to access set methods. Here is the full syntax:

```
<jsp:useBean id="id" class="bean's class" scope="bean's scope">
  <jsp:setProperty name="bean's id" property="property name"
                value="value"/>
  <jsp:getProperty name="bean's id" property="property name"/>
  ...........
</jsp:useBean>
```

The name attribute references the id of a JavaBean previously introduced to the JSP by the useBean action. The property attribute is the name of the get or set methods that should be invoked.

Following is a simple example to access the data using above syntax:

```
<html>
<head>
<title>get and set properties Example</title>
</head>
<body>

<jsp:useBean id="students"
                class="com.tutorialspoint.StudentsBean">
  <jsp:setProperty name="students" property="firstName"
```

```
                value="Zara"/>
   <jsp:setProperty name="students" property="lastName"
                value="Ali"/>
   <jsp:setProperty name="students" property="age"
                value="10"/>
</jsp:useBean>

<p>Student First Name:
   <jsp:getProperty name="students" property="firstName"/>
</p>
<p>Student Last Name:
   <jsp:getProperty name="students" property="lastName"/>
</p>
<p>Student Age:
   <jsp:getProperty name="students" property="age"/>
</p>

</body>
</html>
```

Let us make StudentsBean.class available in CLASSPATH and try to access above JSP. This would produce following result:

```
Student First Name: Zara

Student Last Name: Ali

Student Age: 10
```

SERVLET

Servlet technology is used to create web application (resides at server side and generates dynamic web page).
Servlet technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was popular as a server-side programming language. But there was many disadvantages of this technology. We have discussed these disadvantages below.
There are many interfaces and classes in the servlet API such as Servlet, GenericServlet, HttpServlet,ServletRequest, ServletResponse etc.

What is a Servlet?
Servlet can be described in many ways, depending on the context.

- Servlet is a technology i.e. used to create web application.
- Servlet is an API that provides many interfaces and classes including documentations.
- Servlet is an interface that must be implemented for creating any servlet.
- Servlet is a class that extend the capabilities of the servers and respond to the incoming request. It can respond to any type of requests.
- Servlet is a web component that is deployed on the server to create dynamic web page.

Advantage of Servlet

There are many advantages of servlet over CGI. The web container creates threads for handling the multiple requests to the servlet. Threads have a lot of benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low. The basic benefits of servlet are as follows:

1. **Better performance:** because it creates a thread for each request not process.
2. **Portability:** because it uses java language.
3. **Robust:** Servlets are managed by JVM so we don't need to worry about memory leak, garbage collection etc.
4. **Secure:** because it uses java language..

Java Server Pages(JSP):-Java Server Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems[1], JSP is similar to PHP, but it uses the Java programming language. To deploy and run Java Server Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required. Java Server Pages is a technology which permits software developers to create dynamic request like HTML, XML in order to answer to client request in the net. This technology lets Java code and definite pre-defined procedures to be implanted into static content. The syntax in Java Server Pages includes a supplementary XML tag which is known as JSP actions. It is made use to evoke the utility of the built-in functions. Moreover JSP permits to establish and form the JSP tag libraries which operate as an extension to the standard XML or HTML tags. These JSP tag libraries give a good technique to widen the potentiality of the Web server by providing an independent platform's compiler compiles the JSPs into Java ServletsJSPA JSP compiler may possibly create a servlet in Java code and it is later compiled by the Javacompiler. It might even directly produce the byte code for the servlet. Java Server Pages can beexamined as a high level abstraction of servlets which is practiced as an extension of theServlet2.

Enterprise Java Beans

EJB stands for Enterprise Java Beans. EJB is an essential part of a J2EE platform. J2EE platform have component based architecture to provide multi-tiered, distributed and highly transactional features to enterprise level applications.

EJB provides an architecture to develop and deploy component based enterprise applications considering robustness, high scalability and high performance. An EJB application can be deployed on any of the application server compliant with J2EE 1.3 standard specification..

Benefits

- Simplified development of large scale enterprise level application.
- Application Server/ EJB container provides most of the system level services like transaction handling, logging, load balancing, persistence mechanism, exception handling and so on. Developer has to focus only on business logic of the application.
- EJB container manages life cycle of ejb instances thus developer needs not to worry about when to create/delete ejb objects.

3.1.3 **Database:**

Here MySQL is used as a database because of its platform independence and high compatibility with java. It is world's most popular open source database which requires less memory as compared to other databases.

The MySQL database has become the world's most popular open source database because of its high performance, high reliability and ease of use. It is also the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, MySQL, PHP / Perl / Python.) MySQL runs on more than 20 platforms including Linux, Windows, Mac OS, Solaris, IBM AIX, giving you the kind of flexibility that puts you in control. Whether you're new to database technology or an experienced developer or DBA, MySQL offers a comprehensive range of database tools, support, training and consulting services to make you successful.The MySQL Database powers the most demanding Web, E-commerce and Online Transaction Processing (OLTP) applications. It is a fully integrated transaction-safe, ACID compliant databasewith full commit, rollback, crash recovery and row level locking capabilities. MySQL delivers the ease of use, scalability, and performance that has made MySQL the world's most popular open source database. The MySQL Database 5.5 improves performance and scalability on multi-processor hardware architectures. The new replication monitoring and manageability features provide developers and DBAs with improved tools for building high performance, scalable applications. In addition, the MySQL Performance Schema provides low-level insight into MySQL database performance metrics.

MySQL Database 5.0 delivers enterprise features, including:

Reliability requiring little or no intervention to achieve continuous uptime

Ease of use with "15 minutes to success" installation and configuration

Low administration with very little database maintenance required

Replication providing flexible topologies for scale-out and high availability

Partitioning to improve performance and management of very large database environments

ACID Transactions to build reliable and secure business critical applications

Stored Procedures to improve developer productivity

Triggers to enforce complex business rules at the database level

Views to ensure sensitive information is not compromised

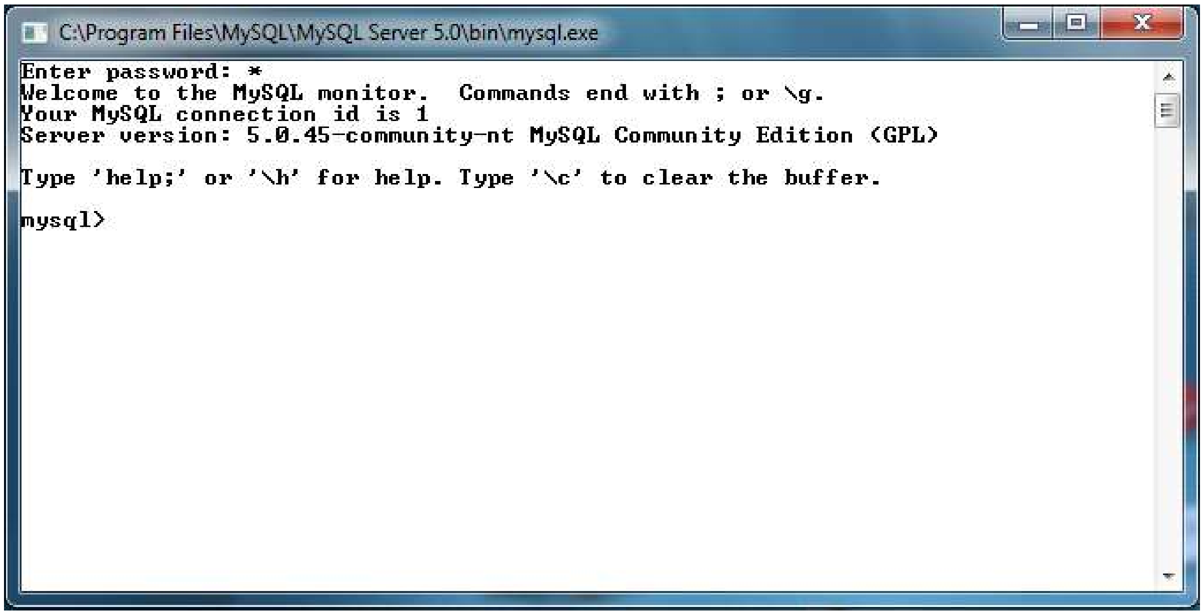
Information Schema to provide easy access to metadata

Connecting to and Disconnecting from the Server

To connect to the server, you will usually need to provide a MySQL user name when you invoke mysql and, most likely, a password.

If the server runs on a machine other than the one where you log in, you will also need to specify a host name. Contact your administrator to find out what connection parameters you should use to connect (that is, what host, user name, and password to use).

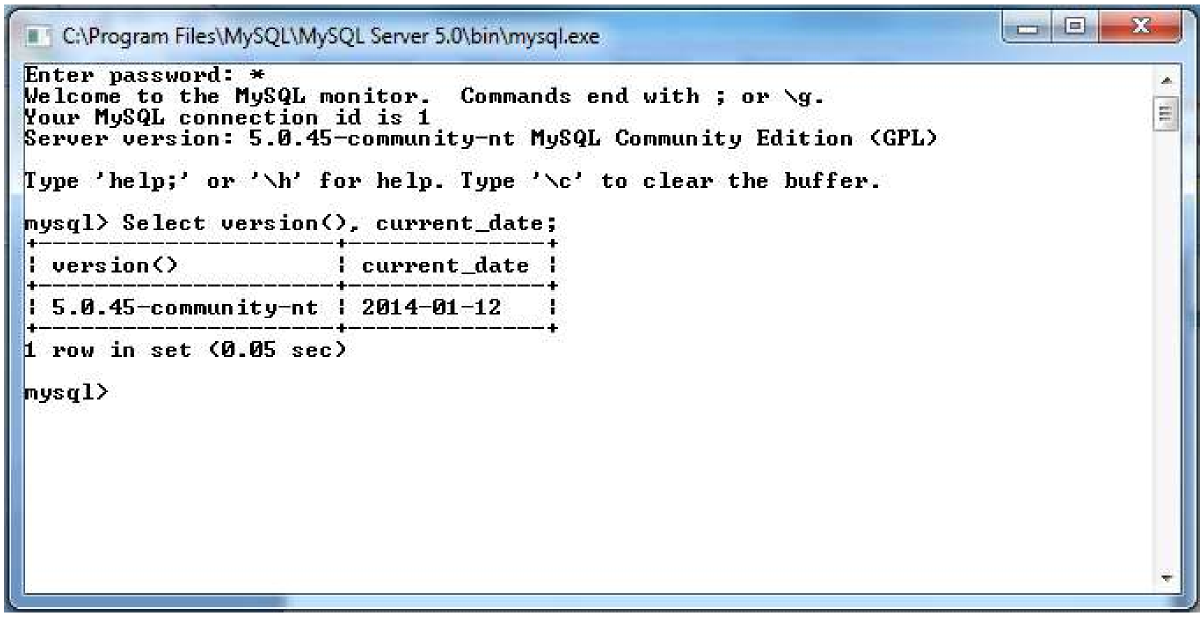
While using MySQL command prompt first of all it asks you to enter the password. Then a query can be written.



```
Enter password: *
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.0.45-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Figure 4.2: Enter password in Mysql

Here is a simple command that asks the server to tell you its version number and the current date. Type it in as shown here following:



```
Enter password: *
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.0.45-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> Select version(), current_date;
+---------------------+--------------+
| version()           | current_date |
+---------------------+--------------+
| 5.0.45-community-nt | 2014-01-12   |
+---------------------+--------------+
1 row in set (0.05 sec)

mysql>
```

Figure 4.3: Screenshot showing current version and date.

This query illustrates several things about mysql:

- A command normally consists of an SQL statement followed by a semicolon. (There are some exceptions where a semicolon may be omitted. QUIT, mentioned earlier, is one of them. We'll get to others later.)
- When you issue a command, MySQL sends it to the server for execution and displays the results, then prints another MySQL prompt to indicate that it is ready for another command.
- MySQLdisplays query output in tabular form (rows and columns). The first row contains labels for the columns. The rows following are the query results. Normally, column labels are the names of the columns you fetch from database tables. If you're retrieving the value of an expression rather than a table column mysql labels the column using the expression itself.
- MySQL shows how many rows were returned and how long the query took to execute, which gives you a rough idea of server performance.
- These values are imprecise because they represent wall clock time (not CPU or machine time), and because they are affected by factors such as server load and network latency.

Creating and Selecting a Database

If the administrator creates your database for you when setting up your permissions, you can begin using it. Otherwise, you need to create it yourself:

```
mysql>CREATE DATABASE pet
```

Creating a database does not select it for use; you must do that explicitly. To make menagerie the current database, use this command:

```
mysql>USE pet
```

Database changed.

Creating a Table

Creating the database is the easy part, but at this point it is empty, as SHOW TABLES tells you:

```
mysql>SHOW TABLES;
```

```
Empty set (0.00 sec)
```

Use a CREATE TABLE statement to specify the layout of your table:

```
mysql>CREATE TABLE dogs (name VARCHAR(20), owner VARCHAR(20));
```

Entering values in a Table

The following query is used to enter values into a table:

When we want to add new records one at a time, the INSERT statement is useful. We supply values for each column, in the order in which the columns were listed in the CREATE TABLE statement. Suppose that Diane gets a new hamster named “Puffball.” We can add a new record using an INSERT statement like this .

```
mysql>INSERT INTO pet

->VALUES ('Puffball','Diane','hamster',1999-03-30',NULL);
```

Retrieving Information from a Table

The SELECT statement is used to pull information from a table. The general form of the statement is:

Creating and Using a Database

SELECT *what_to_select*

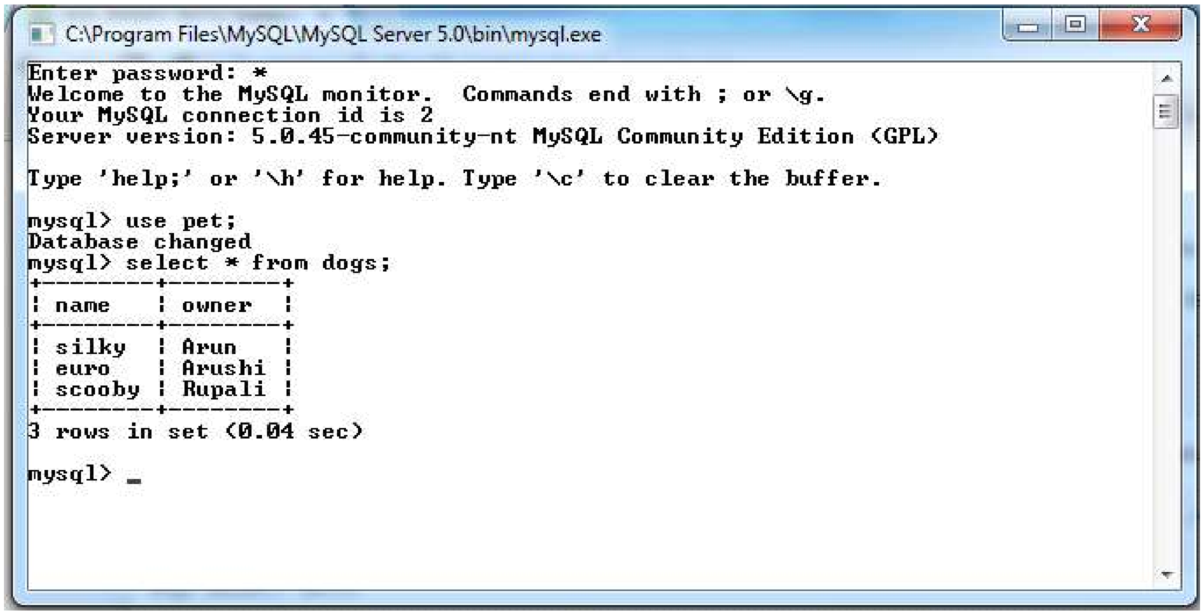
FROM *which_table*

WHERE *conditions_to_satisfy*;

*what_to_select*indicates what we want to see. This can be a list of columns, or * to indicate "all columns." *which_table*indicates the table from which we want to retrieve data. The WHERE clause is optional. If it is present, *conditions_to_satisfy*specifies one or more conditions that rows must satisfy to qualify for retrieval.

Selecting All Data

The simplest form of SELECT retrieves everything from a table:

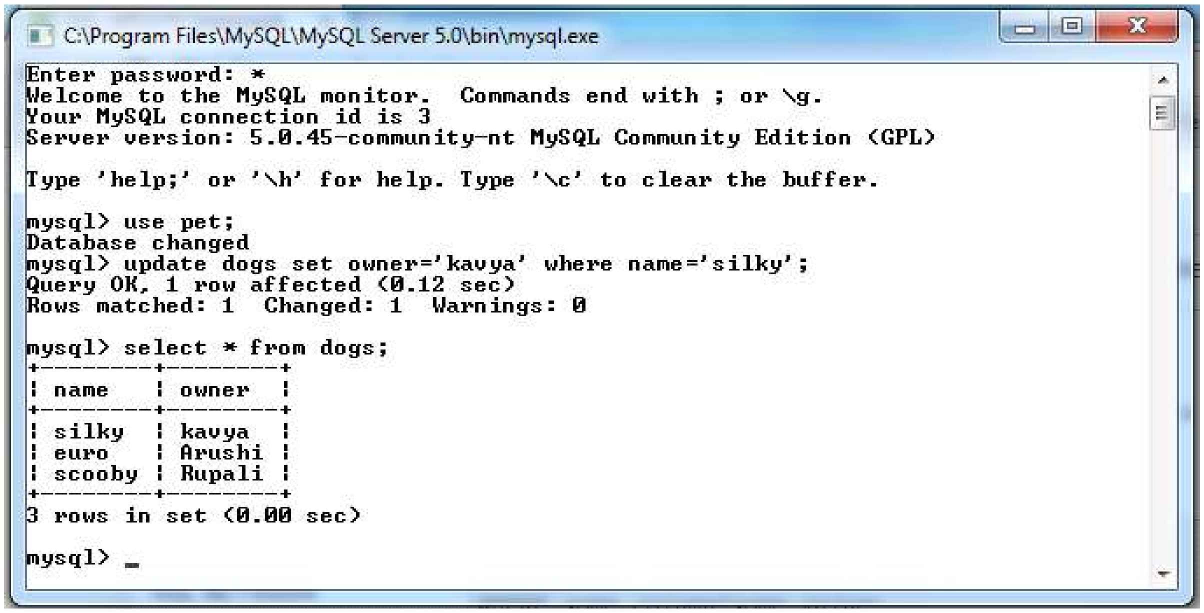

C:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe

```
Enter password: *
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.0.45-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use pet;
Database changed
mysql> select * from dogs;
+---------+--------+
| name    | owner  |
+---------+--------+
| silky   | Arun   |
| euro    | Arushi |
| scooby  | Rupali |
+---------+--------+
3 rows in set (0.04 sec)

mysql> _
```

Fig4.4Selecting all data from table

Updating Data

When you need to update data the query is:

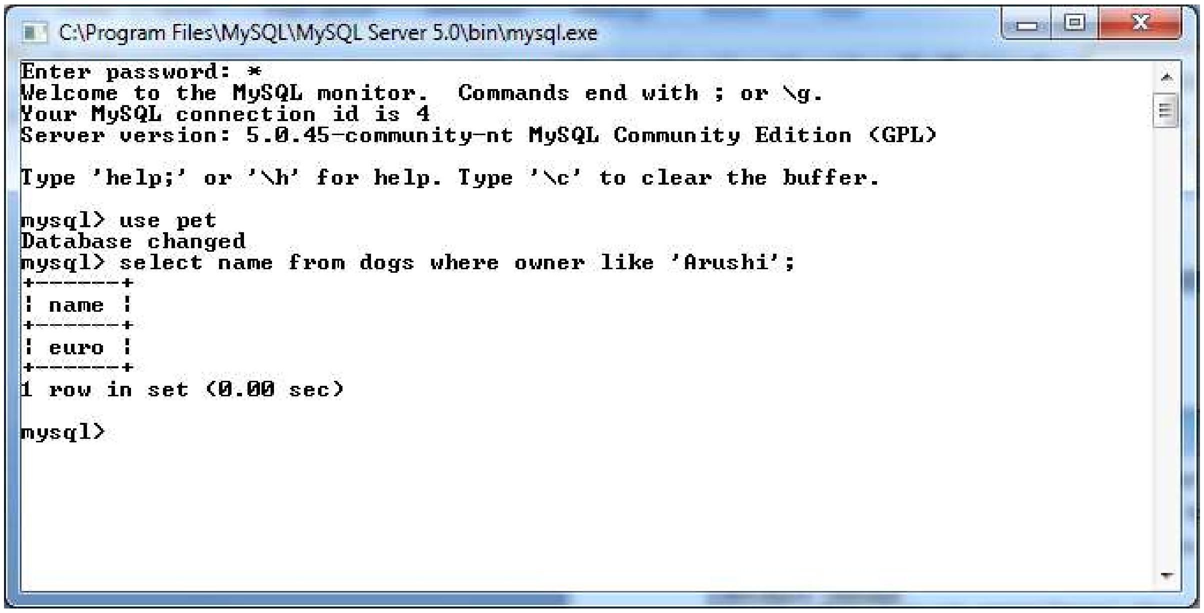


```
Enter password: *
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.0.45-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use pet;
Database changed
mysql> update dogs set owner='kavya' where name='silky';
Query OK, 1 row affected (0.12 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from dogs;
+--------+--------+
| name   | owner  |
+--------+--------+
| silky  | kavya  |
| euro   | Arushi |
| scooby | Rupali |
+--------+--------+
3 rows in set (0.00 sec)

mysql> _
```

Figure 4.5: Updating data in mysql.

SELECTING A PARTICULAR DATA

You can select only particular rows from your table.



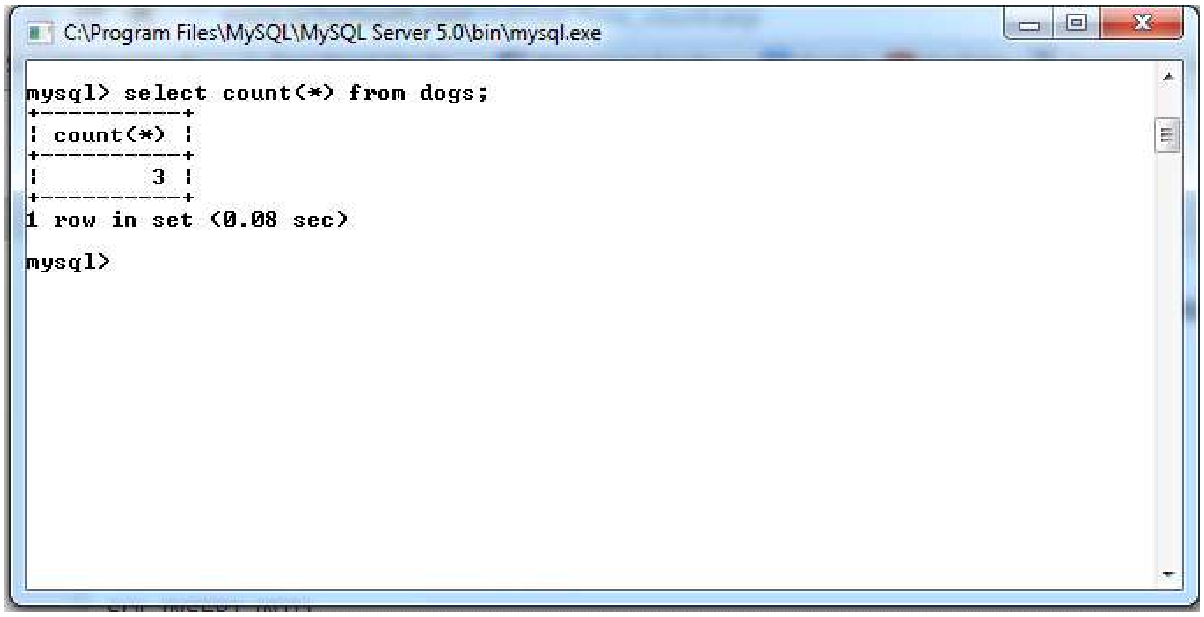
```
Enter password: *
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.0.45-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use pet
Database changed
mysql> select name from dogs where owner like 'Arushi';
+------+
| name |
+------+
| euro |
+------+
1 row in set (0.00 sec)

mysql>
```

Fig4.6Selecting particular data

Counting Rows

Databases are often used to answer the question, “How often does a certain type of data occur in a table?” For example, you might want to know how many pets you have, or how many pets each owner has, or you might want to perform various kinds of census operations on your animals.

Counting the total number of animals you have is the same question as “How many rows are in the pet table?” because there is one record per pet. COUNT(*) counts the number of rows, so the query to count your animals looks like this:



```
mysql> select count(*) from dogs;
+----------+
| count(*) |
+----------+
|        3 |
+----------+
1 row in set (0.08 sec)

mysql>
```

Figure 4.7: Counting number of rows.

Java Database Connectivity

What Is JDBC?

JDBC is a Java 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. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere. 1

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things:

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

ADDING [MY SQL JDBC Driver] Library in NetBeans:

The first step for establishing data connectivity is to add the MySQL JDBC Driver Library for use by your project. This process has to be repeated every time we start a new project but not for new forms or applications added to an existing project. To add the MySQL JDBC Driver Library follow the given steps.

Step 1: Right click on the Project name and select the Properties option:

Step 2: In the Properties dialog box,

i. choose the Libraries option from the Categories pane

ii. click on the Add Library button

iii. From the Add Library dialog box choose the MySQL JDBC Driver

iv. Click on the Create button

The driver is now added to the compile time libraries and this can be verified by clicking on the Compile Time Libraries tab in the Project Properties dialog box. Click on OK to close the Project Properties dialog box.

Basic Libraries Required in the Application for Data Connectivity:

Apart from the MySQL JDBC driver, we also need to import some basic class libraries which are essential for setting up the connection with the database and retrieve data from the database – namely DriverManager Class, Connection Class and Statement Class. These class libraries can be added using the following commands in our application code:

- **import java.sql.DriverManager;**
- **import com.mysql.jdbc.Connection;**
- **import com.mysql.jdbc.Statement;**

The basic purpose for each of these class libraries is:

- The JDBC DriverManager class defines objects which can connect Java applications to a JDBC driver. DriverManager is considered the backbone of JDBC architecture. DriverManager class manages the JDBC drivers that are installed on the system. Its getConnection() method is used to establish a connection to a database. It uses a username, password, and a jdbcurl to establish a connection to the database and returns a connection object.
- A JDBC Connection represents a session/connection with a specific database. Connection interface defines methods for interacting with the database via the established connection. An application can have one or more connections with a single database, or it can have many connections with different databases.
- Once a connection is obtained we can interact with the database. To execute SQL statements, we need to instantiate (create) a Statement object using the connection object. A Statement object is used to send and execute SQL statements to a database.

Running SQL Commands in netbeans:

After executing the Contact List Application, if we need to verify whether the row has been added to the table or not, then we need not go back to the MySQL prompt. Netbeans allows one to directly run SQL commands from the GUI interface. Therefore, the content of the table can directly be tested by running SQL in Netbeans. To execute an SQL command in Netbeans perform the following steps:

Step 1: In the Services tab, right click on the Databases and select the New Connection option from the drop down menu.

Step 2: The New Database Connection dialog window opens up. Provide the values and click OK. Clicking on OK in the above dialog window adds the connection to the existing list. Note that this step is required to be done only once, the first time when creating a connection.

Step 3: Right click on the newly added Connection and select the Connect option from the drop down menu. This step is to be repeated every time we start Netbeans. In the Connect dialog box that opens up, enter the user name and password and select the Remember Password checkbox.

After establishing the connection, right click on the connection driver and select the Execute Command option from the drop down menu. The screen is displayed. Now, type in any SQL to be executed on the database and execute the command. The result is displayed in the bottom half of the window.

Java Server Pages(JSP)

JavaServer Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases.

JavaServer Pages (JSP) is a technology for developing web pages that support dynamic content which helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.

A JavaServer Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.

Using JSP, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.

JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages and sharing information between requests, pages etc.

JSP Processing

The following steps explain how the web server creates the web page using JSP:

- As with a normal page, your browser sends an HTTP request to the web server.
- The web server recognizes that the HTTP request is for a JSP page and forwards it to a JSP

engine. This is done by using the URL or JSP page which ends with **.jsp** instead of .html.

- The JSP engine loads the JSP page from disk and converts it into a servlet content. This conversion is very simple in which all template text is converted to println( ) statements and all JSP elements are converted to Java code that implements the corresponding dynamic behavior of the page.

- The JSP engine compiles the servlet into an executable class and forwards the original request to a servlet engine.

- A part of the web server called the servlet engine loads the Servlet class and executes it. During execution, the servlet produces an output in HTML format, which the servlet engine passes to the web server inside an HTTP response.

- The web server forwards the HTTP response to your browser in terms of static HTML content.

- Finally web browser handles the dynamically generated HTML page inside the HTTP response exactly as if it were a static page.

All the above mentioned steps can be shown below in the following diagram:

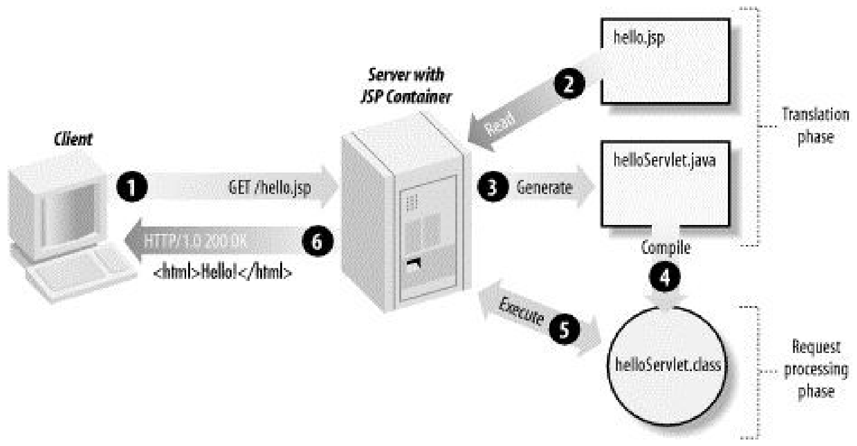


Figure 4.8: JSP Processing

The Scriptlet:

A scriptlet can contain any number of JAVA language statements, variable or method declarations, or expressions that are valid in the page scripting language.

Following is the syntax of Scriptlet:

```
<% CODE FRAGMENT %>
```

Any text, HTML tags, or JSP elements you write must be outside the scriptlet. Following is the simple and first example for JSP:

```
<html>

<head><title>Hello World</title></head>

<body>

Hello World!!

<%

Out.println(“your IP address is”+ request.getRemoteAddr());

</body>

</html>
```

Java Standard Tag Libraries

The JavaServer Pages Standard Tag Library (JSTL) is a collection of useful JSP tags which encapsulates core functionality common to many JSP applications.

JSTL has support for common, structural tasks such as iteration and conditionals, tags for manipulating XML documents, internationalization tags, and SQL tags. It also provides a framework for integrating existing custom tags with JSTL tags.

A library of jstl used in this project is SQL Tags:

The JSTL SQL tag library provides tags for interacting with relational databases (RDBMSs) such as Oracle, mySQL, or Microsoft SQL Server.

Following is the syntax to include JSTL SQL library in your JSP:

<%@ taglib prefix="sql"

url =http://java.sun.com/jsp/jstl/sql%>

Following is the list of SQL JSTL Tags:

Tag	Description
<sql:setDataSource>	Creates a simple DataSource suitable only for prototyping
<sql:query>	Executes the SQL query defined in its body or through the sql attribute.
<sql:update>	Executes the SQL update defined in its body or through the sql attribute.
<sql:param>	Sets a parameter in an SQL statement to the specified value.
<sql:dateParam>	Sets a parameter in an SQL statement to the specified java.util.Date value.
<sql:transaction >	Provides nested database action elements with a shared Connection, set up to execute all statements as one transaction.

Table 2.8: SQL JSTL Tags

Create Table

To create the **Employees** table in EMP database, use the following steps:

Step 1:

Open a **Command Prompt** and change to the installation directory as follows:

```
C:\>
```

```
C:\>cd Program Files\MySQL\bin

C:\Program Files\MySQL\bin>
```

Step 2:

Login to database as follows

```
C:\Program Files\MySQL\bin>mysql -u root -p

Enter password: ********

mysql>
```

Step 3:

Create the table **Employee** in **TEST** database as follows:

```
mysql> use TEST;

mysql> create table Employees

   (

    id int not null,

    age int not null,

    first varchar (255),

    last varchar (255)

   );

Query OK, 0 rows affected (0.08 sec)

mysql>
```

Create Data Records

Finally you create few records in Employee table as follows:

```
mysql> INSERT INTO Employees VALUES (100, 18, 'Zara', 'Ali');

Query OK, 1 row affected (0.05 sec)


mysql> INSERT INTO Employees VALUES (101, 25, 'Mahnaz', 'Fatma');

Query OK, 1 row affected (0.00 sec)
```

```
mysql> INSERT INTO Employees VALUES (102, 30, 'Zaid', 'Khan');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Employees VALUES (103, 28, 'Sumit', 'Mittal');
Query OK, 1 row affected (0.00 sec)

mysql>
```

SELECT Operation:

Following example shows how we can execute SQL SELECT statement using JTSL in JSP programming:

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ page import="javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>

<html>
<head>
<title>SELECT Operation</title>
</head>
<body>

<sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost/TEST"
    user="root"  password="pass123"/>

<sql:query dataSource="${snapshot}" var="result">
SELECT * from Employees;
</sql:query>
```

```
<table border="1" width="100%">
<tr>
  <th>Emp ID</th>
  <th>First Name</th>
  <th>Last Name</th>
  <th>Age</th>
</tr>
<c:forEach var="row" items="${result.rows}">
<tr>
  <td><c:out value="${row.id}"/></td>
  <td><c:out value="${row.first}"/></td>
  <td><c:out value="${row.last}"/></td>
  <td><c:out value="${row.age}"/></td>
</tr>
</c:forEach>
</table>

</body>
</html>
```

Now try to access above JSP, which should display the following result:

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Khan	30
103	Sumit	Mittal	28

Table 2.9 Dispalying tables

INSERT Operation:

Following example shows how we can execute SQL INSERT statement using JTSL in JSP programming:

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ page import="javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>
```

```
<html>
<head>
<title>JINSERT Operation</title>
</head>
<body>

<sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"
     url="jdbc:mysql://localhost/TEST"
     user="root"  password="pass123"/>



<sql:update dataSource="${snapshot}" var="result">
INSERT INTO Employees VALUES (104, 2, 'Nuha', 'Ali');
</sql:update>

<sql:query dataSource="${snapshot}" var="result">
SELECT * from Employees;
</sql:query>

<table border="1" width="100%">
<tr>
   <th>Emp ID</th>
   <th>First Name</th>
   <th>Last Name</th>
   <th>Age</th>
</tr>
<c:forEach var="row" items="${result.rows}">
<tr>
   <td><c:out value="${row.id}"/></td>
   <td><c:out value="${row.first}"/></td>
   <td><c:out value="${row.last}"/></td>
   <td><c:out value="${row.age}"/></td>
</tr>
</c:forEach>
</table>
```

```
</body>

</html>
```

Now try to access above JSP, which should display the following result:

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Khan	30
103	Sumit	Mittal	28
104	Nuha	Ali	2

Table2.10Insert Operation

DELETE Operation:

Following example shows how we can execute SQL DELETE statement using JTSL in JSP programming:

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>

<%@ page import="javax.servlet.http.*,javax.servlet.*" %>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>


<html>

<head>

<title>DELETE Operation</title>

</head>

<body>


<sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"

     url="jdbc:mysql://localhost/TEST"

     user="root"  password="pass123"/>


<c:set var="empId" value="103"/>
```

```
<sql:update dataSource="${snapshot}" var="count">
  DELETE FROM Employees WHERE Id = ?
  <sql:param value="${empId}" />
</sql:update>

<sql:query dataSource="${snapshot}" var="result">
   SELECT * from Employees;
</sql:query>

<table border="1" width="100%">
<tr>
   <th>Emp ID</th>
   <th>First Name</th>
   <th>Last Name</th>
   <th>Age</th>
</tr>
<c:forEach var="row" items="${result.rows}">
<tr>
   <td><c:out value="${row.id}"/></td>
   <td><c:out value="${row.first}"/></td>
   <td><c:out value="${row.last}"/></td>
   <td><c:out value="${row.age}"/></td>
</tr>
</c:forEach>
</table>

</body>
</html>
```

Now try to access above JSP, which should display the following result:

Emp ID	First Name	Last Name	Age
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Khan	30

Table 2.11 Delete Operation

UPDATE Operation:

Following example shows how we can execute SQL UPDATE statement using JTSL in JSP programming:

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>
<%@ page import="javax.servlet.http.*,javax.servlet.*" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>

<html>
<head>
<title>DELETE Operation</title>
</head>
<body>

<sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"
     url="jdbc:mysql://localhost/TEST"
     user="root"  password="pass123"/>

<c:set var="empId" value="102"/>

<sql:update dataSource="${snapshot}" var="count">
  UPDATE Employees SET last = 'Ali'
  <sql:param value="${empId}" />
</sql:update>

<sql:query dataSource="${snapshot}" var="result">
   SELECT * from Employees;
</sql:query>

<table border="1" width="100%">
```

```
<tr>
  <th>Emp ID</th>
  <th>First Name</th>
  <th>Last Name</th>
  <th>Age</th>
</tr>
<c:forEach var="row" items="${result.rows}">
<tr>
  <td><c:out value="${row.id}"/></td>
  <td><c:out value="${row.first}"/></td>
  <td><c:out value="${row.last}"/></td>
  <td><c:out value="${row.age}"/></td>
</tr>
</c:forEach>
</table>

</body>
</html>
```

Now try to access above JSP, which should display the following result:

Emp ID	**First Name**	**Last Name**	**Age**
100	Zara	Ali	18
101	Mahnaz	Fatma	25
102	Zaid	Ali	30

Table 2.12Update Operation

The designing of the website is done with HTML(Hyper Text Markup Language).

HTML or **HyperText Markup Language** is the main markup language for creating web pages and other information that can be displayed in a web browsr.

HTML is written in the form of HTML elements consisting of *tags* enclosed in angle brackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent *empty elements* and so are unpaired, for example <img>. The first tag in a pair is the *start tag*, and the second tag is the *end tag* (they are also called *opening tags* and *closing tags*). In between these tags web designers can add text, further tags, comments and other types of text-based content.

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages.

Web browsers can also refer to Cascading Style Sheets (CSS) to define the appearance and layout of text and other material.

This tutorial will give basic idea on simple syntax (ie. elements) involved with JSP development:

The Scriptlet:

A scriptlet can contain any number of JAVA language statements, variable or method declarations, or expressions that are valid in the page scripting language.

Following is the syntax of Scriptlet:

```
<% code fragment %>
```

You can write XML equivalent of the above syntax as follows:

```
<jsp:scriptlet>
  code fragment
</jsp:scriptlet>
```

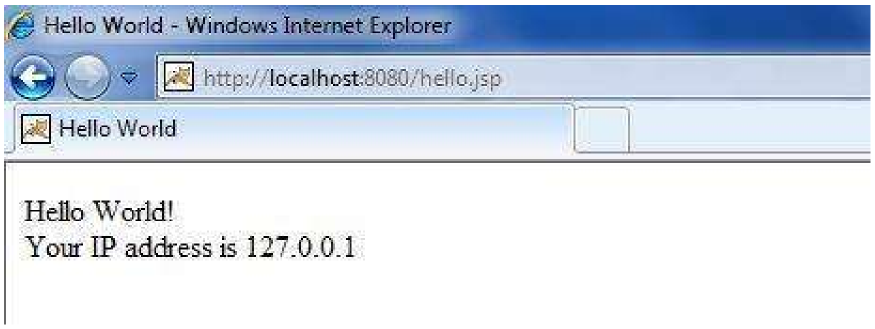
Any text, HTML tags, or JSP elements you write must be outside the scriptlet. Following is the simple and first example for JSP:

```
<html>
<head><title>Hello World</title></head>
```

```
<body>
Hello World!<br/>
<%
out.println("Your IP address is " + request.getRemoteAddr());
%>
</body>
</html>
```

NOTE: Assuming that Apache Tomcat is installed in C:\apache-tomcat-7.0.2 and your environment is setup as per environment setup tutorial.

Let us keep above code in JSP file hello.jsp and put this file in **C:\apache-tomcat-7.0.2\webapps\ROOT** directory and try to browse it by giving URL http://localhost:8080/hello.jsp. This would generate following result:


Hello World - Windows Internet Explorer
http://localhost:8080/hello.jsp
Hello World

```
Hello World!
Your IP address is 127.0.0.1
```

JSP Declarations:

A declaration declares one or more variables or methods that you can use in Java code later in the JSP file. You must declare the variable or method before you use it in the JSP file.

Following is the syntax of JSP Declarations:

```
<%! declaration; [ declaration; ]+ ... %>
```

You can write XML equivalent of the above syntax as follows:

```
<jsp:declaration>
   code fragment
</jsp:declaration>
```

Following is the simple example for JSP Declarations:

```
<%! int i = 0; %>
<%! int a, b, c; %>
<%! Circle a = new Circle(2.0); %>
```

JSP Expression:

A JSP expression element contains a scripting language expression that is evaluated, converted to a String, and inserted where the expression appears in the JSP file.

Because the value of an expression is converted to a String, you can use an expression within a line of text, whether or not it is tagged with HTML, in a JSP file.

The expression element can contain any expression that is valid according to the Java Language Specification but you cannot use a semicolon to end an expression.

Following is the syntax of JSP Expression:

```
<%= expression %>
```

You can write XML equivalent of the above syntax as follows:

```
<jsp:expression>
  expression
</jsp:expression>
```

Following is the simple example for JSP Expression:

```
<html>
<head><title>A Comment Test</title></head>
<body>
<p>
  Today's date: <%= (new java.util.Date()).toLocaleString()%>
</p>
</body>
</html>
```

This would generate following result:

```
Today's date: 11-Sep-2010 21:24:25
```

JSP Comments:

JSP comment marks text or statements that the JSP container should ignore. A JSP comment is useful when you want to hide or "comment out" part of your JSP page.

Following is the syntax of JSP comments:

```
<%-- This is JSP comment --%>
```

Following is the simple example for JSP Comments:

```
<html>
<head><title>A Comment Test</title></head>
<body>
<h2>A Test of Comments</h2>
<%-- This comment will not be visible in the page source --%>
</body>
</html>
```

This would generate following result:

```
A Test of Comments
```

There are a small number of special constructs you can use in various cases to insert comments or characters that would otherwise be treated specially. Here's a summary:

Table 2.8 Comments

Syntax	**Purpose**
<%-- comment --%>	A JSP comment. Ignored by the JSP engine.
<!-- comment -->	An HTML comment. Ignored by the browser.
<\%	Represents static <% literal.
%\>	Represents static %> literal.
\'	A single quote in an attribute that uses single quotes.

\"	A double quote in an attribute that uses double quotes.

JSP Directives:

A JSP directive affects the overall structure of the servlet class. It usually has the following form:

```
<%@ directive attribute="value" %>
```

There are three types of directive tag:

Directive	Description
<%@ page ... %>	Defines page-dependent attributes, such as scripting language, error page, and buffering requirements.
<%@ include ... %>	Includes a file during the translation phase.
<%@ taglib ... %>	Declares a tag library, containing custom actions, used in the page

We would explain JSP directive in separate chapter JSP - Directives

JSP Actions:

JSP actions use constructs in XML syntax to control the behavior of the servlet engine. You can dynamically insert a file, reuse JavaBeans components, forward the user to another page, or generate HTML for the Java plugin.

There is only one syntax for the Action element, as it conforms to the XML standard:

```
<jsp:action_name attribute="value" />
```

Action elements are basically predefined functions and there are following JSP actions available:

Syntax	Purpose
jsp:include	Includes a file at the time the page is requested
jsp:useBean	Finds or instantiates a JavaBean
jsp:setProperty	Sets the property of a JavaBean
jsp:getProperty	Inserts the property of a JavaBean into the output
jsp:forward	Forwards the requester to a new page
jsp:plugin	Generates browser-specific code that makes an OBJECT or EMBED tag for the Java plugin
jsp:element	Defines XML elements dynamically.
jsp:attribute	Defines dynamically defined XML element's attribute.
jsp:body	Defines dynamically defined XML element's body.
jsp:text	Use to write template text in JSP pages and documents.

We would explain JSP actions in separate chapter JSP - Actions

JSP Implicit Objects:

JSP supports nine automatically defined variables, which are also called implicit objects. These variables are:

Objects	Description

request	This is the **HttpServletRequest** object associated with the request.
response	This is the **HttpServletResponse** object associated with the response to the client.
out	This is the **PrintWriter** object used to send output to the client.
session	This is the **HttpSession** object associated with the request.
application	This is the **ServletContext** object associated with application context.
config	This is the **ServletConfig** object associated with the page.
pageContext	This encapsulates use of server-specific features like higher performance **JspWriters**.
page	This is simply a synonym for **this**, and is used to call the methods defined by the translated servlet class.
Exception	The **Exception** object allows the exception data to be accessed by designated JSP.

We would explain JSP Implicit Objects in separate chapter JSP - Implicit Objects.

Control-Flow Statements:

JSP provides full power of Java to be embedded in your web application. You can use all the APIs and building blocks of Java in your JSP programming including decision making statements, loops etc.

Decision-Making Statements:

The **if...else** block starts out like an ordinary Scriptlet, but the Scriptlet is closed at each line with HTML text included between Scriptlet tags.

```
<%! int day = 3; %>
<html>
<head><title>IF...ELSE Example</title></head>
<body>
<% if (day == 1 | day == 7) { %>
    <p> Today is weekend</p>
<% } else { %>
    <p> Today is not weekend</p>
<% } %>
</body>
</html>
```

This would produce following result:

```
Today is not weekend
```

Now look at the following **switch...case** block which has been written a bit differentlty using **out.println()** and inside Scriptletas:

```
<%! int day = 3; %>
<html>
<head><title>SWITCH...CASE Example</title></head>
<body>
<%
switch(day) {
case 0:
  out.println("It\'s Sunday.");
  break;
case 1:
  out.println("It\'s Monday.");
  break;
case 2:
  out.println("It\'s Tuesday.");
  break;
case 3:
  out.println("It\'s Wednesday.");
  break;
case 4:
  out.println("It\'s Thursday.");
  break;
case 5:
  out.println("It\'s Friday.");
  break;
default:
  out.println("It's Saturday.");
}
%>
</body>
</html>
```

This would produce following result:

```
It's Wednesday.
```

Loop Statements:

You can also use three basic types of looping blocks in Java: **for, while,and do…while** blocks in your JSP programming.

Let us look at the following **for** loop example:

```
<%! int fontSize; %>
<html>
<head><title>FOR LOOP Example</title></head>
<body>
<%for ( fontSize = 1; fontSize <= 3; fontSize++){ %>
  <font color="green" size="<%= fontSize %>">
   JSP Tutorial
  </font><br />
<%}%>
</body>
</html>
```

This would produce following result:

```
  JSP Tutorial
```

```
JSP Tutorial

JSP Tutorial
```

Above example can be written using **while** loop as follows:

```
<%! int fontSize; %>
<html>
<head><title>WHILE LOOP Example</title></head>
<body>
<%while ( fontSize <= 3){ %>
   <font color="green" size="<%= fontSize %>">
    JSP Tutorial
   </font><br />
<%fontSize++;%>
<%}%>
</body>
</html>
```

This would also produce following result:

```
JSP Tutorial

JSP Tutorial

JSP Tutorial
```

JSP Operators:

JSP supports all the logical and arithmetic operators supported by Java. Following table give a list of all the operators with the highest precedence appear at the top of the table, those with the lowest appear at the bottom.

Within an expression, higher precedence operators will be evaluated first.

Table2.9 Operators

Category	**Operator**	**Associativity**
Postfix	() [] . (dot operator)	Left to right
Unary	++ - - ! ~	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	>> >>> <<	Left to right
Relational	> >= < <=	Left to right
Equality	== !=	Left to right
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR	|	Left to right
Logical AND	&&	Left to right
Logical OR	||	Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %= >>= <<= &= ^= |=	Right to left
Comma	,	Left to right

JSP Literals:

The JSP expression language defines the following literals:

- **Boolean:** true and false
- **Integer:** as in Java
- **Floating point:** as in Java
- **String:** with single and double quotes; " is escaped as \", ' is escaped as \', and \ is escaped as \\.
- **Null:** null

4.3 Coding Standards of Language Used

- **Guideline 18: Do not expose methods that use reduced-security checks to untrusted code.** Certain methods use a reduced-security check that checks only that the calling method is authorized rather than checking every method in the call stack. Any code that invokes these methods must guarantee that they cannot be invoked on behalf of untrusted code.
- **Guideline 10: Do not use the *clone()*method to copy untrusted method parameters.** Inappropriate use of the clone() method can allow an attacker to exploit vulnerabilities by providing arguments that appear normal but subsequently return unexpected values. Such objects may consequently bypass validation and security checks.
- **Guideline 25: Document thread-safety and use annotations where applicable.** The Java language annotation facility is useful for documenting design intent. Source code annotation is a mechanism for associating metadata with a program element and making it available to the compiler, analyzers, debuggers, or Java Virtual Machine (JVM) for examination. Several annotations are available for documenting thread-safety or the lack thereof.
- **Guideline 29: Be aware of numeric promotion behavior.** Promotions in which the operands are converted from an int to a float or from a long to a double can cause a loss of precision.
- **Guideline 43: Use a try-with-resources statement to safely handle closeable resources.** Using the try-with-resources statement prevents problems that can arise when closing resources with an ordinary try-catch-finally block, such as failing to close a resource because an exception is thrown as a result of closing another resource, or masking an important exception when a resource is closed.
- **Guideline 45: Use the same type for the second and third operands in conditional expressions.** The complexity of the rules that determine the result type of a conditional expression can result in unintended type conversions. Consequently, the second and third operands of each conditional expression should have identical types.
- **Guideline 57: Avoid inadvertent wrapping of loop counters.** Unless coded properly, a while or for loop may execute forever, or until the counter wraps around and reaches its final value.
- **Guideline 64: Strive for logical completeness.** Software vulnerabilities can result when a programmer fails to consider all possible data states.
- **Guideline 69: Do not confuse abstract object equality with reference equality.** Naïve programmers often confuse the intent of the == operation with that of the Object.equals() method. This confusion is frequently evident in the context of processing of String objects.
- **Guideline 71: Understand how escape characters are interpreted when strings are loaded.** Many classes allow inclusion of escape sequences in character and string literals. Correct use of escape sequences in string literals requires understanding how the escape

sequences are interpreted by the Java compiler, as well as how they are interpreted by any subsequent processor.

4.3Project Scheduling using various tools such as PERT, GANTT Charts, OPEN PROJ. Etc.

PERT – Program Evaluation and Review Technique Developed by Lockheed Corporation in participation with the US Navy Polaris Missile/Submarine Project in 1958
CPM – Critical Path Method developed by DuPont c.1958

WBS – Work Breakdown Structure

A complete depiction of the entire task necessary to achieve successful project completion. Project plans that delineate all the tasks that must be accomplished to successfully complete a project from which scheduling, delegating, and budgeting are derived.

Gantt Charts: Henry Gantt who the Gantt chart is named, worked for the department of defense during the First World War. The chart is widely used as a project management tool. The Gantt chart allows you to see start and stop date for project task and subtask.

Gantt Charts are derived from your Work Breakdown Structure WBS

Critical Path – The longest complete path of a project.

Critical Task – A single task along a critical path

Deliverables –Something of value generated by a team or individual as scheduled often taking the form of a plan, report, procedure, product, or service.

Dependant Task – A task or subtask that cannot be initiated until a predecessor task or several predecessor tasks are finished.

Dummy Task – A link that shows an association or relationship between two otherwise parallel tasks along a PERT/CPM network.

Milestone – A significant event or juncture in the project.

Noncritical Task – A task within a CPM network for which slack time is available.

Parallel Task – Two or more tasks that can be undertaken at the same time. This does not imply that they have the same starting and ending times.

Path – A chronological sequence of tasks, each dependant on the predecessors.

Predecessor Task – Task that must be completed before another task can be completed.

Project – The allocation of resources over a specific timeframe and the coordination of interrelated events to accomplish an overall objective while meeting both predictable and unique challenges.

Project Constraint – A critical project element such as money, time, or human resources.

Scope of the project or **scope of the project –** The level of activity and effort necessary to complete a project and achieve the desired outcomes as measured by hours, days, resources consumed, and funds spent.

Slack –**Margin or extra room to accommodate anticipated potential short falls in planning**

Slack Time – **The time interval in which you have leeway as to when a particular task needs to be completed.**

Task or **Event –** A divisible, definable unit of work related to a project, which may or may not include subtasks.

Timeline **–The scheduled start and stop times for a subtask, task, phase or entire project.**

PERT, CPM and GANTT

Before attempting to use these tools, the project's information must be assembled in a certain way. I include a basic description of the preceding steps.

The project planning process consists of the following:

1.Setting the project start date

2.Setting the project completion date

3.Selecting the project methodology or project life cycle to be used

4.Determining the scope of the project in terms of the phases of the selected project methodology or project life cycle

5.Identifying or selecting the project review methods to be used

6.Identifying any predetermined interim milestone or other critical dates which must be met.

7.Listing tasks, by project phase, in the order in which they might be accomplished.

8.Estimating the personnel necessary to accomplish each task

9.Estimating the personnel available to accomplish each task

10.Determining skill level necessary to perform each task

11.Determining task dependencies

Which tasks can be done in parallel?

Which tasks require the completion of other tasks before they can start?

12.Project control or review points

13.Performing project cost estimation and cost-benefit analysis

Work breakdown Structures

The development of a project plan is predicated on having a clear and detailed understanding of both the tasks involved, the estimated length of time each task will take, the dependencies between those tasks, and the sequence in which those tasks have to be performed. Additionally, resource availability must be determined in order to assign each task or group of tasks to the appropriate worker.

One method used to develop the list of tasks is to create what is known as a work breakdown structure.

A definition

A *work breakdown structure* (WBS) is a hierarchic decomposition or breakdown of a project or major activity into successive levels, in which each level is a finer breakdown of the preceding one. In final form a WBS is very similar in structure and layout to a document outline. Each item at a specific level of a WBS is numbered consecutively (e.g., 10, 10, 30, 40, 50). Each item at the next level is numbered within the number of its parent item (e.g., 10.1, 10.2, 10.3, 10.4).

The WBS may be drawn in a diagrammatic form (if automated tools are available) or in a chart resembling an outline.

The WBS begins with a single overall task representing the totality of work to be performed on the project. This becomes the name of the project plan WBS. Using a methodology or system life cycle (analysis, design and implementation) steps as a guide, the project is divided into its major steps. The first phase is project initiation; the second major phase is analysis, followed by design, construction, testing, implementation, and post-implementation follow-up. Each of these phases must be broken in their next level of detail, and each of those, into still finer levels of detail, until a manageable task size is arrived at. The first WBS level for the life cycle would be:

WBS numberTask Description

1.0Project initiation

1.1Draft project plan

2.0Analysis phase

2.1Plan user interviews

2.2Schedule users interviews

3.0Examination and test

4.0Design

5.0Test

6.0Implementation

1. Post-implementation review

Tasks at each successively finer level of detail are numbered to reflect the task from which they were derived. Thus, the first level of tasks would be numbered 1.0, 2.0, 3.0, and so forth. Each of their subtasks would have a two-part number: the first part reflecting the parent task and the second part, the subtasks number itself, such as 1.1, 1.2, or 1.3. As each of these, in turn, decomposed or broken down into its component tasks, each component receives a number comprised of its parent number plus a unique number of its own.

A definition

A *manageable task* is one in which the expected results can be easily identified; success, failure, or completion of the task can be easily ascertained; the time to complete the task can be easily estimated; ant the resource requirements of the task can be easily determined.

Program evaluation and review technique (PERT) charts depict task, duration, and dependency information. Each chart starts with an initiation node from which the first task, or tasks, originates. If multiple tasks begin at the same time, they are all started from the node or branch, or fork out from the starting point. Each task is represented by a line, which states its name or other identifier, its duration, the number of people assigned to it, and in some cases the initials of the personnel assigned. The other end of the task line is terminated by another node, which identifies the start of another task, or the beginning of any slack time, that is, waiting time between tasks.

Each task is connected to its successor tasks in this manner forming a network of nodes and connecting lines. The chart is complete when all final tasks come together at the completion node. When slack time exists between the end of one task and the start of another, the usual method is to draw a broken or dotted line between the end of the first task and the start of the next dependent task.

A PERT chart may have multiple parallel or interconnecting networks of tasks. If the scheduled project has milestones, checkpoints, or review points (all of which are highly recommended in any project schedule), the PERT chart will note that all tasks up to that point terminate at the review node. It should be noted at this point that the project review, approvals, user reviews, and so forth all take time. This time should never be underestimated when drawing up the project plan. It is not unusual for a review to take 1 or 2 weeks. Obtaining management and user approvals may take even longer.

When drawing up the plan, be sure to include tasks for documentation writing, documentation editing, project report writing and editing, and report reproduction. These tasks are usually time-consuming; so don’t underestimate how long it will take to complete them.

PERT charts are usually drawn on ruled paper with the horizontal axis indicating time period divisions in days, weeks, months, and so on. Although it is possible to draw a PERT chart for an entire project, the usual practice is to break the plans into smaller, more meaningful parts. This is very helpful if the chart has to be redrawn for any reason, such as skipped or incorrectly estimated tasks.

Many PERT charts terminate at the major review points, such as at the end of the analysis. Many organizations include funding reviews in the projects life cycle. Where this is the case, each chart terminates in the funding review node.

Funding reviews can affect a project in that they may either increase funding, in which case more people have to make available, or they may decrease funding, in which case fewer people may be available. Obviously more or less people will affect the length of time it takes to complete the project.

Critical Path Method (CPM) charts are similar to PERT charts and are sometimes known as PERT/CPM. In a CPM chart, the critical path is indicated. A critical path consists that set of dependent tasks (each dependent on the preceding one), which together take the longest time to complete. Although it is not normally done, a CPM chart can define multiple, equally critical paths. Tasks, which fall on the critical path, should be noted in some way, so that they may be given special attention. One way is to draw critical path tasks with a double line instead of a single line.

Tasks, which fall on the critical path, should receive special attention by both the project manager and the personnel assigned to them. The critical path for any given method may shift as the project progresses; this can happen when tasks are completed either behind or ahead of schedule, causing other tasks which may still be on schedule to fall on the new critical path.

A Gantt chart is a matrix, which lists on the vertical axis all the tasks to be performed. Each row contains a single task identification, which usually consists of a number and name. The horizontal axis is headed by columns indicating estimated task duration, skill level needed to perform the task, and the name of the person assigned to the task, followed by one column for each period in the project's duration. Each period may be expressed in hours, days, weeks, months, and other time units. In some cases it may be necessary to label the period columns as period 1, period 2, and so on.

The graphics portion of the Gantt chart consists of a horizontal bar for each task connecting the period start and period ending columns. A set of markers is usually used to indicate estimated and actual start and end. Each bar on a separate line, and the name of each person assigned to the task is on a separate line. In many cases when this type of project plan is used, a blank row is left between tasks. When the project is under way, this row is used to indicate progress, indicated by a second bar, which starts in the period column when the task is actually started and continues until the task is actually completed. Comparison between estimated start and end and actual start and end should indicate project status on a task-by-task basis.

Variants of this method include a lower chart, which shows personnel allocations on a person-by-person basis. For this section the vertical axis contains the number of people assigned to the project, and the columns indicating task duration are left blank, as is the column indicating person assigned. The graphics consists of the same bar notation as in the upper chart indicates that the person is working on a task. The value of this lower chart is evident when it shows slack time for the project personnel, that is, times when they are not actually working on any project.

4.4 Testing Techniques and Test Plans:

Testing a program consists of providing the program with a set of test inputs (or test cases) and observing if the program behaves as expected. If the program fails to behave as expected, then the conditions under which failure occurs are noted for later debugging and correction.

Some commonly used terms associated with testing are:

• **Failure:** This is a manifestation of an error (or defect or bug). But, the mere presence of an error may not necessarily lead to a failure.

• **Test case:** This is the triplet [I,S,O] , where I is the data input to the system, S is the state of the system at which the data is input, and O is the expected output of the system.

• **Test suite:** This is the set of all test cases with which a given software product is to be tested.

In this project we followed unit testing. Each and every module is tested individually

Table 4.1: Test case 1: Log- on

Input	***Desire Output***	***Actual Output***	***Status***
Enter the login details.	Specific Menu page should be displayed based on the accessibility criteria.	Specific Menu page is displayed based on the accessibility criteria.	Pass

Table 3.2: Test case 2: Entering data

Input	***Desire Output***	***Actual Output***	***Status***
Click on the	Enter the specific	Specific information is	Pass

particular Tab Item for desired service	information to be stored in the database.	stored in the database based on the fields in the activity	

Table 3.3: Test case 3: Listing of data

Input	***Desire Output***	***Actual Output***	***Status***
Clicking on the My Services in Drawer previews the requested services with their corresponding status and request ids	The specified contents will be displayed.	The specified contents are displayed.	Pass

Chapter 5
Results and Discussion

Main Page

Steps:

For admin login detail click on login And fill username and   password

Login Activity:

The login activity will be displayed after the user has created a profile. Here the user can enter the phone number n password to login into his/her the account and can even user the change the password option if required. If any user already have the account he/she can directly log in.

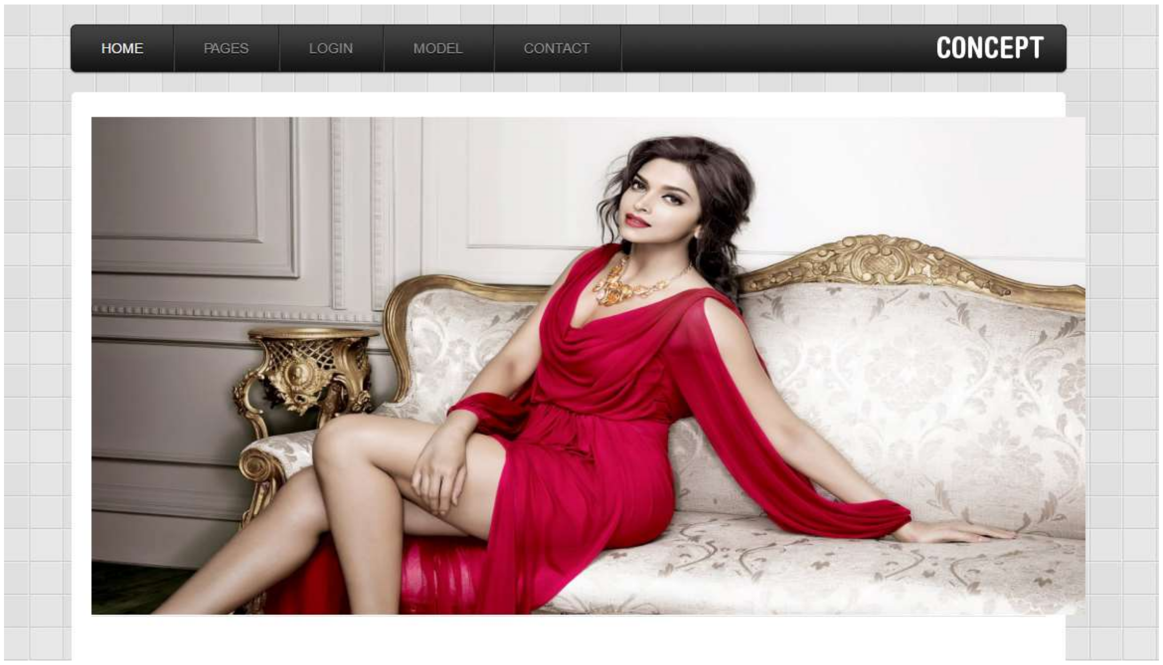


Fig 5.1: Main Page

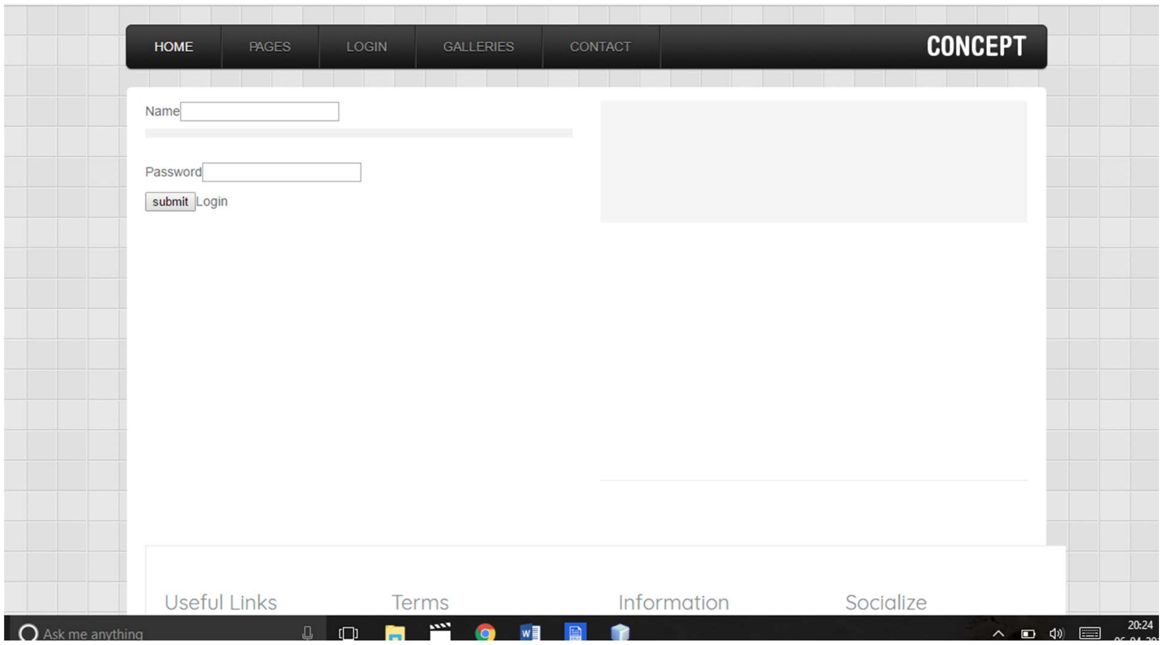
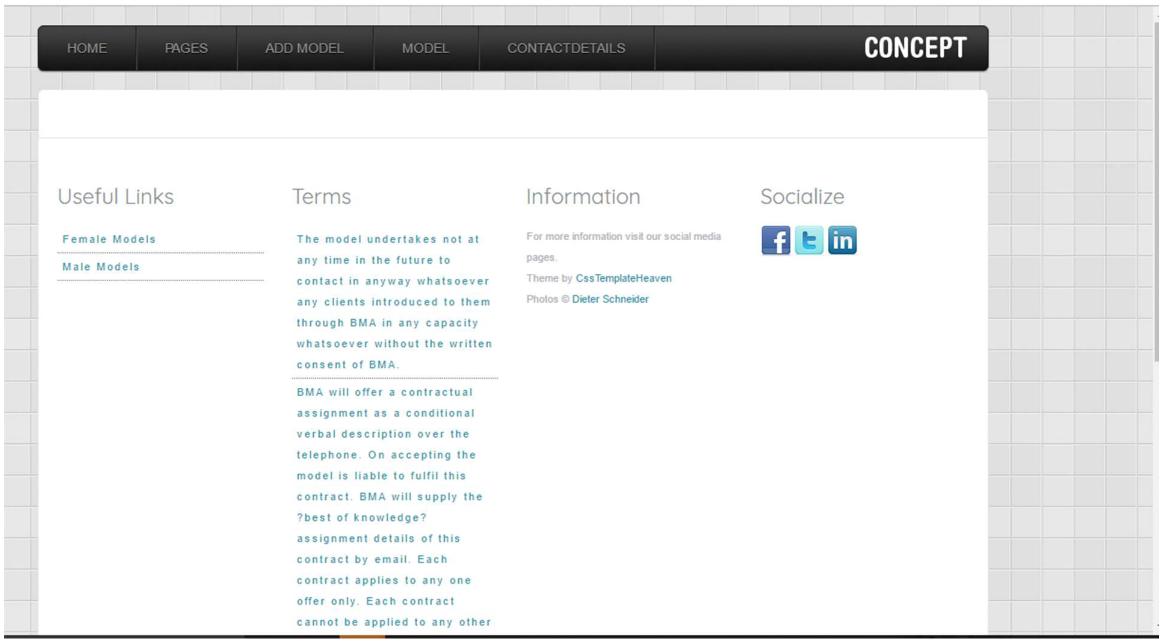


Fig 5.2: Login Activity

Main Activity:

The main activity window displays login window.Here main activity only consists of one module i.e. admin module.The admin login to perform various functions .



The model undertakes not at any time in the future to contact in anyway whatsoever any clients introduced to them through BMA in any capacity whatsoever without the written consent of BMA.

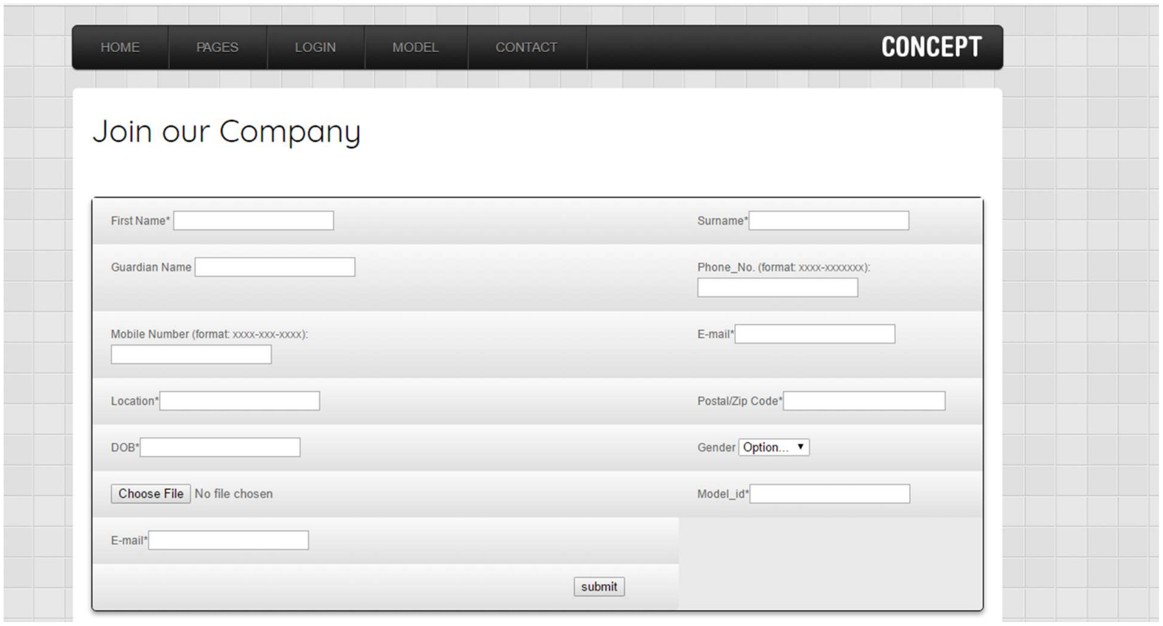
BMA will offer a contractual assignment as a conditional verbal description over the telephone. On accepting the model is liable to fulfil this contract. BMA will supply the ?best of knowledge? assignment details of this contract by email. Each contract applies to any one offer only. Each contract cannot be applied to any other

For more information visit our social media pages.

Theme by CssTemplateHeaven

Photos © Dieter Schneider

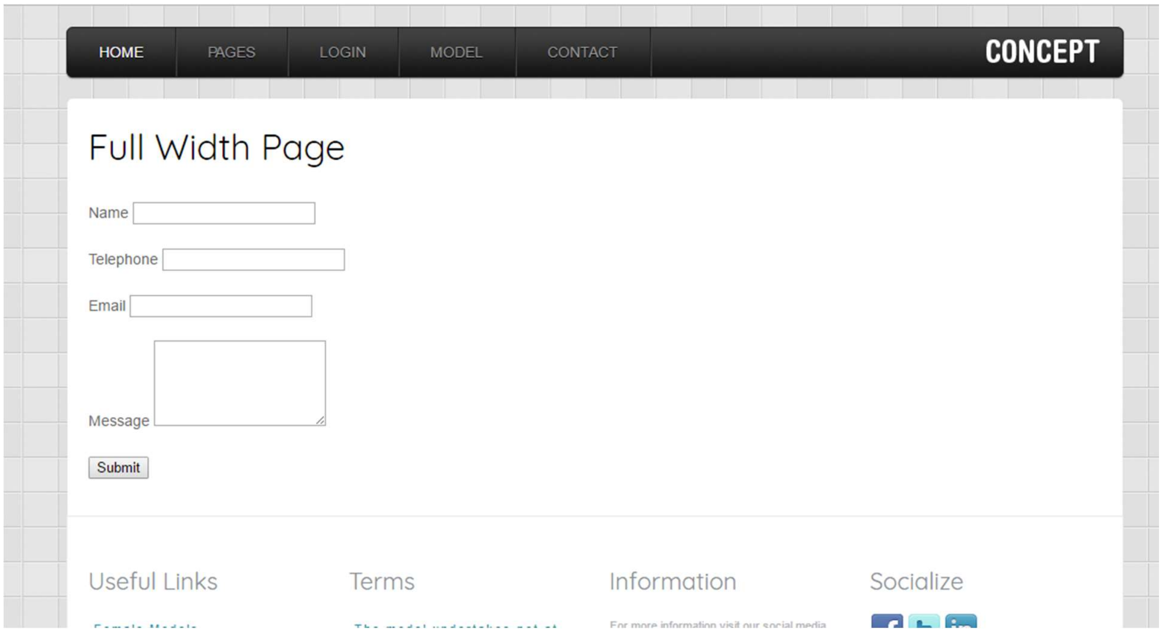
Fig 5.3: Main activity Screen (Admin)



Join our Company

First Name*	Surname*
Guardian Name	Phone_No. (format: xxxx-xxxxxxx):
Mobile Number (format: xxxx-xxx-xxxx):	E-mail*
Location*	Postal/Zip Code*
DOB*	Gender Option...
Choose File No file chosen	Model_id*
E-mail*	
submit	

Fig 5.4: Joining the agency



Full Width Page

Name

Telephone

Email

Message

Submit

Useful Links

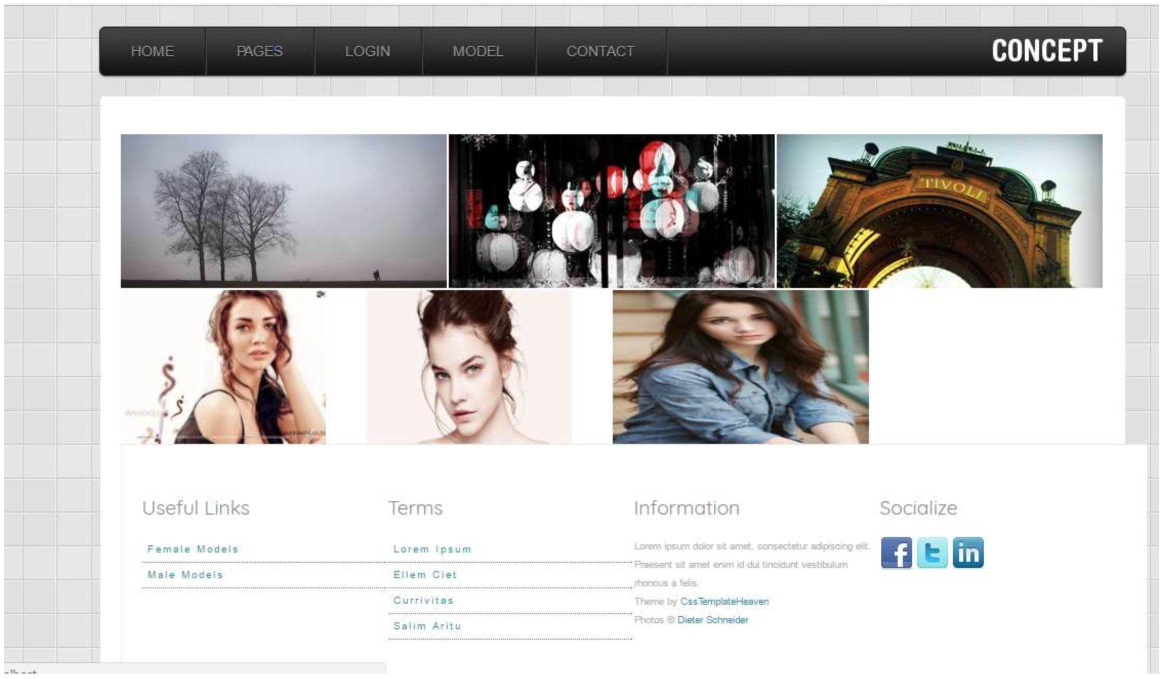
Terms

Information

Socialize

For more information visit our social media

Fig 5.5: Contact details



Useful Links

Female Models

Male Models

Terms

Lorem Ipsum

Ellem Ciet

Currivitas

Salim Aritu

Information

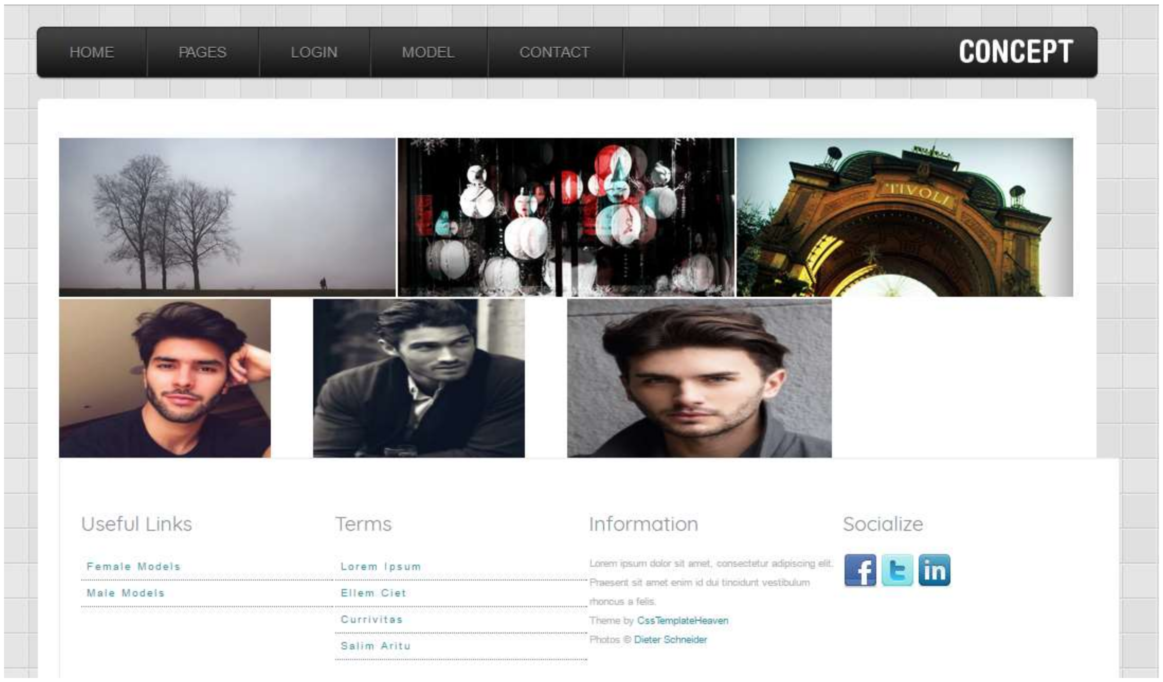
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent sit amet enim id dui tincidunt vestibulum rhoncus a felis.

Theme by CssTemplateHeaven

Photos © Dieter Schneider

Socialize

Fig 5.6: Female Models



Useful Links

Female Models

Male Models

Terms

Lorem Ipsum

Ellem Ciet

Currivitas

Salim Aritu

Information

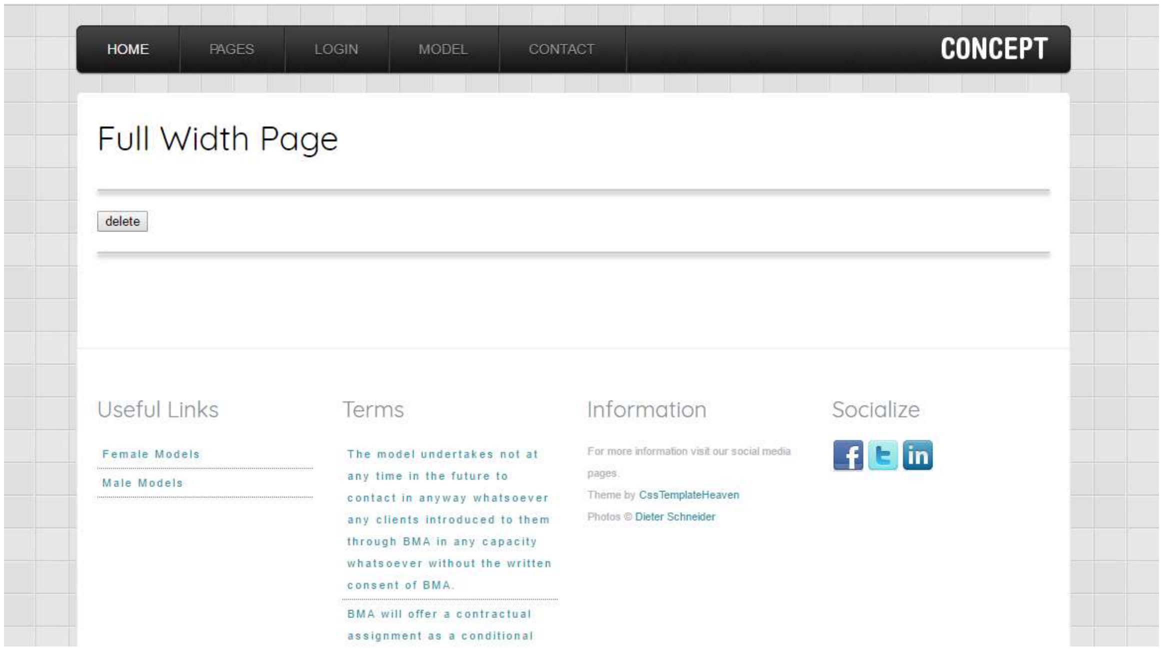
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent sit amet enim id dui tincidunt vestibulum rhoncus a felis.

Theme by CssTemplateHeaven

Photos © Dieter Schneider

Socialize

Fig 5.7: Male Models



Full Width Page

Useful Links

Female Models

Male Models

Terms

The model undertakes not at any time in the future to contact in anyway whatsoever any clients introduced to them through BMA in any capacity whatsoever without the written consent of BMA.

BMA will offer a contractual assignment as a conditional

Information

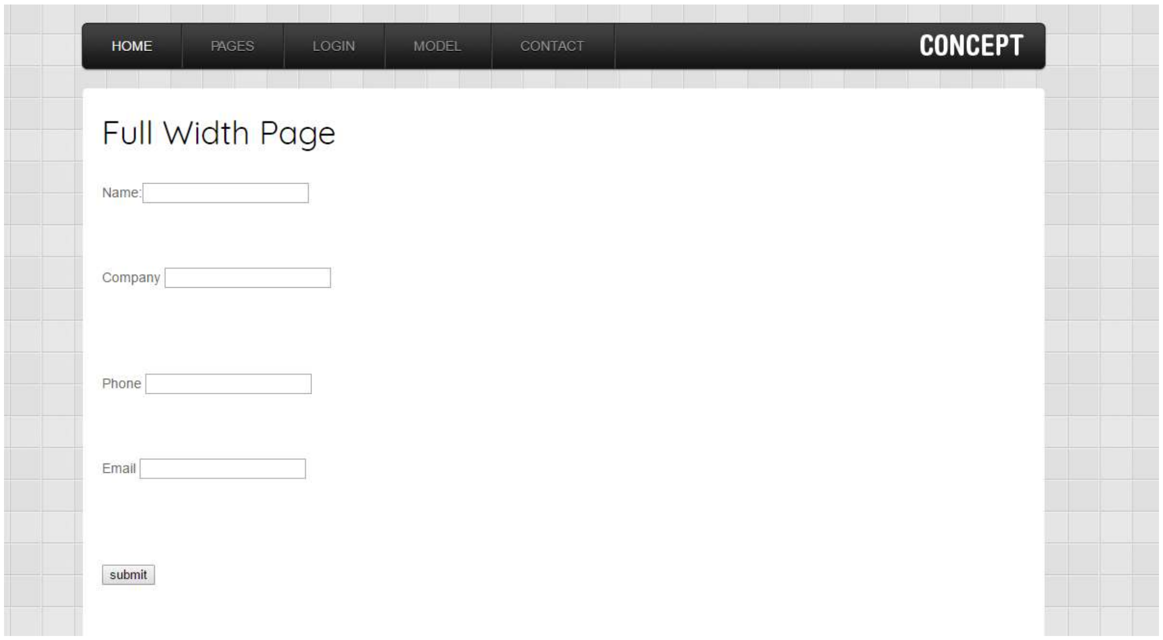
For more information visit our social media pages.

Theme by CssTemplateHeaven

Photos © Dieter Schneider

Socialize

Fig 5.8: Booked Models



Full Width Page

Name:

Company

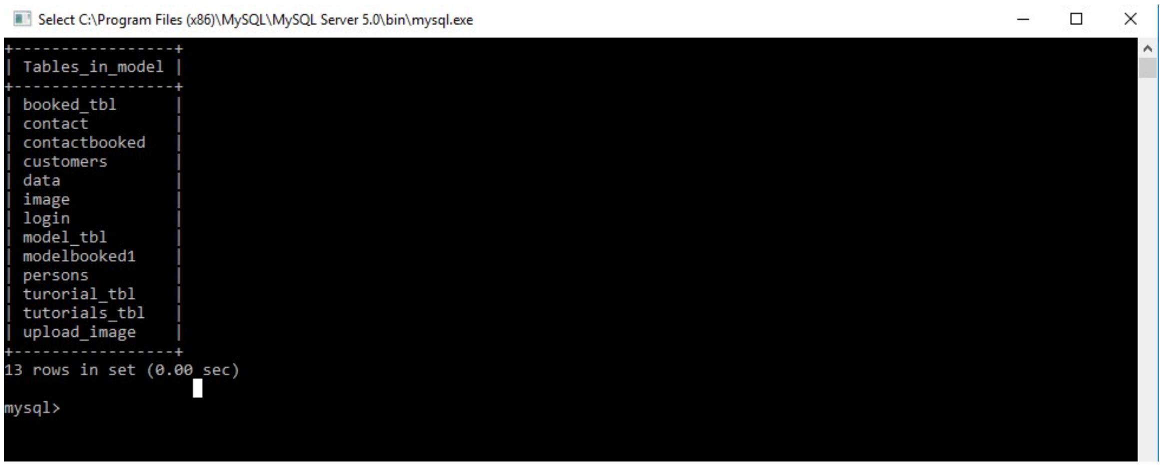
Phone

Email

Fig 5.9 Entering company Details

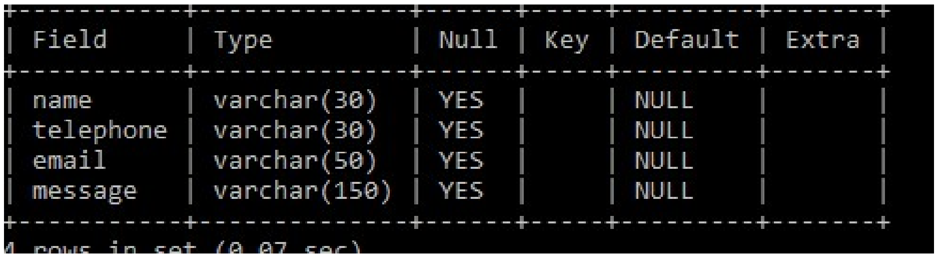
5.5 TABLE STRUCTURE

Figure5.10:ListofDatabaseTables



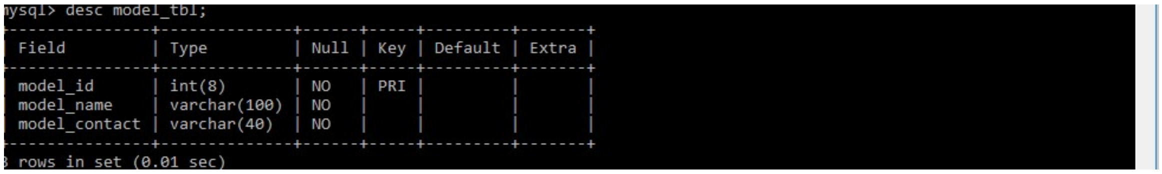
Select C:\Program Files (x86)\MySQL\MySQL Server 5.0\bin\mysql.exe

```
+----------------+
| Tables_in_model |
+----------------+
| booked_tbl     |
| contact        |
| contactbooked  |
| customers      |
| data           |
| image          |
| login          |
| model_tbl      |
| modelbooked1   |
| persons        |
| turorial_tbl   |
| tutorials_tbl  |
| upload_image   |
+----------------+
13 rows in set (0.00 sec)

mysql>
```



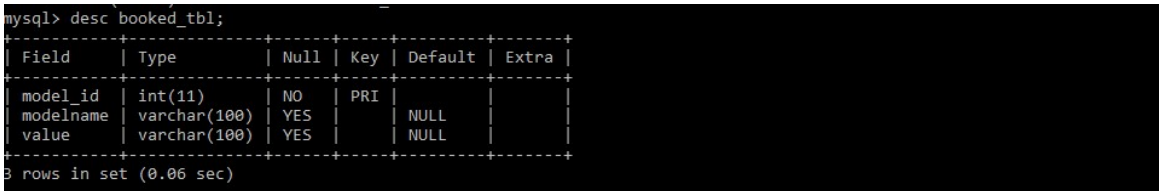
```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| name      | varchar(30)  | YES  |     | NULL    |       |
| telephone | varchar(30)  | YES  |     | NULL    |       |
| email     | varchar(50)  | YES  |     | NULL    |       |
| message   | varchar(150) | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
4 rows in set (0.07 sec)
```

Table: 5.11:Contact Database



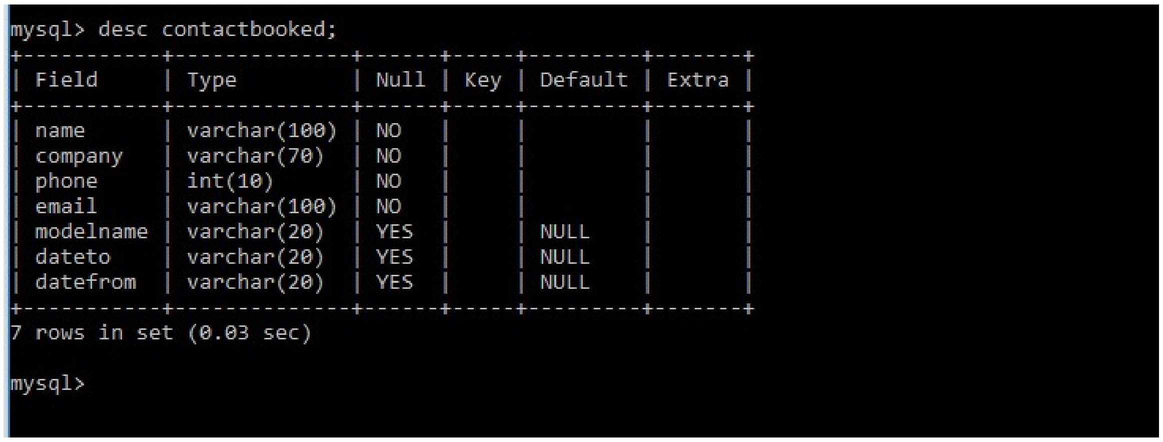
```
ysql> desc model_tbl;
+---------------+--------------+------+-----+---------+-------+
| Field         | Type         | Null | Key | Default | Extra |
+---------------+--------------+------+-----+---------+-------+
| model_id      | int(8)       | NO   | PRI |         |       |
| model_name    | varchar(100) | NO   |     |         |       |
| model_contact | varchar(40)  | NO   |     |         |       |
+---------------+--------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

Table: 5.12: Model Database



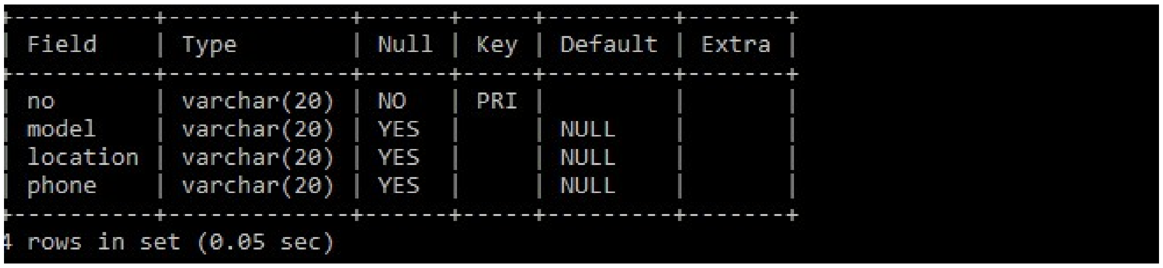
```
mysql> desc booked_tbl;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| model_id  | int(11)      | NO   | PRI |         |       |
| modelname | varchar(100) | YES  |     | NULL    |       |
| value     | varchar(100) | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
3 rows in set (0.06 sec)
```

Table: 5.13: Booked Model Database



```
mysql> desc contactbooked;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| name      | varchar(100) | NO   |     |         |       |
| company   | varchar(70)  | NO   |     |         |       |
| phone     | int(10)      | NO   |     |         |       |
| email     | varchar(100) | NO   |     |         |       |
| modelname | varchar(20)  | YES  |     | NULL    |       |
| dateto    | varchar(20)  | YES  |     | NULL    |       |
| datefrom  | varchar(20)  | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
7 rows in set (0.03 sec)

mysql>
```

Table 5.14 :Booking Model Database



```
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| no       | varchar(20) | NO   | PRI |         |       |
| model    | varchar(20) | YES  |     | NULL    |       |
| location | varchar(20) | YES  |     | NULL    |       |
| phone    | varchar(20) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.05 sec)
```

Table 5.15:Adding Model Database

References

1. Java Programming Language. n.p., n.d. <http://en.wikipedia.org/wiki/Java_%28programming_language%29>
2. http://www.bmamodels.com/
3. https://www.w3schools.com/sql/
4. http://www.itprojectz.com/itprojectz_new/java-projects-on-fashion-model-management-system/
5. Liang, Daniel. Introduction to Java Programming. Prentice Hall, 2013.
6. Schildt, Herbert. Java The Complete Reference. McGraw Hill, 2011.