CHAPTER: 1
COLLEGE MANAGEMENT SYSTEM

INTRODUCTION

1.1Introduction

The main objective of college management system is to automate all functionalities of a college or university. Using this system, you can manage all college management work like admission, fees submission, time table management and result declaration. Using this college management system, you can view or update data and information about students and staff easily. This system helps in managing the activity like student admission, student registration, fees submission. Admin can also retrieve information of employee student.

The COLLEGE MANAGEMENT SYSTEM can be used to store student information like attendance, fees, and student result etc. admin can create report regarding any student any time using this system. Using this system, you can register new student and their course details. You can submit students’ fees and can check fees details anytime. You can create exam result and submit in this system. Student can check their result online by logging to the system. You can also add new employee in the system and can check details of the employee easily. Student can also check course detail online from this system.

Using this system, you can manage all information of all aspects of a college, its students, faculties, Departments, marks and other curricular activities. College management system provides the easiest way to manage all functionalities of a college. This system facilitates colleges to maintain the functionality related to college employees and their students.

College Management System can store and manage all data of the various departments of a college like Administration, Attendance, Staff details etc. using this system user can retrieve any information related to student, teacher and fees. Using this system teacher can check student attendance anytime. This system also help teacher to announce the result. College administration can also manage college work easily. Admin can check leave, salary and other details of teacher any time. They can also create time table of classes from this system. The Library module is used for the data process of library and book accessing for students and staffs.

1.2Objective

This is a web-oriented application allows us to access the whole information about the college, staffs, students, facilities etc. This application provides a virtual tour of Campus. Here we will get the latest information about the students and staffs. This generic application designed for assisting the students of an institute regarding information on the courses, subjects, classes, assignments, grades and timetable. It also provides support that a faculty can also check about his daily schedule, can upload assignments, and notices to the students. Here administrator will manage the accounts of the student and faculties, makes the timetable, and upload the latest information about the campus. It also helps in searching students, staff etc. It increases the throughput.

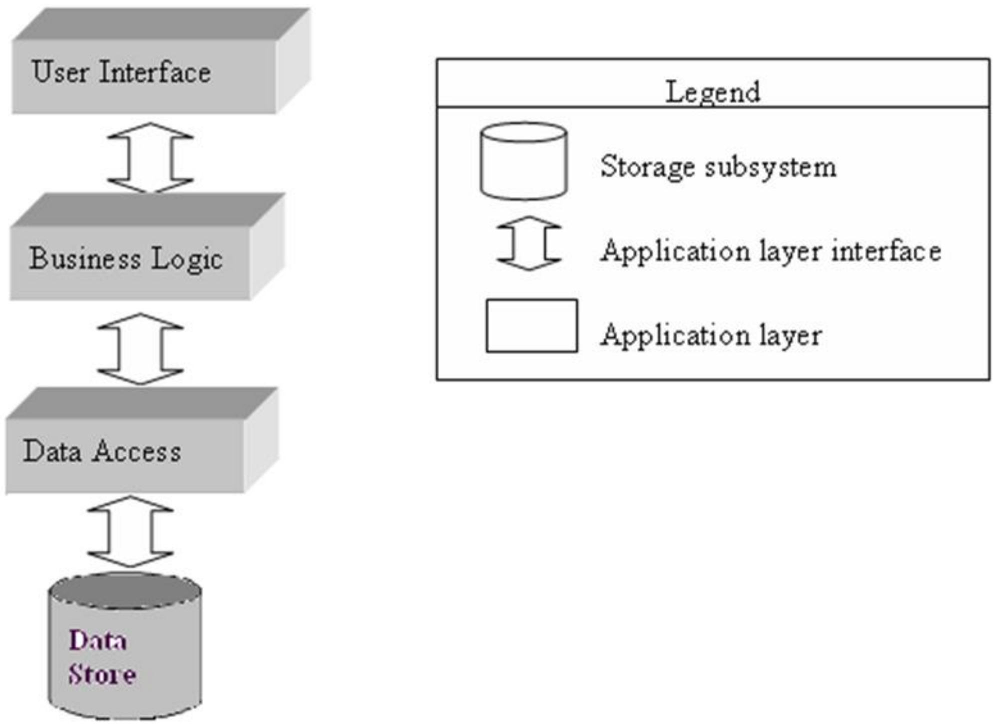
1.3 Scope

- College information: Through this service one can access the complete information about the college campus such as courses available, admission procedure, placements, college events, achievements etc.
- Student tracking: Any company or any organization that want to check the summary about the student of the college, so that they will be able to choose the particular students for their campus placement And for that purpose they will be given a particular link through which they can access the information required.
- Student attendance status: It gives the attendance status of students. Faculty will update the attendance periodically and can be seen by students and parents.
- Student’s performance in exams: This facility provides the performance of the student in each exam which is conducted by university or college such as midterm performance. Marks obtained by students in exams will be updated by faculties that can be access by students and parents.
- Exam Notification: This facility notifies students and parents about examination schedule.
- Events: It will give information about different events that will be conducted by college time to time. Information about these events will be updated by administrator.

- Online assignments: This service provides the facility to faculty to upload assignments and to students to submit these assignments online.
- Information about staff: It will help in maintaining complete information about college faculty members such as their department, cadre, date of joining, salary, etc. Administrator will register new faculties and remove their account when they leave the college.

Logical View:

It provides the user with an abstract view of the overall system functionality.



CHAPTER: 2

Theoretical Background

Today in colleges student details are entered manually. The student details in separate records are tedious task. Referring to all these records and updating is needed. There is a chance for more manual errors.

2.1Problems in existing system:

- It was limited to a single system.
- It was less user-friendly.
- It has a lot of manual work (Manual system does not mean that we are working with pen and paper, it also include working on spread sheets and other simple software's)
- It requires more no of employees need to work.
- It was time consuming process.
- The present system was very less secure.
- It is unable to generate different kinds of report.

2.1.1Solution to these problems:

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

- User friendliness is provided in the application with various controls.
- The system makes the overall project management much easier and flexible.
- It can be accessed over the Internet.
- Various classes have been used to provide file upload and mail features.
- There is no risk of data mismanagement at any level while the project development is under process.
- It provides high level of security using different protocols like https etc.

2.2 Problem Definition

The problem is to provide the complete information about the college campus. In which the college staff members, students and parents can access the information and will be familiar with college campus. It will provide interactive environment for the staff, students and parents by getting knowledge of student attendance, remarks, exams performances, grades, timetables, notices etc.

2.3 Benefits of purposed college management system

The benefits of college management system for the employee is they can create any kind of certificate easily using this system. They can easily retrieve all information related to student and employee. Admin has all the Collective records of students of all the branches. Admin can check all the records of employees of all departments anytime. This system gives easy approach to find the detail information for any student/employee. Using this college management system, it is very easy to handle all functionality of college.

This system is beneficial for both students and employees as they can get all previous or current information when they need. This system is also helpful to maintain the students record like admission record, fees record, exam result records. College management system can help to get all or a particular student attendance information. Also, it can help to maintain the fees and accounting reports of college in proper way. This system also helps to generate mark sheets of current year.

Higher education is no longer confined to the four walls of a classroom. It is moving beyond the limitations of time and geographical boundaries. Along with students, institutions are also advancing on various fronts. They have empowered themselves with advanced technologies, the latest software, superior facilities and so on to attract more and more students and increase retention rates.

CHAPTER: 3

Software Requirement Specification

3.1 User requirements:

The following requirements are raised during the analysis of the needs of the users:

- A Person Should be able to login to the system through the first page of the Application.
- The Administrator can create users as per user requirement.
- Admin can upload the data for a particular Student. On successful completion of upload, user (Student/Parent/Faculty) can view reports.
- A general user will have access to see the status of particular Student id number.
- Student (user) can use all the facilities, same as which are provided to him in the college.
- Student can see attendance, notices, grades, report and other facilities in updated manner.
- There will be a separate page for every student as his account in which he can get notices, attendance, grades, assignments etc.
- Parent can just view the record of student with the username and password provided.
- Faculty can give the attendances and notices for the students.
- The administrator verifies all these reports and generates them for users to view them.

After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

3.2 Analysis Model

This document plays a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

SPIRAL MODEL was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

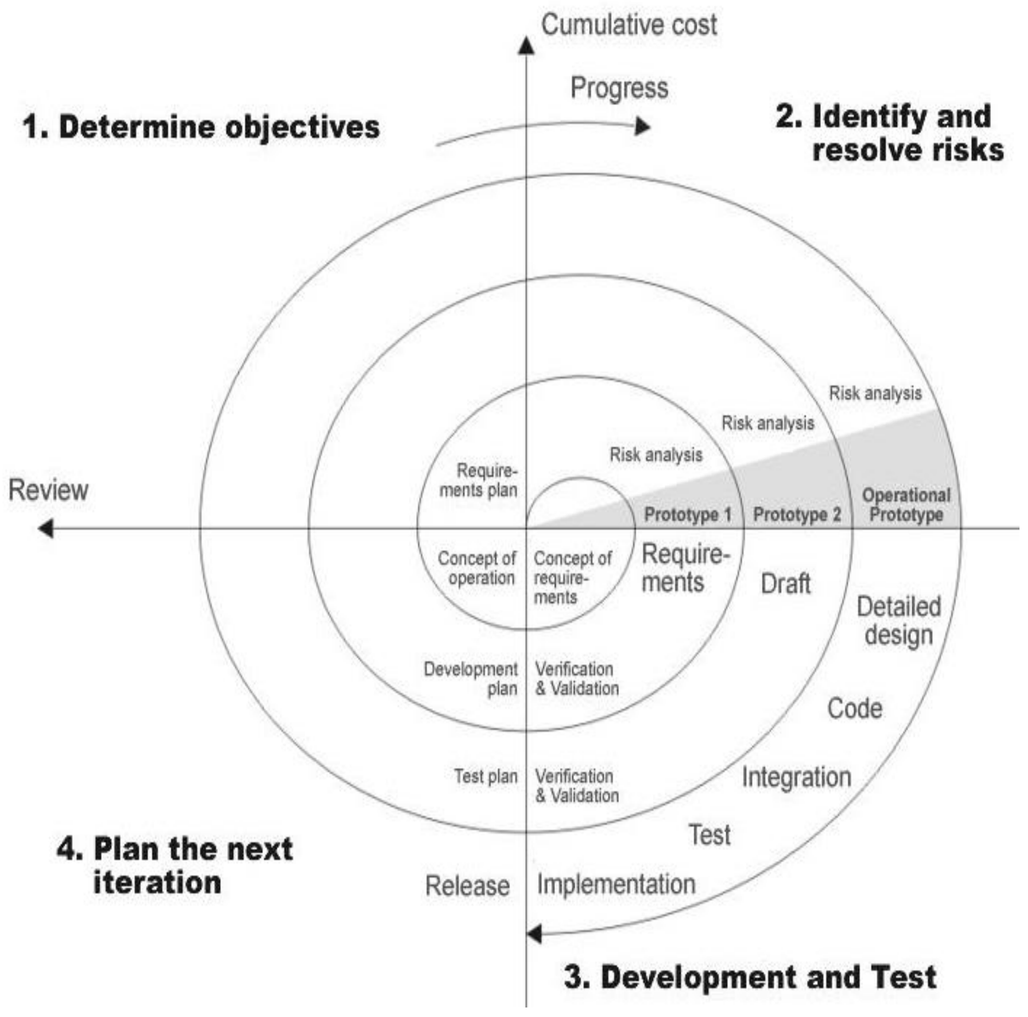
The steps for Spiral Model can be generalized as follows:

- The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
- A preliminary design is created for the new system.
- A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
- A second prototype is evolved by a fourfold procedure:
    1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
    2. Defining the requirements of the second prototype.
    3. Planning and designing the second prototype.
    4. Constructing and testing the second prototype.

- At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer’s judgment, result in a less-than-satisfactory final product.
- The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
- The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
- The final system is constructed, based on the refined prototype.

- The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

The following diagram shows how a spiral model act like:



3.3 Study of the System:

3.3.1 Graphical user interface

In the flexibility of the uses the interface has been developed a graphics concept in mind, associated through a browser interface. The GUI'S at the top level have been categorized as

1. Administrative user interface
2. The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.The operational or generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

3.4 Number of Modules

The system after careful analysis has been identified to be presented with the following modules:

The modules involved are:

1. College information: Through this service one can access the complete information about the college campus such as courses available, admission procedure, placements, college events, achievements etc.
2. Student tracking: Any company or any organization that want to check the summary about the student of the college, so that they will be able to choose the particular students for their campus placement And for that purpose they will be given a particular link through which they can access the information required.
3. Student attendance status: It gives the attendance status of students. Faculty will update the attendance periodically and can be seen by students and parents.
4. Student's performance in exams: This facility provides the performance of the student in each exam which is conducted by university or college such as midterm performance. Marks obtained by students in exams will be updated by faculties that can be access by students and parents.
5. Exam Notification: This facility notifies students and parents about examination schedule.
6. Events: it will give information about different events that will be conducted by college time to time. Information about these events will be updated by administrator.
7. Online assignments: This service provides the facility to faculty to upload assignments and to students to submit these assignments online.
8. Information about staff: It will help in maintaining complete information about college faculty members such as their department, cadre, date of joining, salary, etc. Administrator will register new faculties and remove their account when they leave the college.

CHAPTER: 4
SYSTEM PLANNING (PERT CHART)

Perform and evaluate feasibility studies like cost-benefit analysis, technical feasibility, time feasibility and operational feasibility for the project. Project Scheduling should be made using PERT charts.

Feasibility study is carried out to decide whether the proposed system is feasible for the company. The feasibility study is to serve as a decision document it must answer three key questions:

1. Is there a new and better way to do the job that will benefit the user?
2. What are the cost and the savings of the alternative(s)?
3. What is recommended?

4.1 Technical feasibility:

Technical feasibility determines whether the work for the project can be done with the existing equipment, software technology and available personal Technical feasibility is concerned with specifying equipment and software that will satisfy the user requirement.

This project is feasible on technical remarks also, as the proposed system is more beneficiary in terms of having a sound proof system with new technical components installed on the system he proposed system can run on any machines supporting Windows and Internet services and works on the best software and hardware that had been used while designing the system so it would be feasible in all technical terms of feasibility.

Technical Feasibility Addresses Three Major Issues:

Is the proposed Technology or Solution Practical?

The technologies used are matured enough so that they can be applied to our problems. he practically of the solution we have developed is proved with the use of the technologies we have chosen the technologies such as JAVA USP, Servlet). JavaScript and the compatible HWs are so familiar with the today' s knowledge-based industry that anyone can easily be compatible to the proposed environment.

Do we currently possess the necessary technology?

We first make sure that whether the required theologies are available to us or nor. If they are available then we must ask if we are the capacity. For instance, "Will our Current Printer be able to handle the new reports and forms required of a new system?

Do we posse ss the necessary Technical Expertise and is the Schedule reasonable?

This consideration of technical feasibility is often forgotten during feasibility analysts. We me are the technology, but that doesn't mean we have the skills required to properly apply that technology. As far as our project is concerned, we are the necessary expertise so that the proposed solution can be made feasible.

4.2 Economic feasibility:

Economical feasibility determines whether there are sufficient benefits in creating to make cost accepted, or is the cost of the system too high. As this signifies cost benefit analysis and savings. On the behalf of the cost-benefit analysis, the proposed system is feasible and is economical regarding pre-assumed cost for making system. During the economical feasibility test, we maintained the balance between the Operational and Economical feasibilities, as the two were the conflicting. For example, the solution that provides the best operational impact or the end-users may also be the most expensive and, therefore, the least economical feasible. We classified the costs of Online Counselling according to the phase in which they occur. As we know that the system development costs are usually one-time costs that not recur after the project has been completed. For calculating the Development costs we evaluated certain cost categories viz.

- Personnel costs
- Computer usage
- Training
- Supply and equipment’s costs
- Cost of any new computer equipment and software.

In order to test whether the Proposed System is cost-effective or not we evaluated it through three techniques viz.

- Payback analysis
- Return on Investment:
- Net Present value

4.3 Operational Feasibility :

Operational feasibility criteria measure the urgency of the problem (survey and study phases) or the acceptability of a solution (selection, acquisition and design phases).How do you ensure operational feasibility. There are two aspects of operational feasibility to be considered:

(a) Is the problem worth solving or will the solution to the problem work?

There are certain measures, which decide, the effectiveness of the system These measures can be collectively called as PIECES.

P(Performance):

The sell purchase online provides adequate throughput an response time.

I (Information):

The sell purchase online provides Student and Star with tamely, pertinent, accurate, and usefully formatted information

E (Economy):

The sell purchase online of reduce the cost or the Counselling or Student reporting(selection).

C(Control):

The sell purchase online offer globally controls to protect against fraud and toguarantee the accuracy and security of the data and information.

E (Efficiency):

The sell purchase online makes the maximum use of available resource and minimum processing delays.

S(Services):

The sell purchase online provides desirable and reliable service to those who need it .The sell purchase online is flexible and expandable.

(b) How do the students and staff feel about the problem (Solution)?

It is important to evaluate whether a system can work. We must also evaluate whether a system will work. A workable solution might fail because of students, staff resistance. In case of power project sell purchase online we have examined all the concerns that can be further affect its operational feasibility. The following points will explore the concerns. Itself is complete support of the student/staff online reporting and accessing the information. This purchase online as major role of students is based on the students of feel comfortable and upgraded with it system.

4.4 Software Engineering Paradigm Applied:

The development strategy that encompasses the process, methods, and tools and the generic phases is called Software Engineering Paradigm. The s/w paradigm for software is chosen based on the nature of the project and application, the method and tools to be used, and the controls and desirables that are required. All software development can be character as a problem -solving loops in which four distinct stages are encountered – status, que, problem definition technical development, and solution integration. Status quo represents the current state of affairs, Problem definition identifies the specific problem to be solved technical development solves the problem through the application of some technology, and solution integration delivers the results to those who requested the solution in the first place. There are various software paradigms, but we used Waterfall model (the linear sequential mode).which states that the phases are organized in a linear order. The Waterfall model suggests a systematic, sequential approach to s/w development that the begins at the system level and progresses through analysis, design, coding, testing and maintenance. The sequence of activities formed in a software development project with the waterfall model is: system analysis, system design, coding, testing & integration, installation, and maintenance.

For a successful project resulting in a Successful product, all phases listed in the waterfall model must be performed. Any different ordering the phases will result n a less successful software product.

There are a number of project outputs in waterfall model that is produced to produce a successful product:

- Requirement documents and project plan
- System and detailed design
- Programs (code)
- Test pan, test reports and manuals
- Installation reports

4.5 Benefits of Waterfall model :

The Waterfall model depends upon the sequential approach in which each stage should complete itself to start the next stage. This consecutive order is suitable for smaller projects which are easy to start. The Waterfall model is a useful and predictable system if the requirements are clear, static, and well-documented. Furthermore, the Waterfall model is beneficial if the technology is mature and can easily be understood. The project should be short for the application of the Waterfall model.

The development moves from concept, complete design, deployment, testing, installation, troubleshooting, and ends up in maintenance and operation. Each phase of development has its own value and should be worked properly.

4.6 Limitations of Waterfall Model:

- The waterfall model urgent requirements of a system can be baseline before the design begins. This is possible for the system design to automate and existing manual system. For system, this is a new system, determining the requirement is difficult, as the user does not you are not requirements.
- Freezing the requirements usually required using the hardware.
- The waterfall model triple x requirement be completed before the rest of the development can be proceed.
- It is a document process request form documents at end of the phase. This tends to make the process documentation and it's not suitable for many applications.

CHAPTER: 5

Methodology Adopted, System Implementation, Details of Hardware & Software Used & Technologies Description

5.1 Methodology adopted and System implementation:

1. Apache tomcat is used as a web server to host the application.
2. All the environment variables are set.
3. The application is pasted in the webapps folder.
4. Web server is started now.
5. Application is run using the web browser by typing http://localhost/cis
6. Web.xml file is used to control the flow and user actions.

5.2 Details of hardware & software used:

Hardware Specification (Minimum):

Disc Space:	40 GB
PC Used:	IBM Compatible
Processor:	Pentium 3
Memory:	512 MB RAM
File System:	64 bit

Software Specification:

Operating System (Server Side):	Windows XP.
Operating System (Client Side):	Windows XP.
Client End Language:	HTML
Local Validation:	PHP
Server-Side Language:	PHP
Database:	My Sql 2000
Web Server:	XAMPP server
Web Browser:	Internet Explorer 8/ Mozilla Firefox.

5.3 Technologies Description:

The technology selected for implementing College Information Management System is JSP/MySQL. Glassfish is used as the HTTP server. The development was done in a 'windows operating system' using Eclipse IDE.

5.3.1 HTML

HTML (Hypertext Markup Language) is the most basic building block of the Web. It describes and defines the content of a webpage. Other technologies besides HTML are generally used to describe a webpage's appearance.
"Hypertext" refers to links that connect webpages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web.

By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web.

HTML uses "Markup" to annotate text, images, and other content for display in a Web browser. HTML markup includes special "elements" such as <head>, <title>, <body>, <img>, <p>, <header>,<footer> and many others.

5.3.2 CSS

Cascading Style Sheets (CSS) are a stylesheet language used to describe the presentation of a document written in HTML or XML(including XML dialects like SVG or XHTML).
CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.

CSS is one of the core languages of the open web and has a standardized W3C. Developed in levels, CSS1 is now obsolete, CSS2.1 is a recommendation, and CS$3, now split into smaller modules, is progressing on the standardization track.

5.3.3 JAVASCRIPT

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

5.3.4 MySQL

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. MySQL is a popular choice of database for use in web applications and is an open source product. The process of setting up a MysaL database varies from host to host, however we will end up it a database name, a user name and a password. Before using our database, we must create a table. A table is a section of the database for storing related information. In a table we will set up the different fields which will be used in that table. Creating a table in MySQL is simple, we just type the name, select the number of fields and click the 'go' button. we will then be taken to a setup screen where you must create the fields for the database. Another way of creating databases and tables in MySQL is by executing Simple SQL statements. We have used this method in order to create our database and tables.

5.3.5 J2EE

J2EE is a platform-independent, Java-centric environment from Sun for developing, building and deploying Web-based enterprise applications online. The J2EE platform consists of a set of services, APIs, and protocols that provide the functionality for developing multitier, web-based applications.

Java server 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. JSP enables us to write HTML pages containing tags, inside which we can include powerful Java programs. Using ISP, one can easily separate Presentation and Business logic as a web designer can design and update JSP pages creating the presentation layer and java developer can write server side complex computational code without concerning the web design. And both the layers can easily interact over HTTP requests.

5.3.6 Eclipse IDE:

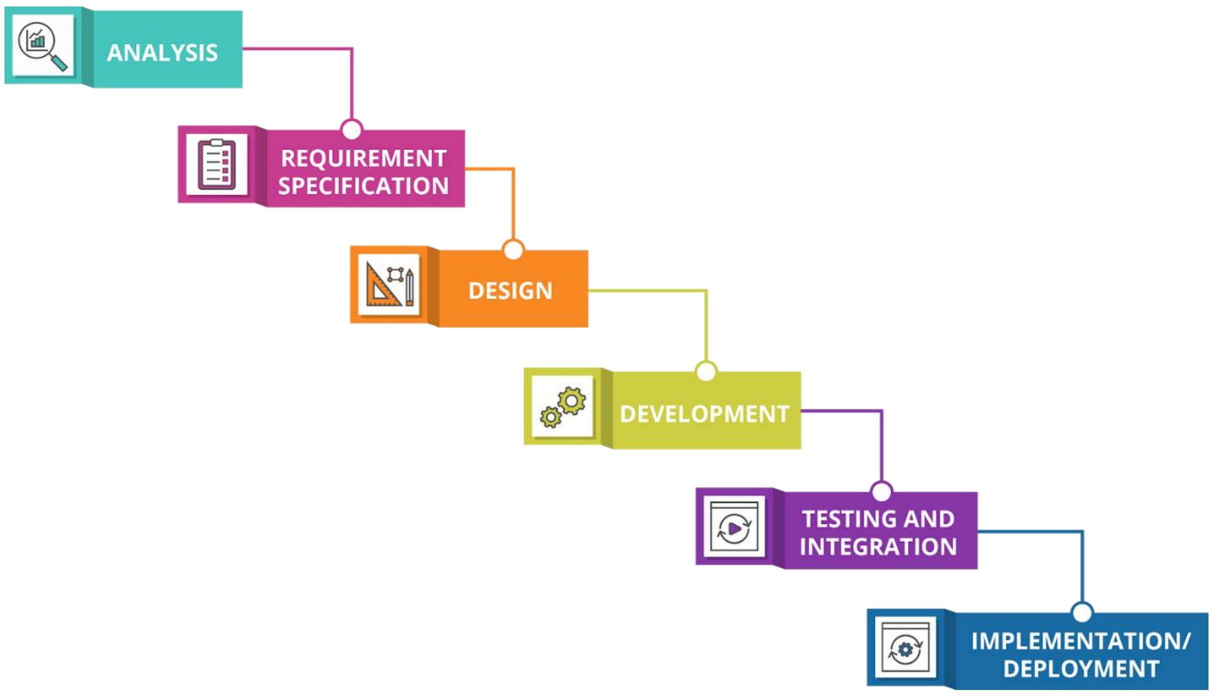
Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. Eclipse is written mostly in Java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Ada, ABAP, C, C++, C#, Clojure, COBOL, D, Erlang, Fortran, Groovy, Haskell, JavaScript, Julia, Lasso, Lua, NATURAL, Perl, PHP, Prolog, Python, R, Ruby (including Ruby on Rails framework), Rust, Scala, and Scheme. It can also be used to develop documents with LaTeX (via a TeXlipse plug-in) and packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

The initial codebase originated from IBM VisualAge. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Since the introduction of the OSGi implementation (Equinox) in version 3 of Eclipse, plug-ins can be plugged-stopped dynamically and are termed (OSGI) bundles.

Eclipse software development kit (SDK) is free and open-source software, released under the terms of the Eclipse Public License, although it is incompatible with the GNU General Public License.[10] It was one of the first IDEs to run under GNU Classpath and it runs without problems under IcedTea.

CHAPTER: 6
Detailed Life Cycle of Project

We have used Waterfall Model as Software Engineering life Cycle Process. It is the simplest; oldest and most widely used process model for software development .This model acquires its name from the fact that classic software life cycle is represented as a sequence of descending steps.



6.1 Requirement Analysis:

This process is also known as feasibility study. In this phase, the development team studied the site requirement. They investigate the need for possible dynamic representation of the site and increase security features. By the end of feasibility study, the team furnishes a document that holds the different specific recommendations for the candidate system. It also includes personnel assignments, costs, project schedules, target dates etc. the requirement gathering process is intensified and focused specially on software. The essential purpose of this phase is to find the need and to define the problem that needs to be solved. During this phase following facts were gathered.

❖ Determined the user need
❖ Identified the facts

- Establish the goals and objective for the proposed system
- Feasibility for the new system

6.2 System Analysis and Design:

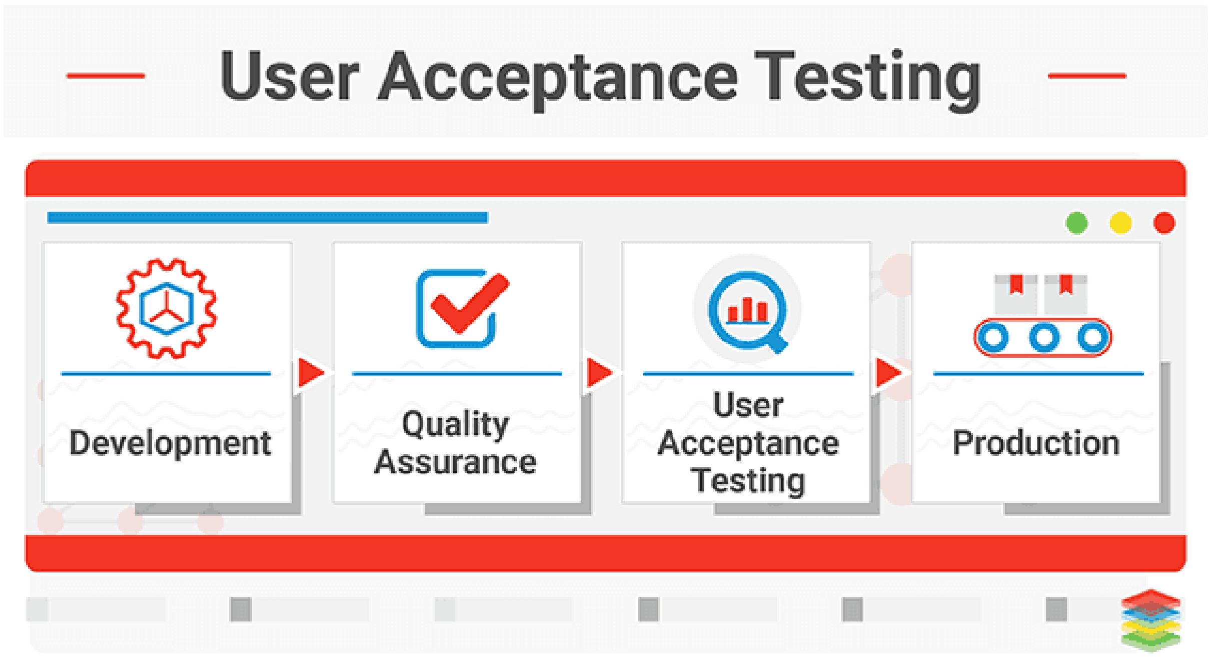
In this phase the software's overall structure and its nuances are defined. In terms of client server technology, the no of tiers needed for the package architecture, database design, data structure design etc are defined in this phase. Analysis and Design are very crucial in entire development cycle. Any glitch in this phase could be expensive to solve in the later stage of software development. Hence following is the essential approach taken during website designing:

- DFD
- Database Designing
- Form Designing
- Pseudo code for methods

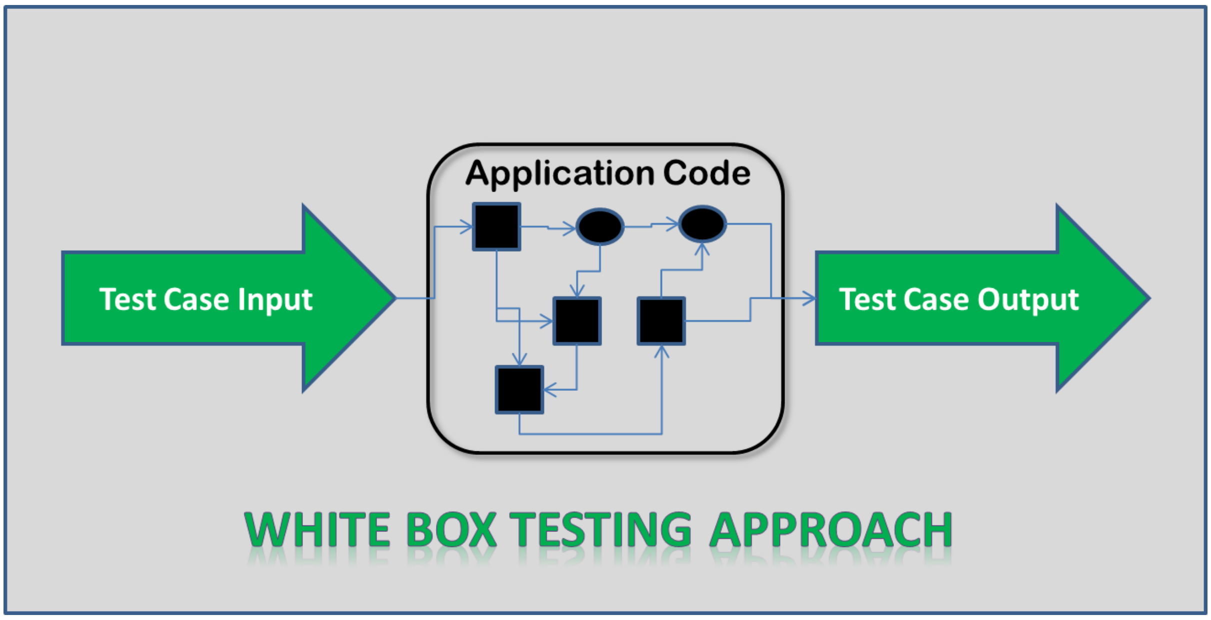
6.3 Testing:

Once the code is generated, the website testing begins. Different testing methodologies are done to unravel the bugs that were committed during the previous phases. Different testing methodologies are used:

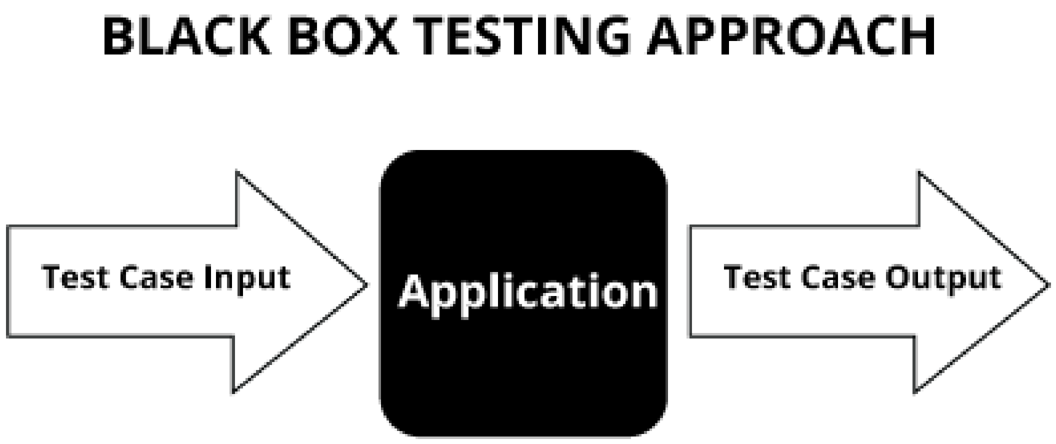
- Acceptance testing



- White Box Testing



- Black Box Testing



Chapter:7

Application Design

7.1 Form Designing

- **Login Form:**

This is login form of the Software, it ask for the id and password of the user, access is given to the authorized persons only. types of logins are there Admin only.

- **Login Page:**

It contains LOGIN, LOG OUT, and EXIT option to exit from software.

- **Registration:**

This module is designed for registration

- **Form list and details**

1. Registration Entry Form: registration entry form can be used to enter the new entry of student.
2. Student Search Form: student search form can be used by admin, for searching particular student related entering branches and year.
3. Notification Form: notification form can be used to add notification related to the exams and schedule of the college. Full authority is given to the admin.
4. Fees Form: this can be generated the fees report of the students.

7.2 Front End - Back End Connectivity

Java as Front end

1. Java offers several benefits to the developer creating front end application for database server.
2. Java is platform independent language hence can be executed on architecture that support on JVM.
3. Java also reduces cost related to deployment & maintenance of hardware & software.
4. Java base clients are think like that uses minimum hardware resources.
5. There are big incentive to create java based solution for corporate as shifting their application across the architecture will not involve overhead or cost.

MySQL-Back end

Databases are the systems that contain many different objects used together to facilitate fast and efficient access to the data.

MySQL is an application program interface form Microsoft that lets a programmer writing Windows applications; get access to a relational as well as non-relational database from both Microsoft and other database providers.

Setting up MySQL Control:

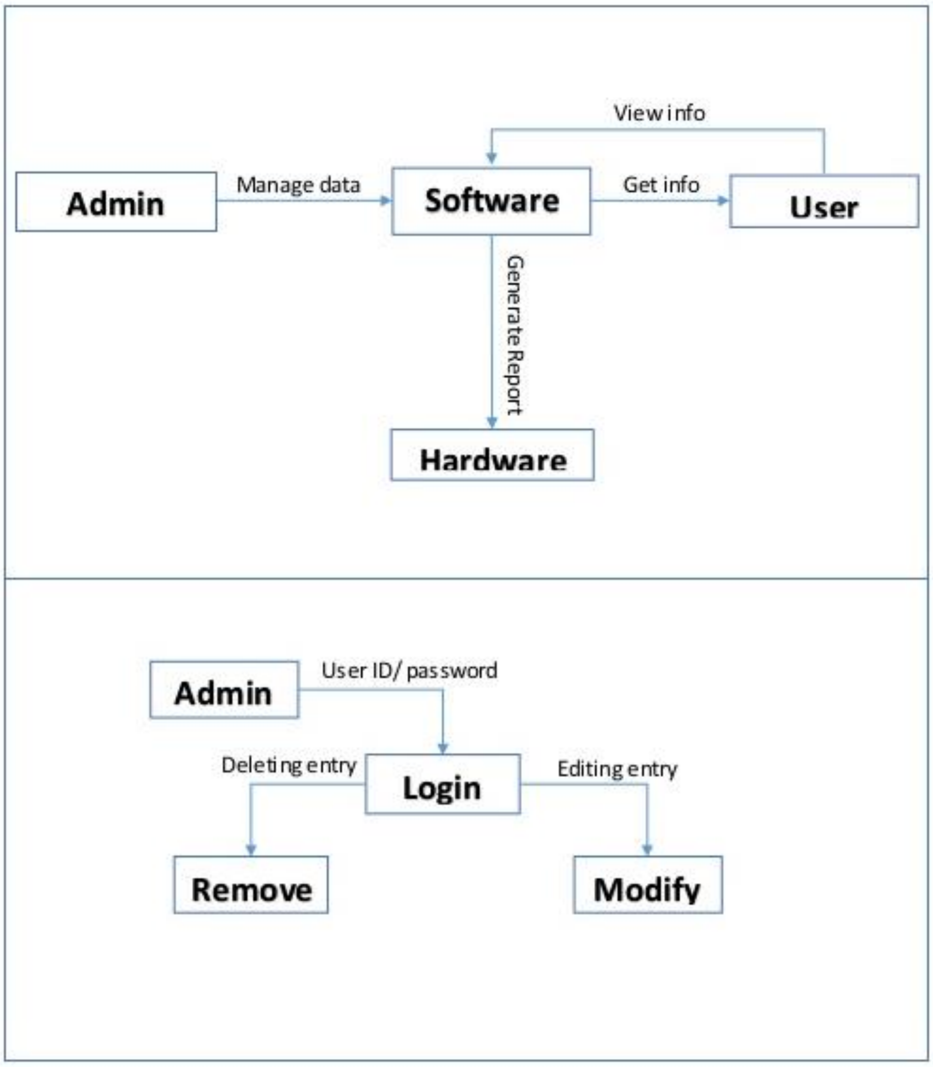
First step to use the MySQL in project is to add the MySQL Data Control to the form. The setting up of MySQL Data Control involves first few steps:

1. Connecting to a data source and,
2. Specifying a command to gain access to the data source.
3. Executing the command.
4. Storing the rows in a cache i.e., the Record set.

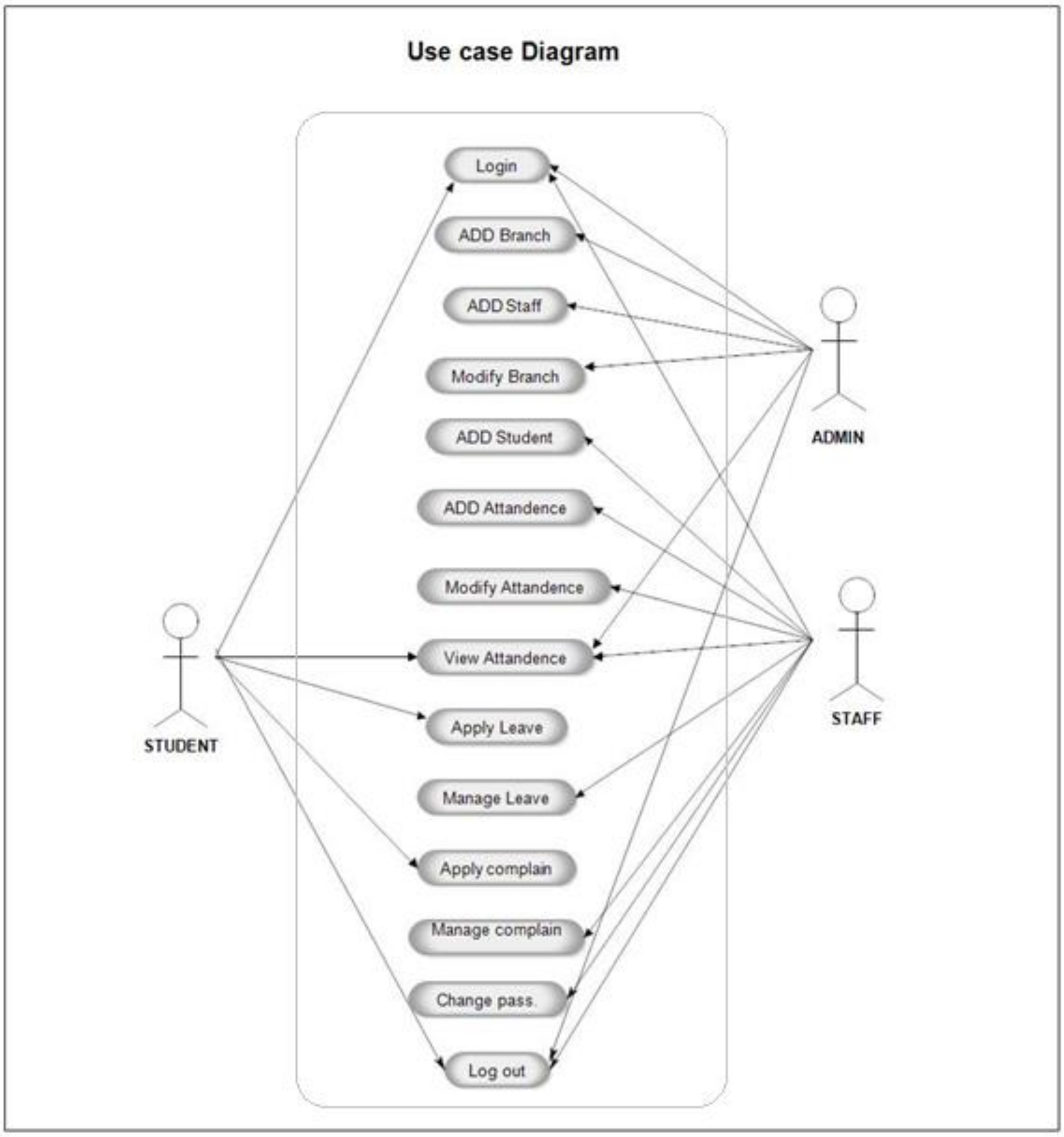
CHAPTER: 8

System Design

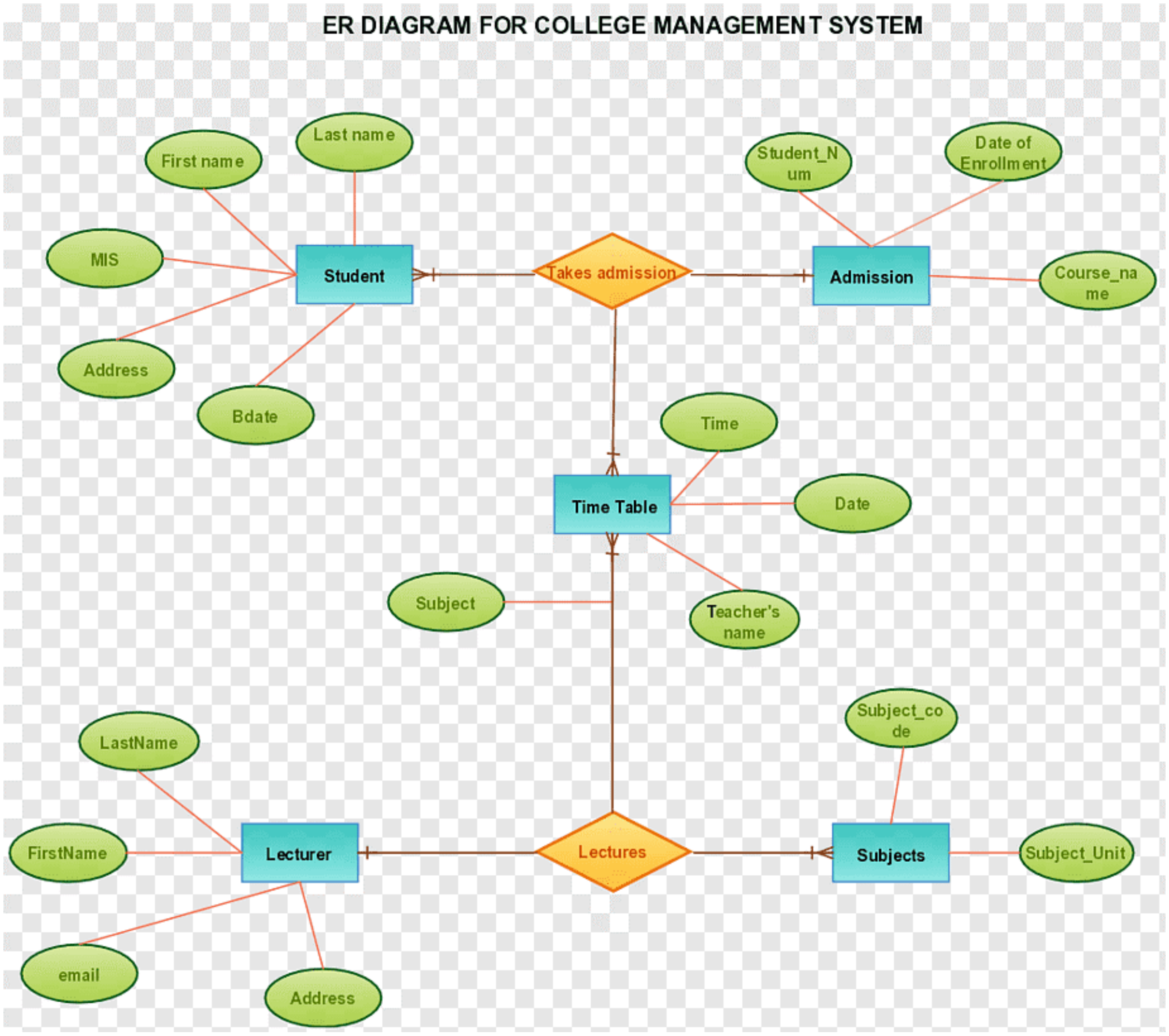
8.1 Architectural Design:

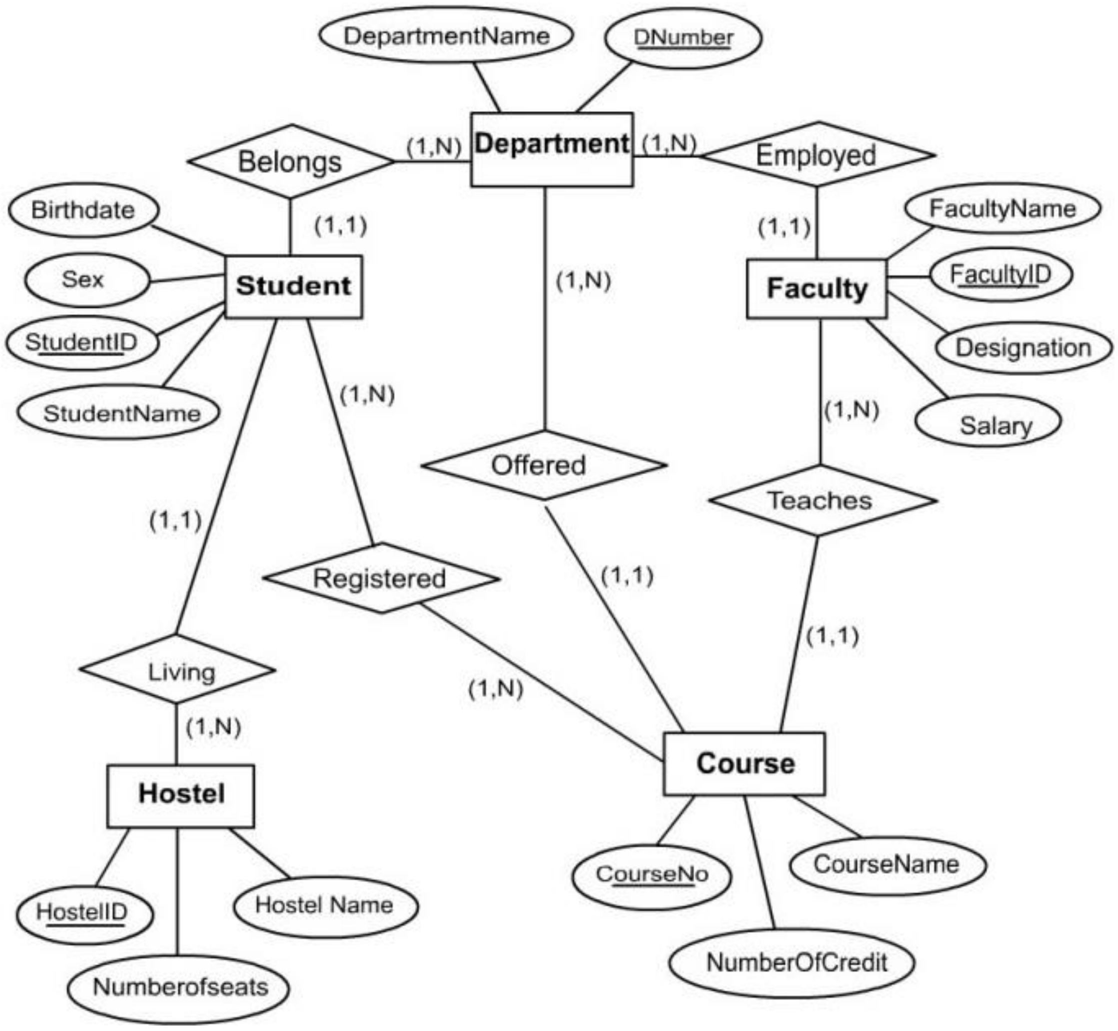


8.2 Use Case Diagram:



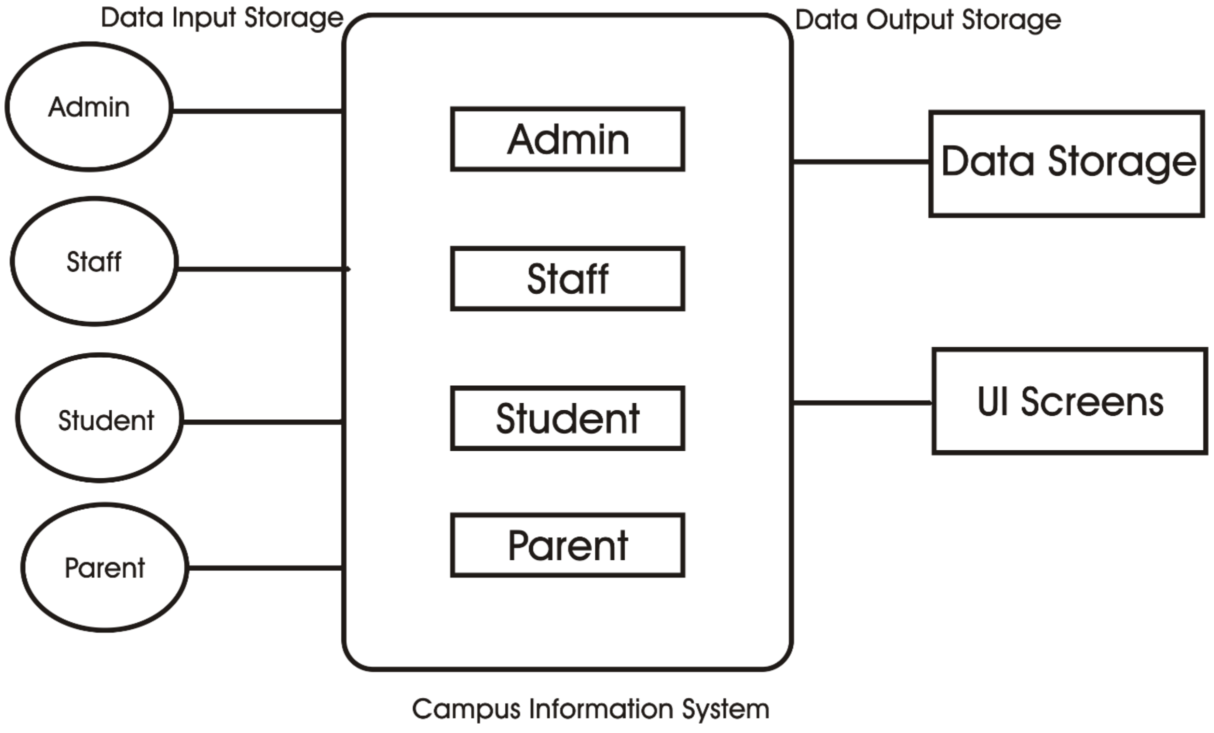
8.3 ER-Diagram:



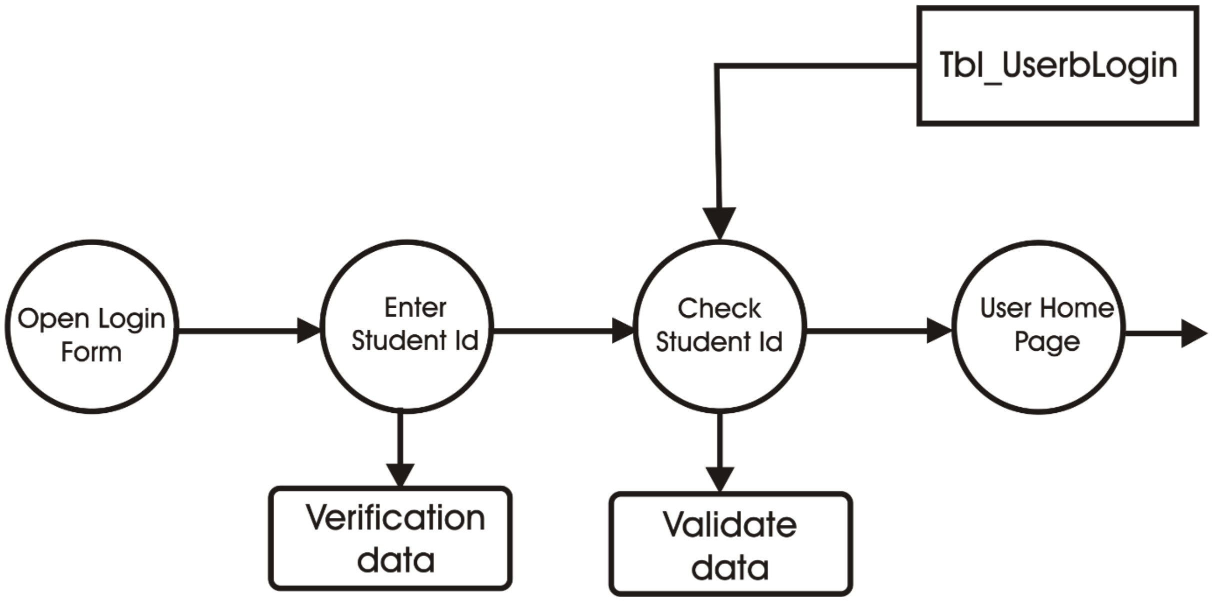

DepartmentName
DNumber
Department
Belongs
(1,N)
(1,N)
Employed
(1,1)
(1,1)
Birthdate
Sex
StudentID
StudentName
Student
FacultyName
FacultyID
Designation
Salary
Faculty
(1,N)
(1,N)
(1,N)
Offered
Teaches
(1,1)
Registered
(1,1)
(1,1)
Living
(1,N)
(1,N)
Hostel
HostelID
Hostel Name
Numberofseats
Course
CourseNo
CourseName
NumberOfCredit

8.4 Data Flow Diagram:

8.4.1 Context 0th Level Diagram:

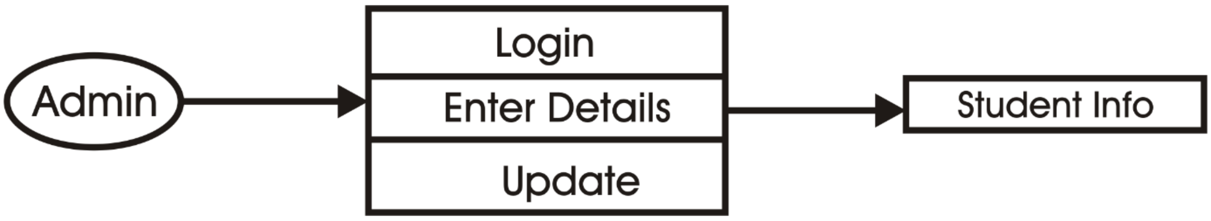


8.4.2 Login DFD Diagram:

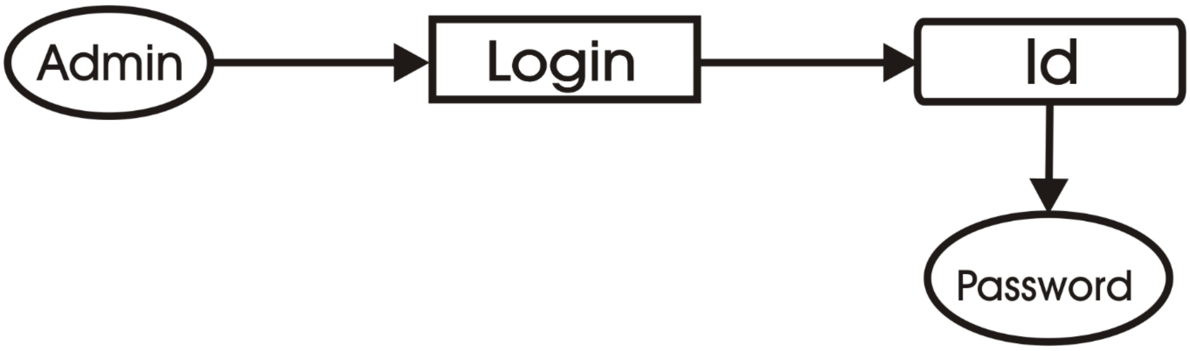


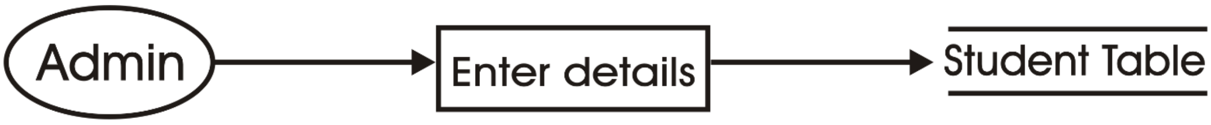
8.4.3 Admin Details Data Flow:

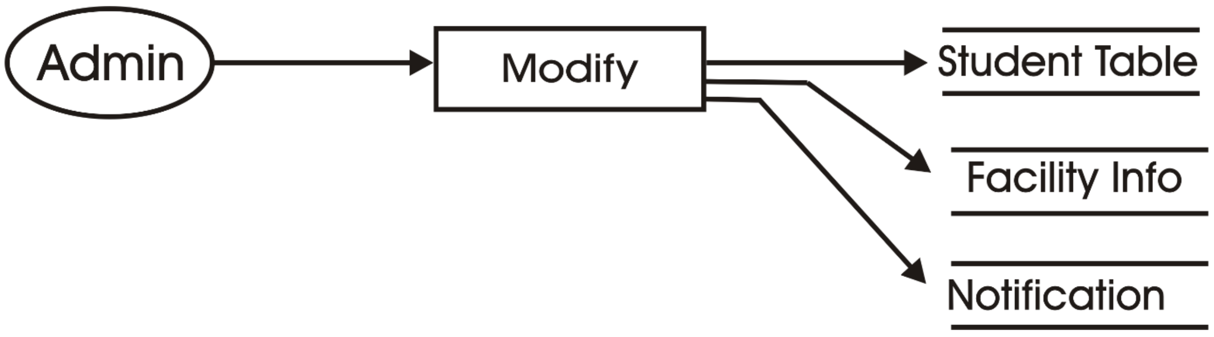
1st level DFD:



2nd Level DFD:

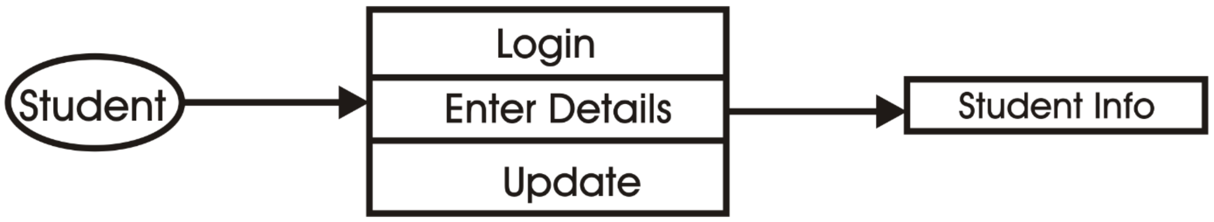




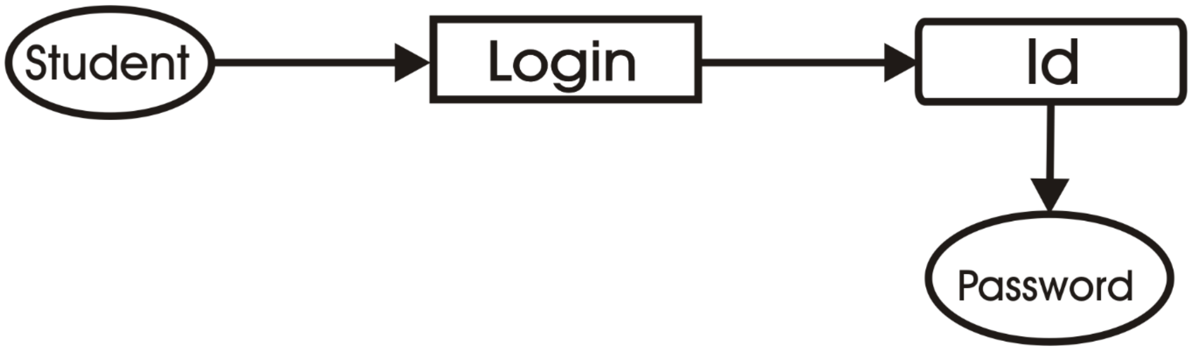


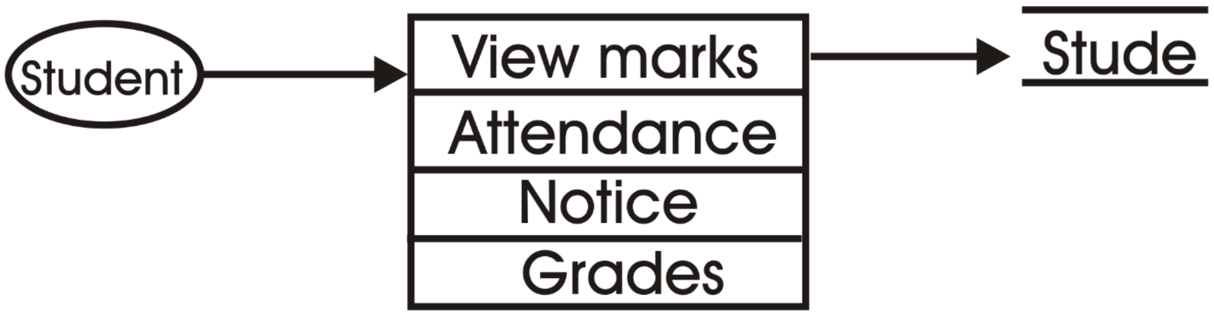
8.4.4 Student Details Data Flow:

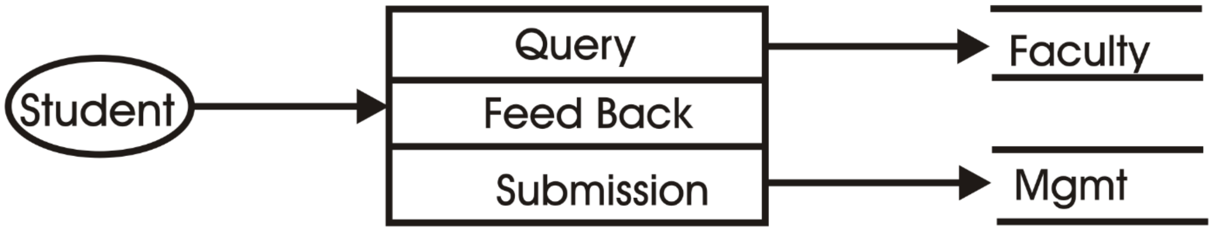
1st level DFD:



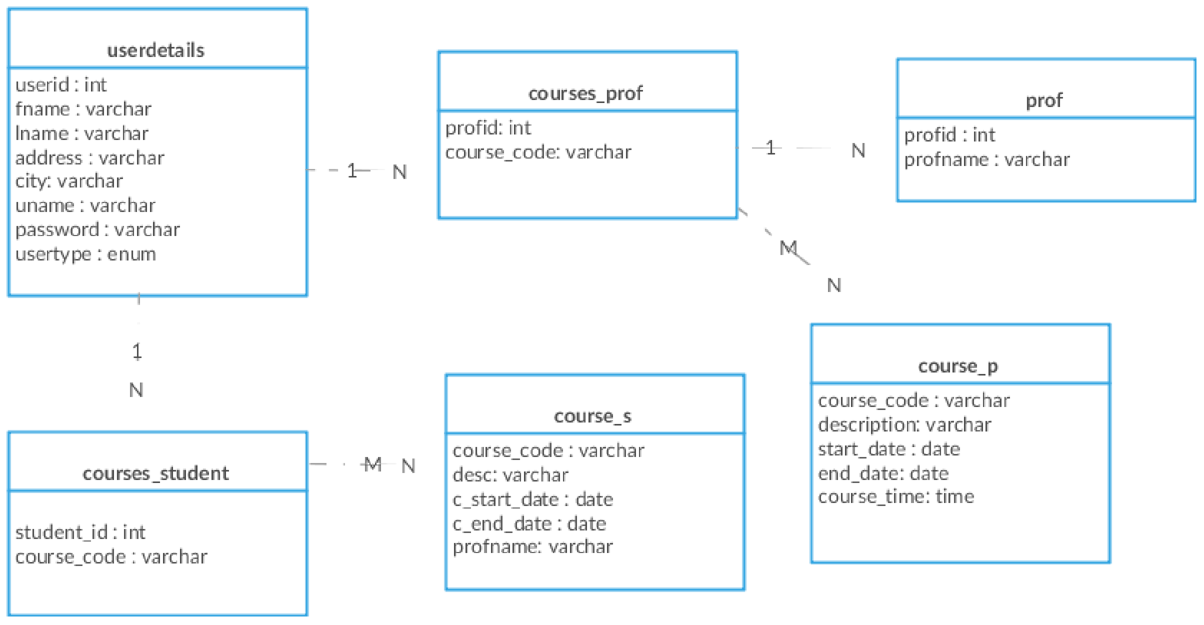
2nd level DFD:







8.5 Database Design:



Chapter: 9

Coding

AddSubject.java

```
package com.adminController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.SubjectModel;
import com.pojo.SubjectDetails;

@WebServlet("/AddSubject")
public class AddSubject extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

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

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

		SubjectDetails subjectDetails=new SubjectDetails();

		subjectDetails.setSubjectCode(subjectCode);
		subjectDetails.setSubjectName(subjectName);

		SubjectModel subjectModel=new SubjectModel();
		subjectModel.addSubject(subjectDetails);
			response.sendRedirect("AdminMainPage.jsp");

	}
	}
```

AddTeacher.java

```
package com.adminController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.TeacherModel;
import com.pojo.TeacherDetails;

@WebServlet("/AddTeacher")
```

```
public class AddTeacher extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		String teacherName=request.getParameter("teacherName");
		String password="123456789";
		String teacherEmailId=request.getParameter("teacherEmailId");
		String subject1=request.getParameter("subject1");
		String subject2=request.getParameter("subject2");
		String subject3=request.getParameter("subject3");

		TeacherDetails teacherDetails=new TeacherDetails();
		teacherDetails.setTeacherName(teacherName);
		teacherDetails.setPassword(password);
		teacherDetails.setEmailId(teacherEmailId);
		teacherDetails.setSubject1(subject1);
		teacherDetails.setSubject2(subject2);
		teacherDetails.setSubject3(subject3);

		TeacherModel teacherModel=new TeacherModel();
		teacherModel.addTeacher(teacherDetails);
			response.sendRedirect("AdminMainPage.jsp");

	}
}
```

AddTimetable.java

```
package com.adminController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.TimeTableModel;
import com.pojo.TimeTable;

@WebServlet("/AddTimetable")
public class AddTimetable extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

		String branch=request.getParameter("branch");
		String semId1=request.getParameter("semId");
		int semId=Integer.parseInt(semId1);

		String mon1=request.getParameter("mon1");
		String mon2=request.getParameter("mon2");
		String mon3=request.getParameter("mon3");
String mon4=request.getParameter("mon4");
String mon5=request.getParameter("mon5");

String tue1=request.getParameter("tue1");
String tue2=request.getParameter("tue2");
String tue3=request.getParameter("tue3");
String tue4=request.getParameter("tue4");
String tue5=request.getParameter("tue5");

String wed1=request.getParameter("wed1");
String wed2=request.getParameter("wed2");
String wed3=request.getParameter("wed3");
String wed4=request.getParameter("wed4");
String wed5=request.getParameter("wed5");

String thu1=request.getParameter("thu1");
String thu2=request.getParameter("thu2");
String thu3=request.getParameter("thu3");
String thu4=request.getParameter("thu4");
String thu5=request.getParameter("thu5");

String fri1=request.getParameter("fri1");
String fri2=request.getParameter("fri2");
String fri3=request.getParameter("fri3");
String fri4=request.getParameter("fri4");
String fri5=request.getParameter("fri5");

String sat1=request.getParameter("sat1");
String sat2=request.getParameter("sat2");
String sat3=request.getParameter("sat3");
String sat4=request.getParameter("sat4");
String sat5=request.getParameter("sat5");

TimeTable timetablePojo=new TimeTable();

timetablePojo.setBranch(branch);
timetablePojo.setSemId(semId);
timetablePojo.setMon1(mon1);
timetablePojo.setMon2(mon2);
timetablePojo.setMon3(mon3);
timetablePojo.setMon4(mon4);
timetablePojo.setMon5(mon5);

timetablePojo.setTue1(tue1);
timetablePojo.setTue2(tue2);
timetablePojo.setTue3(tue3);
timetablePojo.setTue4(tue4);
timetablePojo.setTue5(tue5);

timetablePojo.setWed1(wed1);
```

```
		timetablePojo.setWed2(wed2);
		timetablePojo.setWed3(wed3);
		timetablePojo.setWed4(wed4);
		timetablePojo.setWed5(wed5);

		timetablePojo.setThu1(thu1);
		timetablePojo.setThu2(thu2);
		timetablePojo.setThu3(thu3);
		timetablePojo.setThu4(thu4);
		timetablePojo.setThu5(thu5);

		timetablePojo.setFri1(fri1);
		timetablePojo.setFri2(fri2);
		timetablePojo.setFri3(fri3);
		timetablePojo.setFri4(fri4);
		timetablePojo.setFri5(fri5);

		timetablePojo.setSat1(sat1);
		timetablePojo.setSat2(sat2);
		timetablePojo.setSat3(sat3);
		timetablePojo.setSat4(sat4);
		timetablePojo.setSat5(sat5);

		TimeTableModel timeTableModel=new TimeTableModel();
		timeTableModel.addTimeTable(timetablePojo);

		response.sendRedirect("AdminMainPage.jsp");

	}

}
```

AdminChangePassword.java

```
 package com.adminController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/AdminChangePassword")
public class AdminChangePassword extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

	}
}
```

AdminLogin.java

```
package com.adminController;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/AdminLogin")
public class AdminLogin extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

                String adminId1=request.getParameter("adminId");
                int adminId=Integer.parseInt(adminId1);
                String adminName=request.getParameter("adminName");

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

                try {
                        Class.forName("com.mysql.jdbc.Driver");

                                Connection con=
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem",
"root", "root");
                           String sql="select * from Admin where id=?";
                                PreparedStatement ps=con.prepareStatement(sql);

                                ps.setInt(1, 1);
                           ResultSet rs=ps.executeQuery();
                          if(rs!=null)
                          {
                                  while(rs.next()) {
                                  if(rs.getString("name").equalsIgnoreCase(adminName) &&
rs.getString("password").equalsIgnoreCase(adminPassword)){
                                          response.sendRedirect("AdminMainPage.jsp");
                                  }
                                  }
                                          }else
                          {
                                  response.sendRedirect("AdminLogin.jsp");
                          }
```

```
                                        } catch (ClassNotFoundException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (SQLException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        }
}
```

NewStudentAdmission .java

```
package com.adminController;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.AdmissionDetailsModel;
import com.pojo.AdmissionDetails;

@WebServlet("/NewStudentAdmission")
public class NewStudentAdmission extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

                String studentName=request.getParameter("studentName");
                String semId1=request.getParameter("semId");
                int semId=Integer.parseInt(semId1);
                String branch=request.getParameter("branch");
                String semTotalFee1=request.getParameter("semTotalFee");
                int semTotalFee=Integer.parseInt(semTotalFee1);
                String studentPaid1=request.getParameter("studentPaid");
                int studentPaid=Integer.parseInt(studentPaid1);
                String balance1=request.getParameter("balance");
                int balance=Integer.parseInt(balance1);
                String studentEmail=request.getParameter("studentEmail");
                String studentDateOfBirth=request.getParameter("studentDateOfBirth");
                String student10thPercentage=request.getParameter("student10thPercentage");
                String student12thPercentage=request.getParameter("student12thPercentage");
                String studentDiplomo=request.getParameter("studentDiplomo");

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

                Date date = new Date();
        SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yyyy HH:mm:ss");
                System.out.println(formatter.format(date));
                AdmissionDetails admissionDetails=new AdmissionDetails();
                admissionDetails.setStudentName(studentName);
                admissionDetails.setAadharNumber(studentAadharNumber);
                admissionDetails.setSemesterJoining(semId);
                admissionDetails.setPassword("123456789");
                admissionDetails.setBranch(branch);
                admissionDetails.setDate(formatter.format(date));
                admissionDetails.setEmail(studentEmail);
                admissionDetails.setDateOfBirth(studentDateOfBirth);
                admissionDetails.setPercentageIn10th(student10thPercentage);
                admissionDetails.setVerified(0);

                if(semId==3) {
                        admissionDetails.setTotalFeesSem3(semTotalFee);
                        admissionDetails.setBalanceFeesSem3(balance);
                        admissionDetails.setPaidFeesSem3(studentPaid);
                        admissionDetails.setLateralEntry(1);
                }else {
                        admissionDetails.setTotalFeesSem1(semTotalFee);
                        admissionDetails.setBalanceFeesSem1(balance);
                        admissionDetails.setPaidFeesSem1(studentPaid);
                }

        AdmissionDetailsModel admissionDetailsModel=new AdmissionDetailsModel();
                boolean action=admissionDetailsModel.addNewStudent(admissionDetails);
                if(action==true) {
                        response.sendRedirect("AdminMainPage.jsp");
                }else {
                        response.sendRedirect("newStudentAdmission.jsp");
                }
        }
}
```

OldStudentAdmission .java

```
package com.adminController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.AdmissionDetailsModel;
import com.pojo.AdmissionDetails;

@WebServlet("/OldStudentAdmission")
public class OldStudentAdmission extends HttpServlet {

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		String studentAdmissionId1=request.getParameter("studentAdmissionId");
		int studentAdmissionId=Integer.parseInt(studentAdmissionId1);
		String semId1=request.getParameter("semId");
		int semId=Integer.parseInt(semId1);
		String totalSemFee1=request.getParameter("totalSemFee");
		int totalSemFee=Integer.parseInt(totalSemFee1);
		String paidSemFee1=request.getParameter("paidSemFee");
		int paidSemFee=Integer.parseInt(paidSemFee1);
		String balanceSemFee1=request.getParameter("balanceSemFee");
		int balanceSemFee=Integer.parseInt(balanceSemFee1);

		AdmissionDetails admissionDetailspojo=new AdmissionDetails();
		admissionDetailspojo.setAdmissionId(studentAdmissionId);

AdmissionDetailsModel admissionDetailsModel=new AdmissionDetailsModel();

		boolean action=false;
		if(semId==1) {
			admissionDetailspojo.setTotalFeesSem1(totalSemFee);
			admissionDetailspojo.setPaidFeesSem1(paidSemFee);
			admissionDetailspojo.setBalanceFeesSem1(balanceSemFee);

action=admissionDetailsModel.updateStudentAdmissionSem1(admissionDetailspojo);
		}else if(semId==2) {
			admissionDetailspojo.setTotalFeesSem2(totalSemFee);
			admissionDetailspojo.setPaidFeesSem2(paidSemFee);
			admissionDetailspojo.setBalanceFeesSem2(balanceSemFee);

action=admissionDetailsModel.updateStudentAdmissionSem2(admissionDetailspojo);
		}else if(semId==3) {
			admissionDetailspojo.setTotalFeesSem3(totalSemFee);
			admissionDetailspojo.setPaidFeesSem3(paidSemFee);
			admissionDetailspojo.setBalanceFeesSem3(balanceSemFee);

action=admissionDetailsModel.updateStudentAdmissionSem3(admissionDetailspojo);
		}else if(semId==4) {
			admissionDetailspojo.setTotalFeesSem4(totalSemFee);
			admissionDetailspojo.setPaidFeesSem4(paidSemFee);
			admissionDetailspojo.setBalanceFeesSem4(balanceSemFee);

action=admissionDetailsModel.updateStudentAdmissionSem4(admissionDetailspojo);
		}else if(semId==5) {
			admissionDetailspojo.setTotalFeesSem5(totalSemFee);
			admissionDetailspojo.setPaidFeesSem5(paidSemFee);
			admissionDetailspojo.setBalanceFeesSem5(balanceSemFee);

action=admissionDetailsModel.updateStudentAdmissionSem5(admissionDetailspojo);
		}else if(semId==6) {
			admissionDetailspojo.setTotalFeesSem6(totalSemFee);
			admissionDetailspojo.setPaidFeesSem6(paidSemFee);
```

```
                              admissionDetailspojo.setBalanceFeesSem6(balanceSemFee);

action=admissionDetailsModel.updateStudentAdmissionSem6(admissionDetailspojo);
                    }
                                        response.sendRedirect("AdminMainPage.jsp");
          }
}
```

UpdateStudentDetails.java

```
package com.adminController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.AdmissionDetailsModel;
import com.pojo.AdmissionDetails;

@WebServlet("/UpdateStudentDetails")
public class UpdateStudentDetails extends HttpServlet {
                    /* (non-Javadoc)
           * @see javax.servlet.http.HttpServlet#doGet(javax.servlet.http.HttpServletRequest,
javax.servlet.http.HttpServletResponse)
           */
          protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                    String admissionId1=request.getParameter("admissionId");
                    int admissionId=Integer.parseInt(admissionId1);
                    String studentName=request.getParameter("studentName");
                    String semesterJoining1=request.getParameter("semesterJoining");
                    int semesterJoining=Integer.parseInt(semesterJoining1);
                    String branch=request.getParameter("branch");
                    String date=request.getParameter("date");
                    String totalFeesSem11=request.getParameter("totalFeesSem1");
                    int totalFeesSem1=Integer.parseInt(totalFeesSem11);
                    String paidFeesSem11=request.getParameter("paidFeesSem1");
                    int paidFeesSem1=Integer.parseInt(paidFeesSem11);
                    String balanceFeesSem11=request.getParameter("balanceFeesSem1");
                    int balanceFeesSem1=Integer.parseInt(balanceFeesSem11);
                    String totalFeesSem21=request.getParameter("totalFeesSem2");
                    int totalFeesSem2=Integer.parseInt(totalFeesSem21);
                    String paidFeesSem21=request.getParameter("paidFeesSem2");
                    int paidFeesSem2=Integer.parseInt(paidFeesSem21);
                    String balanceFeesSem21=request.getParameter("balanceFeesSem2");
                    int balanceFeesSem2=Integer.parseInt(balanceFeesSem21);
                    String totalFeesSem31=request.getParameter("totalFeesSem3");
                    int totalFeesSem3=Integer.parseInt(totalFeesSem31);
                    String paidFeesSem31=request.getParameter("paidFeesSem3");
                    int paidFeesSem3=Integer.parseInt(paidFeesSem31);
```

```
String balanceFeesSem31=request.getParameter("balanceFeesSem3");
int balanceFeesSem3=Integer.parseInt(balanceFeesSem31);
String totalFeesSem41=request.getParameter("totalFeesSem4");
int totalFeesSem4=Integer.parseInt(totalFeesSem41);
String paidFeesSem41=request.getParameter("paidFeesSem4");
int paidFeesSem4=Integer.parseInt(paidFeesSem41);
String balanceFeesSem41=request.getParameter("balanceFeesSem4");
int balanceFeesSem4=Integer.parseInt(balanceFeesSem41);
String totalFeesSem51=request.getParameter("totalFeesSem5");
int totalFeesSem5=Integer.parseInt(totalFeesSem51);
String paidFeesSem51=request.getParameter("paidFeesSem5");
int paidFeesSem5=Integer.parseInt(paidFeesSem51);
String balanceFeesSem51=request.getParameter("balanceFeesSem5");
int balanceFeesSem5=Integer.parseInt(balanceFeesSem51);
String totalFeesSem61=request.getParameter("totalFeesSem6");
int totalFeesSem6=Integer.parseInt(totalFeesSem61);
String paidFeesSem61=request.getParameter("paidFeesSem6");
int paidFeesSem6=Integer.parseInt(paidFeesSem61);
String balanceFeesSem61=request.getParameter("balanceFeesSem6");
int balanceFeesSem6=Integer.parseInt(balanceFeesSem61);
String dateOfBirth=request.getParameter("dateOfBirth");
String percentageIn10th=request.getParameter("percentageIn10th");
String percentageIn12th=request.getParameter("percentageIn12th");
String percentageInDip=request.getParameter("percentageInDip");
String verified1=request.getParameter("verified");
int verified=Integer.parseInt(verified1);

AdmissionDetails admissionDetails=new AdmissionDetails();
admissionDetails.setAdmissionId(admissionId);
admissionDetails.setStudentName(studentName);
admissionDetails.setSemesterJoining(semesterJoining);
admissionDetails.setBranch(branch);
admissionDetails.setDate(date);

admissionDetails.setTotalFeesSem1(totalFeesSem1);
admissionDetails.setPaidFeesSem1(paidFeesSem1);
admissionDetails.setBalanceFeesSem1(balanceFeesSem1);

admissionDetails.setTotalFeesSem2(totalFeesSem2);
admissionDetails.setPaidFeesSem2(paidFeesSem2);
admissionDetails.setBalanceFeesSem2(balanceFeesSem2);

admissionDetails.setTotalFeesSem3(totalFeesSem3);
admissionDetails.setPaidFeesSem3(paidFeesSem3);
admissionDetails.setBalanceFeesSem3(balanceFeesSem3);

admissionDetails.setTotalFeesSem4(totalFeesSem4);
admissionDetails.setPaidFeesSem4(paidFeesSem4);
admissionDetails.setBalanceFeesSem4(balanceFeesSem4);
```

```
                admissionDetails.setTotalFeesSem5(totalFeesSem5);
                admissionDetails.setPaidFeesSem5(paidFeesSem5);
                admissionDetails.setBalanceFeesSem5(balanceFeesSem5);

                admissionDetails.setTotalFeesSem6(totalFeesSem6);
                admissionDetails.setPaidFeesSem6(paidFeesSem6);
                admissionDetails.setBalanceFeesSem6(balanceFeesSem6);

                admissionDetails.setDateOfBirth(dateOfBirth);
                admissionDetails.setPercentageIn10th(percentageIn10th);
                admissionDetails.setPercentageIn12th(percentageIn12th);
                admissionDetails.setPercentageInDip(percentageInDip);

                admissionDetails.setVerified(verified);

        AdmissionDetailsModel admissionDetailsModel=new AdmissionDetailsModel();
                boolean action=admissionDetailsModel.UpdateStudent(admissionDetails);
                                response.sendRedirect("AdminMainPage.jsp");

        }
}
```

com.model
AdmissionDetailsModel.java

```
package com.model;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;
import com.pojo.AdmissionDetails;
public class AdmissionDetailsModel {

        public boolean addNewStudent(AdmissionDetails admissionDetails) {
                AnnotationConfiguration cfg = new AnnotationConfiguration();
                cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
                SessionFactory sf = cfg.buildSessionFactory();
                Session session = sf.openSession();
                Transaction tx = session.beginTransaction();

                session.persist(admissionDetails);
                tx.commit();
                return true;
        }

        public boolean updateStudentAdmissionSem1(AdmissionDetails admissionDetails) {
                AnnotationConfiguration cfg = new AnnotationConfiguration();
                cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
                SessionFactory sf = cfg.buildSessionFactory();
                Session session = sf.openSession();
```

```
		Transaction tx = session.beginTransaction();
		Query q = session.createQuery(
				"update AdmissionDetails set
totalFeesSem1=:totalFeesSem1,paidFeesSem1=:paidFeesSem1,balanceFeesSem1=:balanceF
eesSem1 where admissionId=:admissionId");
		q.setParameter("totalFeesSem1", admissionDetails.getTotalFeesSem1());
		q.setParameter("paidFeesSem1", admissionDetails.getPaidFeesSem1());
		q.setParameter("balanceFeesSem1",
admissionDetails.getBalanceFeesSem1());
		q.setParameter("admissionId", admissionDetails.getAdmissionId());
		int status = q.executeUpdate();
		System.out.println(status);
		tx.commit();
		return true;
	}

	public boolean updateStudentAdmissionSem2(AdmissionDetails admissionDetails) {
		AnnotationConfiguration cfg = new AnnotationConfiguration();
		cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
		SessionFactory sf = cfg.buildSessionFactory();
		Session session = sf.openSession();
		Transaction tx = session.beginTransaction();
		Query q = session.createQuery(
				"update AdmissionDetails set
totalFeesSem2=:totalFeesSem2,paidFeesSem2=:paidFeesSem2,balanceFeesSem2=:balanceF
eesSem2 where admissionId=:admissionId");
		q.setParameter("totalFeesSem2", admissionDetails.getTotalFeesSem2());
		q.setParameter("paidFeesSem2", admissionDetails.getPaidFeesSem2());
		q.setParameter("balanceFeesSem2",
admissionDetails.getBalanceFeesSem2());
		q.setParameter("admissionId", admissionDetails.getAdmissionId());
		int status = q.executeUpdate();
		System.out.println(status);
		tx.commit();
		return true;
	}

	public boolean updateStudentAdmissionSem3(AdmissionDetails admissionDetails) {
		AnnotationConfiguration cfg = new AnnotationConfiguration();
		cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
		SessionFactory sf = cfg.buildSessionFactory();
		Session session = sf.openSession();
		Transaction tx = session.beginTransaction();
		Query q = session.createQuery(
				"update AdmissionDetails set
totalFeesSem3=:totalFeesSem3,paidFeesSem3=:paidFeesSem3,balanceFeesSem3=:balanceF
eesSem3 where admissionId=:admissionId");
		q.setParameter("totalFeesSem3", admissionDetails.getTotalFeesSem3());
		q.setParameter("paidFeesSem3", admissionDetails.getPaidFeesSem3());
		q.setParameter("balanceFeesSem3",
admissionDetails.getBalanceFeesSem3());
		q.setParameter("admissionId", admissionDetails.getAdmissionId());

		int status = q.executeUpdate();
		System.out.println(status);
		tx.commit();
		return true;
	}

	public boolean updateStudentAdmissionSem4(AdmissionDetails admissionDetails) {

		AnnotationConfiguration cfg = new AnnotationConfiguration();
		cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
		SessionFactory sf = cfg.buildSessionFactory();
		Session session = sf.openSession();
		Transaction tx = session.beginTransaction();
		Query q = session.createQuery(
				"update AdmissionDetails set
totalFeesSem4=:totalFeesSem4,paidFeesSem4=:paidFeesSem4,balanceFeesSem4=:balanceF
eesSem4 where admissionId=:admissionId");
		q.setParameter("totalFeesSem4", admissionDetails.getTotalFeesSem4());
		q.setParameter("paidFeesSem4", admissionDetails.getPaidFeesSem4());
		q.setParameter("balanceFeesSem4",
admissionDetails.getBalanceFeesSem4());
		q.setParameter("admissionId", admissionDetails.getAdmissionId());
		int status = q.executeUpdate();
		System.out.println(status);
		tx.commit();

		return true;
	}

	public boolean updateStudentAdmissionSem5(AdmissionDetails admissionDetails) {

		AnnotationConfiguration cfg = new AnnotationConfiguration();
		cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
		SessionFactory sf = cfg.buildSessionFactory();
		Session session = sf.openSession();
		Transaction tx = session.beginTransaction();
		Query q = session.createQuery(
				"update AdmissionDetails set
totalFeesSem5=:totalFeesSem5,paidFeesSem5=:paidFeesSem5,balanceFeesSem5=:balanceF
eesSem5 where admissionId=:admissionId");
		q.setParameter("totalFeesSem5", admissionDetails.getTotalFeesSem5());
		q.setParameter("paidFeesSem5", admissionDetails.getPaidFeesSem5());
		q.setParameter("balanceFeesSem5",
admissionDetails.getBalanceFeesSem5());
		q.setParameter("admissionId", admissionDetails.getAdmissionId());
		int status = q.executeUpdate();
```

```
                System.out.println(status);
                tx.commit();
                return true;
        }

        public boolean updateStudentAdmissionSem6(AdmissionDetails admissionDetails) {
                AnnotationConfiguration cfg = new AnnotationConfiguration();
                cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
                SessionFactory sf = cfg.buildSessionFactory();
                Session session = sf.openSession();
                Transaction tx = session.beginTransaction();
                Query q = session.createQuery(
                                "update AdmissionDetails set
totalFeesSem6=:totalFeesSem6,paidFeesSem6=:paidFeesSem6,balanceFeesSem6=:balanceF
eesSem6 where admissionId=:admissionId");
                q.setParameter("totalFeesSem6", admissionDetails.getTotalFeesSem6());
                q.setParameter("paidFeesSem6", admissionDetails.getPaidFeesSem6());
                q.setParameter("balanceFeesSem6",
admissionDetails.getBalanceFeesSem6());
                q.setParameter("admissionId", admissionDetails.getAdmissionId());
                int status = q.executeUpdate();
                System.out.println(status);
                tx.commit();
                return true;
        }

        public boolean UpdateStudent(AdmissionDetails admissionDetails) {
                AnnotationConfiguration cfg = new AnnotationConfiguration();
                cfg.configure("com/configureFiles/AdmissionDetails.cfg.xml");
                SessionFactory sf = cfg.buildSessionFactory();
                Session session = sf.openSession();
                Transaction tx = session.beginTransaction();
                Query q = session.createQuery(
                                "update AdmissionDetails set
verified=:verified,totalFeesSem1=:totalFeesSem1,paidFeesSem1=:paidFeesSem1,balanceFee
sSem1=:balanceFeesSem1,totalFeesSem2=:totalFeesSem2,paidFeesSem2=:paidFeesSem2,ba
lanceFeesSem2=:balanceFeesSem2,totalFeesSem3=:totalFeesSem3,paidFeesSem3=:paidFees
Sem3,balanceFeesSem3=:balanceFeesSem3,totalFeesSem4=:totalFeesSem4,paidFeesSem4=:
paidFeesSem4,balanceFeesSem4=:balanceFeesSem4,totalFeesSem5=:totalFeesSem5,paidFee
sSem5=:paidFeesSem5,balanceFeesSem5=:balanceFeesSem5,totalFeesSem6=:totalFeesSem
6,paidFeesSem6=:paidFeesSem6,balanceFeesSem6=:balanceFeesSem6 where
admissionId=:admissionId");

                q.setParameter("verified", admissionDetails.getVerified());

                q.setParameter("totalFeesSem1", admissionDetails.getTotalFeesSem1());
                q.setParameter("totalFeesSem1", admissionDetails.getTotalFeesSem1());
                q.setParameter("totalFeesSem1", admissionDetails.getTotalFeesSem1());

                q.setParameter("totalFeesSem1", admissionDetails.getTotalFeesSem1());
```

```
			q.setParameter("paidFeesSem1", admissionDetails.getPaidFeesSem1());
			q.setParameter("balanceFeesSem1",
admissionDetails.getBalanceFeesSem1());

			q.setParameter("totalFeesSem2", admissionDetails.getTotalFeesSem2());
			q.setParameter("paidFeesSem2", admissionDetails.getPaidFeesSem2());
			q.setParameter("balanceFeesSem2",
admissionDetails.getBalanceFeesSem2());

			q.setParameter("totalFeesSem3", admissionDetails.getTotalFeesSem3());
			q.setParameter("paidFeesSem3", admissionDetails.getPaidFeesSem3());
			q.setParameter("balanceFeesSem3",
admissionDetails.getBalanceFeesSem3());

			q.setParameter("totalFeesSem4", admissionDetails.getTotalFeesSem4());
			q.setParameter("paidFeesSem4", admissionDetails.getPaidFeesSem4());
			q.setParameter("balanceFeesSem4",
admissionDetails.getBalanceFeesSem4());

			q.setParameter("totalFeesSem5", admissionDetails.getTotalFeesSem5());
			q.setParameter("paidFeesSem5", admissionDetails.getPaidFeesSem5());
			q.setParameter("balanceFeesSem5",
admissionDetails.getBalanceFeesSem5());

			q.setParameter("totalFeesSem6", admissionDetails.getTotalFeesSem6());
			q.setParameter("paidFeesSem6", admissionDetails.getPaidFeesSem6());
			q.setParameter("balanceFeesSem6",
admissionDetails.getBalanceFeesSem6());

			q.setParameter("admissionId", admissionDetails.getAdmissionId());

			int status = q.executeUpdate();
			System.out.println(status);
			tx.commit();
			return true;
		}
}
```

ResultModel.java

```
package com.model;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;
import com.pojo.ResultDetails;
import com.pojo.SubjectDetails;

public class ResultModel {
public boolean addResult(ResultDetails resultDetails) {
				AnnotationConfiguration cfg=new AnnotationConfiguration();
```

```
                    cfg.configure("com/configureFiles/ResultDetails.cfg.xml");
                    SessionFactory sf=cfg.buildSessionFactory();
                    Session session=sf.openSession();
                    Transaction tx=session.beginTransaction();
                              session.persist(resultDetails);
                    tx.commit();
                    return true;
          }
}
```

SubjectModel.java

```
package com.model;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;
import com.pojo.SubjectDetails;
import com.pojo.TeacherDetails;

public class SubjectModel {
public boolean addSubject(SubjectDetails subjectDetails) {
                                        AnnotationConfiguration cfg=new AnnotationConfiguration();
                    cfg.configure("com/configureFiles/SubjectDetails.cfg.xml");
                    SessionFactory sf=cfg.buildSessionFactory();
                    Session session=sf.openSession();
                    Transaction tx=session.beginTransaction();
                              session.persist(subjectDetails);
                    tx.commit();
                    return true;
          }
}
```

TeachertModel.java

```
package com.model;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;
import com.pojo.TeacherDetails;

public class TeacherModel {
public boolean addTeacher(TeacherDetails teacherDetails) {
                    AnnotationConfiguration cfg=new AnnotationConfiguration();
                    cfg.configure("com/configureFiles/Teacher.cfg.xml");
                    SessionFactory sf=cfg.buildSessionFactory();
                    Session session=sf.openSession();
                    Transaction tx=session.beginTransaction();
                    session.persist(teacherDetails);
                    tx.commit();
                    return true;
```

```
	}
}
```

TimetabelModel.java

```
package com.model;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.AnnotationConfiguration;
import com.pojo.TeacherDetails;
import com.pojo.TimeTable;

public class TimeTableModel {
public boolean addTimeTable(TimeTable timeTable) {
			AnnotationConfiguration cfg=new AnnotationConfiguration();
			cfg.configure("com/configureFiles/TimeTable.cfg.xml");
			SessionFactory sf=cfg.buildSessionFactory();
			Session session=sf.openSession();
			Transaction tx=session.beginTransaction();
			session.persist(timeTable);
			tx.commit();
			return true;
	}
}
```

com.pojo
AdmissionDetails.java

```
package com.pojo;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table
public class AdmissionDetails {
	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	int admissionId;

	String studentName;

	int semesterJoining;

	String branch;

	String date;

	int totalFeesSem1;
int paidFeesSem1;

int balanceFeesSem1;

int totalFeesSem2;

int paidFeesSem2;

int balanceFeesSem2;

int totalFeesSem3;

int paidFeesSem3;

int balanceFeesSem3;

int totalFeesSem4;

int paidFeesSem4;

int balanceFeesSem4;

int totalFeesSem5;

int paidFeesSem5;

int balanceFeesSem5;

int totalFeesSem6;

int paidFeesSem6;

int balanceFeesSem6;

int feesRemainingStatus;

String password;

String usnNumber;

String Email;

String dateOfBirth;

String percentageIn10th;

String percentageIn12th;

String percentageInDip;
```

```
int lateralEntry;

String aadharNumber;

int verified;

public int getAdmissionId() {
        return admissionId;
}

public void setAdmissionId(int admissionId) {
        this.admissionId = admissionId;
}

public String getStudentName() {
        return studentName;
}

public void setStudentName(String studentName) {
        this.studentName = studentName;
}

public int getSemesterJoining() {
        return semesterJoining;
}

public void setSemesterJoining(int semesterJoining) {
        this.semesterJoining = semesterJoining;
}

public String getBranch() {
        return branch;
}

public void setBranch(String branch) {
        this.branch = branch;
}

public String getDate() {
        return date;
}

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

public int getTotalFeesSem1() {
        return totalFeesSem1;
}
```

```
public void setTotalFeesSem1(int totalFeesSem1) {
        this.totalFeesSem1 = totalFeesSem1;
}

public int getPaidFeesSem1() {
        return paidFeesSem1;
}

public void setPaidFeesSem1(int paidFeesSem1) {
        this.paidFeesSem1 = paidFeesSem1;
}

public int getBalanceFeesSem1() {
        return balanceFeesSem1;
}

public void setBalanceFeesSem1(int balanceFeesSem1) {
        this.balanceFeesSem1 = balanceFeesSem1;
}

public int getTotalFeesSem2() {
        return totalFeesSem2;
}

public void setTotalFeesSem2(int totalFeesSem2) {
        this.totalFeesSem2 = totalFeesSem2;
}

public int getPaidFeesSem2() {
        return paidFeesSem2;
}

public void setPaidFeesSem2(int paidFeesSem2) {
        this.paidFeesSem2 = paidFeesSem2;
}

public int getBalanceFeesSem2() {
        return balanceFeesSem2;
}

public void setBalanceFeesSem2(int balanceFeesSem2) {
        this.balanceFeesSem2 = balanceFeesSem2;
}

public int getTotalFeesSem3() {
        return totalFeesSem3;
}

public void setTotalFeesSem3(int totalFeesSem3) {
		this.totalFeesSem3 = totalFeesSem3;
	}

	public int getPaidFeesSem3() {
		return paidFeesSem3;
	}

	public void setPaidFeesSem3(int paidFeesSem3) {
		this.paidFeesSem3 = paidFeesSem3;
	}

	public int getBalanceFeesSem3() {
		return balanceFeesSem3;
	}

	public void setBalanceFeesSem3(int balanceFeesSem3) {
		this.balanceFeesSem3 = balanceFeesSem3;
	}

	public int getTotalFeesSem4() {
		return totalFeesSem4;
	}

	public void setTotalFeesSem4(int totalFeesSem4) {
		this.totalFeesSem4 = totalFeesSem4;
	}

	public int getPaidFeesSem4() {
		return paidFeesSem4;
	}

	public void setPaidFeesSem4(int paidFeesSem4) {
		this.paidFeesSem4 = paidFeesSem4;
	}

	public int getBalanceFeesSem4() {
		return balanceFeesSem4;
	}

	public void setBalanceFeesSem4(int balanceFeesSem4) {
		this.balanceFeesSem4 = balanceFeesSem4;
	}

	public int getTotalFeesSem5() {
		return totalFeesSem5;
	}

	public void setTotalFeesSem5(int totalFeesSem5) {
		this.totalFeesSem5 = totalFeesSem5;
	}
```

```
public int getPaidFeesSem5() {
        return paidFeesSem5;
}

public void setPaidFeesSem5(int paidFeesSem5) {
        this.paidFeesSem5 = paidFeesSem5;
}

public int getBalanceFeesSem5() {
        return balanceFeesSem5;
}

public void setBalanceFeesSem5(int balanceFeesSem5) {
        this.balanceFeesSem5 = balanceFeesSem5;
}

public int getTotalFeesSem6() {
        return totalFeesSem6;
}

public void setTotalFeesSem6(int totalFeesSem6) {
        this.totalFeesSem6 = totalFeesSem6;
}

public int getPaidFeesSem6() {
        return paidFeesSem6;
}

public void setPaidFeesSem6(int paidFeesSem6) {
        this.paidFeesSem6 = paidFeesSem6;
}

public int getBalanceFeesSem6() {
        return balanceFeesSem6;
}

public void setBalanceFeesSem6(int balanceFeesSem6) {
        this.balanceFeesSem6 = balanceFeesSem6;
}

public int getFeesRemainingStatus() {
        return feesRemainingStatus;
}

public void setFeesRemainingStatus(int feesRemainingStatus) {
        this.feesRemainingStatus = feesRemainingStatus;
}

public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

	public String getUsnNumber() {
		return usnNumber;
	}

	public void setUsnNumber(String usnNumber) {
		this.usnNumber = usnNumber;
	}

	public String getEmail() {
		return Email;
	}

	public void setEmail(String email) {
		Email = email;
	}

	public String getDateOfBirth() {
		return dateOfBirth;
	}

	public void setDateOfBirth(String dateOfBirth) {
		this.dateOfBirth = dateOfBirth;
	}

	public String getPercentageIn10th() {
		return percentageIn10th;
	}

	public void setPercentageIn10th(String percentageIn10th) {
		this.percentageIn10th = percentageIn10th;
	}

	public String getPercentageIn12th() {
		return percentageIn12th;
	}

	public void setPercentageIn12th(String percentageIn12th) {
		this.percentageIn12th = percentageIn12th;
	}

	public String getPercentageInDip() {
		return percentageInDip;
	}
	public void setPercentageInDip(String percentageInDip) {
		this.percentageInDip = percentageInDip;
	}

	public int getLateralEntry() {
		return lateralEntry;
	}

	public void setLateralEntry(int lateralEntry) {
		this.lateralEntry = lateralEntry;
	}

	public String getAadharNumber() {
		return aadharNumber;
	}

	public void setAadharNumber(String aadharNumber) {
		this.aadharNumber = aadharNumber;
	}

	public int getVerified() {
		return verified;
	}

	public void setVerified(int verified) {
		this.verified = verified;
	}
	}
```

ResultDetails.java

```
package com.pojo;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table
public class ResultDetails {
	@Id
	int studentId;

	String branch;

	int semId;

	String subject1;
String subject2;

String subject3;

String subject4;

String subject5;

String subject6;

String praticals1;

String praticals2;

String praticals3;

String totalMarks;

public int getStudentId() {
        return studentId;
}

public void setStudentId(int studentId) {
        this.studentId = studentId;
}

public String getBranch() {
        return branch;
}

public void setBranch(String branch) {
        this.branch = branch;
}

public int getSemId() {
        return semId;
}

public void setSemId(int semId) {
        this.semId = semId;
}

public String getSubject1() {
        return subject1;
}

public void setSubject1(String subject1) {
        this.subject1 = subject1;
}
```

```
	public String getSubject2() {
		return subject2;
	}

	public void setSubject2(String subject2) {
		this.subject2 = subject2;
	}

	public String getSubject3() {
		return subject3;
	}

	public void setSubject3(String subject3) {
		this.subject3 = subject3;
	}

	public String getSubject4() {
		return subject4;
	}

	public void setSubject4(String subject4) {
		this.subject4 = subject4;
	}

	public String getSubject5() {
		return subject5;
	}

	public void setSubject5(String subject5) {
		this.subject5 = subject5;
	}

	public String getSubject6() {
		return subject6;
	}

	public void setSubject6(String subject6) {
		this.subject6 = subject6;
	}

	public String getPraticals1() {
		return praticals1;
	}

	public void setPraticals1(String praticals1) {
		this.praticals1 = praticals1;
	}

	public String getPraticals2() {
		return praticals2;
	}

	public void setPraticals2(String praticals2) {
		this.praticals2 = praticals2;
	}

	public String getPraticals3() {
		return praticals3;
	}

	public void setPraticals3(String praticals3) {
		this.praticals3 = praticals3;
	}

	public String getTotalMarks() {
		return totalMarks;
	}

	public void setTotalMarks(String totalMarks) {
		this.totalMarks = totalMarks;
	}
	}
```

SemesterSubjectDetails .java

```
package com.pojo;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table
public class SemesterSubjectDetails {
	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	int subjectId;

	String branchName;

	int semId;

	String subject2Code;

	String subject2Name;

	String subject3Code;

	String subject3Name;
String subject4Code;

String subject4Name;

String subject5Code;

String subject5Name;

String subject6Code;

String subject6Name;

String pritical1Code;

String pritical1Name;

String pritical2Code;

String pritical2Name;

String pritical3Code;

String pritical3Name;

public int getSubjectId() {
        return subjectId;
}

public void setSubjectId(int subjectId) {
        this.subjectId = subjectId;
}

public String getBranchName() {
        return branchName;
}

public void setBranchName(String branchName) {
        this.branchName = branchName;
}

public int getSemId() {
        return semId;
}

public void setSemId(int semId) {
        this.semId = semId;
}

public String getSubject2Code() {
        return subject2Code;
}

public void setSubject2Code(String subject2Code) {
	this.subject2Code = subject2Code;
}

public String getSubject2Name() {
	return subject2Name;
}

public void setSubject2Name(String subject2Name) {
	this.subject2Name = subject2Name;
}

public String getSubject3Code() {
	return subject3Code;
}

public void setSubject3Code(String subject3Code) {
	this.subject3Code = subject3Code;
}

public String getSubject3Name() {
	return subject3Name;
}

public void setSubject3Name(String subject3Name) {
	this.subject3Name = subject3Name;
}

public String getSubject4Code() {
	return subject4Code;
}

public void setSubject4Code(String subject4Code) {
	this.subject4Code = subject4Code;
}

public String getSubject4Name() {
	return subject4Name;
}

public void setSubject4Name(String subject4Name) {
	this.subject4Name = subject4Name;
}

public String getSubject5Code() {
	return subject5Code;
}
```

```
public void setSubject5Code(String subject5Code) {
	this.subject5Code = subject5Code;
}

public String getSubject5Name() {
	return subject5Name;
}

public void setSubject5Name(String subject5Name) {
	this.subject5Name = subject5Name;
}

public String getSubject6Code() {
	return subject6Code;
}

public void setSubject6Code(String subject6Code) {
	this.subject6Code = subject6Code;
}

public String getSubject6Name() {
	return subject6Name;
}

public void setSubject6Name(String subject6Name) {
	this.subject6Name = subject6Name;
}

public String getPritical1Code() {
	return pritical1Code;
}

public void setPritical1Code(String pritical1Code) {
	this.pritical1Code = pritical1Code;
}

public String getPritical1Name() {
	return pritical1Name;
}

public void setPritical1Name(String pritical1Name) {
	this.pritical1Name = pritical1Name;
}

public String getPritical2Code() {
	return pritical2Code;
}

public void setPritical2Code(String pritical2Code) {
	this.pritical2Code = pritical2Code;
	}

	public String getPritical2Name() {
		return pritical2Name;
	}

	public void setPritical2Name(String pritical2Name) {
		this.pritical2Name = pritical2Name;
	}

	public String getPritical3Code() {
		return pritical3Code;
	}

	public void setPritical3Code(String pritical3Code) {
		this.pritical3Code = pritical3Code;
	}

	public String getPritical3Name() {
		return pritical3Name;
	}

	public void setPritical3Name(String pritical3Name) {
		this.pritical3Name = pritical3Name;
	}
}
```

SubjectDetails.java

```
package com.pojo;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table
public class SubjectDetails {
	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	int subjectId;

	String subjectCode;

	String subjectName;

	public int getSubjectId() {
		return subjectId;
	}
	public void setSubjectId(int subjectId) {
		this.subjectId = subjectId;
	}

	public String getSubjectCode() {
		return subjectCode;
	}

	public void setSubjectCode(String subjectCode) {
		this.subjectCode = subjectCode;
	}

	public String getSubjectName() {
		return subjectName;
	}

	public void setSubjectName(String subjectName) {
		this.subjectName = subjectName;
	}

}
```

TeacherDetails.java

```
package com.pojo;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table
public class TeacherDetails {
	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	int id;

	String teacherName;

	String password;

	String emailId;;

	String subject1;

	String subject2;

	String subject3;

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}

	public String getTeacherName() {
		return teacherName;
	}

	public void setTeacherName(String teacherName) {
		this.teacherName = teacherName;
	}


	public String getEmailId() {
		return emailId;
	}

	public void setEmailId(String emailId) {
		this.emailId = emailId;
	}

	public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

	public String getSubject1() {
		return subject1;
	}
	public void setSubject1(String subject1) {
		this.subject1 = subject1;
	}

	public String getSubject2() {
		return subject2;
	}

	public void setSubject2(String subject2) {
		this.subject2 = subject2;
	}
	public String getSubject3() {
		return subject3;
	}
	public void setSubject3(String subject3) {
		this.subject3 = subject3;
	}
}
```

TimeTable.Java

```
package com.pojo;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table
public class TimeTable {
	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	int id;

	String branch;

	int semId;

	String mon1;

	String mon2;

	String mon3;

	String mon4;

	String mon5;

	String tue1;

	String tue2;

	String tue3;

	String tue4;

	String tue5;

	String wed1;

	String wed2;

	String wed3;

	String wed4;
String wed5;

String thu1;

String thu2;

String thu3;

String thu4;

String thu5;

String fri1;

String fri2;

String fri3;

String fri4;

String fri5;

String sat1;

String sat2;

String sat3;

String sat4;

String sat5;

public int getId() {
        return id;
}

public void setId(int id) {
        this.id = id;
}

public String getBranch() {
        return branch;
}

public void setBranch(String branch) {
        this.branch = branch;
}

public int getSemId() {
        return semId;
}

public void setSemId(int semId) {
        this.semId = semId;
}

public String getMon1() {
        return mon1;
}

public void setMon1(String mon1) {
        this.mon1 = mon1;
}

public String getMon2() {
        return mon2;
}

public void setMon2(String mon2) {
        this.mon2 = mon2;
}

public String getMon3() {
        return mon3;
}

public void setMon3(String mon3) {
        this.mon3 = mon3;
}

public String getMon4() {
        return mon4;
}

public void setMon4(String mon4) {
        this.mon4 = mon4;
}

public String getMon5() {
        return mon5;
}

public void setMon5(String mon5) {
        this.mon5 = mon5;
}

public String getTue1() {
        return tue1;
}
```

```
	public void setTue1(String tue1) {
		this.tue1 = tue1;
	}

	public String getTue2() {
		return tue2;
	}

	public void setTue2(String tue2) {
		this.tue2 = tue2;
	}

	public String getTue3() {
		return tue3;
	}

	public void setTue3(String tue3) {
		this.tue3 = tue3;
	}

	public String getTue4() {
		return tue4;
	}

	public void setTue4(String tue4) {
		this.tue4 = tue4;
	}

	public String getTue5() {
		return tue5;
	}

	public void setTue5(String tue5) {
		this.tue5 = tue5;
	}

	public String getWed1() {
		return wed1;
	}

	public void setWed1(String wed1) {
		this.wed1 = wed1;
	}

	public String getWed2() {
		return wed2;
	}

	public void setWed2(String wed2) {
		this.wed2 = wed2;
```

```
}

public String getWed3() {
        return wed3;
}

public void setWed3(String wed3) {
        this.wed3 = wed3;
}

public String getWed4() {
        return wed4;
}

public void setWed4(String wed4) {
        this.wed4 = wed4;
}

public String getWed5() {
        return wed5;
}

public void setWed5(String wed5) {
        this.wed5 = wed5;
}

public String getThu1() {
        return thu1;
}

public void setThu1(String thu1) {
        this.thu1 = thu1;
}

public String getThu2() {
        return thu2;
}

public void setThu2(String thu2) {
        this.thu2 = thu2;
}

public String getThu3() {
        return thu3;
}

public void setThu3(String thu3) {
        this.thu3 = thu3;
}
```

```
public String getThu4() {
        return thu4;
}

public void setThu4(String thu4) {
        this.thu4 = thu4;
}

public String getThu5() {
        return thu5;
}

public void setThu5(String thu5) {
        this.thu5 = thu5;
}

public String getFri1() {
        return fri1;
}

public void setFri1(String fri1) {
        this.fri1 = fri1;
}

public String getFri2() {
        return fri2;
}

public void setFri2(String fri2) {
        this.fri2 = fri2;
}

public String getFri3() {
        return fri3;
}

public void setFri3(String fri3) {
        this.fri3 = fri3;
}

public String getFri4() {
        return fri4;
}

public void setFri4(String fri4) {
        this.fri4 = fri4;
}

public String getFri5() {
        return fri5;
```

```
	}

	public void setFri5(String fri5) {
		this.fri5 = fri5;
	}

	public String getSat1() {
		return sat1;
	}

	public void setSat1(String sat1) {
		this.sat1 = sat1;
	}

	public String getSat2() {
		return sat2;
	}

	public void setSat2(String sat2) {
		this.sat2 = sat2;
	}

	public String getSat3() {
		return sat3;
	}
	public void setSat3(String sat3) {
		this.sat3 = sat3;
	}
	public String getSat4() {
		return sat4;
	}
	public void setSat4(String sat4) {
		this.sat4 = sat4;
	}
	public String getSat5() {
		return sat5;
	}
	public void setSat5(String sat5) {
		this.sat5 = sat5;
	}

}
```

com.studentController
CandidatePasswordChange.java

```
package com.studentController;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@WebServlet("/CandidatePasswordChange")
public class CandidatePasswordChange extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		HttpSession session=request.getSession(false);

		int studentId=(Integer) session.getAttribute("studentId");

		String password1=request.getParameter("newPassword1");
		String password2=request.getParameter("newPassword2");

		if(password1.equalsIgnoreCase(password2)) {

		}else {
			response.sendRedirect("StudentChangePassword.jsp");
			return;
		}
				try {
			Class.forName("com.mysql.jdbc.Driver");
				Connection con=
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem",
"root", "root");
		String sql="update AdmissionDetails set password=?  where admissionId=?";
				PreparedStatement ps=con.prepareStatement(sql);
				ps.setString(1, password2);
				ps.setInt(2, studentId);
			   int i=ps.executeUpdate();

		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (SQLException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
			response.sendRedirect("StudentMainPage.jsp");
			}
}
```

SeeMyAdmissionDetails.java

```
package com.studentController;
```

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@WebServlet("/SeeMyAdmissionDetails")
public class SeeMyAdmissionDetails extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		HttpSession session=request.getSession(false);

		int studentId=(Integer) session.getAttribute("studentId");
		response.setContentType("text/html");
		 PrintWriter pw = response.getWriter();
		 pw.println("<html>");
		 pw.println("<body>");
		try{
			Class.forName("com.mysql.jdbc.Driver");
			Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem","roo
t","root");
			PreparedStatement ps = conn.prepareStatement("select * from
AdmissionDetails where admissionId=?");
			ps.setInt(1, studentId);
			ResultSet rs = ps.executeQuery();

			if(rs!=null){
				while(rs.next()){
					pw.println("<table align='center' border='1'>");

					pw.println("<tr>");
					 pw.println("<td>");
					 pw.println("Student Id");
					 pw.println("</td>");
					 pw.println("<td>");
					 pw.println(rs.getInt("admissionId"));
					 pw.println("</td>");
					 pw.println("</tr>");

					 pw.println("<tr>");
pw.println("<td>");
pw.println("Branch");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("branch"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("student Name");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("studentName"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("semester Joining");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("semesterJoining"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("total Fees Sem1");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("totalFeesSem1"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("paid Fees Sem1");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("paidFeesSem1"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Balance Fees Sem1");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("balanceFeesSem1"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("total Fees Sem2");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("totalFeesSem2"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("paid Fees Sem2");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("paidFeesSem2"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Balance Fees Sem2");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("balanceFeesSem2"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("total Fees Sem3");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("totalFeesSem3"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("paid Fees Sem3");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("paidFeesSem3"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Balance Fees Sem3");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("balanceFeesSem3"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("total Fees Sem4");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("totalFeesSem4"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("paid Fees Sem4");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("paidFeesSem4"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Balance Fees Sem4");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("balanceFeesSem4"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("total Fees Sem5");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("totalFeesSem5"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("paid Fees Sem5");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("paidFeesSem5"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Balance Fees Sem5");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("balanceFeesSem5"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("total Fees Sem6");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("totalFeesSem6"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("paid Fees Sem6");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("paidFeesSem6"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Balance Fees Sem6");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("balanceFeesSem6"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("Email");
pw.println("</td>");
pw.println("<td>");
pw.println(rs.getString("Email"));
pw.println("</td>");
pw.println("</tr>");

pw.println("<tr>");
pw.println("<td>");
pw.println("date Of Birth");
```

```
                                        pw.println("</td>");
                                        pw.println("<td>");
                                        pw.println(rs.getString("dateOfBirth"));
                                        pw.println("</td>");
                                        pw.println("</tr>");

                                        pw.println("<tr>");
                                        pw.println("<td>");
                                        pw.println("percentage In 10th");
                                        pw.println("</td>");
                                        pw.println("<td>");
                                        pw.println(rs.getString("percentageIn10th"));
                                        pw.println("</td>");
                                        pw.println("</tr>");

                                        pw.println("<tr>");
                                        pw.println("<td>");
                                        pw.println("percentage In 12th");
                                        pw.println("</td>");
                                        pw.println("<td>");
                                        pw.println(rs.getString("percentageIn12th"));
                                        pw.println("</td>");
                                        pw.println("</tr>");

                                        pw.println("<tr>");
                                        pw.println("<td>");
                                        pw.println("aadhar Number");
                                        pw.println("</td>");
                                        pw.println("<td>");
                                        pw.println(rs.getString("aadharNumber"));
                                        pw.println("</td>");
                                        pw.println("</tr>");

                                        pw.println("</table>");
                                }
                        }
                        }catch(SQLException | ClassNotFoundException e) {
                        }
                                }
}
```

SeeMyDetails.java

```
package com.studentController;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
```

```
@WebServlet("/SeeMyDetails")
public class SeeMyDetails extends HttpServlet {
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		HttpSession session=request.getSession(false);

		int studentId=(Integer) session.getAttribute("studentId");
	}
}
```

SeeMyResultDetails,java

```
package com.studentController;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@WebServlet("/SeeMyResultDetails")
public class SeeMyResultDetails extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		HttpSession session=request.getSession(false);

		int studentId=(Integer) session.getAttribute("studentId");

		response.setContentType("text/html");
		 PrintWriter pw = response.getWriter();

		 pw.println("<html>");
		 pw.println("<body>");

		try{
			Class.forName("com.mysql.jdbc.Driver");
			Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem","ro
ot","root");
PreparedStatement ps = conn.prepareStatement("select * from ResultDetails where
studentId=?");
			ps.setInt(1, studentId);
```

```
ResultSet rs = ps.executeQuery();

if(rs!=null){
        while(rs.next()){
                pw.println("<table align='center' border='1'>");

                pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("Student Id");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getInt("studentId"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("Branch");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("branch"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("sem Id");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getInt("semId"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("subject 1");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("subject1"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("subject 2");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("subject2"));
                 pw.println("</td>");
                 pw.println("</tr>");
                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("subject 3");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("subject3"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("subject 4");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("subject4"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("subject 5");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("subject5"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("subject 6");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("subject6"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("Praticals 1");
                 pw.println("</td>");
                 pw.println("<td>");
                 pw.println(rs.getString("praticals1"));
                 pw.println("</td>");
                 pw.println("</tr>");

                 pw.println("<tr>");
                 pw.println("<td>");
                 pw.println("Praticals 2");
                 pw.println("</td>");
```

```
                                        pw.println("<td>");
                                        pw.println(rs.getString("praticals2"));
                                        pw.println("</td>");
                                        pw.println("</tr>");

                                        pw.println("<tr>");
                                        pw.println("<td>");
                                        pw.println("Praticals 3");
                                        pw.println("</td>");
                                        pw.println("<td>");
                                        pw.println(rs.getString("praticals3"));
                                        pw.println("</td>");
                                        pw.println("</tr>");

                                        pw.println("<tr>");
                                        pw.println("<td>");
                                        pw.println("Total Marks");
                                        pw.println("</td>");
                                        pw.println("<td>");
                                        pw.println(rs.getString("totalMarks"));
                                        pw.println("</td>");
                                        pw.println("</tr>");
                                        pw.println("</table>");
                                }
                        }
                        }catch(SQLException | ClassNotFoundException e) {

                        }
                pw.println("</body>");
                pw.println("</html>");
        }
}
```

StudentLogin.java

```
package com.studentController;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@WebServlet("/StudentLogin")
public class StudentLogin extends HttpServlet {
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                String studentId1=request.getParameter("studentId");
                int studentId=Integer.parseInt(studentId1);

                HttpSession session=request.getSession(true);
     session.setAttribute("studentId",studentId);

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

                try {
                        Class.forName("com.mysql.jdbc.Driver");
                Connection con=
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem",
"root", "root");
                            String sql="select * from AdmissionDetails where admissionId=?";
                                 PreparedStatement ps=con.prepareStatement(sql);

                                 ps.setInt(1, studentId);
                            ResultSet rs=ps.executeQuery();
                           if(rs!=null)
                           {
                                   while(rs.next()) {

if(rs.getString("password").equalsIgnoreCase(studentPassword) &&
rs.getInt("verified")==1){
                                           response.sendRedirect("StudentMainPage.jsp");
                                                           }else
                           {
                                   response.sendRedirect("StudentLogin1.jsp");
                           }
                                   }
                           }

                } catch (ClassNotFoundException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                } catch (SQLException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        }
}
```

AddResult.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@page import="java.sql.*"%>
<!DOCTYPE html>
```

```
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
        <form action="AddResult">
                <table align="center">
                        <tr>
                        <td>Admission Id</td>
                        <td><select name="admisssionId">
                        <option>-----</option>
                        <%
                                try {
        Class.forName("com.mysql.jdbc.Driver");
                                                Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem",
                                        "root", "root");
        PreparedStatement ps = conn.prepareStatement("select * from AdmissionDetails");
                        ResultSet rs = ps.executeQuery();
                        if (rs != null) {
                                        while (rs.next()) {
                                %>
                        <option>
                        <%
                        out.print(rs.getInt("admissionId"));
                                %>
                        </option>
                        <%
                                }
                                        }
                                                } catch (SQLException e) {
                                        }
                                                %>
                </select>
                        </tr>
                        <tr>
                                <td>Branch</td>
                                <td><select name="branch">
                                        <option>Electrical And Electronics</option>
                                        <option>Electrical And Communication</option>
                <option>Computer Science</option>
                <option>Information Technology</option>
            </select></td>
        </tr>
        <tr>
            <td>Sem Id</td>
            <td>
                <select name="semId">
                    <option>----</option>
                    <option>1</option>
                    <option>2</option>
                    <option>3</option>
                    <option>4</option>
                    <option>5</option>
                    <option>6</option>
                </select>
            </td>
        </tr>
        <tr>
            <td>subject 1</td>
            <td><input type="text" name="subject1"></td>
        </tr>
        <tr>
            <td>subject 2</td>
            <td><input type="text" name="subject2"></td>
        </tr>
        <tr>
            <td>subject 3</td>
            <td><input type="text" name="subject3"></td>
        </tr>
        <tr>
            <td>subject 4</td>
            <td><input type="text" name="subject4"></td>
        </tr>
        <tr>
            <td>subject 5</td>
            <td><input type="text" name="subject5"></td>
        </tr>
        <tr>
            <td>subject 6</td>
            <td><input type="text" name="subject6"></td>
        </tr>
        <tr>
            <td>Pratical 1</td>
            <td><input type="text" name="pratical1"></td>
        </tr>
        <tr>
            <td>Pratical 2</td>
            <td><input type="text" name="pratical2"></td>
        </tr>
```

```
                        <tr>
                                <td>Pratical 3</td>
                                <td><input type="text" name="pratical3"></td>
                        </tr>
                        <tr>
                                <td>Total Marks</td>
                                <td><input type="text" name="totalMarks"></td>
                        </tr>
                        <tr>
                                <td colspan="2" align="center"><input type="submit"
                                        value="submit"></td>
                        </tr>
                </table>
        </form>
</body>
</html>
```

AddSubjects.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
        <form action="AddSubject">
                <table border="1" align="center">
                        <tr>
                                <td>subject Name</td>
                                <td><input type="text" name="subjectName"></td>
                        </tr>
                        <tr>
                                <td>subject Code</td>
                                <td><input type="text" name="subjectCode"></td>
                        </tr>
                        <tr>
                <td colspan="2" align="center"><input type="submit" value="submit"></td>
                        </tr>
                </table>
        </form>
</body>
</html>
```

AddTeachers.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="AddTeacher">
        <table align="center">
                        <tr>
                                <td>Teacher Name</td>
                                <td><input type="text" name="teacherName"></td>
                        </tr>
                        <tr>
                                <td>E-mail Id</td>
                                <td><input type="text" name="teacherEmailId"></td>
                        </tr>
                        <tr>
                                <td>Subject 1</td>
                                <td><input type="text" name="subject1"></td>
                        </tr>
                        <tr>
                                <td>Subject 2</td>
                                <td><input type="text" name="subject2"></td>
                        </tr>
                        <tr>
                                <td>Subject 3</td>
                                <td><input type="text" name="subject3"></td>
                        </tr>
                        <tr>
                <td colspan="2" align="center"><input type="submit" value="submit"></td>
                        </tr>
                </table>
        </form>
```

```
</body>
</html>
```

AddTimeTable.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="AddTimetable">
        <table border="1" align="center">
                <tr>
                        <td>Branch</td>
                        <td><input type="text" name="branch"></td>
                </tr>
                <tr>
                        <td>sem</td>
                        <td>
                                <select name="semId">
                                        <option>----</option>
                                        <option>1</option>
                                        <option>2</option>
                                        <option>3</option>
                                        <option>4</option>
                                        <option>5</option>
                                        <option>6</option>
                                </select>
                        </td>
                </tr>
                <tr>
                        <td>day</td>
                        <td>session 1</td>
                        <td>session 2</td>
                        <td>session 3</td>
                        <td>session 4</td>
                        <td>session 5</td>
                </tr>
```

```
<tr>
        <td>Monday</td>
        <td><input type="text" name="mon1"></td>
        <td><input type="text" name="mon2"></td>
        <td><input type="text" name="mon3"></td>
        <td><input type="text" name="mon4"></td>
        <td><input type="text" name="mon5"></td>
</tr>
<tr>
        <td>Tuesday</td>
        <td><input type="text" name="tue1"></td>
        <td><input type="text" name="tue2"></td>
        <td><input type="text" name="tue3"></td>
        <td><input type="text" name="tue4"></td>
        <td><input type="text" name="tue5"></td>
</tr>
<tr>
        <td>Wednesday</td>
        <td><input type="text" name="wed1"></td>
        <td><input type="text" name="wed2"></td>
        <td><input type="text" name="wed3"></td>
        <td><input type="text" name="wed4"></td>
        <td><input type="text" name="wed5"></td>
</tr>
<tr>
        <td>Thursday</td>
        <td><input type="text" name="thu1"></td>
        <td><input type="text" name="thu2"></td>
        <td><input type="text" name="thu3"></td>
        <td><input type="text" name="thu4"></td>
        <td><input type="text" name="thu5"></td>
</tr>
<tr>
        <td>Friday</td>
        <td><input type="text" name="fri1"></td>
        <td><input type="text" name="fri2"></td>
        <td><input type="text" name="fri3"></td>
        <td><input type="text" name="fri4"></td>
        <td><input type="text" name="fri5"></td>
</tr>
<tr>
        <td>Saturday</td>
        <td><input type="text" name="sat1"></td>
        <td><input type="text" name="sat2"></td>
        <td><input type="text" name="sat3"></td>
        <td><input type="text" name="sat4"></td>
        <td><input type="text" name="sat5"></td>
</tr>

<tr>
```

```
                <td colspan="6" align="center">
                    <input type="submit" value="submit">
                </td>
            </tr>
        </table>
</form>
</body>
</html>
```

AdminChangePassword.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
        <form action="AdminPasswordChange">
            <table align="center">
                <tr>
                    <td>Current Password</td>
                    <td><input type="password" name="currentPassword"></td>
                </tr>
                <tr>
                    <td>New Password</td>
                    <td><input type="password" name="newPassword1"></td>
                </tr>
                <tr>
                    <td>Re-enter New Password</td>
                    <td><input type="password" name="newPassword2"></td>
                </tr>
                <tr>
        <td colspan="2" align="center"><input type="submit" value="submit"></td>
                </tr>
            </table>
        </form>
</body>
</html>
```

AdminLogin.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
        <form action="AdminLogin">
                <table>
                        <tr>
                                <td>Admin Id</td>
                                <td><input type="text" name="adminId"></td>
                        </tr>
                        <tr>
                                <td>Admin Name</td>
                                <td><input type="text" name="adminName"></td>
                        </tr>
                        <tr>
                                <td>Admin Password</td>
                                <td><input type="password" name="adminPassword"></td>
                        </tr>
                        <tr>
                <td colspan="2" align="center"><input type="submit" value="login"></td>
                        </tr>
                </table>
        </form>
</body>
</html>
```

AdminMainPage.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<style>
```

```
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
}
.topnav {
  overflow: hidden;
  background-color: #333;
}
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}
.topnav a:hover {
  background-color: #ddd;
  color: black;
}
.topnav a.active {
  background-color: #4CAF50;
  color: white;
}
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
</head>
<body>
	<div class="topnav">
  <a class="active" href="AdminChangePassword.jsp">change Password</a>
	<a href="newStudentAdmission.jsp">new Student Admission</a>
	<a href="OldStudentAdmission.jsp">Old Student Admission</a>
<a href="SeeStudentsAdmissionAndRecordDetails.jsp">See Students Admission And
RecordDetails</a><br>
	<a href="ChangeRecordOfStudents.jsp">Change Record Of Students</a>
	<a href="AddTeachers.jsp">Add Teachers</a>
	<a href="SeeTeachers.jsp">See Teachers</a>
	<a href="AddSubjects.jsp">Add  Subjects</a>
	<a href="SeeSubjects.jsp">See Subjects</a>
	<a href="AddTimeTable.jsp">Add Timetable</a>
	<a href="SeeTimetable.jsp">See Timetable</a>
	</div>
</body>
</html>
```

ChangeRecordOfStudent.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
        <form action="UpdateRecordOfStudents.jsp">
                <table align="center">
                        <tr>
                                <td>Admission Id</td>
                                <td><select name="admisssionId">
                                                <option>-----</option>
                                                <%
                                                        try {
                Class.forName("com.mysql.jdbc.Driver");
Connection conn
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem",
        "root", "root");
        PreparedStatement ps = conn.prepareStatement("select * from AdmissionDetails");
                                ResultSet rs = ps.executeQuery();
                                                                if (rs != null) {
                                                                        while (rs.next()) {
                                                %>
                                                <option>
                                                        <%
                                                        out.print(rs.getInt("admissionId"));
                                                        %>
                                                </option>
                                                <%
                                                        }
                                                                }
                                                        } catch (SQLException e) {
                                                        }
                                                %>
                                </select>
```

```
                              </tr>
                              <tr>
                    <td colspan="2" align="center"><input type="submit" value="submit"></td>
                              </tr>
                    </table>
          </form>
</body>
</html>
```

index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<center>
<a href="AdminLogin.jsp"><input type="button" value="Admin Login"></a><br><br>
<a href="TeacherLogin.jsp"><input type="button" value="Teacher Login"></a><br><br>
<a href="StudentLogin.jsp"><input type="button" value="Student Login"></a>
</center>
</body>
</html>
```

newStudentAdmission.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="NewStudentAdmission">
        <table border="1">
        <tr>
                <td>Student Name</td>
                <td><input type="text" name="studentName"></td>
        </tr>
                <tr>
                <td>Semester Joining</td>
                <td><select name="semId">
                        <option>1</option>
                        <option>3</option>
                </select></td>
        </tr>
                <tr>
                <td>Branch</td>
                <td>
                <select name="branch">
                        <option>Electrical And Electronics</option>
                        <option>Electrical And Communication</option>
                        <option>Computer Science</option>
                        <option>Information Technology</option>
                </select>
                </td>
        </tr>
                <tr>
                <td>sem total fee</td>
                <td><input type="text" name="semTotalFee"></td>
        </tr>
                <tr>
                <td>paid</td>
                <td><input type="text" name="studentPaid"></td>
        </tr>
                <tr>
                <td>Balance</td>
                <td><input type="text" name="balance"></td>
        </tr>
                <tr>
                <td>e-mail</td>
                <td><input type="text" name="studentEmail"></td>
        </tr>
        <tr>
                <td>Date Of Birth</td>
                <td><input type="date" name="studentDateOfBirth"></td>
        </tr>
        <tr>
                <td>10th Percentage</td>
                <td><input type="text" name="student10thPercentage"></td>
        </tr>
        <tr>
                <td>12th Percentage</td>
                <td><input type="text" name="student12thPercentage"></td>
        </tr>
        <tr>
                <td>Diplomo</td>
                <td><input type="text" name="studentDiplomo"></td>
        </tr>

        <tr>
                <td>Aadhar Number</td>
                <td><input type="text" name="studentAadharNumber"></td>
        </tr>
        <tr>
                <td colspan="2" align="center"><input type="submit" value="submit"></td>
        </tr>
        </table>
</form>
</body>
</html>
```

OldStudentAdmission.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
   <%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="OldStudentAdmission">
        <table>
                <tr>
                        <td>Select Student Admission Id</td>
                        <td><select name="studentAdmissionId">
```

```
				<%
try{
Class.forName("com.mysql.jdbc.Driver");
Connection conn = 
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem",
"root","root");
PreparedStatement ps = conn.prepareStatement("select * from AdmissionDetails");
ResultSet rs = ps.executeQuery();
if(rs!=null){
	while(rs.next()){
		%>
			<option><%out.print(rs.getInt("admissionId")); %></option>
		<%
	}
	}
	}catch(SQLException e) {
	}
	%>
			</select>
			</td>
		</tr>
		<tr>
			<td>sem</td>
			<td>
			<select name="semId">
			<option>1</option>
			<option>2</option>
			<option>3</option>
			<option>4</option>
			<option>5</option>
			<option>6</option>
			</select>
			</td>
		</tr>
		<tr>
			<td>Total sem Fee</td>
			<td><input type="text" name="totalSemFee"></td>
	</tr>
		<tr>
			<td>paid sem Fee</td>
			<td><input type="text" name="paidSemFee"></td>
		</tr>
		<tr>
			<td>balance sem Fee</td>
			<td><input type="text" name="balanceSemFee"></td>
		</tr>
		<tr>
			<td colspan="2" align="center"><input type="submit" 
value="submit"></td>
		</tr>
```

```
        </table>
</form>
</body>
</html>
```

SeeMyDetails.jsp

```
   pageEncoding="ISO-8859-1"%>
   <%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
</body>
</html>
```

SeeParticularStudentResultDetails.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
   <%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="SeeParticularStudentResultDetails">
        <table border="1" align="center">
                <tr>
                        <td>Student Id</td>
                        <td><select name="admisssionId">
                                        <option>-----</option>
                                        <%
                                                try {
                        Class.forName("com.mysql.jdbc.Driver");
                                                        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem",
                                                                        "root", "root");
```

```
          PreparedStatement ps = conn.prepareStatement("select * from AdmissionDetails");
                              ResultSet rs = ps.executeQuery();
                              if (rs != null) {
                                        while (rs.next()) {
                                        %>
                                        <option>
                                             <%
               out.print(rs.getInt("admissionId"));
                                             %>
                                        </option>
                                        <%
                                             }
                                                  }
                                             } catch (SQLException e) {
                                             }
                                        %>
                              </select>
                         </tr>
          <tr>
                              <td colspan="2" align="center"><input type="submit"
                                   value="submit"></td>
                         </tr>
               </table>
</form>
</body>
</html>
```

SeeStudentsAdmissionAndRecordDetails.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
   <%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<table border=1>
          <tr>
               <td>Admission Id</td>
```

```
                <td>Student Name</td>
                <td>SemesterJoining</td>
                <td>Branch</td>
                <td>Date</td>
                <td>Total Fees Sem1</td>
                <td>Paid Fees Sem1</td>
                <td>Balance Fees Sem1</td>
                <td>Total Fees Sem2</td>
                <td>Paid Fees Sem2</td>
                <td>Balance Fees Sem2</td>
                <td>Total Fees Sem3</td>
                <td>Paid Fees Sem3</td>
                <td>Balance Fees Sem3</td>
                <td>Total Fees Sem4</td>
                <td>Paid Fees Sem4</td>
                <td>Balance Fees Sem4</td>
                <td>Total Fees Sem5</td>
                <td>Paid Fees Sem5</td>
                <td>Balance Fees Sem5</td>
                <td>Total Fees Sem6</td>
                <td>Paid Fees Sem6</td>
                <td>Balance Fees Sem6</td>
                <td>Fees Remaining Status</td>
                <td>Date Of Birth</td>
                <td>Percentage In 10th</td>
                <td>percentage In 12th</td>
                <td>percentage In Dip</td>
                <td>verified</td>
                        </tr>
        <%
try{
Class.forName("com.mysql.jdbc.Driver");
Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem",
"root","root");
PreparedStatement ps = conn.prepareStatement("select * from AdmissionDetails");
ResultSet rs = ps.executeQuery();

if(rs!=null){
        while(rs.next()){
                %>
                <tr>
                        <td><%out.print(rs.getInt("admissionId")); %></td>
                        <td><%out.print(rs.getString("studentName")); %></td>
                        <td><%out.print(rs.getInt("semesterJoining")); %></td>
                        <td><%out.print(rs.getString("branch")); %></td>
                        <td><%out.print(rs.getString("date")); %></td>
                        <td><%out.print(rs.getInt("totalFeesSem1")); %></td>
                        <td><%out.print(rs.getInt("paidFeesSem1")); %></td>
                        <td><%out.print(rs.getInt("balanceFeesSem1")); %></td>
```

```
                              <td><%out.print(rs.getInt("totalFeesSem2")); %></td>
                              <td><%out.print(rs.getInt("paidFeesSem2")); %></td>
                              <td><%out.print(rs.getInt("balanceFeesSem2")); %></td>
                              <td><%out.print(rs.getInt("totalFeesSem3")); %></td>
                              <td><%out.print(rs.getInt("paidFeesSem3")); %></td>
                              <td><%out.print(rs.getInt("balanceFeesSem3")); %></td>
                              <td><%out.print(rs.getInt("totalFeesSem4")); %></td>
                              <td><%out.print(rs.getInt("paidFeesSem4")); %></td>
                              <td><%out.print(rs.getInt("balanceFeesSem4")); %></td>
                              <td><%out.print(rs.getInt("totalFeesSem5")); %></td>
                              <td><%out.print(rs.getInt("paidFeesSem5")); %></td>
                              <td><%out.print(rs.getInt("balanceFeesSem5")); %></td>
                              <td><%out.print(rs.getInt("totalFeesSem6")); %></td>
                              <td><%out.print(rs.getInt("paidFeesSem6")); %></td>
                              <td><%out.print(rs.getInt("balanceFeesSem6")); %></td>
                              <%int i=0;
                              if(rs.getInt("balanceFeesSem1")==0 &&
rs.getInt("balanceFeesSem2")==0 && rs.getInt("balanceFeesSem3")==0 &&
rs.getInt("balanceFeesSem4")==0 && rs.getInt("balanceFeesSem5")==0 &&
rs.getInt("balanceFeesSem6")==0){
                                        i=1;
                              }
                                        %>
                              <td><%
                              if(i==1){
                                        out.print("No Balance");
                              }else{
                                        out.print("Balance");
                              }
                               %></td>
                              <td><%out.print(rs.getString("dateOfBirth")); %></td>
                              <td><%out.print(rs.getString("percentageIn10th")); %></td>
                              <td><%out.print(rs.getString("percentageIn12th")); %></td>
                              <td><%out.print(rs.getString("percentageInDip")); %></td>
                              <td><%out.print(rs.getInt("verified")); %></td>
                              </tr>
                    <%
          }
          }
          }catch(SQLException e) {
          }
          %>
          </tr>
          </table>
</body>
</html>
```

SeeSubjects.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
   <%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
	<table border=1 align="center">
	<tr>
		<td>Subject code</td>
		<td>Subject Name</td>
			</tr>
		<%
try{
Class.forName("com.mysql.jdbc.Driver");
Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem",
"root","root");
PreparedStatement ps = conn.prepareStatement("select * from SubjectDetails");
ResultSet rs = ps.executeQuery();
if(rs!=null){
	while(rs.next()){
		%><tr>
		<td><%
		out.println(rs.getString("subjectCode"));
		%></td>
		<td><%
		out.println(rs.getString("subjectName"));
		%></td>
		</tr><%
	}
}
}catch(SQLException e) {
}
%>
</table>
</body>
</html>
```

SeeTeachers.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
    <%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<table border=1 align="center">
	<tr>
		<td>Teacher Id</td>
		<td>Teacher Name</td>
		<td>E-mail Id</td>
		<td>Subject 1</td>
		<td>Subject 2</td>
		<td>Subject 3</td>
	</tr>
		<%
try{
Class.forName("com.mysql.jdbc.Driver");
Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem","ro
ot","root");
PreparedStatement ps = conn.prepareStatement("select * from TeacherDetails");
ResultSet rs = ps.executeQuery();

if(rs!=null){
	while(rs.next()){
		%><tr>
		<td><%
		out.println(rs.getInt("id"));
		%></td>
		<td><%
		out.println(rs.getString("teacherName"));
		%></td>
		<td><%
		out.println(rs.getString("emailId"));
		%></td>
		<td><%
```

```
                    out.println(rs.getString("Subject1"));
                    %></td>
                    <td><%
                    out.println(rs.getString("Subject2"));
                    %></td>

                    <td><%
                    out.println(rs.getString("Subject3"));
                    %></td></tr><%
          }
}
}catch(SQLException e) {
}
%>
</table>
</body>
</html>
```

SeeTimetable.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style><body>
<form action="SemTimeTable1.jsp">
          <table align="center" border="1">
                    <tr>
                              <td>branch</td>
                              <td><select name="branch">
                                                  <option>---------</option>
                                                  <%
                                                            try {
                                        Class.forName("com.mysql.jdbc.Driver");
                                        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem",
"root", "root");
PreparedStatement ps = conn.prepareStatement("select distinct branch from timetable");
                    ResultSet rs = ps.executeQuery();
```

```
                                        if (rs != null) {
                                                while (rs.next()) {
                                        %>
                                        <option>
                                                <%
                                                        out.print(rs.getString("branch"));
                                                %>
                                        </option>
                                <%
                                                }
                                                        }
                                                } catch (SQLException e) {

                                                }
                                        %>
                        </select></td>
                </tr>
                <tr>
                        <td>sem</td>
                        <td>
                                <select name="semId">
                                        <option>-----</option>
                                        <option>1</option>
                                        <option>2</option>
                                        <option>3</option>
                                        <option>4</option>
                                        <option>5</option>
                                        <option>6</option>
                                </select>
                        </td>
                </tr>
                <tr>
        <td colspan="2" align="center"><input type="submit" value="submit"></td>
                </tr>
        </table>
</form>
</body>
</html>
```

SemTimeTable1.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style><body>
<form action="SemTimeTable1.jsp">
        <table align="center" border="1">
                <tr>
                        <td>branch</td>
                        <td><select name="branch">
                                        <option>---------</option>
                                        <%
                                                try {
                Class.forName("com.mysql.jdbc.Driver");
                        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegeManagementSystem",
                "root", "root");
PreparedStatement ps = conn.prepareStatement("select distinct branch from timetable");
                        ResultSet rs = ps.executeQuery();
                        if (rs != null) {
                                        while (rs.next()) {
                                        %>
                                        <option>
                                                <%
                                                        out.print(rs.getString("branch"));
                                                %>
                                        </option>
                                        <%
                                                }
                                                        }
                                                } catch (SQLException e) {
                                                }
                                        %>
                        </select></td>
                </tr>
                <tr>
                        <td>sem</td>
                        <td>
                                <select name="semId">
                                        <option>-----</option>
                                        <option>1</option>
                                        <option>2</option>
                                        <option>3</option>
                                        <option>4</option>
                                        <option>5</option>
                                        <option>6</option>
                                </select>
                        </td>
```

```
			</tr>
			<tr>
				<td colspan="2" align="center"><input type="submit"
value="submit"></td>
			</tr>
		</table>
</form>
</body>
</html>
```

StudentChangePassword.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
	<form action="CandidatePasswordChange">
		<table align="center">
			<tr>
				<td>Current Password</td>
				<td><input type="password" name="currentPassword"></td>
			</tr>
			<tr>
				<td>New Password</td>
				<td><input type="password" name="newPassword1"></td>
			</tr>
			<tr>
				<td>Re-enter New Password</td>
				<td><input type="password" name="newPassword2"></td>
			</tr>
			<tr>
				<td colspan="2" align="center"><input type="submit"
					value="submit"></td>
			</tr>
		</table>
	</form>
</body>
```

```
</html>
```

StudentLogin.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="StudentLogin">
		<table>
			<tr>
				<td>Student Admission Id</td>
				<td><input type="text" name="studentId"></td>
			</tr>
			<tr>
				<td>Student Password</td>
				<td><input type="password" name="studentPassword"></td>
			</tr>
			<tr>
				<td colspan="2" align="center"><input type="submit"
value="login"></td>
			</tr>
		</table>
	</form>
</body>
</html>
```

StudentLogin1.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
```

```
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<script type="text/javascript">
alert("check your admission id , password or verification")
</script>
<body>
<form action="StudentLogin">
		<table>
			<tr>
				<td>Student Admission Id</td>
				<td><input type="text" name="studentId"></td>
			</tr>
			<tr>
				<td>Student Password</td>
				<td><input type="text" name="studentPassword"></td>
			</tr>
			<tr>
				<td colspan="2" align="center"><input type="submit"
value="login"></td>
			</tr>
		</table>
	</form>

</body>
</html>
```

StudentMainPage.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<style>
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
}
.topnav {
  overflow: hidden;
  background-color: #333;
}
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}
.topnav a:hover {
  background-color: #ddd;
  color: black;
}
.topnav a.active {
  background-color: #4CAF50;
  color: white;
}
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
</head>
<body>
<div class="topnav">
  <a class="active" href="StudentChangePassword.jsp">Change Password</a>
<a href="SeeTimetable.jsp">See Timetable</a>
</div>
<form action="SeeMyAdmissionDetails">
<input type="submit" value="See My Admission Details">
</form>
<form action="SeeMyResultDetails">
<input type="submit" value="See My Result Details">
</form>
</body>
</html>
```

TeacherChangePassword.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="TeacherChangePassword">
	<table border="1" align="center">
		<tr>
			<td>Enter new password</td>
			<td><input type="text" name="newPassword"></td>
		</tr>
		<tr>
			<td>Re-Enter new password</td>
			<td><input type="text" name="Password"></td>
		</tr>
		<tr>
			<td colspan="2" align="center"><input type="submit"
value="submit"></td>
		</tr>
	</table>
</form>
</body>
</html>
```

TeacherLogin.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
<form action="TeacherLogin">
		<table>
			<tr>
				<td>Teacher Id</td>
				<td><input type="text" name="teacherId"></td>
```

```
						</tr>
						<tr>
							<td>Teacher Name</td>
							<td><input type="text" name="teacherName"></td>
						</tr>
						<tr>
							<td>Teacher Password</td>
							<td><input type="password" name="teacherPassword"></td>
						</tr>
						<tr>
							<td colspan="2" align="center"><input type="submit"
value="login"></td>
						</tr>
				</table>
		</form>

</body>
</html>
```

TeacherMainPage.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
<style>
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
}
.topnav {
  overflow: hidden;
  background-color: #333;
}
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}
.topnav a:hover {
  background-color: #ddd;
  color: black;
}
.topnav a.active {
  background-color: #4CAF50;
  color: white;
}
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
</head>
<body>
            <div class="topnav">
  <a class="active" href="TeacherChangePassword.jsp">Change Password</a>
<a href="SeeSubjects.jsp">See Subjects</a><br>
<a href="SeeStudentsAdmissionAndRecordDetails.jsp">See Students Admission And
RecordDetails</a><br>
<a href="SeeTimetable.jsp">See Timetable</a><br>
<a href="SeeTeachers.jsp">See Teachers</a><br>
<a href="AddResult.jsp">Add Result</a><br>
<a href="SeeParticularStudentResultDetails.jsp">See Particular Student Result
Details</a><br>
</div>
</body>
</html>
```

UpdateAdmissionOfStudent.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>

</body>
</html>
```

UpdateRecordOfStudents.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
            pageEncoding="ISO-8859-1"%>
<%@page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
```

```
<style>
body{
 background-repeat: no-repeat;
 background-image: url("images1.jpg");
 background-size:cover;
 background-position:top center;
 background-attachment: fixed;
}
</style>
<body>
        <form action="UpdateStudentDetails">
                <table border=1>
                        <%
                        String admissionId1=request.getParameter("admisssionId");
                        int admissionId=Integer.parseInt(admissionId1);
                                try {
                                        Class.forName("com.mysql.jdbc.Driver");
                                        Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/collegemanagementsystem",
                                                        "root", "root");
                                        PreparedStatement ps = conn.prepareStatement("select
* from AdmissionDetails where admissionId=?");
                                        ps.setInt(1,admissionId);
                                        ResultSet rs = ps.executeQuery();
                                        if (rs != null) {
                                                while (rs.next()) {
                        %>
                        <tr>
                                <td>Admission Id</td>
                                <td><input type="text" name="admissionId"
                        value="<%out.print(rs.getInt("admissionId"));%>"></td>
                        </tr>
                        <tr>
                                <td>Student Name</td>
                                <td><input type="text" name="studentName"
                        value="<%out.print(rs.getString("studentName"));%>"></td>
                        </tr>
                        <tr>
                                <td>SemesterJoining</td>
                                <td><input type="text" name="semesterJoining"
                        value="<%out.print(rs.getInt("semesterJoining"));%>"></td>
                        </tr>
                        <tr>
                                <td>Branch</td>
                                <td><input type="text" name="branch"
                                        value="<%out.print(rs.getString("branch"));%>"></td>
                        </tr>
                        <tr>
                                <td>Date</td>
                                <td><input type="text" name="date"
```

```
                    value="<%out.print(rs.getString("date"));%>"></td>
</tr>
<tr>
          <td>Total Fees Sem1</td>
          <td><input type="text" name="totalFeesSem1"
value="<%out.print(rs.getInt("totalFeesSem1"));%>"></td>
</tr>
<tr>
          <td>Paid Fees Sem1</td>
          <td><input type="text" name="paidFeesSem1"
value="<%out.print(rs.getInt("paidFeesSem1"));%>"></td>
</tr>
<tr>
          <td>Balance Fees Sem1</td>
          <td><input type="text" name="balanceFeesSem1"
value="<%out.print(rs.getInt("balanceFeesSem1"));%>"></td>
</tr>
<tr>
          <td>Total Fees Sem2</td>
          <td><input type="text" name="totalFeesSem2"
value="<%out.print(rs.getInt("totalFeesSem2"));%>"></td>
</tr>
<tr>
          <td>Paid Fees Sem2</td>
          <td><input type="text" name="paidFeesSem2"
value="<%out.print(rs.getInt("paidFeesSem2"));%>"></td>
</tr>
<tr>
          <td>Balance Fees Sem2</td>
          <td><input type="text" name="balanceFeesSem2"
value="<%out.print(rs.getInt("balanceFeesSem2"));%>"></td>
</tr>
<tr>
          <td>Total Fees Sem3</td>
          <td><input type="text" name="totalFeesSem3"
value="<%out.print(rs.getInt("totalFeesSem3"));%>"></td>
</tr>
<tr>
          <td>Paid Fees Sem3</td>
          <td><input type="text" name="paidFeesSem3"
value="<%out.print(rs.getInt("paidFeesSem3"));%>"></td>
</tr>
<tr>
          <td>Balance Fees Sem3</td>
          <td><input type="text" name="balanceFeesSem3"
value="<%out.print(rs.getInt("balanceFeesSem3"));%>"></td>
</tr>
<tr>
          <td>Total Fees Sem4</td>
          <td><input type="text" name="totalFeesSem4"
```

```
value="<%out.print(rs.getInt("totalFeesSem4"));%>"></td>
</tr>
<tr>
        <td>Paid Fees Sem4</td>
        <td><input type="text" name="paidFeesSem4"
value="<%out.print(rs.getInt("paidFeesSem4"));%>"></td>
</tr>
<tr>
        <td>Balance Fees Sem4</td>
        <td><input type="text" name="balanceFeesSem4"
value="<%out.print(rs.getInt("balanceFeesSem4"));%>"></td>
</tr>
<tr>
        <td>Total Fees Sem5</td>
        <td><input type="text" name="totalFeesSem5"
value="<%out.print(rs.getInt("totalFeesSem5"));%>"></td>
</tr>
<tr>
        <td>Paid Fees Sem5</td>
        <td><input type="text" name="paidFeesSem5"
value="<%out.print(rs.getInt("paidFeesSem5"));%>"></td>
</tr>
<tr>
        <td>Balance Fees Sem5</td>
        <td><input type="text" name="balanceFeesSem5"
value="<%out.print(rs.getInt("balanceFeesSem5"));%>"></td>
</tr>
<tr>
        <td>Total Fees Sem6</td>
        <td><input type="text" name="totalFeesSem6"
value="<%out.print(rs.getInt("totalFeesSem6"));%>"></td>
</tr>
<tr>
        <td>Paid Fees Sem6</td>
        <td><input type="text" name="paidFeesSem6"
value="<%out.print(rs.getInt("paidFeesSem6"));%>"></td>
</tr>
<tr>
        <td>Balance Fees Sem6</td>
        <td><input type="text" name="balanceFeesSem6"
value="<%out.print(rs.getInt("balanceFeesSem6"));%>"></td>
</tr>
<tr>
        <td>Date Of Birth</td>
        <td><input type="text" name="dateOfBirth"
value="<%out.print(rs.getString("dateOfBirth"));%>"></td>
</tr>
<tr>
        <td>Percentage In 10th</td>
        <td><input type="text" name="percentageIn10th"
```

```
                        value="<%out.print(rs.getString("percentageIn10th"));%>"></td>
                        </tr>
                        <tr>
                                <td>percentage In 12th</td>
                                <td><input type="text" name="percentageIn12th"
                        value="<%out.print(rs.getString("percentageIn12th"));%>"></td>
                        </tr>
                        <tr>
                                <td>percentage In Dip</td>
                                <td><input type="text" name="percentageInDip"
                        value="<%out.print(rs.getString("percentageInDip")); %>"></td>
                        </tr>
                        <tr>
                                <td>verified</td>
                                <td><select name="verified">
                                                <option>
                                                        <%out.print(rs.getInt("verified")); %>
                                                </option>
                                                <option>0</option>
                                                <option>1</option>
                                </select></td>
                        </tr>
                        <%
        }
        }
        }catch(SQLException e) {
        }
        %>
                        </tr>
                        <tr>
        <td colspan="2" align="center"><input type="submit" value="update"></td>
                        </tr>
                </table>
        </form>
</body>
</html>
```

Chapter: 10

Methodology used for testing

The completion of a system will be achieved only after it has been thoroughly tested. Though this gives a feel the project is completed, there cannot be any project without going through this stage. Hence in this stage it is decided whether the project can undergo the real time environment execution without any break downs, therefore a package can be rejected even at this stage.

10.1 Testing methods

Software testing methods are traditionally divided into black box testing and white box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

1) **Black box testing** - Black box testing treats the software as a "black box," without any knowledge of internal implementation. Black box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, traceability matrix, exploratory testing and specification-based testing.

2) **White box testing** - White box testing, by contrast to black box testing, is when the tester has access to the internal data structures and algorithms (and the code that implement these).White box testing methods can also be used to evaluate the completeness of a test suite that was created with black box testing methods. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested.

3) **Grey Box Testing** - Grey box testing involves having access to internal data structures and algorithms for purposes of designing the test cases, but testing at the user, or black-box level. Manipulating input data and formatting output do not qualify as "grey box," because the input and output are clearly outside of the "black-box" that we are calling the system under test. This distinction is particularly important when conducting integration testing between two modules of code written by two different developers,

where only the interfaces are exposed for test. Grey box testing may also include reverse engineering to determine, for instance, boundary values or error messages.

4) **Acceptance testing** - Acceptance testing can mean one of two things:

1. A smoke test is used as an acceptance test prior to introducing a build to the main testing process.
2. Acceptance testing performed by the customer is known as user acceptance testing (UAT).

5) **Regression Testing -** Regression testing is any type of **software testing** that seeks to uncover software regressions. Such regression occurs whenever software functionality that was previously working correctly stops working as intended. Typically regressions occur as an unintended consequence of program changes. Common methods of regression testing include re-running previously run tests and checking whether previously fixed faults have re-emerged.

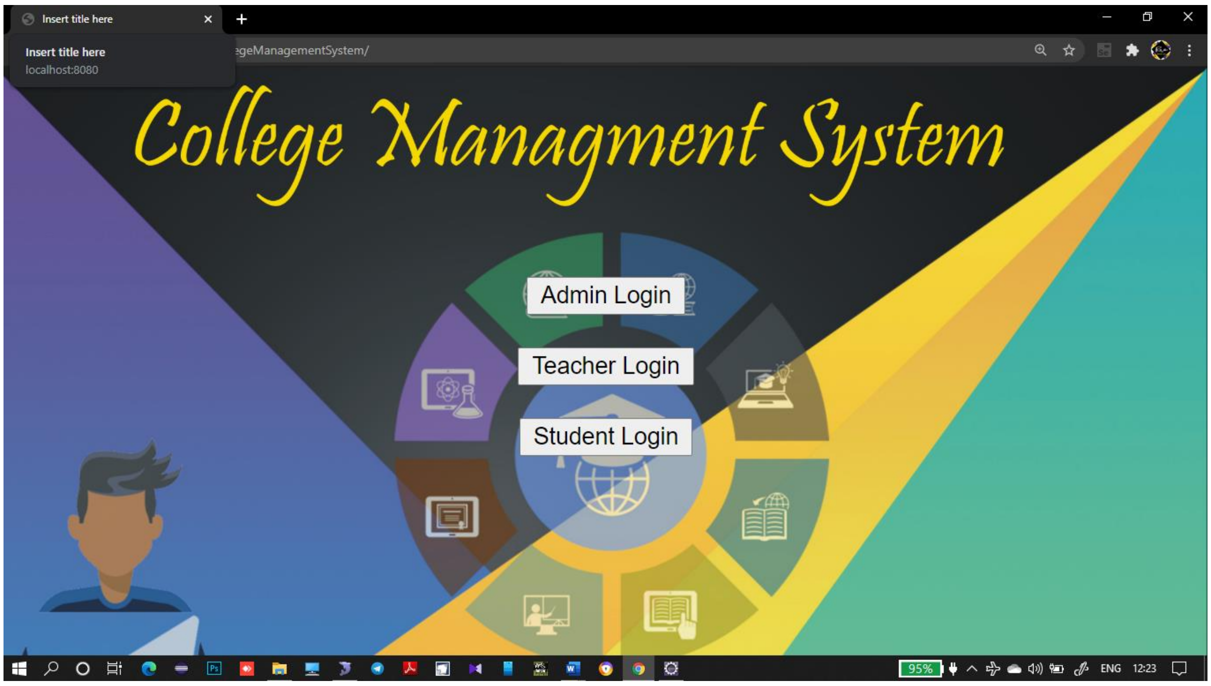
6) **Non Functional Software Testing** - Special methods exist to test non-functional aspects of software.

- Performance testing checks to see if the software can handle large quantities of data or users. This is generally referred to as software scalability. This activity of Non Functional Software Testing is often times referred to as Load Testing.
- Stability testing checks to see if the software can continuously function well in or above an acceptable period. This activity of Non Functional Software Testing is often times referred to as indurations test.
- Usability testing is needed to check if the user interface is easy to use and understand.
- Security testing is essential for software which processes confidential data and to prevent system intrusion by hackers.
- Internationalization and localization is needed to test these aspects of software, for which a pseudo localization method can be used.

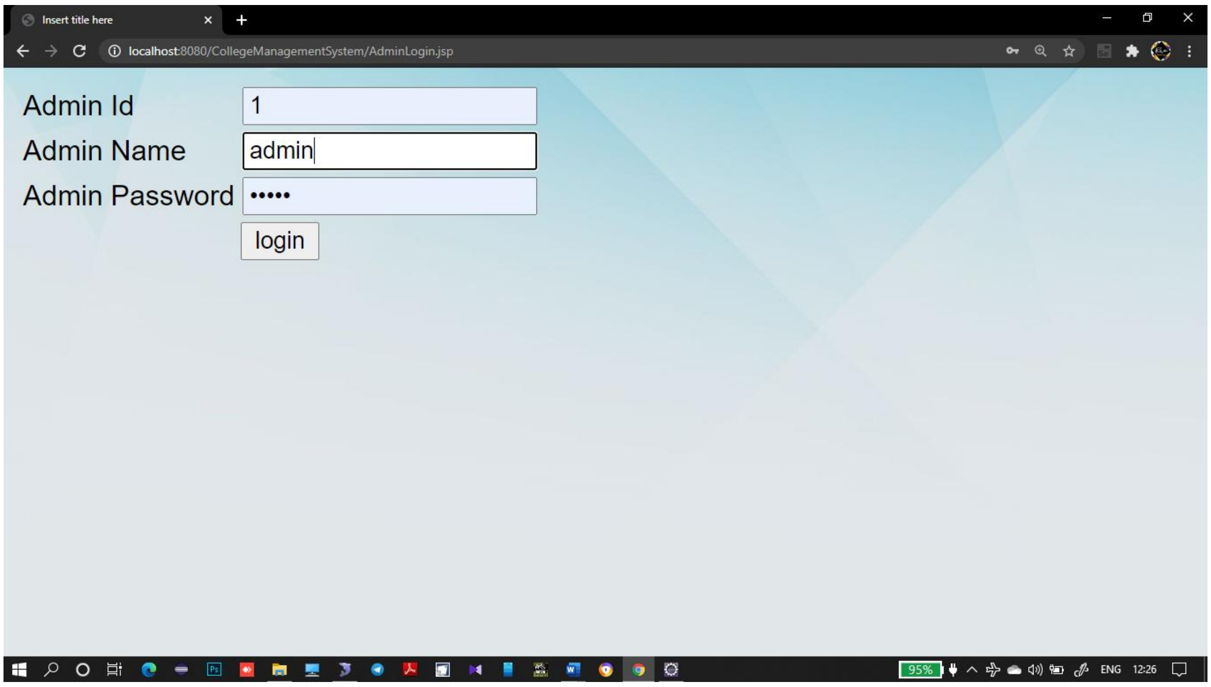
Chapter: 11

Results

Index :

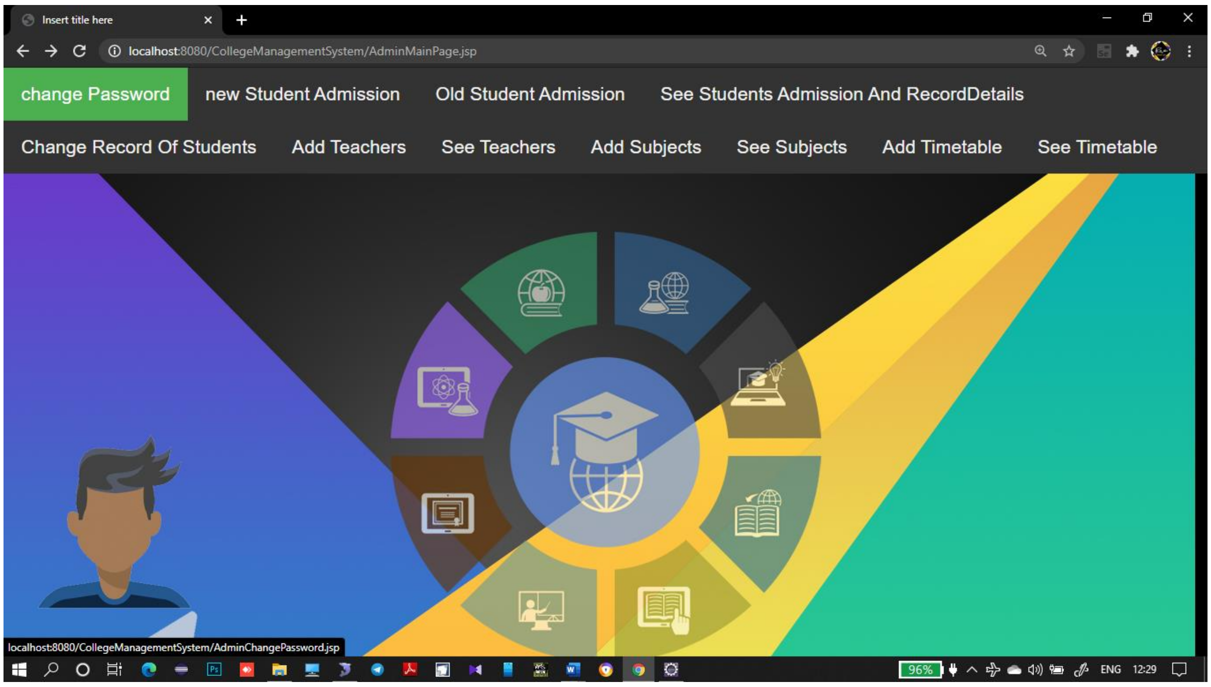


Login page:

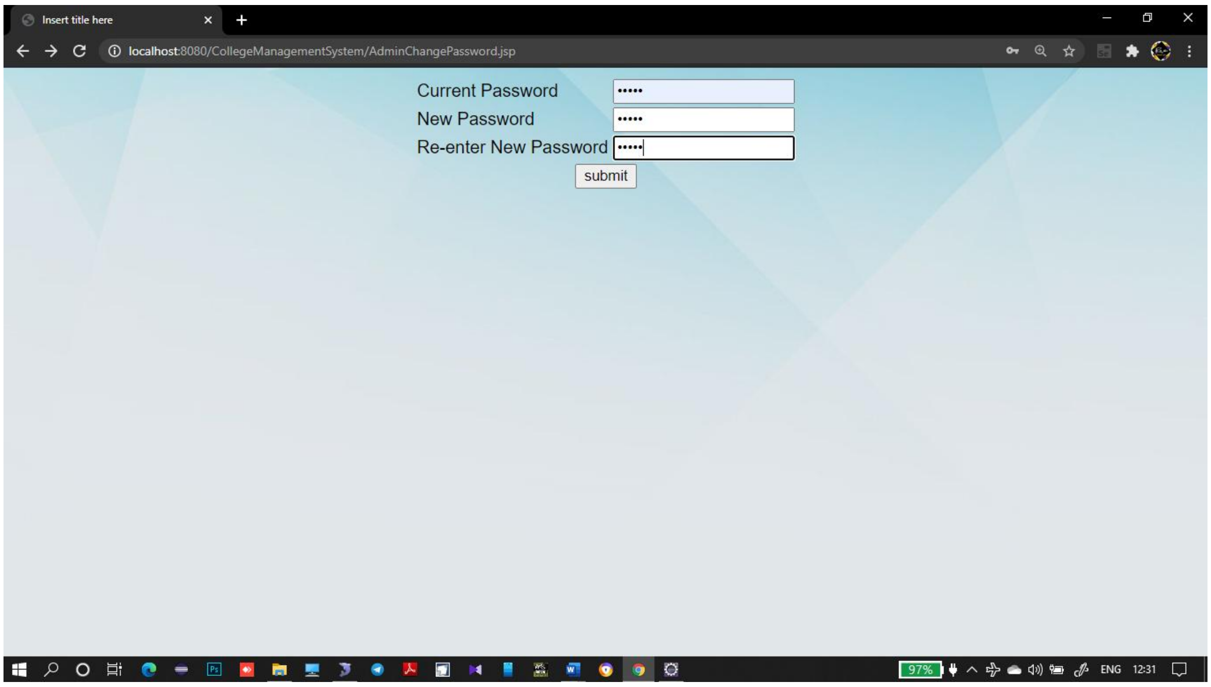


Admin Id 1
Admin Name admin
Admin Password •••••
login

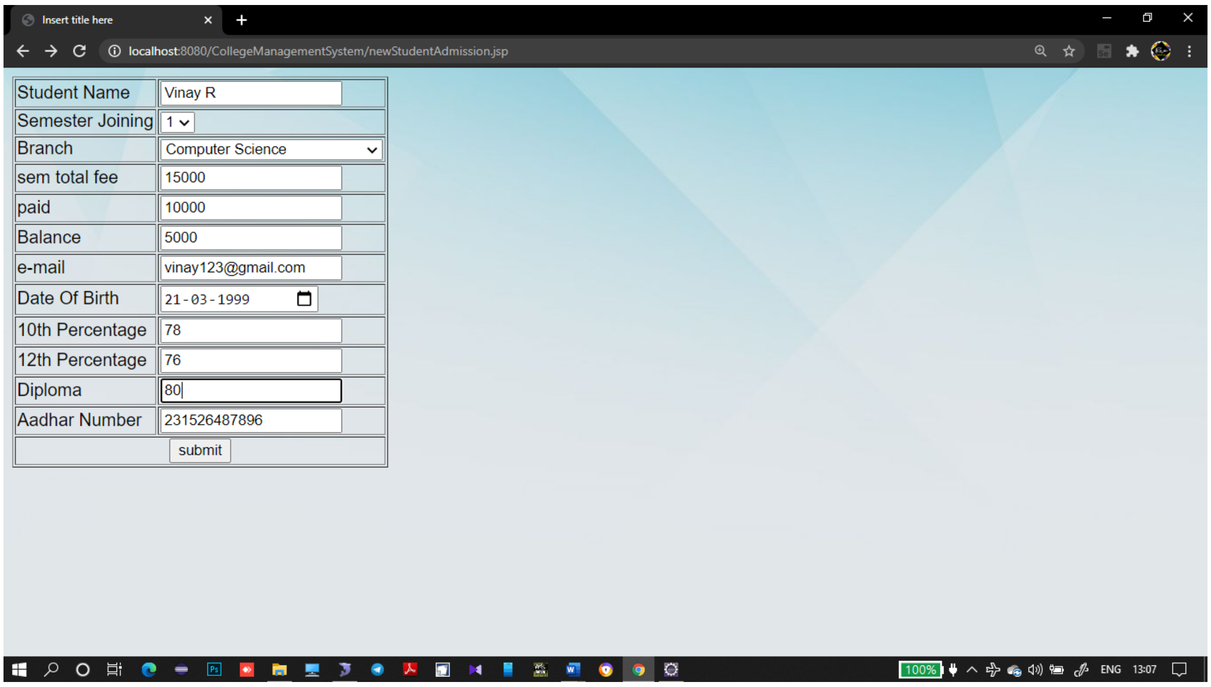
Admin Portal:


Insert title here
localhost:8080/CollegeManagementSystem/AdminMainPage.jsp
change Password
new Student Admission
Old Student Admission
See Students Admission And RecordDetails
Change Record Of Students
Add Teachers
See Teachers
Add Subjects
See Subjects
Add Timetable
See Timetable
localhost:8080/CollegeManagementSystem/AdminChangePassword.jsp

Change Password:



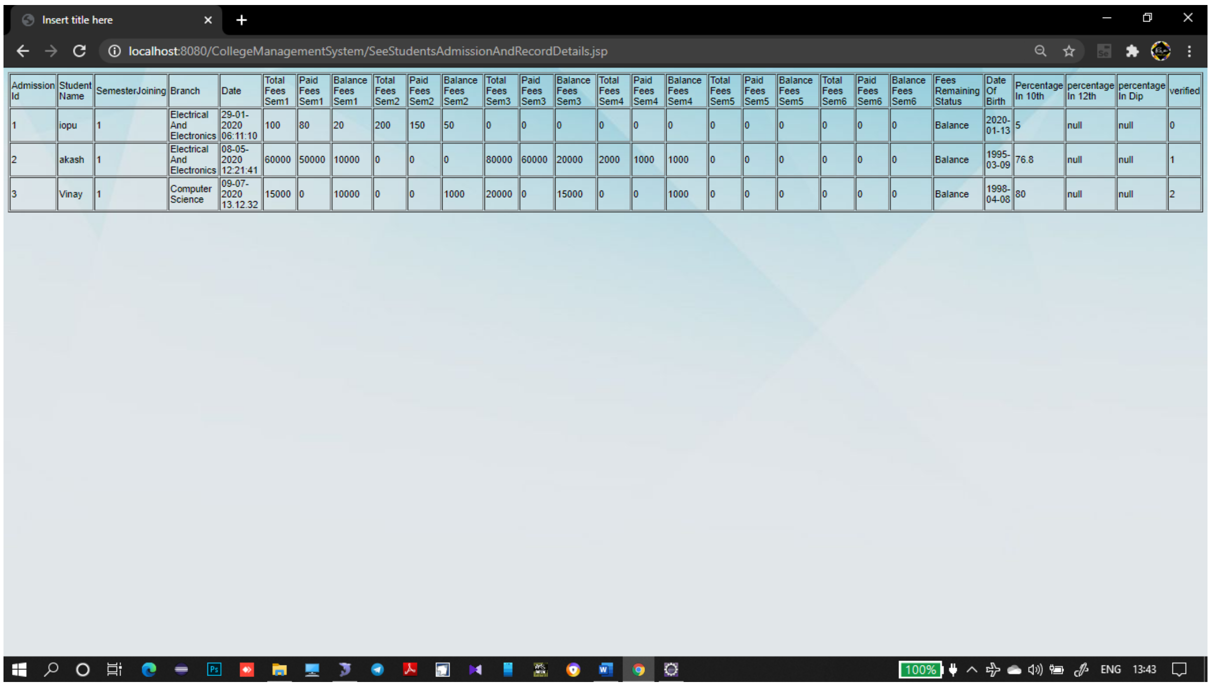
Student Admission Form:



localhost:8080/CollegeManagementSystem/newStudentAdmission.jsp

Student Name	Vinay R
Semester Joining	1
Branch	Computer Science
sem total fee	15000
paid	10000
Balance	5000
e-mail	vinay123@gmail.com
Date Of Birth	21-03-1999
10th Percentage	78
12th Percentage	76
Diploma	80
Aadhar Number	231526487896
	submit

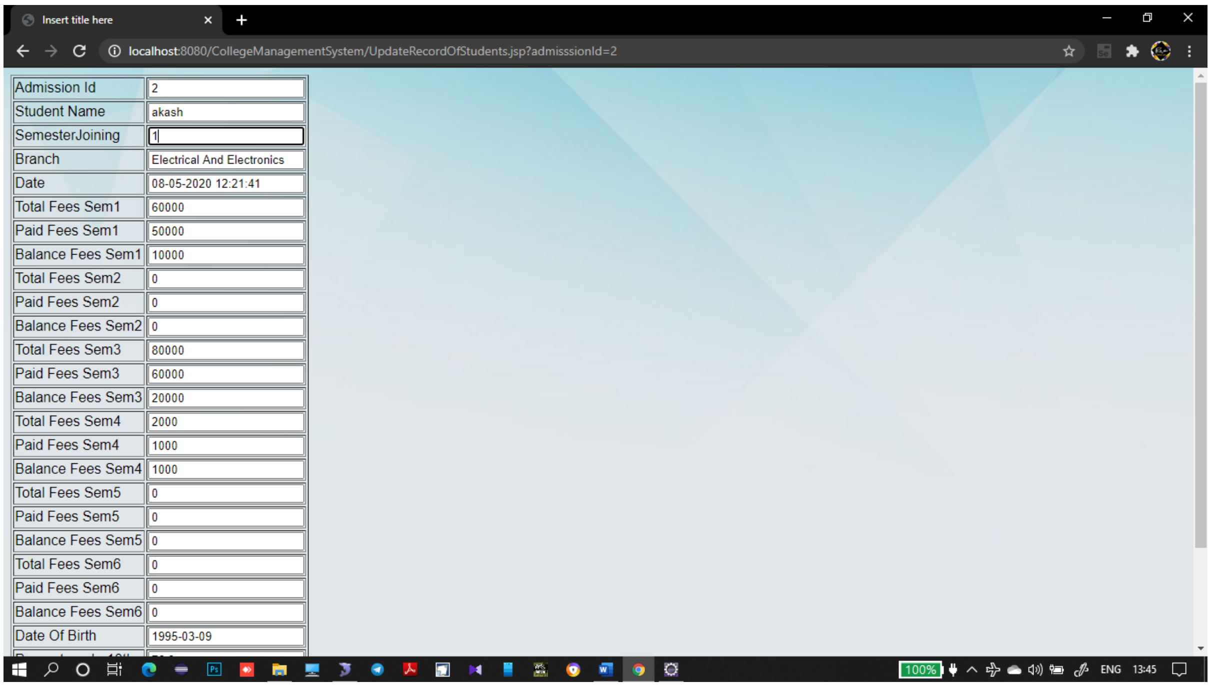
All Student Details:



localhost:8080/CollegeManagementSystem/SeeStudentsAdmissionAndRecordDetails.jsp

Admission Id	Student Name	SemesterJoining	Branch	Date	Total Fees Sem1	Paid Fees Sem1	Balance Fees Sem1	Total Fees Sem2	Paid Fees Sem2	Balance Fees Sem2	Total Fees Sem3	Paid Fees Sem3	Balance Fees Sem3	Total Fees Sem4	Paid Fees Sem4	Balance Fees Sem4	Total Fees Sem5	Paid Fees Sem5	Balance Fees Sem5	Total Fees Sem6	Paid Fees Sem6	Balance Fees Sem6	Fees Remaining Status	Date Of Birth	Percentage In 10th	percentage In 12th	percentage In Dip	verified
1	iopu	1	Electrical And Electronics	29-01-2020 06:11:10	100	80	20	200	150	50	0	0	0	0	0	0	0	0	0	0	0	0	Balance	2020-01-13	5	null	null	0
2	akash	1	Electrical And Electronics	08-05-2020 12:21:41	60000	50000	10000	0	0	0	80000	60000	20000	2000	1000	1000	0	0	0	0	0	0	Balance	1995-03-09	76.8	null	null	1
3	Vinay	1	Computer Science	09-07-2020 13.12.32	15000	0	10000	0	0	1000	20000	0	15000	0	0	1000	0	0	0	0	0	0	Balance	1998-04-08	80	null	null	2

Change Student Details Form:

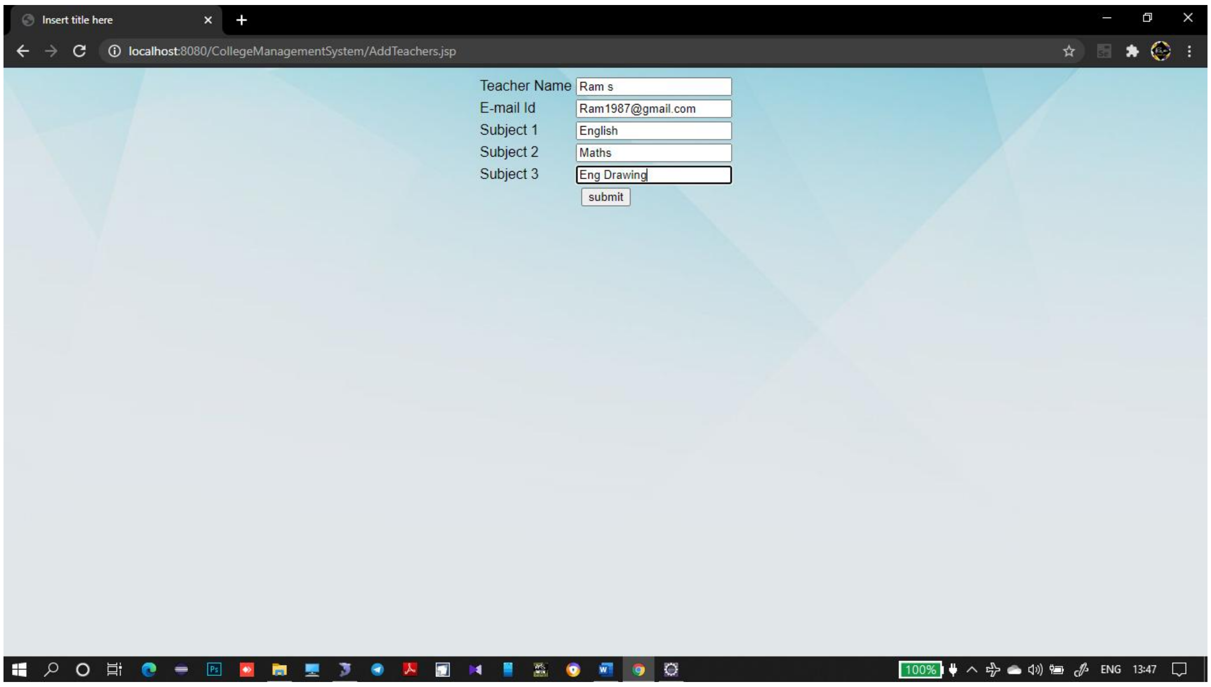


Insert title here

localhost:8080/CollegeManagementSystem/UpdateRecordOfStudents.jsp?admisssionId=2

Admission Id	2
Student Name	akash
SemesterJoining	1
Branch	Electrical And Electronics
Date	08-05-2020 12:21:41
Total Fees Sem1	60000
Paid Fees Sem1	50000
Balance Fees Sem1	10000
Total Fees Sem2	0
Paid Fees Sem2	0
Balance Fees Sem2	0
Total Fees Sem3	80000
Paid Fees Sem3	60000
Balance Fees Sem3	20000
Total Fees Sem4	2000
Paid Fees Sem4	1000
Balance Fees Sem4	1000
Total Fees Sem5	0
Paid Fees Sem5	0
Balance Fees Sem5	0
Total Fees Sem6	0
Paid Fees Sem6	0
Balance Fees Sem6	0
Date Of Birth	1995-03-09

Add Teacher Form:



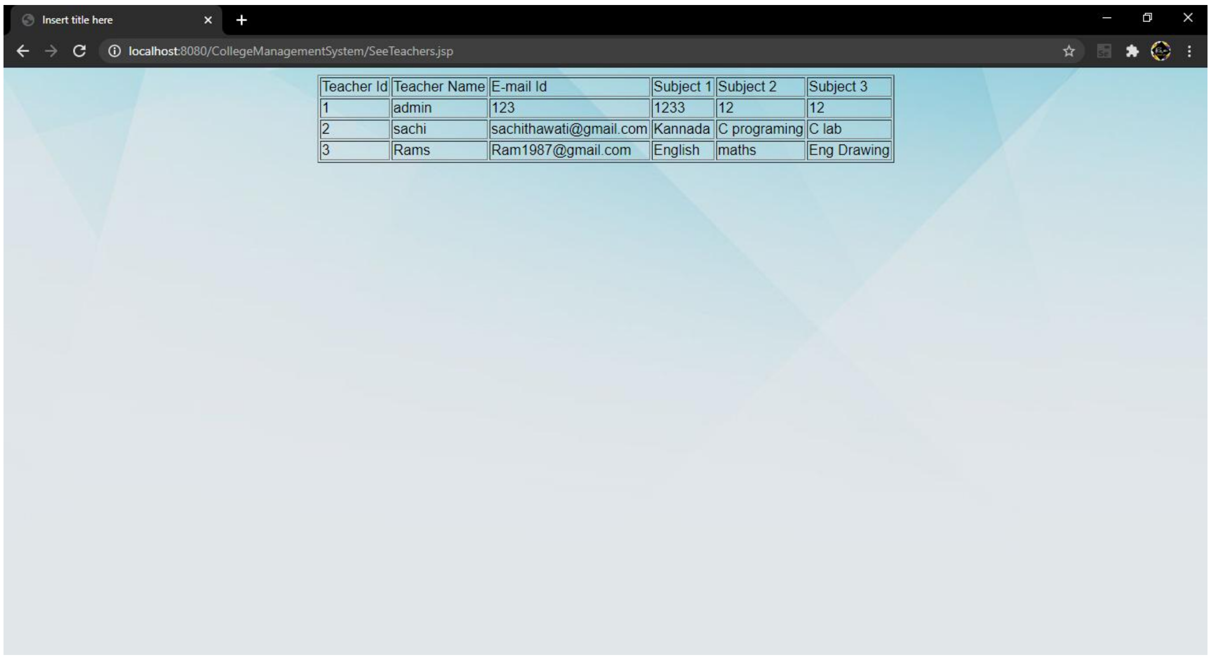
Insert title here

localhost:8080/CollegeManagementSystem/AddTeachers.jsp

Teacher Name	Ram s
E-mail Id	Ram1987@gmail.com
Subject 1	English
Subject 2	Maths
Subject 3	Eng Drawing

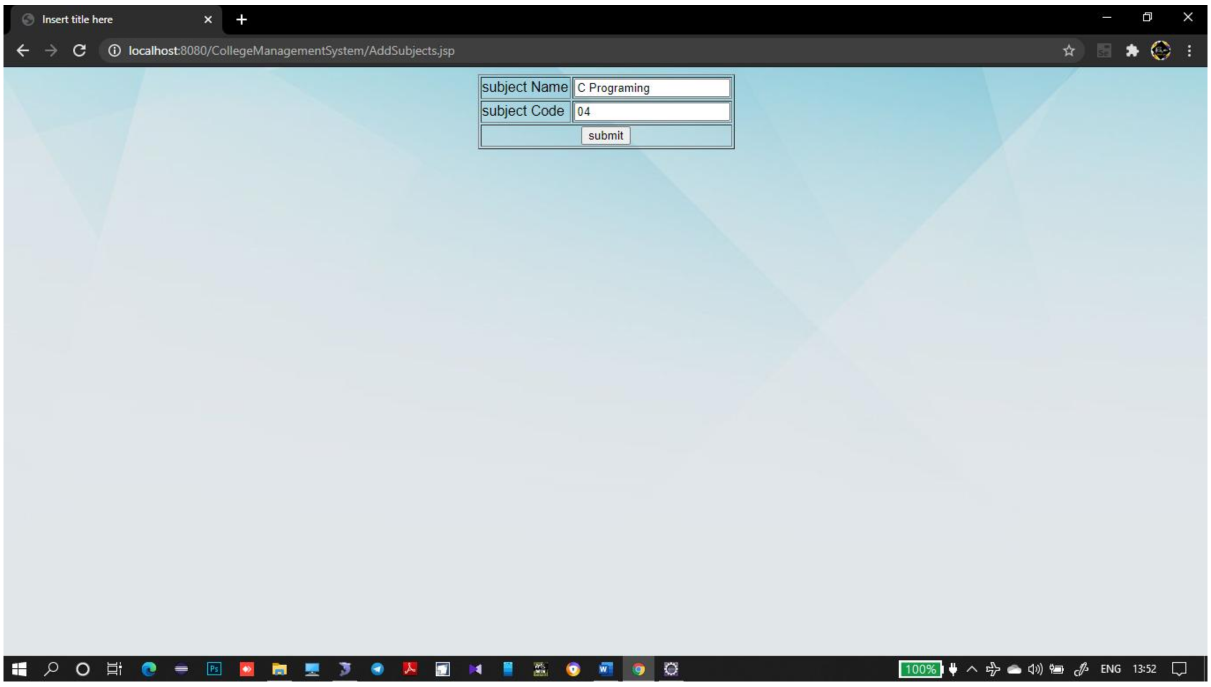
submit

View Teacher Form:



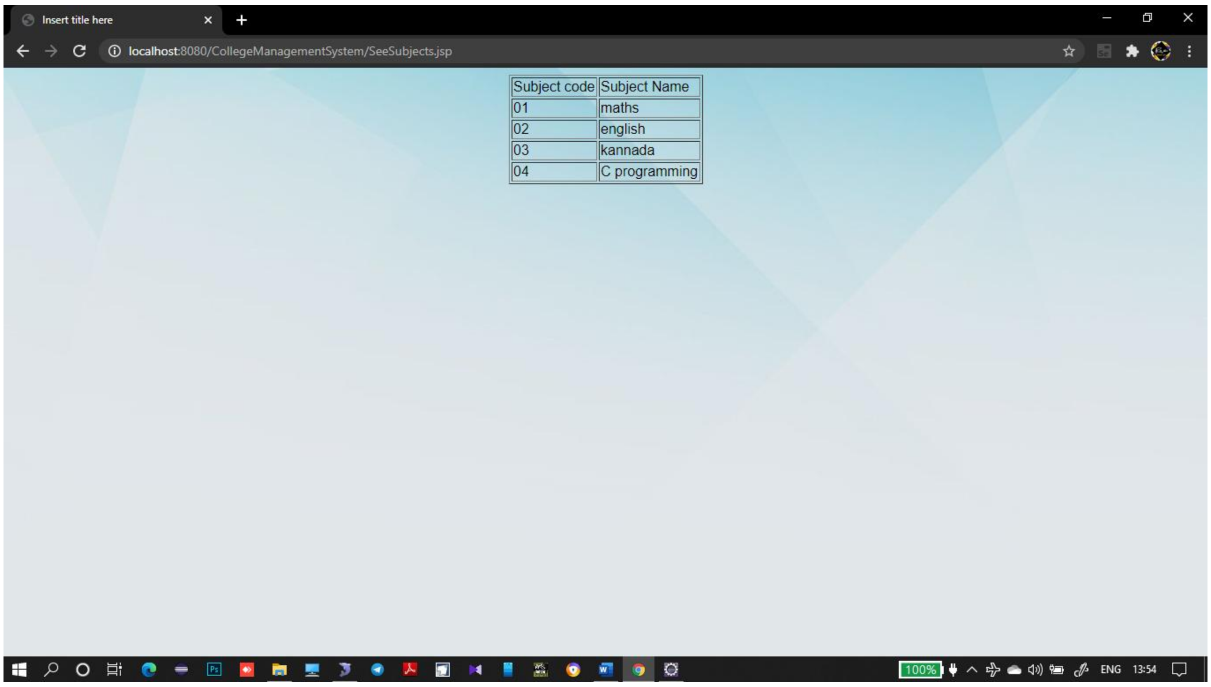
Teacher Id	Teacher Name	E-mail Id	Subject 1	Subject 2	Subject 3
1	admin	123	1233	12	12
2	sachi	sachithawati@gmail.com	Kannada	C programing	C lab
3	Rams	Ram1987@gmail.com	English	maths	Eng Drawing

Add Subject Form:



subject Name	C Programing
subject Code	04

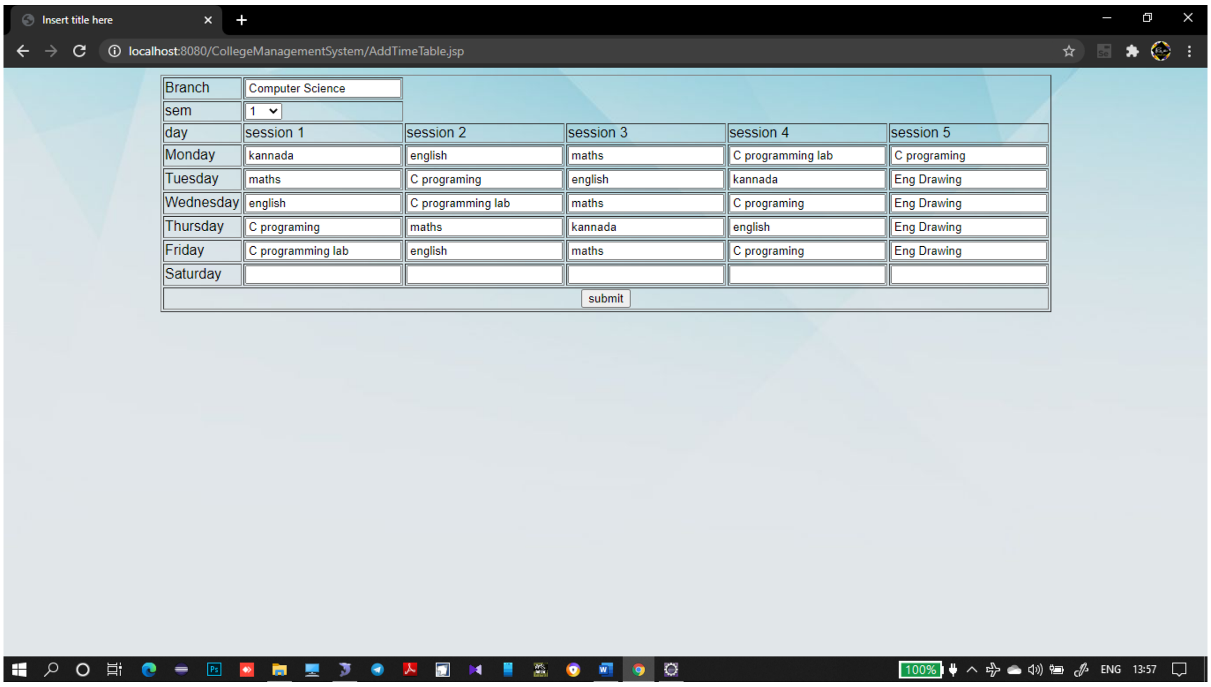
View Subject Form:



localhost:8080/CollegeManagementSystem/SeeSubjects.jsp

Subject code	Subject Name
01	maths
02	english
03	kannada
04	C programming

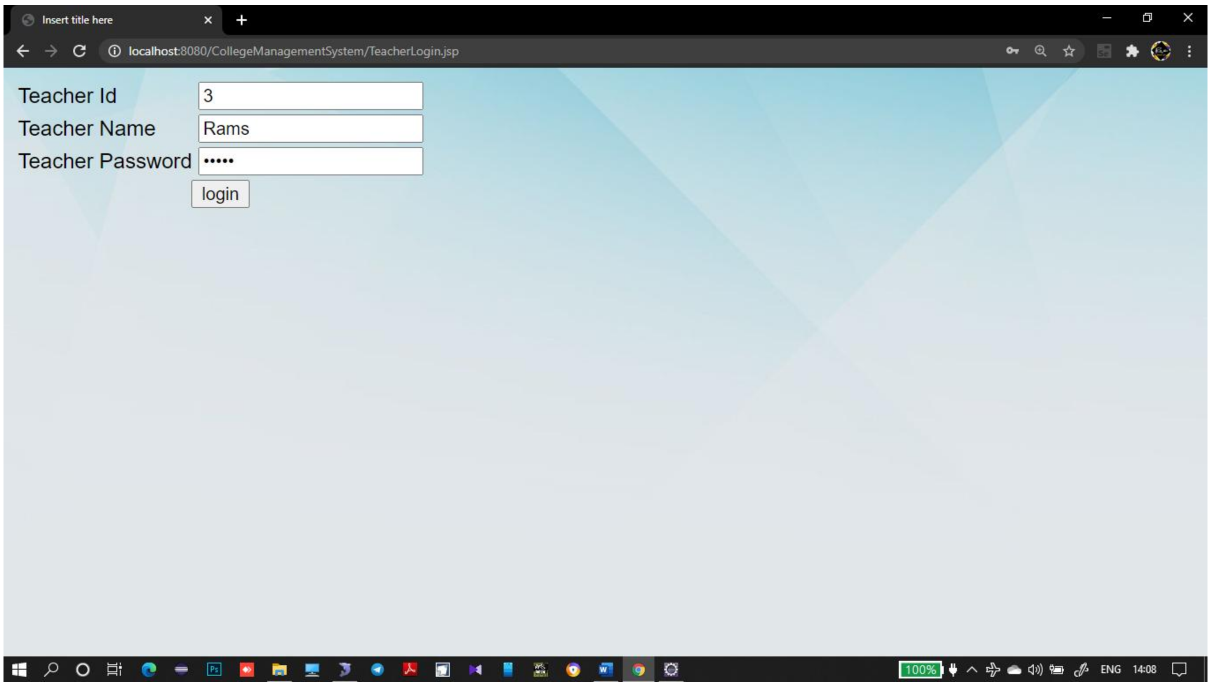
Add Time Table Form:



localhost:8080/CollegeManagementSystem/AddTimeTable.jsp

Branch	Computer Science				
sem	1				
day	session 1	session 2	session 3	session 4	session 5
Monday	kannada	english	maths	C programming lab	C programing
Tuesday	maths	C programing	english	kannada	Eng Drawing
Wednesday	english	C programming lab	maths	C programing	Eng Drawing
Thursday	C programing	maths	kannada	english	Eng Drawing
Friday	C programming lab	english	maths	C programing	Eng Drawing
Saturday					

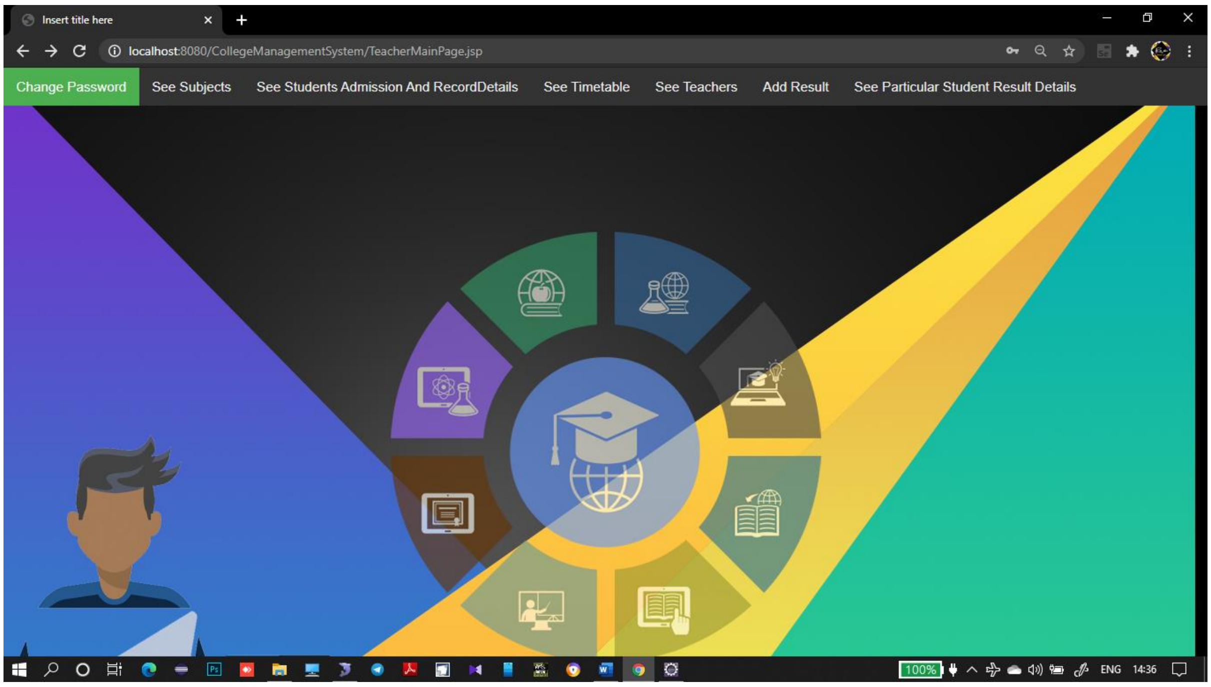
Teacher Login:



Teacher Id: 3

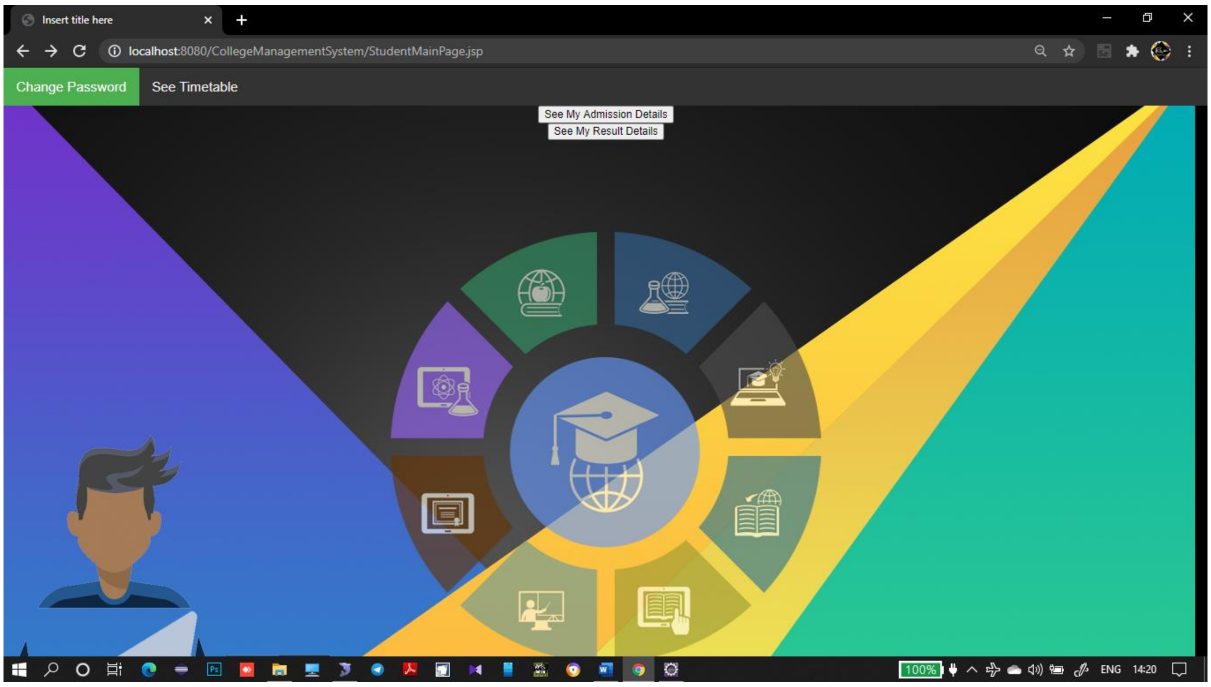
Teacher Name: Rams

Teacher Password: •••••

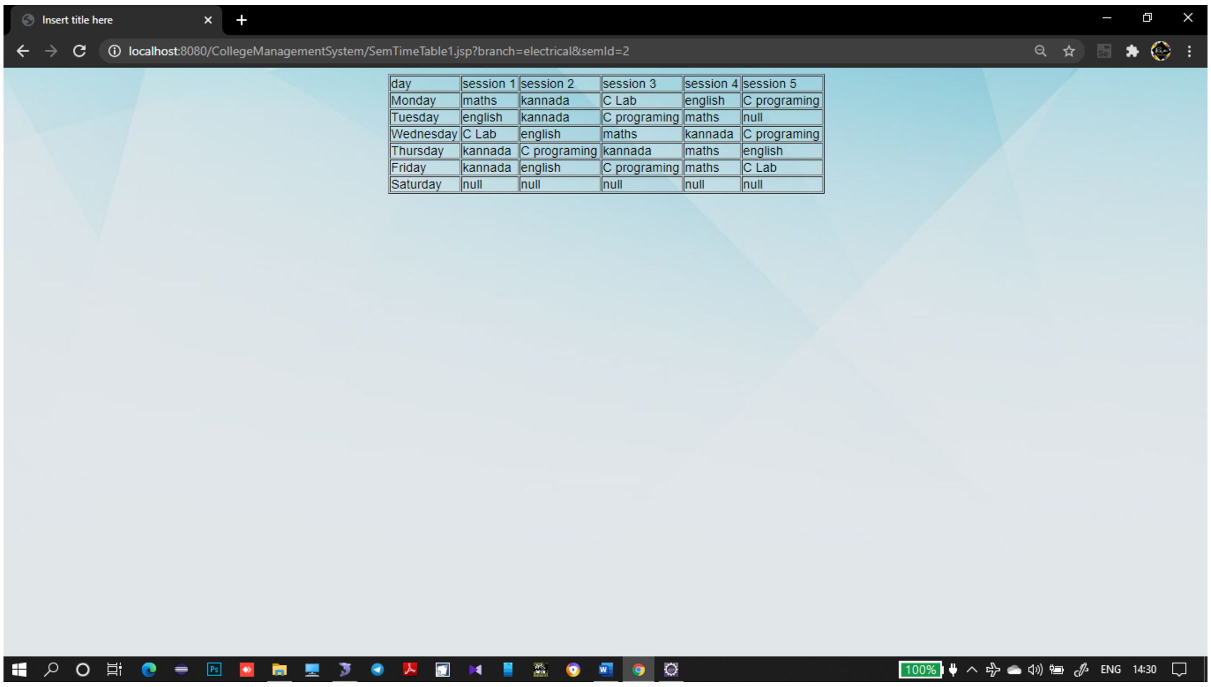


Insert title here
localhost:8080/CollegeManagementSystem/TeacherMainPage.jsp
Change Password
See Subjects
See Students Admission And RecordDetails
See Timetable
See Teachers
Add Result
See Particular Student Result Details
100%
ENG
14:36

Student Portal:



Student TimeTable:


Insert title here
localhost:8080/CollegeManagementSystem/SemTimeTable1.jsp?branch=electrical&semId=2

day	session 1	session 2	session 3	session 4	session 5
Monday	maths	kannada	C Lab	english	C programing
Tuesday	english	kannada	C programing	maths	null
Wednesday	C Lab	english	maths	kannada	C programing
Thursday	kannada	C programing	kannada	maths	english
Friday	kannada	english	C programing	maths	C Lab
Saturday	null	null	null	null	null

Chapter: 12

Future Enhancement

- Online examination module would be introduced to conduct online examination
- Scheduling of the staff. i.e., time table setting of the staff
- Further, the faculty can upload the videos of their lectures on to this site and students who had missed those classes can view those videos.

CONCLUSION

The project entitled as **College Management System** is the system that deals with the issues related to a particular institution.

- This project is successfully implemented with all the features mentioned in system requirements specification.
- The application provides appropriate information to users according to the chosen service.
- The project is designed keeping in view the day to day problems faced by a college.
- Deployment of our application will certainly help the college to reduce unnecessary wastage of time in personally going to each department for some information.

Awareness and right information about any college is essential for both the development of student as well as faculty. So this serves the right purpose in achieving the desired requirements of both the communities.

Chapter:13

REFERENCES

Books:

1. Internet & World Wide Web: How to Program Deitel, PJ Deitel.
2. Web Development with Java Server Pages BY Duane K.Fields and Mark A.Kolb.
3. The Complete Reference Java2 HerbertSchildt.
4. Core Servlets and Java Server Pages By Marty Hall.
5. Apache Jakarta-Tomcat by James Goodwill.
6. Practical PostgreSQL by John Worsley, Joshua Drake

Web Sites:

1. www.sves-srpt.ac.in
2. www.kings.cam.ac.uk
3. www.wellington-college.school.nz
4. www.mysqltutorial.org.