1210311414

**INTERNSHIP PROJECT**

**BUSINESS WORKFLOW MAINTAINANCE APPLICATION**

**EVANA DINESH RAM**

**ABSTRACT**

**BUSINESS WORKFLOW MAINTAINANCE APPLICATION**- is the project developed for for chilukuri housing projects pvt. Limited .This project helps for the digitalization of the data required for the a housing project company to add the necessary data and to retrieve the data when needed within seconds rather than searching for data going through the files which may take very long time.

All the data such as phase management ,employee management ,the total expenditure incurred during the course of development of phase,coustomer management and getting the details of the financial structure –all of these could be maintained by using this application.This helps a lot when it comes to saving the time and making the details entry error-free.This is not any other kind of web application which could be accessed by many online,rather it is a application which could be accessed by only those who are authorized to access. This application is developed to be more interactive with the user even if he have minimal computer knowledge.

**INDEX**

1. INTRODUCTION
   1. INTRODUCTION TO PROJECT
   2. INTRODUCTION TO CHILUKURI BRUNDAVAN HOUSING PVT. LMTD.
2. SYSTEM ANALYSIS
   1. SYSTEM REQUIREMENT SPECIFICATION
      1. HARDWARE REQUIREMENTS
      2. SOFTWARE REQUIREMENTS
3. IMPLEMENTATION
   1. INTRODUCTION TO HTML
   2. INTRODUCTION TO JAVA
   3. INTRODUCTION TO JAVASCRIPT
   4. INTRODUCTION TO JDBC

3.5.DATA BASE MODELS

3.6MYSQL

3.7 NETBEANS

1. SYSTEM DESIGN
   1. INTRODUCTION
   2. USECASE DIAGRAM
   3. CLASS DIAGRAM
2. SYSTEM IMPLEMENTATION
3. OUTPUT SCREENS
4. DATABASE MODEL
5. TESTING
   1. INTRODUCTION
   2. TEST-CASES

**INTRODUCTION**

* 1. **INTRODUCTION TO PROJECT:**

The housing companies need to maintain large data such as sales of the flats,the data related to the dimensions of the plots,salaries paid to the employees of the company and their coustomer details ,maintainance fees and many more related.There has been days where number of files required to enter this data has grown up exponentially into large pile,especially when the development is done on large scale.In such case with the usage of this type of application saves lot of space and could be maintained easily with minimal labour,and data retrieval is far more easier when compared to those files maintainance.

The major benefit of this application project is maintaining all the records of data of the entire housing projects undertaken by the chilukuri housing pvt. Limited .these records could be accessed with in few seconds,whenever necessary by those who were authorized to.The security is added feature where only those who knows the correct user name and password could only accesss this application and maintain the data .

Another benefit is the efficiency with which it works and the code is developed which provides zero tolerance for errors.The source code is well tested to perform in the best way possible.

There is also a feature where all the expenditure incurred in all aspects in the development course of phase are calculated accordingly and can be exported to an excel sheet.The maintainance receipts and the coustomer receipts are generated and could be available for printing in an much readable format.

**1.2 INTRODUCTION TO CHILUKURI BRUNDAVAN HOUSING PVT. LIMITED**:

**Chilukuri Housing Projects Pvt. Ltd.**, a leading construction company in the city of Vizag. It is Started in the year 1997 dealing with Realestates and Constructions ,15 years with more than 10 projects in Construction of Residential and Commercial Apartments in 2,90,000 sft, and VUDA approved Residential Layouts nearly in 200 Acers. Chilukuri group with a Professional team at highly qualified architects civil engineers, Management pool and experienced work force who are skilled in concerned fields, All the Projects promoted by the firm so far have been highlysuccessful.  
  
The firm is currently carrying out extensive construction activity at various locations in and around the City. The company is proud of the reputation it has built for honesty and the quality of construction of its projects. Its management has set out a strict code of conduct to abide by in its dealings with customers.

**Chilukuri Housing Projects Pvt. Ltd.** has a well-deserved reputation for exemplary levels of service in real estate sector, gained through a commitment to the highest professional standards. Respected to our customers and competitors alike, we have grown to become one of the most admired full-service real estate company in the Coastal Region of Andhra Pradesh, India.

At **Chilukuri Housing Projects Pvt. Ltd.**, uncompromising facilities is our standard. We provide our representatives with the best support and most productive working environments necessary to offer you first-class service.

**Chilukuri Housing Projects Pvt. Ltd.** is premier Realtors having 15 years of market stand with impressive track record. The company is floated by a team of young professionals with vast experience in real estate and Construction field. With ever-growing clientele and best wishes from thousands of its satisfied customers around the world. We strongly support the belief that a long-term relationship is what sustains a successful business. Constantly reinforcing our customers faith in us, is what we always strive for. Our longstanding relationship with our customers is further enhanced by our transparent and honest approach. Unlike other real estate companies, Chilukuri Group was designed to reward agents for working together, to serve our customers better.

* 1. **SYSTEM REQUIREMENT SPECIFICATION:**

**2.1.1Hardware Requirements:**

Processor : Pentium IV

Hard Disk : 40GB

RAM : 512MB or more

**2.1.2 Software Requirements:**

Operating System : Windows XP/7/8 or Linux

Programming Language : Java

Web Applications : JDBC, Servlets, JSP

IDE/Workbench : Netbeans 7.1.2

Database : mysql

Server Deployment : Tomcat 5.x

**IMPLEMENTATION**

3.1. INTRODUCTION TO HTML

The hypertext markup language (HTML) is a simple markup language. Used to create a hypertext documents that are portable from one platform to another HTML documents are SGML (Standard generalized mark up language) documents with generic semantics that are appropriate for representing information from a wide range of applications. This specification defines HTML version 3.2.HTML 3.2 aims to capture recommended practice as of early ’96 and as such a replacement for HTML2.0 (RFC 1866).

A set of instructions embedded in a document is called mark up language. These instructions describe what the document text means and hoe it should look like in a display. Hyper Text Mark Up language (HTML) is the language used to encode World Wide Web documents.

**WHY TO USE HTML:**

Website is a collection of pages, publications, and documents that reside on web server. While these pages publications and a document as a formatted in a single format, you should use HTML for home page and all primary pages and the site. This will enable the millions of web users can easily access and to take advantage of your website.

HTML is considered first for formatting any new material you plan to publish on the web. HTML documents are platform independent, meaning that they don’t confirm to any standard. If they are created properly you can move home page to any server platform or you can access them with any complaint www browser.

**STRUCTURE OF HTML:**

HTML elements perform a defined task. HTML uses two types of elements

* Empty Tags
* Container Tags

Empty (or open) tags and container tags. These tags differ because of what they represent. Empty tags represent formatting constricts such as line breaks and horizontal rules. Container tags define a section of text and specify the formatting the containers dot all of the selected text. A container tag has both a beginning and an ending.

**HTML LAYOUT:**

An HTML document consists of text, which comprises the content of the document and tags, which, defines the structure, and appearance of the document. The structure of an HTML document is simple, consists of outer.

<HTML>tag enclosing the document header and body

<HTML>

<HEAD>

<TITLE>the title of HTML document</TITLE>

</HEAD>

<BODY>

This is where the actual HTML documents

Text lies, which is displayed in the browser

</BODY>

</HTML>

Each document has a head and body delimited by the <HEAD> and <BODY> tag. The head is where you give your HTML document a title and where you indicate other parameters the browser may use when displaying the document. This includes the text for displaying the text. Tag also references special and indicates the hot spots that link your document to other documents.

**HTML FORMS:**

Create a form usually involves two independent steps: create the layout for the form itself and then writing a script program on the server side to process the formation you get back from a form.

To create a form. You use the <FORM> tag. Inside the opening and closing FORM tags are each of the individual form elements plus any other HTML content to create a layout for that form.

The opening tag of the FORM element usually includes the attributes: METHOD and ACTION. The METHOD attributes can be either GET or POST which determines how your form data is sent to the script to process it.

The ACTION attribute is a pointer to the script that processes the form on the server side. The ACTION can be included by a relative path or by a full URL to a script on your server or somewhere else. For example, the following <FORM> tag would call a script called form-name in cgi-bin directory on server www.myservser.com

<FORM Method= post action=” ”>

……………….

</FORM>

**METHOD ATTRIBUTE:**

The other required attribute for the <form> tag sets the methods by which the browser form’s data to the server for processing. There are two ways: the POST method and GET method. With POST method, the browser sends the data in two steps: the browser first contacts the form-processing server specified in the action attributes, and once contact is made, sends the data to the in to a separate transmission.

The GET method in the other hand, contacts the form processing server and sends the form data in a single transaction step: the browser appends the data to the form’s action URL, separated by the question mark (?) character.

3.2. INTRODUCTION TO JAVA

CREATION OF JAVA:

**J**ava was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan at SUN Microsystems Incorporation in the year 1991.It took 18 months to develop the 1st working version. This language was initially called “OAK”, but was renamed “JAVA” in 1995, many more contributed to the design and evolution of the language.

**JAVA OVERVIEW:**

**J**ava is a powerful but lean object-oriented programming language. It has generated a lot of excitement because it makes it possible to program for Internet by creating Applets. Programs that can be embedded in web page. The context of an applet can be an animation with sound, an interactive game or a ticker tape. With constantly updated stock prices. Applets can be just little decorations to liven up web page, or they can be serious applications like Word processor or Spreadsheet.

But Java is more than a programming language for writing Applets. It is being used more and more for writing standalone applications as well. It is becoming so popular that many people believe it will become standard language for both general purpose and Internet programming.

There are many buzzwords associated with Java, but because of its spectacular growth in popularity, a new buzzword has appeared ubiquitous. Indeed, all indications are that it will soon be everywhere.

Java builds on the strength of C++. It has taken the best features of C++ and discarded the more problematic and error prone parts. To this lean core, it has added garbage collection (automatic memory management), multithreading (the capacity for one program to do more than one thing at a time), security capabilities. This result is that Java is simple, elegant, and powerful and easy-to-use.

Java is actually a platform consisting of 3 components:

1. Java Programming Language.
2. Java Library of Classes and Interfaces.
3. Java Virtual Machine

The following sections will say more about these components.

**JAVA IS PORTABLE:**

One of the biggest advantages Java offers is that it is portable. An application written in Java will run on all the major platforms. Any computer with a Java-based browser can run the applications or Applets written in the Java-Programming-Language. A programmer no longer has to write one program to run on a Macintosh, another program to run on a Windows-machine still another to run on a UNIX-machine and so on. In other words, with Java developers write their programs only once.

The Virtual Machine is what gives Java is cross platform capabilities. Rather being compiled into machine language, which is different for each OS’s and computer architecture, Java code is compiled into Byte codes.

With other languages, the program code is compiled into a language that the computer can understand. The problem is that other computers with different machine instruction set cannot understand that language. Java code on the other hand is compiled into Byte-Code rather than a machine language. These byte codes go to the JVM, which executes them directly or translates them into the language that is understood by the machine running it.

In summary, these means that with the JDBC API extending Java, a programmer writing Java code can access all the major RDBMS on any platform that supports the JVM.

**JAVA IS OBJECT-ORIENTED:**

# The Java programming language is OBJECT-ORIENTED, which makes program design focus on what you are dealing with, rather than on how your are

# going to do something. This makes it more useful for programming in sophisticated projects, because one can break the things into understandable components. A big benefit is that these components can then be reused.

Object-Oriented Languages use the paradigm of classes. In simplest term, a class includes both the data and the functions to operate on data. You can create an instance of a class, also called an object, which will have all the data members and functionality of its class. Because of this, you can think of a class as being like template, with each object being a specific instance of a particular type of class.

The class paradigm allows one to encapsulate data so that specific data values are those using the data cannot see the function implementation. Encapsulation makes it possible to make the changes in code without breaking other programs that use that code.

If for example, the implementation of a function is changed, the change is invisible to any programmer who invokes that function, and does not affect his/her program, except hopefully to improve it.

Java includes inheritance, or the ability to derive new classes from existing classes. The derived class, is also called as Sub-Class, inherits all the data in the functions of the existing class.

**JAVA DEVOLPMENT EVNIRONMENT:**

To code, edit, debug and test the java programs, one needs to have a java development environment. At the minimum this will consists of a java compiler interpreter and applet viewer where applets can be tested.

Sun’s java development kit (JDK) latest version is 2.2 can be freely downloaded from the Internet.

Java compiler is available on DOS, Win95, WIN’NT, Solaris and MAC etc.

3.3. INTRODUCTION TO JAVASCRIPT

**JAVASCRIPT:**

JavaScript is a new scripting language for WebPages scripts written with java script can be embedded into your HTML pages. With java script you have very many possibilities for enhancing your HTML page with interesting elements. For example you are able to respond to user-initiated events quite easily. Some effects that are now possible with java script were some time ago only possible with CGI. So you can create really sophisticated pages with the help of java script script on the Internet.

**DIFFERENCE BETWEEN JAVA AND JAVA SCRIPT:**

Although the names are almost the same Java is not the same as Java Script. These are two different techniques for Internet programming. Java is programming language. JavaScript is a scripting language as the name implies. The difference is that we can create real programs with java. But about real programming. So Java Script is meant to be easy to understand and easy to use. JavaScript authors should not have to care too much about programming. We could say that Java Script is rather an extension to HTML than a separate computer language. Of course this is not the official definition but it makes it easier to understand the difference between java and java script.

**HOW CAN JAVA SCRIPT SCRIPTS RUN?**

The first browser to support java script was the Netscape Navigator 2.0 of course the higher versions do have java script as well. You might know that java does not run on all Netscape Navigators 2.0 (or higher versions) versions. But this is not true for java script -although there are some problems with the different versions.

The Mac version for example seems to have many bugs. In the near future there are going to be some other browsers, which support java script. The Microsoft Internet explorer 3.0 is going to support java script. JavaScript enabled browsers are going to spread soon - it is worth learning this new technique now. You might realize that is really easy to write Java Script scripts. We have to know is some basic techniques and some work-around for problems you might encounter. Of course we need a basic. Understanding HTML before reading this tutorial you can find many really good online resources about HTML. Best you make an online search about ‘html’ at yahoo if you want to get informed about HTML. Now I want to show some small script s so you can learn how they are implemented into HTML-documents and to show which possibilities you have with the new scripting language. The following is a very small script, which will only print a text into an HTML document.

<html>

<head>

My first JavaScript

</head>

<body><br>

This is a normal HTML document

<br>

<script language=”JavaScript”>

Document.write (“this is a java script”)

</script><b r>

Backing HTML again

</body>

</html>

If you are using a java script enabled-browser at the moment then you will have the possibility to see this script working. If your browser doesn’t support Java Script then this output might be some kind of strange…

This is a normal HTML document

This is java script!

Back in HTML again.

**FUNCTIONS:**

Functions are bet declared between the <Head> tag of HTML page. Functions are called by user-initiated events. Seems reasonable to keep the functions between the <Head> tags. They are loaded first before a user can do anything that might call a function. Scripts can be placed between inside comment fields to ensure that older browser do not display the script itself.

<html>

<head>

<script language=”JavaScript”>

function pushbutton (){

alert (“Hello!”);

}

</script>

</head>

<body>

<form>

<input type=”button” name=”Button1” value=”push me” onclick=”pushbutton ()”>

</form>

</body>

</html>

If we want to test this one immediately and you are using a Java Script enabled browser then please go ahead and push the button.

This script will create a button and when you press it a window will pop up saying “hello!”. In fact we have a lot of possibilities just by adding functions to our scripts.

The common browsers transmit the form information by either method: here’s the complete tag including the GET transmission method attribute for the previous form

Example

<Form method =GET action=http://www.mycompany.com/cgi-bin/upfdate.pl>

………

</form>

**INPUT ELEMENTS:**

Use the <input> tag to define any one of a number of common form elements including text fields multiple choice lists click able images and submission buttons. There are many attributers for this tag only that types and name attributes are required for each element, each type of input element uses only a subset of the followed attributes. Additional <input> attributes may be required based upon which type of the form element you specify.

**SUBMIT BUTTON:**

The submit button (<input type=submit> ) does what its name implies, settings in motion the form’s submission to the server from the browser. We many have more than submit buttons will be added to the parameter list the browser sends along to the server.

Example

< Input type =”submit”>

<Input type=”submit” value=”submit” name=”name”>

**RESET BUTTON:**

The reset button if firm <input> button is nearly self- explanatory; it lets the user reset erase or set to some default value all elements in the form. By default the browser displays a reset button worth the label “reset”. We can change that by specifying a value attribute with tour own button label.

## **3.4. INTRODUCTION TO JDBC**

JDBC (Java Database connectivity) is a front-end tool for connecting to a server to ODBC in that respect, however JDBC can connect only java client and it uses ODBC for the connectivity. JDBC is essentially a low-level API since any data manipulation, storage and retrieval has to be done by the program itself. Some tools, which provide a higher-level abstraction, are expected shortly.

The next question that needs to be answered is why we need JDBC, once we have ODBC on hand. We can use the same ODBC to connect the entire database and ODBC is a proven technology.

Problem for doing this is ODBC gives a ‘c’ language API, which uses pointers extensively. Since java does not have any pointes and is object-oriented sun Microsystems, inventor of java developed to suit its needs.

**REQUIREMENTS TO USE JDBC:**

To use JDBC you need a basic knowledge of databases and SQL.A part from this you need the jdk1.1 (Java Development Kit1.1 available javasoft’s website) or a version of Java since jdk1.1 and above come bundled with JDBC software.

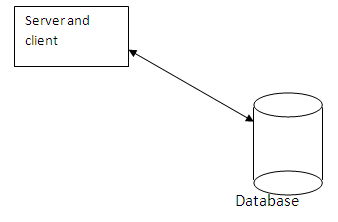
After that you need to have a back-end database engine for which a JDBC driver is available. When JDBC drivers are not available JDBC-ODBC bridge drivers are used to access the database through ODBC.Back-end is not needed when JDBC driver is capable of storing and retrieving the data itself, or if JDBC-ODBC Bridge and the ODBC driver can be used to store and retrieve the information.

# 3.5.DATA BASE MODELS

JDBC and accessing the database through applets and JDBC.API via an intermediate server resulted server resulted in a new type of database model which is different from the client-server model. Based on number of intermediate server through the request should go it is named as single tire, two tire and multi tire architecture

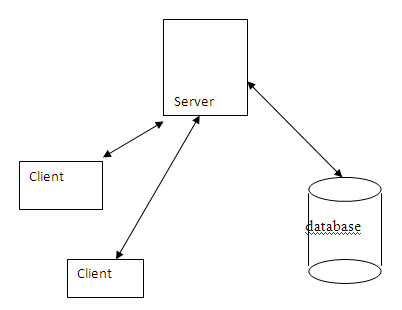
**SINGLE TIER:**

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



**TWO TIER (CLIENT-SERVER):**

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



**THREE TIER AND N-TIER:**

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

**JDBC-ODBC BRIDGE PLUS ODBC DRIVER**

The java soft bridge product provides JDBC access via ODBC drivers. Note that ODBC binary code end in many cases database client code must be loaded on each client machine that uses this driver. As a result, this kind of driver is most appropriate on a corporate network where client installations are not major problem, or for application server code written in java in a 3-tier architecture.

**3.6 MYSQL:**the world's second most widely used open-source relational database management system (RDBMS). It is named after co-founder Michael Widenius's daughter, My. The SQL phrase stands for Structured Query Language.

The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack (and other 'AMP' stacks). LAMP is an acronym for "Linux, Apache, MySQL, [Perl](http://en.wikipedia.org/wiki/Perl)/[PHP](http://en.wikipedia.org/wiki/PHP)/[Python](http://en.wikipedia.org/wiki/Python_(programming_language))." Free-software-open source projects that require a full-featured database management system often use MySQL.

For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, [MODx](http://en.wikipedia.org/wiki/MODx), [Joomla](http://en.wikipedia.org/wiki/Joomla), [WordPress](http://en.wikipedia.org/wiki/WordPress), [phpBB](http://en.wikipedia.org/wiki/PhpBB), [MyBB](http://en.wikipedia.org/wiki/MyBB), [Drupal](http://en.wikipedia.org/wiki/Drupal) and other software. MySQL is also used in many high-profile, large-scale [websites](http://en.wikipedia.org/wiki/Website), including [Google](http://en.wikipedia.org/wiki/Google) (though not for searches), [Facebook](http://en.wikipedia.org/wiki/Facebook), [Twitter](http://en.wikipedia.org/wiki/Twitter), [Flickr](http://en.wikipedia.org/wiki/Flickr) and [YouTube](http://en.wikipedia.org/wiki/YouTube).

MySQL is a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS), and ships with no [GUI](http://en.wikipedia.org/wiki/Graphical_user_interface) tools to administer MySQL databases or manage data contained within the databases. Users may use the included [command line](http://en.wikipedia.org/wiki/Command_line) tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data record. The official set of MySQL front-end tools, [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is actively developed by Oracle, and is freely available for use

The official [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is a free integrated environment developed by MySQL AB, that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, [MySQL GUI Tools](http://en.wikipedia.org/wiki/MySQL_GUI_Tools). Similar to other third-party packages, but still considered the authoritative MySQL front end, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).

MySQL Workbench is available in two editions, the regular [free and open source](http://en.wikipedia.org/wiki/Free_and_open_source_software) *Community Edition* which may be downloaded from the MySQL website, and the proprietary *Standard Edition* which extends and improves the feature set of the Community Edition.

Third-party proprietary and free graphical administration applications (or "front ends") are available that integrate with MySQL and enable users to work with database structure and data visually. Some well-known front ends, in alphabetical order, are:

* [Adminer](http://en.wikipedia.org/wiki/Adminer) – a free MySQL front end written in one [PHP](http://en.wikipedia.org/wiki/PHP) script, capable of managing multiple databases, with many [CSS](http://en.wikipedia.org/wiki/Cascading_Style_Sheets) skins available.
* [DBEdit](http://en.wikipedia.org/wiki/DBEdit) – a free front end for MySQL and other databases.
* [HeidiSQL](http://en.wikipedia.org/wiki/HeidiSQL) – a full featured free front end that runs on [Windows](http://en.wikipedia.org/wiki/Windows), and can connect to local or remote MySQL servers to manage databases, tables, column structure, and individual data records. Also supports specialised GUI features for date/time fields and enumerated multiple-value fields.
* [LibreOffice Base](http://en.wikipedia.org/wiki/LibreOffice) – LibreOffice Base allows the creation and management of databases, preparation of forms and reports that provide end users easy access to data. Like [Microsoft Access](http://en.wikipedia.org/wiki/Microsoft_Access), it can be used as a front-end for various database systems, including Access databases (JET), ODBC data sources, and MySQL or PostgreSQL]
* [Navicat](http://en.wikipedia.org/wiki/Navicat) – a series of proprietary graphical database management applications, developed for Windows, Macintosh and Linux.
* [OpenOffice.org](http://en.wikipedia.org/wiki/OpenOffice.org) – OpenOffice.org Base can manage MySQL databases if the entire suite is installed. Free and open-source.
* [phpMyAdmin](http://en.wikipedia.org/wiki/PhpMyAdmin) – a free Web-based front end widely installed] by [web hosts](http://en.wikipedia.org/wiki/Web_hosting_service), since it is developed in PHP and is included in the LAMP stack, [MAMP](http://en.wikipedia.org/wiki/MAMP), [XAMPP](http://en.wikipedia.org/wiki/XAMPP) and [WAMP](http://en.wikipedia.org/wiki/WAMP_(software_bundle)) software bundle installers.
* [Webmin](http://en.wikipedia.org/wiki/Webmin) – a free Web-based management utility and a MySQL front end, developed in Perl with some parts written in Java.
* [SQLBuddy](http://en.wikipedia.org/wiki/SQLBuddy) – a free Web-based front end, developed in PHP.
* [SQLyog](http://en.wikipedia.org/wiki/SQLyog) – commercial, but there is also a free 'community' edition available.
* [Toad for MySQL](http://en.wikipedia.org/wiki/TOAD_(software)) – a free development and administration front end for MySQL from [Quest Software](http://en.wikipedia.org/wiki/Quest_Software)
* [Chive](http://en.wikipedia.org/wiki/Chive) – is a free, open source, web-based database management tool designed as an alternative to [phpMyAdmin](http://en.wikipedia.org/wiki/PhpMyAdmin).

Other available proprietary MySQL front ends include [dbForge Studio for MySQL](http://en.wikipedia.org/w/index.php?title=DbForge_Studio_for_MySQL&action=edit&redlink=1), DBStudio, [Epictetus](http://en.wikipedia.org/wiki/Epictetus_Database_Client), [Microsoft Access](http://en.wikipedia.org/wiki/Microsoft_Access), [Oracle SQL Developer](http://en.wikipedia.org/wiki/Oracle_SQL_Developer), SchemaBank, [SQLPro SQL Client](http://en.wikipedia.org/wiki/SQLPro_SQL_Client), [Toad Data Modeler](http://en.wikipedia.org/wiki/Toad_Data_Modeler) and [DaDaBIK](http://en.wikipedia.org/wiki/DaDaBIK).

**3.7 NETBEANS IDE:**

**NetBeans** is an [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for developing primarily with [Java](http://en.wikipedia.org/wiki/Java_(programming_language)), but also with other languages, in particular PHP, C,C++, and [HTML5](http://en.wikipedia.org/wiki/HTML5).] It is also an [application platform](http://en.wikipedia.org/wiki/Platform_(computing)) framework for Java desktop applications and others.

The NetBeans IDE is written in Java and can run on Windows, OS X, Linux, Solaris and other platforms supporting a compatible [JVM](http://en.wikipedia.org/wiki/Java_Virtual_Machine).

The NetBeans Platform allows applications to be developed from a set of modular [software components](http://en.wikipedia.org/wiki/Software_component) called *modules*. Applications based on the NetBeans Platform (including the NetBeans IDE itself) can be extended by [third party developers](http://en.wikipedia.org/wiki/Third_party_developer).]

The NetBeans Team actively support the product and seek feature suggestions from the wider community. Every release is preceded by a time for Community testing and feedback

**HISTORY:**

NetBeans began in 1996 as Xelfi , a Java IDE student project under the guidance of the Faculty of Mathematics and Physics at [Charles University](http://en.wikipedia.org/wiki/Charles_University) in [Prague](http://en.wikipedia.org/wiki/Prague). In 1997 Roman Staněk formed a company around the project and produced commercial versions of the NetBeans IDE until it was bought by [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) in 1999. Sun open-sourced the NetBeans IDE in June of the following year. Since then, the NetBeans community has continued to grow.] In 2010, Sun (and thus NetBeans) [was acquired](http://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle) by [Oracle](http://en.wikipedia.org/wiki/Oracle_Corporation).

**VERSIONS RELEASED:**

NetBeans IDE 6.0 introduced support for developing IDE modules and rich client applications based on the NetBeans platform, a Java Swing [GUI](http://en.wikipedia.org/wiki/GUI) builder (formerly known as "Project Matisse"), improved [CVS](http://en.wikipedia.org/wiki/Concurrent_Versions_System) support, [WebLogic](http://en.wikipedia.org/wiki/WebLogic) 9 and [JBoss](http://en.wikipedia.org/wiki/JBoss) 4 support, and many editor enhancements. NetBeans 6 is available in official repositories of major Linux distributions.

NetBeans IDE 6.5, released in November 2008, extended the existing [Java EE](http://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) features (including Java Persistence support, EJB 3 and JAX-WS). Additionally, the NetBeans Enterprise Pack supports development of Java EE 5 enterprise applications, including [SOA](http://en.wikipedia.org/wiki/Service-oriented_architecture) visual design tools, XML schema tools, web services orchestration (for BPEL), and [UML](http://en.wikipedia.org/wiki/Unified_Modeling_Language) modeling. The NetBeans IDE Bundle for C/C++ supports C/C++ and FORTRAN development.

NetBeans IDE 6.8 is the first IDE to provide complete support of Java EE 6 and the [GlassFish Enterprise Server v3](http://en.wikipedia.org/wiki/GlassFish). Developers hosting their open-source projects on [kenai.com](http://en.wikipedia.org/wiki/Project_Kenai) additionally benefit from instant messaging and issue tracking integration and navigation right in the IDE, support for web application development with PHP 5.3 and the Symfony framework, and improved code completion, layouting, hints and navigation in JavaFX projects.

NetBeans IDE 6.9, released in June 2010, added support for [OSGi](http://en.wikipedia.org/wiki/OSGi), [Spring Framework](http://en.wikipedia.org/wiki/Spring_Framework) 3.0, Java EE dependency injection (JSR-299), [Zend Framework](http://en.wikipedia.org/wiki/Zend_Framework) for [PHP](http://en.wikipedia.org/wiki/PHP), and easier code navigation (such as "Is Overridden/Implemented" annotations), formatting, hints, and refactoring across several languages.

NetBeans IDE 7.0 was released in April 2011. On August 1, 2011, the NetBeans Team released NetBeans IDE 7.0.1, which has full support for the official release of the [Java SE 7](http://en.wikipedia.org/wiki/Java_version_history#Java_SE_7_.28July_28.2C_2011.29) platform.

NetBeans IDE 7.3 was released in February 2013 which added support for HTML5 and web technologies.]

NetBeans IDE 7.4 was released on October 15, 2013.

NetBeans IDE 8.0 was released on March 18, 2014.

### NetBeans IDE Bundle for Web and Java EE:

The **NetBeans IDE Bundle for Web & Java EE**] provides complete tools for all the latest Java EE 6 standards, including the new Java EE 6 Web Profile, Enterprise Java Beans (EJBs), servlets, Java Persistence API, web services, and annotations. NetBeans also supports the JSF 2.0 (Facelets), JavaServer Pages (JSP), Hibernate, Spring, and Struts frameworks, and the Java EE 5 and J2EE 1.4 platforms. It includes [GlassFish](http://en.wikipedia.org/wiki/GlassFish) and [Apache Tomcat](http://en.wikipedia.org/wiki/Apache_Tomcat). Some of its features with javaEE includes

* Improved support for CDI, REST services and Java Persistence
* New support for Bean Validation
* Support for JSF component libraries, including bundled PrimeFaces library
* Improved editing for Expression Language in JSF, including code completion, refactoring and hints

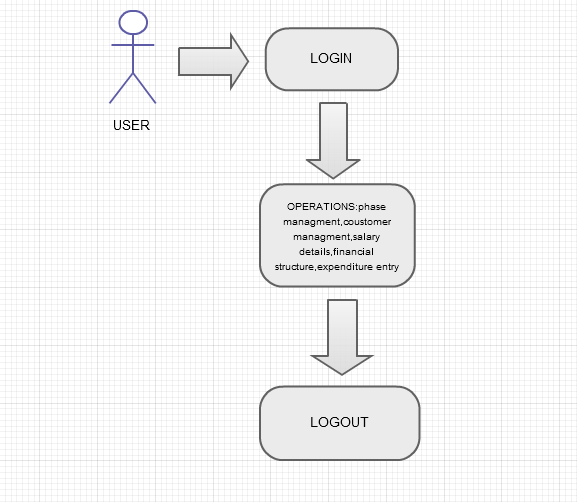
**SystemDesign**

**4.1 Introduction:**

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development[3], then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user. Until the 1990s systems design had a crucial and respected role in the data processing industry. In the 1990s standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering.Object-oriented analysis and design methods are becoming the most widely used methods for computer system design [citation needed]. The UML has become the standard language used in Object-oriented analysis and design [citation needed]. It is widely used for modeling software systems and is increasingly used for high designing non-software systems and organizations

**4.2 Use Case Diagram:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.



**4.3 CLASS DIAGRAM:**

**Class:** A Class is a description for a set of objects that shares the same attributes, and has similar operations, relationships, behaviors and semantics.

**Generalization:** Generalization is a relationship between a general element and a more specific kind of that element. It means that the more specific element can be used whenever the general element appears. This relation is also known as specialization

or inheritance link.

**Realization:** Realization is the relationship between a specialization and its implementation. It is an indication of the inheritance of behavior without the inheritance of structure.

**Association:** Association is represented by drawing a line between classes. Associations represent structural relationships between classes and can be named to facilitate model under standing. If two classes are associated, you can navigate from an object of one class to an object of the class.

**Aggregation:** Aggregation is a special kind of association in which one class represents as the larger class that consists of a smaller class. It has the meaning of “has-a” relationship.

**5.0 SYSTEM IMPLEMENTATION**

**Header.html:**

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Heading</title>

<link href="../styles.css" rel="stylesheet" />

</head>

<body>

<div class="topPanel">

<div style="float: left">

<img src="Logo\_51.jpg" height="60" />

</div>

<div

style="float: right; margin-top: 10px; font-weight: bold; font-size: 14px; color: navy; letter-spacing: .5px; margin-right: 5px;">

BUSINESS WORKFLOW MAINTENANCE APPLICATION- CHILUKURI HOUSTING

PROJECTS PVT LTD.&nbsp; &nbsp;</div>

</div>

</body>

</html>

Login.html:

<%@page import="com.lws.beans.LoginBean"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Login Page</title>

<link href="../styles.css" rel="stylesheet"/>

<style>

body {

font-family: Arial, Verdana, sans-serif;

font-size: 11px !important;

}

</style>

</head>

<body style="background-color: white">

<form method="post">

<div class="centeredPanel">

<img src="Logo\_51.jpg"/>

<br/>

<h3>Business Work-flow Management Application- Sign In.</h3>

<table cellpadding="5" cellspacing="5" style="border-collapse: collapse;border: 1px solid #eee; padding: 5px; border-radius: 3px; -moz-border-radius: 3px; -webkit-border-radius: 3px;">

<tbody style="background-color:#eee ">

<tr>

<td><label for="loginid">

Login Id: </label></td>

<td><input id="loginid" name="user" type="text" /></td>

</tr>

<tr>

<td><label for="pwd">

Password : </label></td>

<td><input id="pwd" name="password" type="password"/></td>

</tr>

<tr><td>&nbsp;</td>

<td> <input type="submit" value="Login"/></td>

</tr>

<tr style="background-color:white">

<td colspan="2">

Powered By Living Word Solutions.. Copyright &copy; All rights reserved.

</td>

</tr>

<%

try {

String uname = request.getParameter("user");

String pwd = request.getParameter("password");

if(uname != null && pwd != null) {

if(LoginBean.loginCheck(uname, pwd)) {

session.setAttribute("uname", uname);

response.sendRedirect("../phase.jsp");

} else {

%>

<tfoot>

<tr>

<td colspan="2" style="color:red">

Login Failed.. Please try again..!

</td>

</tr>

</tfoot>

<%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</tbody>

</table>

</div></form></body>

</html>

**Menu.jsp:**

&nbsp; <span style="color:navy;font-size:13px;">Welcome ${uname}</span> <br/>

<dl class="demos-nav">

<dt>Main Menu

</dt>

<dd><a href="phase.jsp">Phase Management</a></dd>

<dd><a href="otherPhase.jsp">Miscellaneous-Phase</a></dd>

<dd><a href="employee.jsp">Employee Management</a></dd>

<dd><a href="changePWd.jsp">Change Password</a></dd>

<dd><a href="logout.jsp">Logout</a></dd>

<dt>Expenditure</dt>

<dd><a href="addSalary.jsp">Add Salary </a></dd>

<dd><a href="viewSalary.jsp">View Salary </a></dd>

<dd><a href="otherExp.jsp">Other Expenditure</a></dd>

<dt>Customer Management</dt>

<dd>

<a href="customer.jsp">Add Customer Details</a>

</dd>

<dd>

<a href="modifyCustomer.jsp">Modify Customer Details</a>

</dd>

<dd>

<a href="viewCustomer.jsp">View Customer Details</a>

</dd>

<dt>Maintenance Options</dt>

<dd>

<a href="addMaintenance.jsp">Add Maintenance Fee</a>

</dd>

<dd>

<a href="modifyMaintenance.jsp">Modify Maintenance Fee</a>

</dd>

<dd>

<a href="viewMaintenance.jsp">View Maintenance Fee</a>

</dd>

<dt>Finance Structure</dt>

<dd>

<a href="overallIncome.jsp">Overall Income</a>

</dd>

<dd>

<a href="overallExpenditure.jsp">Overall Expenditure</a>

</dd>

<dd>

<a href="balanceSheet.jsp">Balance Sheet1</a>

</dd>

<dd>

<a href="balanceSheet1.jsp">Balance Sheet2</a>

</dd></dl>

**Phase.jsp:**

<%@page import="com.lws.beans.PhaseBean"%>

<%@page import="com.lws.entities.PhaseEntity"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Phase Management</title>

<link rel="stylesheet" href="styles.css" />

<script src="details.js" type="text/javascript" ></script>

<style>

a {

color : blue;

text-decoration: none;

}

a:hover {

color : blue;

text-decoration: underline;

}

</style>

<script>

function modifyDetails(id, phase, size, plot) {

document.getElementById("id").value = id;

document.getElementById("phase").value = phase;

document.getElementById("size").value = size;

document.getElementById("plot").value = plot;

document.getElementById("sub").value = "Modify";

}

function deleteDetails(id, phase, size, plot) {

document.getElementById("id").value = id;

document.getElementById("phase").value = phase;

document.getElementById("size").value = size;

document.getElementById("plot").value = plot;

document.getElementById("sub").value = "Delete";

}

</script>

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top;">

<span style="color: navy;font-size: 16px;">Phase Management</span>

<br/>

<br/>

<div class="insertPanel">

<input type="hidden" id="id" name="id" />

<table cellpadding="5" cellspacing="5">

<tr>

<td>Phase</td><td><input type="text" id="phase" name="phase" /></td>

<td>Size</td><td><input type="text" id="size" name="size" /></td>

<td>Plot Numbers</td>

<td colspan="3"><input type="text" id="plot" style="width:250px;" name="plot" /> &nbsp; \*(User comma(",") as a separator)</td>

</tr><tr>

<td colspan="4"><input type="submit" id="sub" name="action" value="Register" />

<jsp:useBean id="add" scope="page" class="com.lws.entities.PhaseEntity"/>

<jsp:setProperty property="\*" name="add"/>

<%

try {

if(add.getPhase() != null) {

String action = request.getParameter("action");

if(action.equalsIgnoreCase("register")) {

if(PhaseBean.register(add)) {

%>

<span style="color:navy">Phase Details added successfully..!</span>

<%

} else {

%>

<span style="color:red">Phase details are not added..!</span>

<%

}

} else if(action.equalsIgnoreCase("modify")) {

if(PhaseBean.update(add)) {

%>

<span style="color:navy">Phase Details modified successfully..!</span>

<%

} else {

%>

<span style="color:red">Phase Details not modified..!</span>

<%

}

} else if(action.equalsIgnoreCase("delete")) {

if(PhaseBean.delete(add)) {

%>

<span style="color:navy">Phase Details deleted successfully..!</span>

<%

} else {

%>

<span style="color:red">Phase Details not deleted..!</span>

<%

}

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</td>

</tr>

</table>

</div>

<div class="mainContent">

<table style="width:100%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Phase</th>

<th>Size</th>

<th>Area(Sq.Yards)</th>

<th>Price</th>

<th>Plot Numbers</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</thead> <tbody>

<%

try {

java.util.List<PhaseEntity> list = PhaseBean.getList();

if( !list.isEmpty() && list != null) { int i = 0;

for(PhaseEntity pe : list) {

%>

<tr>

<td><%=++i %></td>

<td><%=pe.getPhase() %></td>

<td><%=pe.getSize() %></td>

<td><%=pe.getArea() %></td>

<td><%=pe.getPrice() %></td>

<td><%=pe.getPlot() %></td>

<td><a href="#" onclick="modifyDetails(<%=pe.getId() %>, '<%=pe.getPhase() %>', '<%=pe.getSize() %>', '<%=pe.getPlot() %>')">Edit</a></td>

<td><a href="#" onclick="deleteDetails(<%=pe.getId() %>, '<%=pe.getPhase() %>', '<%=pe.getSize() %>', '<%=pe.getPlot() %>')">Delete</a></td>

</tr>

<%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</tbody>

</table>

</div>

<br/>

<br/>

</td></tr></table> </div>

</center>

</form>

</body>

</html>

**OtherPhase.jsp:**

<%@page import="com.lws.entities.PhaseEntity"%>

<%@page import="com.lws.beans.PhaseBean"%>

<%@page import="java.util.List"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Miscellaneous Phase Management</title>

<link rel="stylesheet" href="styles.css" />

<script src="details.js" type="text/javascript"></script>

<style>

a {

color: blue;

text-decoration: none;

}

a:hover {

color: blue;

text-decoration: underline;

}

</style>

<script>

function modifyDetails(id, phase, size, plot, area, price) {

document.getElementById("id").value = id;

document.getElementById("phase").value = phase;

document.getElementById("size").value = size;

document.getElementById("plot").value = plot;

document.getElementById("area").value = area;

document.getElementById("price").value = price;

document.getElementById("sub").value = "Modify";

}

function deleteDetails(id, phase, size, plot, area, price) {

document.getElementById("id").value = id;

document.getElementById("phase").value = phase;

document.getElementById("size").value = size;

document.getElementById("plot").value = plot;

document.getElementById("area").value = area;

document.getElementById("price").value = price;

document.getElementById("sub").value = "Delete";

}

</script>

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include

file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top"><span

style="color: navy; font-size: 16px;">Miscellaneous Phase

Management</span> <br /> <br /> <input type="hidden" name="id" id="id" />

<div class="insertPanel">

<table cellpadding="3" cellspacing="3">

<tr>

<td>Phase</td>

<td><input type="text" name="phase" id="phase" /></td>

<td>Size</td>

<td><input type="text" name="size" id="size" /></td>

<td>Area</td>

<td><input type="text" name="area" id="area" /></td>

<td>Price</td>

<td><input type="text" name="price" id="price" /></td>

<td>Plot Number</td>

<td><input type="text" name="plot" id="plot" /></td>

</tr>

<tr>

<td colspan="6"><input type="submit" id="sub"

name="action" value="Register" /> <jsp:useBean id="mis"

scope="page" class="com.lws.entities.PhaseEntity" /> <jsp:setProperty

property="\*" name="mis" /> <%

try {

if (mis.getPhase() != null) {

String action = request.getParameter("action");

if (action.equalsIgnoreCase("register")) {

if (PhaseBean.registerMiscellaneous(mis)) {

%> <span style="color: navy">Phase Details added

successfully..!</span> <%

} else {

%> <span style="color: red">Phase Details are

not added..!</span> <%

}

} else if (action.equalsIgnoreCase("modify")) {

if (PhaseBean.modifyMiscellaneous(mis)) {

%> <span style="color: navy">Phase Details

modified successfully..!</span> <%

} else {

%> <span style="color: red">Phase Details are

not modified..!</span> <%

}

} else if (action.equalsIgnoreCase("delete")) {

if (PhaseBean.delete(mis)) {

%> <span style="color: navy">Phase Details

deleted successfully..!</span> <%

} else {

%> <span style="color: red">Phase Details are

not deleted..!</span> <%

}

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%></td>

</tr>

</table>

</div>

<div class="mainContent">

<table style="width: 100%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Phase</th>

<th>Size</th>

<th>Area(Sq.Yards)</th>

<th>Price</th>

<th>Plot Numbers</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</thead>

<tbody>

<%

try {

java.util.List<PhaseEntity> list = PhaseBean.getMisList();

if (!list.isEmpty() && list != null) {

int i = 0;

for (PhaseEntity pe : list) {

%>

<tr>

<td><%=++i%></td>

<td><%=pe.getPhase()%></td>

<td><%=pe.getSize()%></td>

<td><%=pe.getArea()%></td>

<td><%=pe.getPrice()%></td>

<td><%=pe.getPlot()%></td>

<td><a href="#"

onclick="modifyDetails(<%=pe.getId()%>, '<%=pe.getPhase()%>', '<%=pe.getSize()%>', '<%=pe.getPlot()%>', '<%=pe.getArea()%>','<%=pe.getPrice()%>')">Edit</a></td>

<td><a href="#"

onclick="deleteDetails(<%=pe.getId()%>, '<%=pe.getPhase()%>', '<%=pe.getSize()%>', '<%=pe.getPlot()%>', '<%=pe.getArea()%>','<%=pe.getPrice()%>')">Delete</a></td>

</tr>

<%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</tbody>

</table>

</div> <br /> <br /></td>

</tr>

</table>

</div>

</center>

</form>

</body>

</html>

**Employee.jsp:**

<%@page import="com.lws.entities.EmployeeEntity"%>

<%@page import="java.util.List"%>

<%@page import="com.lws.beans.EmployeeBean"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Employee Management</title>

<link rel="stylesheet" href="styles.css" />

<script src="details.js" type="text/javascript"></script>

<script>

function changeDetails(id, name, address, phone, email, desig, regdate,

action) {

document.getElementById("oldEid").value = id;

document.getElementById("eid").value = id;

document.getElementById("name").value = name;

document.getElementById("address").value = address;

document.getElementById("phone").value = phone;

document.getElementById("email").value = email;

document.getElementById("desig").value = desig;

document.getElementById("regDate").value = regdate;

document.getElementById("sub").value = action;

}

</script>

<style>

a {

color: blue;

text-decoration: none;

}

a:hover {

color: blue;

text-decoration: underline;

}

</style>

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include

file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top"><span

style="color: navy; font-size: 16px;">Employee Management</span>

<br /> <br /> <input type="hidden" id="oldEid" name="oldEid" />

<div class="insertPanel">

<table cellpadding="5" cellspacing="5">

<tr>

<td>Id</td>

<td><input type="text" id="eid" name="eid" /></td>

<td>Name</td>

<td><input type="text" id="name" name="name" /></td>

<td>Address</td>

<td><input type="text" id="address" name="address" /></td>

<td>Phone</td>

<td><input type="text" id="phone" name="phone" /></td>

</tr>

<tr>

<td>Email</td>

<td><input type="text" id="email" name="email" /></td>

<td>Designation</td>

<td><input type="text" id="desig" name="desig" /></td>

<td>Reg.Date</td>

<td><input type="text" id="regDate" name="regDate" /></td>

<td colspan="2"><input type="submit" id="sub"

name="action" value="Register" /> <jsp:useBean id="emp"

scope="page" class="com.lws.entities.EmployeeEntity" /> <jsp:setProperty

property="\*" name="emp" /> <%

try {

if (emp.getEid() != null) {

String action = request.getParameter("action");

if (action.equalsIgnoreCase("register")) {

if (EmployeeBean.validateEmployee(emp.getEid())) {

%> <span style="color: red">Id exists.. Please

try with new Id value.</span> <%

} else {

if (EmployeeBean.registerEmployee(emp)) {

%> <span style="color: navy"> Employee values

registered successfully..! </span> <%

} else {

%> <span style="color: red"> Employee values are

not registered..! </span> <%

}

}

} else if (action.equalsIgnoreCase("modify")) {

if (EmployeeBean.modifyEmployee(emp)) {

%> <span style="color: navy"> Employee values

modified successfully..! </span> <%

} else {

%> <span style="color: red"> Employee values are

not modified..! </span> <%

}

} else if (action.equalsIgnoreCase("delete")) {

if (EmployeeBean.deleteEmployee(emp)) {

%> <span style="color: navy"> Employee values

deleted successfully..! </span> <%

} else {

%> <span style="color: red"> Employee values are

not deleted..! </span> <%

}

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%></td>

</tr>

</table>

</div>

<div class="mainContent">

<table style="width: 150%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Modify</th>

<th>Delete</th>

<th>Id</th>

<th>Name</th>

<th>Address</th>

<th>Phone</th>

<th>Email</th>

<th>Designation</th>

<th>Reg. Date</th>

</tr>

</thead>

<tbody>

<%

try {

List<EmployeeEntity> list = EmployeeBean.getList();

if (!list.isEmpty() && list != null) {

int index = 0;

for (EmployeeEntity e : list) {

%>

<tr>

<td><%=++index%></td>

<td><a href="#"

onclick="changeDetails('<%=e.getEid()%>', '<%=e.getName()%>', '<%=e.getAddress()%>', '<%=e.getPhone()%>', '<%=e.getEmail()%>', '<%=e.getDesig()%>', '<%=e.getRegDate()%>', 'Modify')">Modify</a></td>

<td><a href="#"

onclick="changeDetails('<%=e.getEid()%>', '<%=e.getName()%>', '<%=e.getAddress()%>', '<%=e.getPhone()%>', '<%=e.getEmail()%>', '<%=e.getDesig()%>', '<%=e.getRegDate()%>', 'Delete')">Delete</a></td>

<td><%=e.getEid()%></td>

<td><%=e.getName()%></td>

<td><%=e.getAddress()%></td>

<td><%=e.getPhone()%></td>

<td><%=e.getEmail()%></td>

<td><%=e.getDesig()%></td>

<td><%=e.getRegDate()%></td>

</tr>

<%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</tbody>

</table>

</div></td>

</tr>

</table>

</div>

</center>

</form>

</body>

</html>

**changePWD.jsp:**

<%@page import="com.lws.beans.LoginBean"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Change Passwword</title>

<link rel="stylesheet" href="styles.css" />

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top">

<span style="color: navy;font-size: 16px;">Change Password</span>

<br/>

<br/>

<div class="insertPanel">

<table cellpadding="5" cellspacing="5">

<tr>

<td>Old Password</td><td><input type="password" name="oldPwd" /></td>

<td>New Password</td><td><input type="password" name="newPwd" /></td>

</tr>

<tr>

<td colspan="4">

<input type="submit" value="Change Password"/> &nbsp;

<%

try {

String newPwd = request.getParameter("newPwd");

String pwd = request.getParameter("oldPwd");

String uname = session.getAttribute("uname").toString();

if(newPwd != null && pwd != null) {

if(LoginBean.changePassword(uname, pwd, newPwd))

{

%>

<span style="color:navy">Password Changed Successfully..!</span>

<%

} else {

%>

<span style="color:red">Password Not Changed.. Please Try Again..!</span>

<%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</td>

</tr>

</table>

</div>

<br/>

<br/>

</td>

</tr>

</table>

</div>

</center>

</form>

</body>

</html>

addSalary.jsp:

<%@page import="com.lws.beans.SalaryBean"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1" import="com.lws.entities.EmployeeEntity, com.lws.beans.EmployeeBean"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Add Salary Management</title>

<link rel="stylesheet" href="styles.css" />

<script src="jquery-1.8.2.min.js" type="text/javascript" ></script>

<script src="details.js" type="text/javascript" ></script>

<link rel="stylesheet" href="http://code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css" />

<script src="http://code.jquery.com/jquery-1.9.1.js"></script>

<script src="http://code.jquery.com/ui/1.10.3/jquery-ui.js"></script>

<script>

$(function() {

$("#datepicker").datepicker({dateFormat : "dd-mm-yy"});

});

function addDetails(a) {

$("#eid").val(a);

}

</script>

<style>

div.ui-datepicker{

font-size:10px;

}

a {

color :blue;

text-decoration: none;

}

a:hover {

color:blue ; text-decoration : underline;

}

</style>

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top">

<span style="color: navy;font-size: 16px;">Add Salary </span>

<br/>

<br/>

<div class="insertPanel">

<table cellpadding="5" cellspacing="5">

<tr>

<td>Id</td><td><input type="text" name="eid" id="eid"/></td>

<td>Amount Paid:</td><td><input type="text" name="paid" id="paid" /></td>

<td>Paid Date:</td><td><input type="text" id="datepicker" name="dateFormat"/></td>

<td>Month:</td><td><select name="month">

<option value="January">January</option>

<option value="February">February</option>

<option value="March">March</option>

<option value="April">April</option>

<option value="May">May</option>

<option value="June">June</option>

<option value="July">July</option>

<option value="August">August</option>

<option value="September">September</option>

<option value="October">October</option>

<option value="November">November</option>

<option value="December">December</option>

</select></td>

</tr>

<tr>

<td colspan="8">

<input type="submit" value="Add Salary" /> &nbsp;

&nbsp;

<jsp:useBean id="add" class="com.lws.entities.SalaryEntity" scope="page" />

<jsp:setProperty property="\*" name="add"/>

<%

try {

if(add.getEid() != null) {

if(SalaryBean.registerSalary(add)) {

%>

<span style="color:navy">Salary Generated</span>

<%

} else {

%> <span style="color:red">Salary Not Generated</span><%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</td>

</tr>

</table>

</div>

<div class="mainContent">

<table style="width: 150%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Assign</th>

<th>Id</th>

<th>Name</th>

<th>Address</th>

<th>Phone</th>

<th>Email</th>

<th>Designation</th>

<th>Reg. Date</th>

</tr>

</thead>

<tbody>

<%

try {

java.util.List<EmployeeEntity> list = EmployeeBean.getList();

if (!list.isEmpty() && list != null) {

int index = 0;

for (EmployeeEntity e : list) {

%>

<tr>

<td><%=++index%></td>

<td><a href="#" onclick="addDetails('<%=e.getEid() %>')">Click to Assign</a></td>

<td><%=e.getEid()%></td>

<td><%=e.getName()%></td>

<td><%=e.getAddress()%></td>

<td><%=e.getPhone()%></td>

<td><%=e.getEmail()%></td>

<td><%=e.getDesig()%></td>

<td><%=e.getRegDate()%></td>

</tr>

<%

}

}

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</tbody>

</table>

</div>

</td>

</tr>

</table>

</div>

</center>

</form>

</body>

</html>

Similarly viewSalary.jsp, otherExp.jsp, customer.jsp, modifyCustomer.jsp , viewCustomer.jsp are coded to diplay salary data,other expenditures incurred,coustmer management modify coustomers and view coustomer details respectively.

Under the maintenance options addMaintenance.jsp, modifyMaintenance.jsp, viewMaintenance.jsp are coded to display Add Maintenance Fee, Modify Maintenance Fee and view maintenance fee respectively.

overallIncome.jsp:

<%@page import="com.lws.beans.MaintenanceBean"%>

<%@page import="com.lws.entities.MaintenanceEntity"%>

<%@page import="java.util.List"%>

<%@page import="com.lws.beans.CustomerBean"%>

<%@page import="com.lws.entities.CustomerEntity"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Overall Income</title>

<link rel="stylesheet" href="styles.css" />

<style>

a {

color :blue;

text-decoration : none;

}

a:hover {

text-decoration : underline;

color :blue;

}

</style>

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include

file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top"><span

style="color: navy; font-size: 16px;">Overall Income</span> <br />

<br />

<div class="insertPanel">

<table cellpadding="5" cellspacing="5">

<tr>

<td>Select Year:</td>

<td><select name="year">

<option value="no">-- Select --</option>

<option value="2012">2012-2013</option>

<option value="2013">2013-2014</option>

<option value="2014">2014-2015</option>

<option value="2015">2015-2016</option>

</select></td>

<td><input type="submit" value="Get Details" /></td>

</tr>

</table>

</div> <br />

<div class="mainContent">

<%

try {int main = 0, mem = 0;

String year = request.getParameter("year");

if (year != null && !year.equals("no")) {

%>

<span style="color: navy; font-size: 14px;">Membership

Registrations - (<%=year %>)</span> <br />

<br />

<table id="background-image" style="min-width: 200%">

<thead>

<tr>

<th>S.No</th>

<th>Name</th>

<th colspan="2">Address</th>

<th>State</th>

<th>Country</th>

<th>Phone</th>

<th>Email-Id</th>

<th>Club</th>

<th>Phase</th>

<th>Size</th>

<th>Area</th>

<th>Plot</th>

<th>Paid</th>

<th>Reg Date</th>

<th>Remarks</th>

</tr>

</thead>

<tbody>

<%

java.util.List<CustomerEntity> list = CustomerBean

.getList(" year(rdate) = " + year);

if (!list.isEmpty() && list != null) {

int index = 0;

for (CustomerEntity ce : list) { mem += ce.getPaid();

%>

<tr>

<td><%=++index%></td>

<td><%=ce.getName()%></td>

<td colspan="2"><%=ce.getAddress()%></td>

<td><%=ce.getState() %></td>

<td><%=ce.getCountry() %></td>

<td><%=ce.getPhone()%></td>

<td><%=ce.getEmai()%></td>

<td><%=ce.getClub()%></td>

<td><%=ce.getPhase()%></td>

<td><%=ce.getSize()%></td>

<td><%=ce.getArea()%></td>

<td><%=ce.getPlot()%></td>

<td><%=ce.getPaid()%></td>

<td><%=ce.getDateFormat()%></td>

<td><%=ce.getRemarks()%></td>

</tr>

<%

}

}

%>

</tbody>

</table>

<br/>

<span style="color: navy; font-size: 14px;">Maintenance

Registrations - (<%=year %>)</span> <br />

<br />

<table style="width:250%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Name</th>

<th>Address</th>

<th>State</th>

<th>Country</th>

<th>Phone</th>

<th>Email</th>

<th>Club</th>

<th>Phase</th>

<th>Size</th>

<th>Area</th>

<th>Plot</th>

<th>Receivable Amt</th>

<th>Received Amt</th>

<th>Total Received</th>

<th>Due</th>

<th>Received Date</th>

</tr>

</thead>

<tbody>

<%

List<MaintenanceEntity> lists = MaintenanceBean.getList(" year(m.pdate) = "+year+" ");

if( !lists.isEmpty() && lists != null) {

int index = 0;

for(MaintenanceEntity me : lists) { main += me.getPaid();

%>

<tr>

<td><%=++index %></td>

<td><%=me.getCe().getName() %></td>

<td><%=me.getCe().getAddress() %></td>

<td><%=me.getCe().getState()%></td>

<td><%=me.getCe().getCountry()%></td>

<td><%=me.getCe().getPhone() %></td>

<td><%=me.getCe().getEmai() %></td>

<td><%=me.getCe().getClub() %></td>

<td><%=me.getCe().getPhase() %></td>

<td><%=me.getCe().getSize() %></td>

<td><%=me.getCe().getArea() %></td>

<td><%=me.getCe().getPlot() %></td>

<td><%=me.getMainAmt() %></td>

<td><%=me.getPaid() %></td>

<td><%=me.getTotalAmt() %></td>

<td><%=me.getDue() %></td>

<td><%=me.getDateFormat() %></td>

</tr>

<%

}

}

%>

</tbody>

</table>

<%

} %>

</div>

<br/>

<span style="color: navy; font-size: 14px;">Membership

Registrations : <%=mem%></span> &nbsp; <span style="color: navy; font-size: 14px;">Maintenance

Registrations : <%=main %></span> &nbsp; <span style="color: navy; font-size: 14px;">Overall Income : <%= (main+mem) %></span>

<br/>

<a href="excelIncome.jsp?exportToExcel=yes&year=<%=year%>">Click to get Excel Report</a>

<br/>

<%

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</td></tr></table></div>

</center>

</form>

</body></html>

Balancesheet.jsp:

<%@page import="com.lws.beans.MaintenanceBean"%>

<%@page import="com.lws.entities.MaintenanceEntity"%>

<%@page import="java.util.List"%>

<%@page import="com.lws.beans.CustomerBean"%>

<%@page import="com.lws.entities.CustomerEntity"%>

<%@page import="com.lws.beans.ExpenditureBean"%>

<%@page import="com.lws.entities.ExpenditureEntity"%>

<%@page import="com.lws.beans.SalaryBean"%>

<%@page import="com.lws.entities.SalaryEntity"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<title>Balance Sheet</title>

<link rel="stylesheet" href="styles.css" />

</head>

<body>

<form method="post">

<center>

<div class="panel">

<%@include file="header.html"%>

<table

style="width: 100%; border-collapse: collapse; border-color: #ccc;">

<tr>

<td style="width: 20%; vertical-align: top"><%@include

file="menu.jsp"%></td>

<td style="border-left: 1px; vertical-align: top"><span

style="color: navy; font-size: 16px;">Balance Sheet</span> <br />

<br />

<div class="insertPanel">

<table cellpadding="5" cellspacing="5">

<tr>

<td>Select Year:</td>

<td><select name="year">

<option value="no">-- Select --</option>

<option value="2012">2012-2013</option>

<option value="2013">2013-2014</option>

<option value="2014">2014-2015</option>

<option value="2015">2015-2016</option>

</select></td>

<td><input type="submit" value="Get Details" /></td>

</tr>

</table>

</div> <br />

<div class="mainContent">

<%

try {int main = 0, mem = 0;int sal = 0, exp = 0;

String year = request.getParameter("year");

if (year != null && !year.equals("no")) {

%>

<span style="color: navy; font-size: 14px;">Membership

Registrations - (<%=year %>)</span> <br />

<br />

<table id="background-image" style="min-width: 200%">

<thead>

<tr>

<th>S.No</th>

<th>Name</th>

<th colspan="2">Address</th>

<th>State</th>

<th>Country</th>

<th>Phone</th>

<th>Email-Id</th>

<th>Club</th>

<th>Phase</th>

<th>Size</th>

<th>Area</th>

<th>Plot</th>

<th>Paid</th>

<th>Reg Date</th>

<th>Remarks</th>

</tr>

</thead>

<tbody>

<%

java.util.List<CustomerEntity> list = CustomerBean

.getList(" year(rdate) = " + year);

if (!list.isEmpty() && list != null) {

int index = 0;

for (CustomerEntity ce : list) { mem += ce.getPaid();

%>

<tr>

<td><%=++index%></td>

<td><%=ce.getName()%></td>

<td colspan="2"><%=ce.getAddress()%></td>

<td><%=ce.getState() %></td>

<td><%=ce.getCountry() %></td>

<td><%=ce.getPhone()%></td>

<td><%=ce.getEmai()%></td>

<td><%=ce.getClub()%></td>

<td><%=ce.getPhase()%></td>

<td><%=ce.getSize()%></td>

<td><%=ce.getArea()%></td>

<td><%=ce.getPlot()%></td>

<td><%=ce.getPaid()%></td>

<td><%=ce.getDateFormat()%></td>

<td><%=ce.getRemarks()%></td>

</tr>

<%

}

}

%>

</tbody>

</table>

<br/>

<span style="color: navy; font-size: 14px;">Maintenance

Registrations - (<%=year %>)</span> <br />

<br />

<table style="width:250%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Name</th>

<th>Address</th>

<th>State</th>

<th>Country</th>

<th>Phone</th>

<th>Email</th>

<th>Club</th>

<th>Phase</th>

<th>Size</th>

<th>Area</th>

<th>Plot</th>

<th>Receivable Amt</th>

<th>Received Amt</th>

<th>Total Received</th>

<th>Due</th>

<th>Received Date</th>

</tr>

</thead>

<tbody>

<%

List<MaintenanceEntity> lists = MaintenanceBean.getList(" year(m.pdate) = "+year+" ");

if( !lists.isEmpty() && lists != null) {

int index = 0;

for(MaintenanceEntity me : lists) { main += me.getPaid();

%>

<tr>

<td><%=++index %></td>

<td><%=me.getCe().getName() %></td>

<td><%=me.getCe().getAddress() %></td>

<td><%=me.getCe().getState() %></td>

<td><%=me.getCe().getCountry() %></td>

<td><%=me.getCe().getPhone() %></td>

<td><%=me.getCe().getEmai() %></td>

<td><%=me.getCe().getClub() %></td>

<td><%=me.getCe().getPhase() %></td>

<td><%=me.getCe().getSize() %></td>

<td><%=me.getCe().getArea() %></td>

<td><%=me.getCe().getPlot() %></td>

<td><%=me.getMainAmt() %></td>

<td><%=me.getPaid() %></td>

<td><%=me.getTotalAmt() %></td>

<td><%=me.getDue() %></td>

<td><%=me.getDateFormat() %></td>

</tr>

<%

}

}

%>

</tbody>

</table>

<span style="color: navy; font-size: 14px;">Salary Details - (<%=year %>)</span> <br />

<br />

<table style="width:150%" id="background-image" >

<thead>

<tr>

<th>Id</th>

<th>Name</th>

<th>Address</th>

<th>Phone</th>

<th>Email</th>

<th>Designation</th>

<th>Paid</th>

<th>Paid Date</th>

<th>Month</th>

</tr>

</thead>

<tbody>

<%

List<SalaryEntity> list1 = SalaryBean.getList(" year(s.pdate) = "+year+"");

if( !list1.isEmpty() && list1 != null) {

int index = 0;

for(SalaryEntity se : list1) { sal += se.getPaid();

%>

<tr>

<td><%=++index %></td>

<td><%=se.getEmployee().getName() %></td>

<td><%=se.getEmployee().getAddress() %></td>

<td><%=se.getEmployee().getPhone() %></td>

<td><%=se.getEmployee().getEmail() %></td>

<td><%=se.getEmployee().getDesig() %></td>

<td><%=se.getPaid() %></td>

<td><%=se.getDateFormat() %></td>

<td><%=se.getMonth() %></td>

</tr>

<%

}

}

%>

</tbody>

</table>

<br/>

<span style="color: navy; font-size: 14px;">Other Expenditure - (<%=year %>)</span> <br />

<br />

<table style="width:100%" id="background-image">

<thead>

<tr>

<th>S.No</th>

<th>Details</th>

<th>Paid Amount</th>

<th>Paid Date</th>

</tr>

</thead>

<tbody>

<%

List<ExpenditureEntity> lists1 = ExpenditureBean.getList(" year (pdate) = "+year+" ");

if( !lists1.isEmpty() && lists1 != null) {

int index = 0;

for(ExpenditureEntity ee : lists1) { exp += ee.getPaid();

%>

<tr>

<td><%=++index%> </td>

<td><%=ee.getDetails() %></td>

<td><%=ee.getPaid() %></td>

<td><%=ee.getDateFormat() %></td>

</tr>

<%

}

}

%>

</tbody>

</table>

<%

} %>

</div>

<br/>

<span style="color: navy; font-size: 14px;">Membership(Amt)

: <%=mem%></span> &nbsp; <span style="color: navy; font-size: 14px;">Maintenance(Amt)

: <%=main %></span> &nbsp; <span style="color: navy; font-size: 14px;">Overall Income : <%= (main+mem) %></span>

&nbsp;

<span style="color: navy; font-size: 14px;">Salary(Amt) : <%=sal%></span> &nbsp; <span style="color: navy; font-size: 14px;">Other

Expenditures : <%=exp%></span> &nbsp; <span style="color: navy; font-size: 14px;">Overall Expenditure : <%= (sal+exp) %></span>

<br/>

<span style="color: navy; font-size: 14px;">Balance Sheet : <%= (mem+main)-(sal+exp) %></span>

<%

} catch (Exception ex) {

System.out.println(ex.getMessage());

}

%>

</td>

</tr>

</table>

</div>

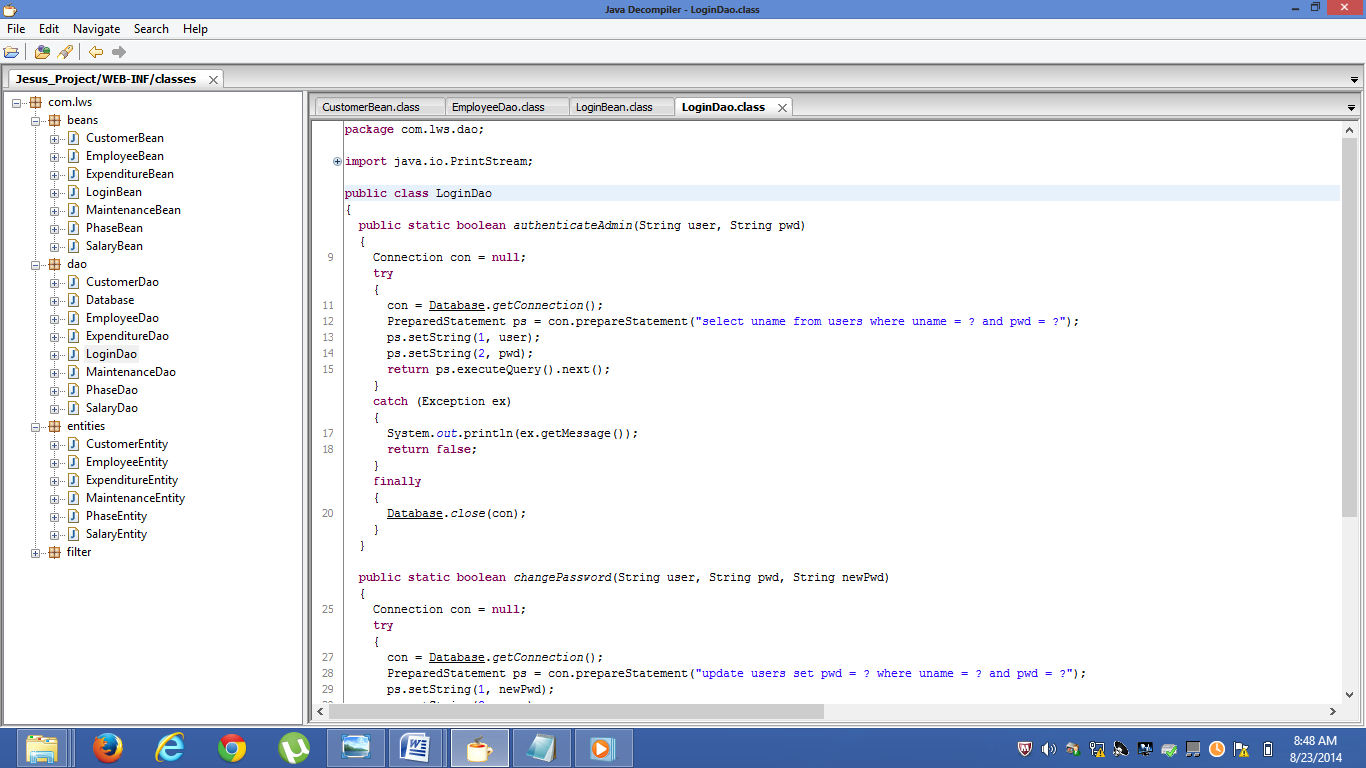
</center>

</form>

</body>

</html>

**THE DISTRIBUTION OF “java files” IN THE PROJECT SOURCE CODE:**



**loginDao.java:**

package com.lws.dao;

import java.io.PrintStream;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

public class LoginDao

{

public static boolean authenticateAdmin(String user, String pwd)

{

Connection con = null;

try

{

con = Database.getConnection();

PreparedStatement ps = con.prepareStatement("select uname from users where uname = ? and pwd = ?");

ps.setString(1, user);

ps.setString(2, pwd);

return ps.executeQuery().next();

}

catch (Exception ex)

{

System.out.println(ex.getMessage());

return false;

}

finally

{

Database.close(con);

}

}

public static boolean changePassword(String user, String pwd, String newPwd)

{

Connection con = null;

try

{

con = Database.getConnection();

PreparedStatement ps = con.prepareStatement("update users set pwd = ? where uname = ? and pwd = ?");

ps.setString(1, newPwd);

ps.setString(2, user);

ps.setString(3, pwd);

return ps.executeUpdate() == 1;

}

catch (Exception ex)

{

System.out.println(ex.getMessage());

return false;

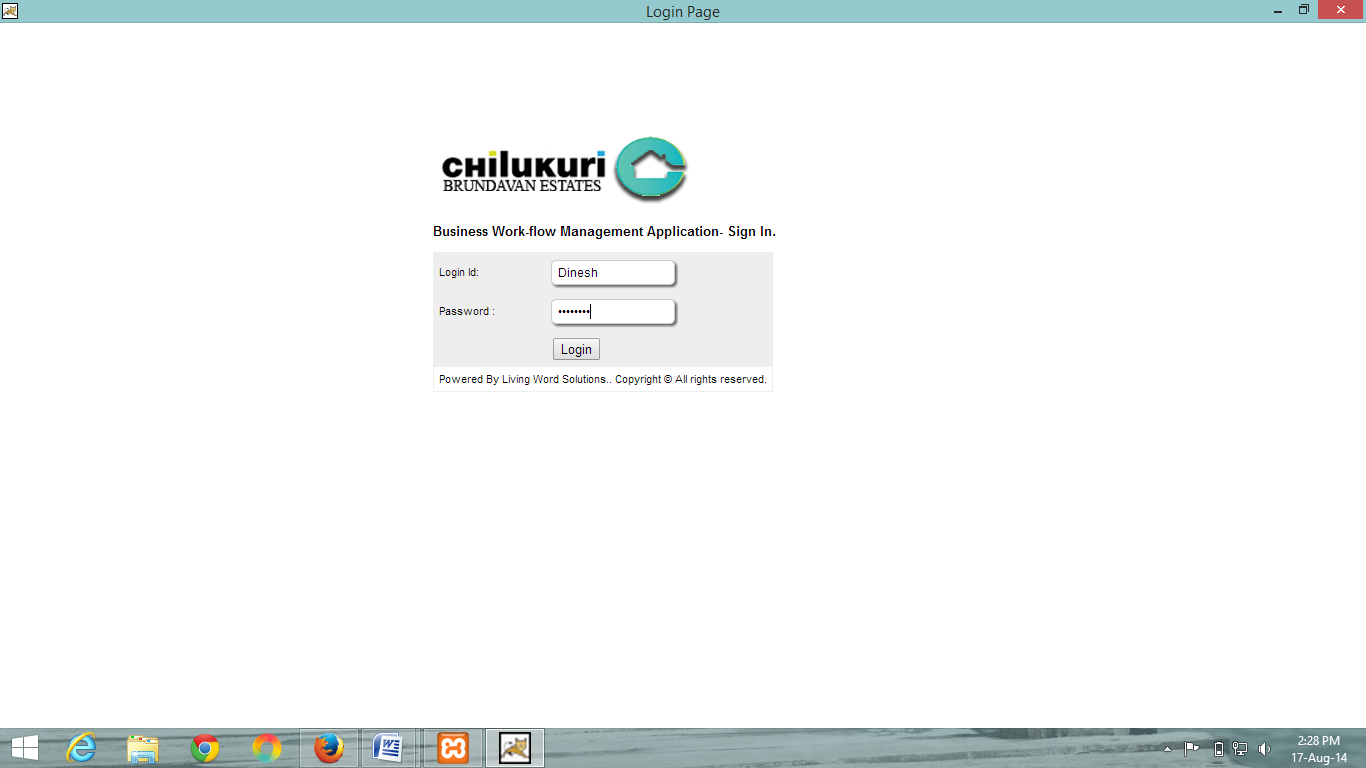
}

finally

{

Database.close(con);

}}}

6.OUTPUT-SCREENS:

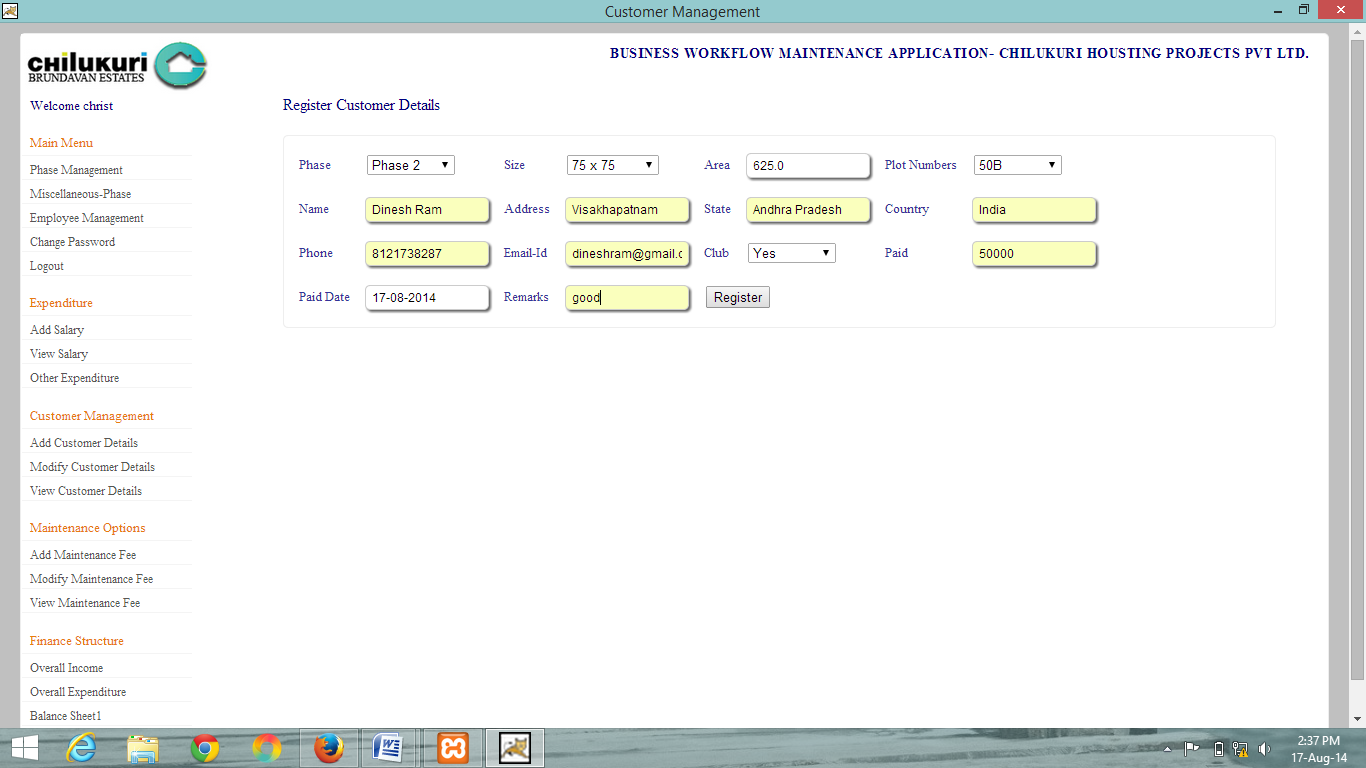


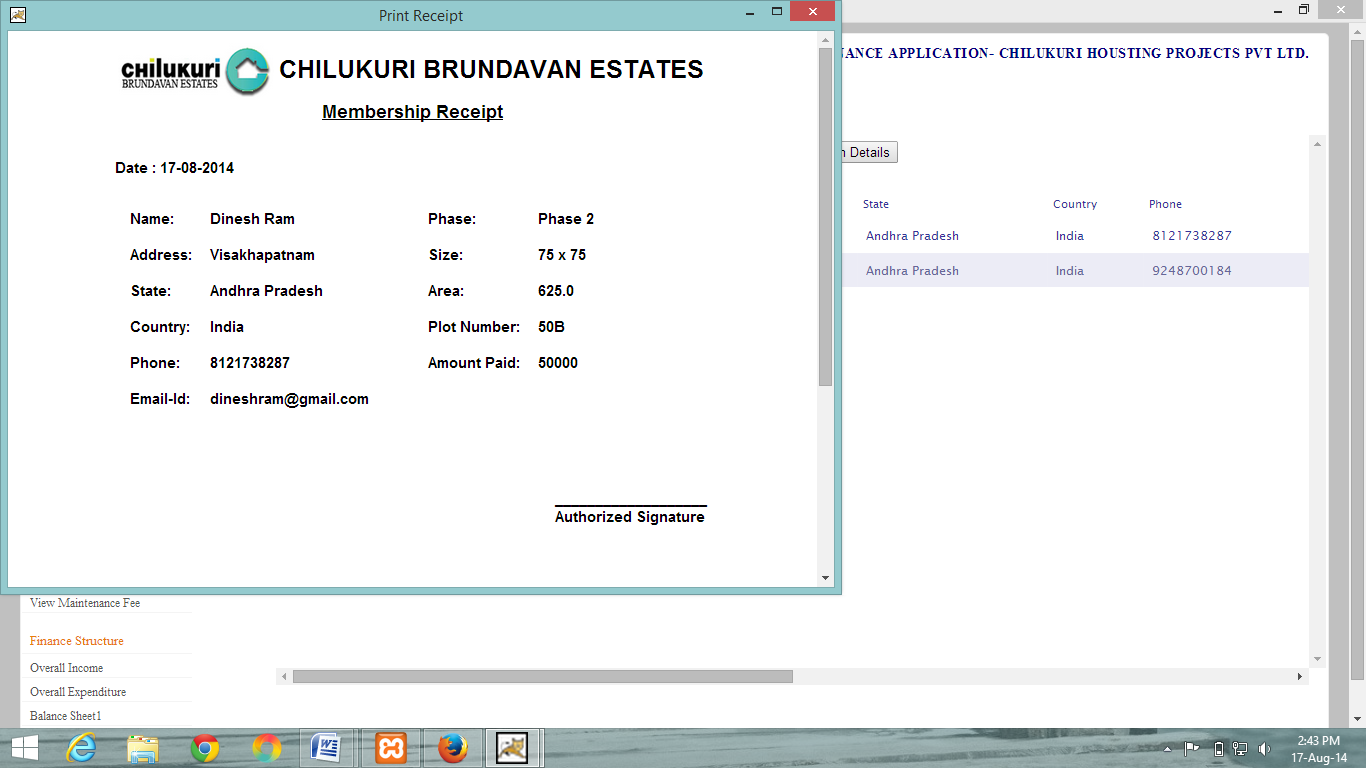




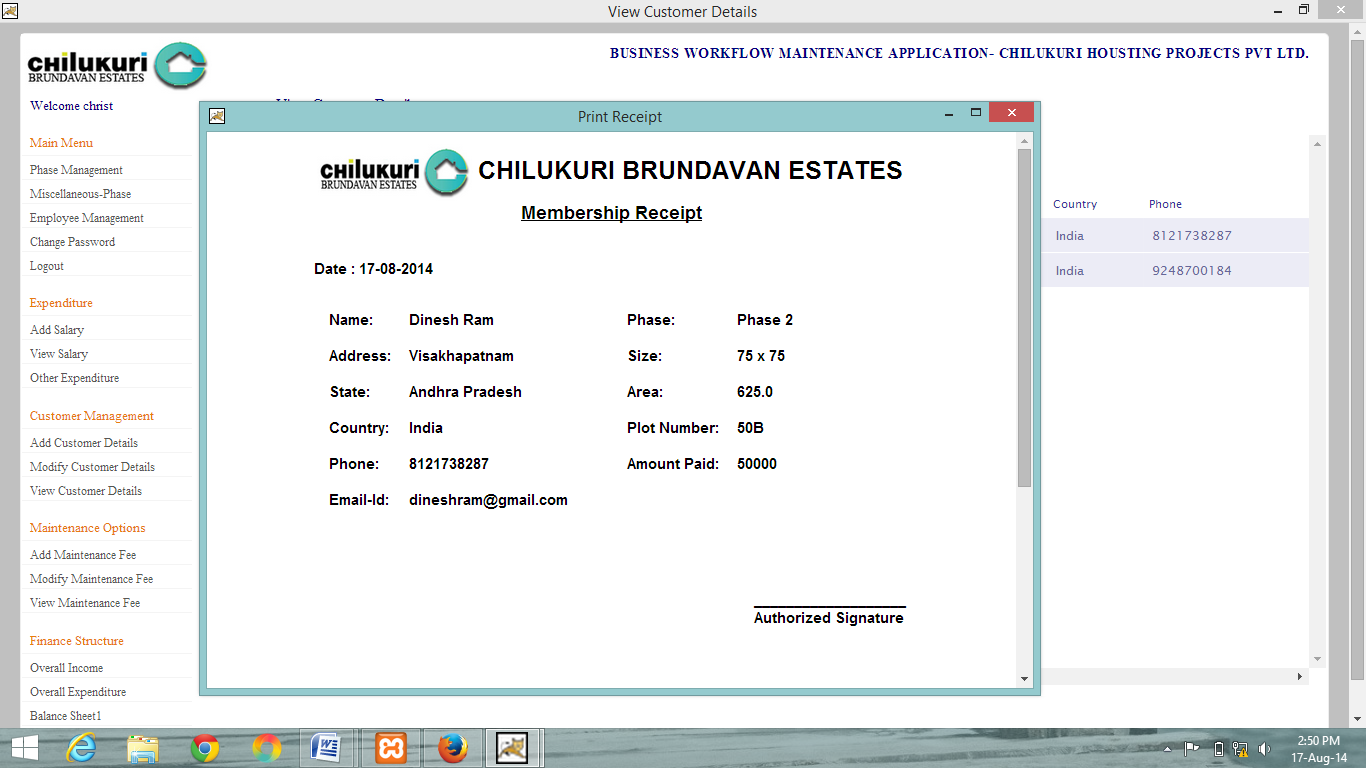






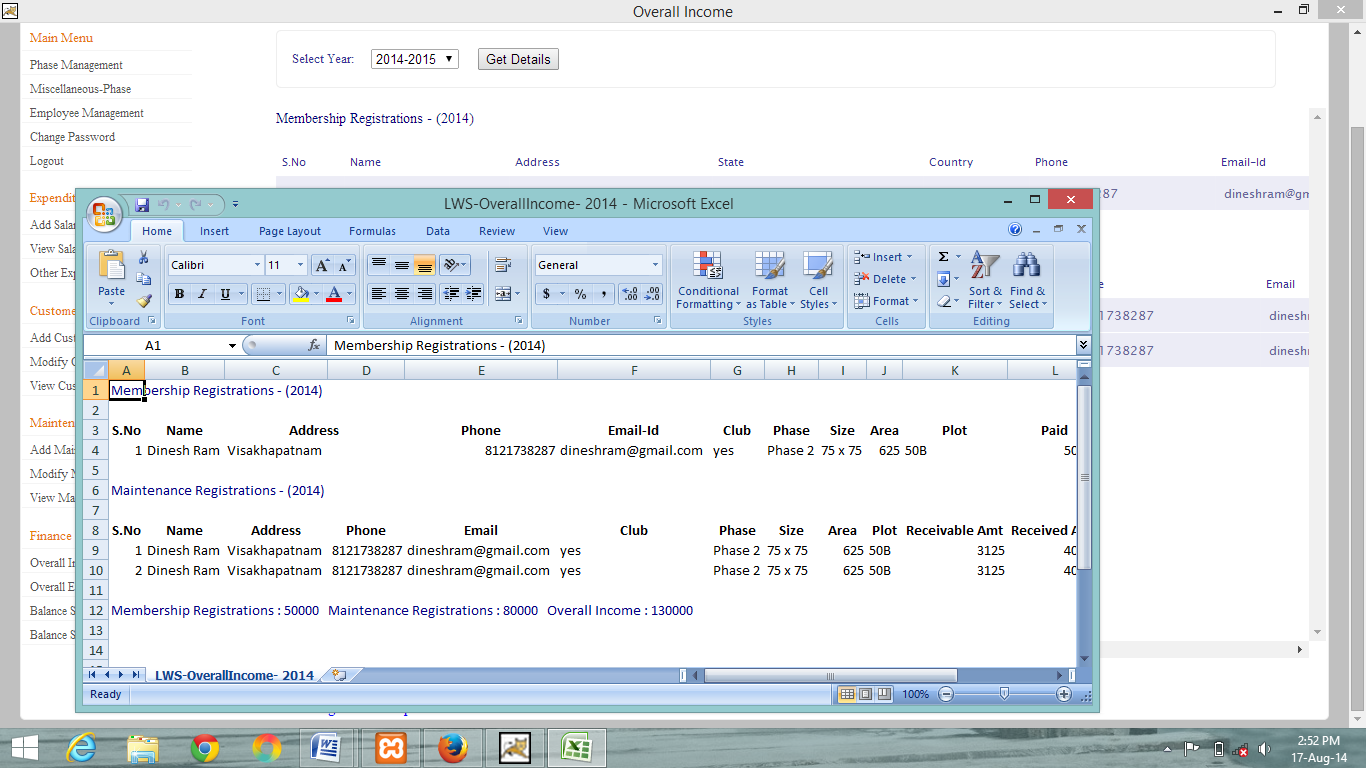




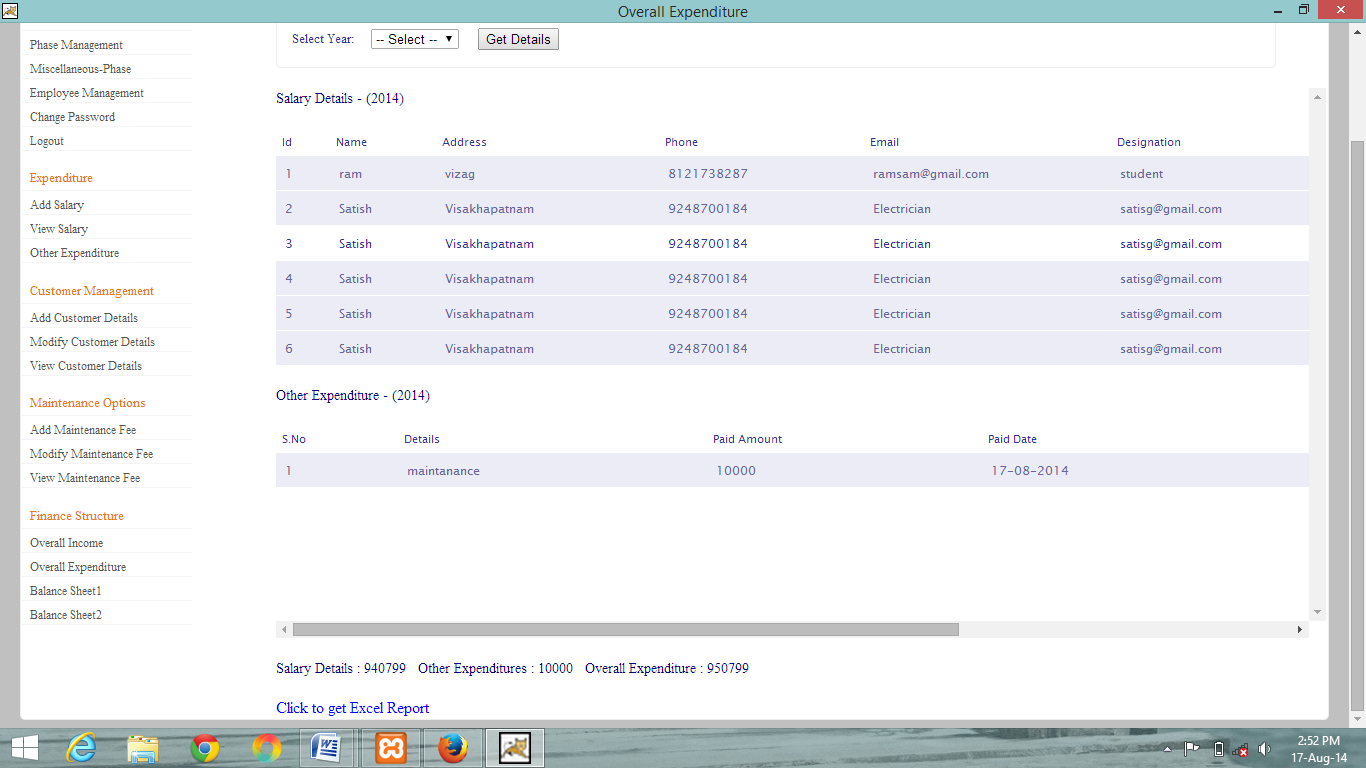






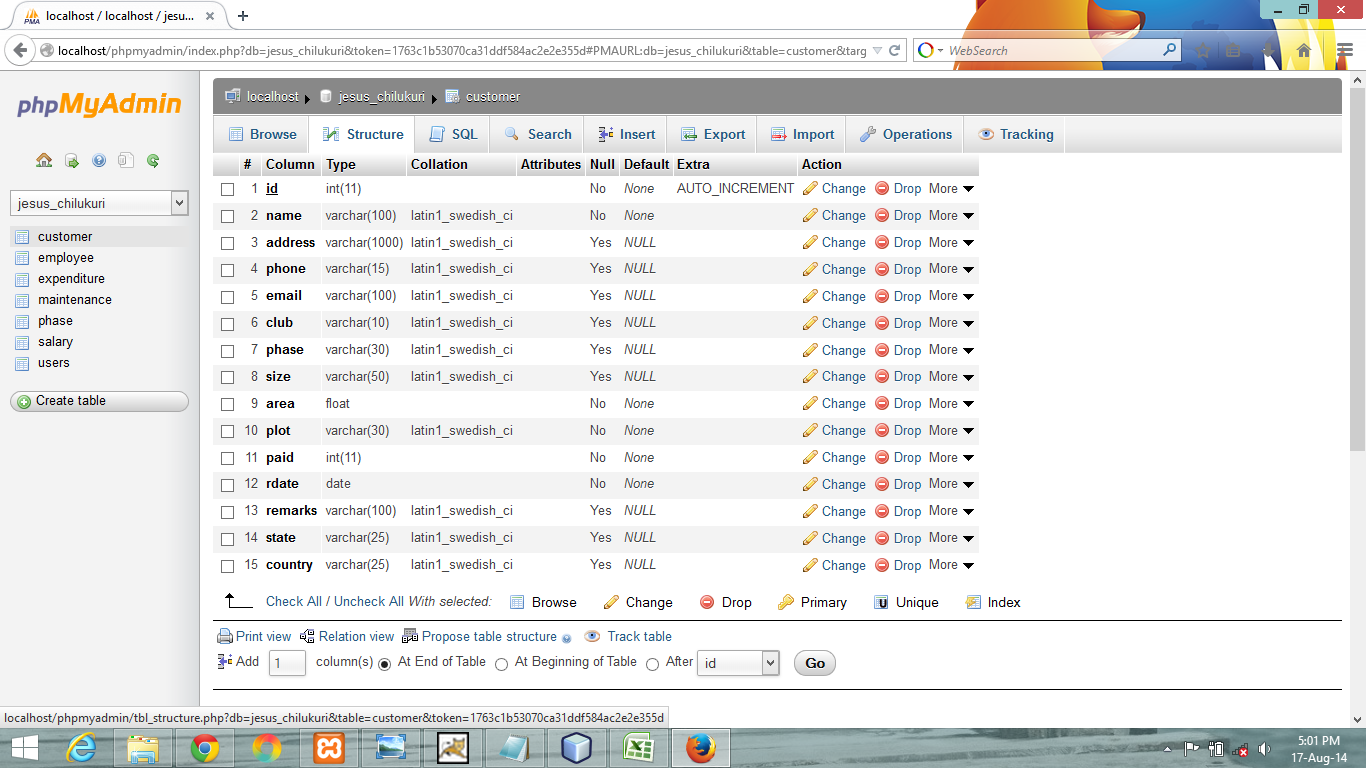








**7.0 DATABASE MODEL:**

****

**8.0 Testing**

**8.1 Introduction:**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

Software testing can also be stated as the process of validating and verifying that a software program/application/product:

1.meets the business and technical requirements that guided its design and development;

2.Works as expected; and

3.Can be implemented with the same characteristics.Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.

**8.2Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test No. | Test Case | Expected Output | Actual Output | Result |
| 1. | Invalid entries | Message | Showing a message invalid | passed |
| 2. | Valid input | Navigation to next page | Navigation to next page | passed |
| 3. | Null entries | Error | Navigation to next page | passed |