A Project Report on

KMIT ONLINE GATE PASS APPLICATION

Submitted to

Jawaharlal Nehru Technological University, Hyderabad in partial fulfillment of requirements for the award of the degree of **BACHELOR OF TECHNOLOGY**

In

COMPUTER SCIENCE AND ENGINEERING

By

Vavilala Sai Vardhan Sagar	16BD1A05D1
Thumma John Vivek Reddy	16BD1A05CK
Vangaru Shanmukha	16BD1A05CU
T Bhargav Narayana	16BD1A05C6

Under the esteemed guidance of SHANKER M MACCHA

(Assistant/Associate)

Professor Department of CSE

Department of Computer Science and Engineering

KESHAV MEMORIAL INSTITUE OF TECHNOLOGY

Approved by AICTE, Affiliated to JNTUH 3-5-1206, Narayanaguda, Hyderabad - 500029

2019 – 2020

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

Approved by AICTE, Affiliated to JNTU,

Hyderabad 3-5-1206, Narayanaguda,

Hyderabad - 500029.

CERTIFICATE

This is to certify that the project entitled "**KMIT ONLINE GATE PASS APPLICATION**" being submitted by **Mr. VAVILALA SAI VARDHAN SAGAR (16BD1A05D1), Mr. THUMMA JOHN VIVEK REDDY (16BD1A05CK), Mr. VANGARU SHANMUKHA (16BD1A05CU), Mr. T BHARGAV NARAYANA (16BD1A05C6)** students of **Keshav Memorial Institute of Technology, JNTUH** in partial fulfillment of the requirements of the award of the Degree of **Bachelor of Technology in Computer Science and Engineering** as a specialization is a record of bonafide work carried out by them under my guidance and supervision in the academic year 2019 – 2020.

GUIDE	**HEAD OF THE DEPARTMENT**
SHANKER M MACCHA	**Dr. S.Padmaja**
(Assistant/ Associate) Professor	HoD-CSE

Submitted for the Project Viva Voce examination held on …………….

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project report entitled **“KMIT ONLINE GATE PASS APPLICATION”** is done in the partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering affiliated to Jawaharlal Nehru Technological University, Hyderabad. This project has not been submitted anywhere else.

Vavilala Sai Vardhan Sagar	**16BD1A05D1**
Thumma John Vivek Reddy	**16BD1A05CK**
Vangaru Shanmukha	**16BD1A05CU**
T Bhargav Narayana	**16BD1A05C6**

ACKNOWLEDGEMENT

We take this opportunity to thank all the people who have rendered their full support to our project work. We render our thanks to **Dr. Maheshwar Dutta, B.E., M Tech., Ph.D**., Principal who encouraged us to do the Project. We are grateful to **Mr. Neil Gogte,** Director for facilitating all the amenities required for carrying out this project. We express our sincere gratitude to **Mr. S. Nitin**, Director and **Mrs. Deepa Ganu**, Dean Academics for providing an excellent environment in the college. We are also thankful to **Dr. S. Padmaja**, Head of the Department for providing us with both time and amenities to make this project a success within the given schedule. We are also thankful to our guide **Mr. Shanker M Maccha**, for his valuable guidance and encouragement given to us throughout the project work. We would like to thank the entire CSE Department faculty, who helped us directly and indirectly in the completion of the project. We sincerely thank our friends and family for their constant motivation during the project work.

Vavilala Sai Vardhan Sagar	**16BD1A05D1**
Thumma John Vivek Reddy	**16BD1A05CK**
Vangaru Shanmukha	**16BD1A05CU**
T Bhargav Narayana	**16BD1A05C6**

CONTENTS

ABSTRACT

Abstract:

As we all know that getting a gate pass is a tedious task in any of the colleges. First, we must take the gate pass slip and sign it with our coordinators, teacher and HOD. So, instead of this tedious process, we have devised a suitable web application that allows you to obtain gate pass without putting much effort. The application is used by three end users – Student, Coordinator and Watchmen. Therefore, database consists of three tables. Whenever a student requests for a gate pass, based on the department of the student the request will be directed to the respective coordinator. Based on the reason specified in the request, the coordinator may approve or reject it. Until then, the request will be under processing state. Based on the action of the coordinator the value of the status field will change accordingly along the watchmen database. Student should show his ID card as a proof.

1. INTRODUCTION

1.1 Purpose of the Project

In some organizations, even in this modern era watchmen using register to the exit about the person/student which means no authentication is done by guard which results in many miss happenings.

The problem with the current system:

- It involves lot of paper usage.
- As the gate pass is a simple paper document, any student can easily duplicate it which results in undesired people getting out of the premises.
- There is no way of knowing who and when had gone out.
- There is no authentication process.
- It involves a tedious process of getting an approval for gate pass.
- Noting data in the register is very laborious as well as time consuming.

Our current project provides a solution to the above problems or limitations in the following manner:

- It reduces the paper usage as it is online based.
- It removes the cause of miss happenings as the approval comes directly from the coordinator to the watchmen.
- Each request and its result – approve/reject is stored in database.
- The student must verify his identity through ID card while going out.
- As it is online based, it reduces the overload on the student to get the approval of the coordinator.
- It removes the need of data entry in the registers.

1.2 How it Works?

An end user must first register himself/herself in order to use the application. For registration the end-user must provide his mail-id to which an OTP will be sent to verify his identity. Once registered user can login into the application and explore various events. The application is used by three types of end-users – student, coordinator and watchmen. Based on the user credentials he will be directed to the respective page.

End-users:

Student:

- Student will login into the application using his roll number and password.
- Student dashboard consists of gate pass form and other set of options.
- The student needs to fill the gate pass form in order to raise a request.
- Once the student submits the form, the request will be directed to the respective coordinator.

Coordinator:

- Coordinator will login into the application using his mail id.
- Once the coordinator login into the application he /she will be displayed with the list of pending requests.
- He/she will be provided with approve/reject button for every request.
- Based on his/her action the request will be directed to the watchmen dashboard/database.

Watchmen:

- Watchmen will login into the application using his id number.
- Once the watchmen login into the application he/she will be displayed with list of latest requests.

Administrator:

The admin will be responsible for adding and removing the details of end-users. Checking the authenticity of request, updating the details of the end-users and other miscellaneous functions will come under the purview of admin.

2. SOFTWARE REQUIREMENT SPECIFICATIONS

2.1 What is SRS?

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.)

The SRS phase consists of two basic activities:

Problem/Requirement Analysis:

The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

Requirement Specification:

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SR document. Producing the SRS document is the basic goal of this phase.

2.2 Role of SRS

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium though which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

2.3 Requirements Specification Document

A Software Requirements Specification (SRS) is a document that describes the nature of a project, software or application. In simple words, SRS document is a manual of a project provided it is prepared before you kick-start a project/application. This document is also known by the names SRS report, software document. A software document is primarily prepared for a project, software or any kind of application.

There are a set of guidelines to be followed while preparing the software requirement specification document. This includes the purpose, scope, functional and nonfunctional requirements, software and hardware requirements of the project. In addition to this, it also contains the information about environmental conditions required, safety and security requirements, software quality attributes of the project etc.

The purpose of SRS (Software Requirement Specification) document is to describe the external behavior of the application developed or software. It defines the operations, performance and interfaces and quality assurance requirement of the application or software. The complete software requirements for the system are captured by the SRS.

This section introduces the requirement specification document for Web Scraping and Sentimental Analysis for amazon products using Python which enlists functional as well as non-functional requirements.

2.4 Functional Requirements

For documenting the functional requirements, the set of functionalities supported by the system are to be specified. A function can be specified by identifying the state at which data is to be input to the system, its input data domain, the output domain, and the type of processing to be carried on the input data to obtain the output data. Functional requirements define specific behavior or function of the application. Following are the functional requirements: The input design is the link between the information system and user. It compromises the developing specification and procedures for data preparation and those steps are necessary to put transaction data into a useable form for processing can be achieved by inspecting the computer to read data from a written or printed documented or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input design considered for the following things:

- What data should be given as input?
- How the data should be arranged or recorded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Input design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management correct information from the computerized system. It is achieved by creating user-friendly screens for data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulations can be performed. It also provides viewing facilities. When the data is entered, it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that user will not be in maize of instant.

2.4.1 Functional Requirements for our project:

- User should be able to log into the application.
- Student should be able make a request for gate pass.
- Coordinator should be able to act for the request made by the student.
- Watchmen should be able to view the latest requests.
- Admin should be able to alter the contents of the database.

2.5 Non-Functional Requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Especially these are the constraints the system must work within.

Following are the non-functional requirements:

Performance:

The performance of the developed applications can be calculated by using following methods:

Measuring enables you to identify how the performance of your application stands in relation to your defined performance goals and helps you to identify the bottlenecks that affect your application performance. It helps you identify whether your application is moving toward or away from your performance goals. Defining what you will measure, that is, your metrics, and defining the objectives for each metric is a critical part of your testing plan.

Performance objectives include the following:

- Response time or latency
- Throughput
- Resource utilization

2.5.1 Non-Functional Requirements for our project:

- Availability: The system should be available 24 hours per 6 days.
- Security: The system should hide users' password.
- Usability: The Users should learn how to use the system at most in 2hours.
- Maintainability: The system should be closed 24 hours per week for maintenance and elaboration (e.g. Mean Time to Repair -MTTR).
- Reliability: The system must be 100% reliable due to the importance of data and the damages that can be caused by incorrect or incomplete data.
- Recover ability: The system should never crash more than 10 minutes per month. (e.g. Mean Time Between/To Failures -MTBF/MTTF)

2.6 Software Requirements

Platform	:	Eclipse
Database	:	MySQL Server, Workbench
Operation System	:	Windows XP, Windows 7, Windows
8or 10 Web Server	:	Apache tomcat server

2.7 Hardware Requirements

Processor	:	Intel Core Duo 2.0 GHz or more
RAM	:	2 GB or More
Hard disk	:	80GB or more
Monitor	:	15” CRT, or LCD monitor
Internet	:	speed of 1Mbps or higher
Keyboard	:	Normal or Multimedia
Mouse	:	Compatible mouse

3. LITERATURE SURVEY

3.1 SDLC (Software Development Life Cycle):

The process of testing software in a well-planned and systematic way is known as software testing life cycle (STLC). Different organizations have different phases in STLC however generic Software Test Life Cycle (STLC) for waterfall development model consists of the following phases.

1. Requirements Analysis
2. Test Planning
3. Test Analysis
4. Test Design
5. Test Construction and Verification
6. Test Execution and Bug Reporting
7. Final Testing and Implementation
8. Post Implementation

1. Requirements Analysis:

In this phase testers analyze the customer requirements and work with developers during the design phase to see which requirements are testable and how they are going to test those requirements. It is very important to start testing activities from the requirements phase itself because the cost of fixing defect is very less if it is found in requirements phase rather than in future phases. In this phase all the planning about testing is done like what needs to be tested, how the testing will be done, test strategy to be followed, what will be the test environment, what test methodologies will be followed, hardware and software availability, resources, risks etc. A high-level test plan document is created which includes all the planning inputs mentioned above and circulated to the stakeholders.

2. Test Analysis:

After test planning phase is over test analysis phase starts, in this phase we need to dig deeper into project and figure out what testing needs to be carried out in each SDLC phase. Automation activities are also decided in this phase, if automation needs to be done for software product, how will the automation be done, how much time will it take to automate, and which features need to be automated. Nonfunctional testing areas (Stress and performance testing) are also analyzed and defined in this phase.

3. Test Design:

   In this phase various black-box and white-box test design techniques are used to design the test cases for testing, testers start writing test cases by following those design techniques, if automation testing needs to be done then automation scripts also needs to write in this phase.

4. Test Construction and Verification:

   In this phase testers prepare more test cases by keeping in mind the positive and negative scenarios, end user scenarios etc. All the test cases and automation scripts need to be completed in this phase and got reviewed by the stakeholders. The test plan document should also be finalized and verified by reviewers.

5. Test Execution and Bug Reporting:

   Once the unit testing is done by the developers and test team gets the test build, the test cases are executed and defects are reported in bug tracking tool, after the test execution is complete and all the defects are reported. Test execution reports are created and circulated to project stakeholders. After developers fix the bugs raised by testers, they give another build with fixes to testers, testers do re- testing and regression testing to ensure that the defect has been fixed and not affected any other areas of software. Testing is an iterative process i.e. If defect is found and fixed, testing needs to be done after every defect fix. After tester assures that defects have been fixed and no more critical defects remain in software the build is given for final testing.

6. Final Testing and Implementation:

   In this phase the final testing is done for the software, non-functional testing like stress, load and performance testing are performed in this phase. The software is also verified in the production kind of environment. Final test execution reports and documents are prepared in this phase.

7. Post Implementation:

   In this phase the test environment is cleaned up and restored to default state, the process review meetings are done, and lessons learn are documented. A document is prepared to cope up similar problems in future releases.

3.2 Technologies used

1. HTML:

Hypertext Markup Language is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img/> and <input/> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

2. CSS:

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

3. JavaScript:

JavaScript, often abbreviated as JS, is a high-level, interpreted scripting language that conforms to the ECMA Script specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object orientation, and first-class functions. As a multi-paradigm language, java script supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles.

It has APIs for working with text, arrays, dates, regular expressions, and the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities. It relies upon the host environment in which it is embedded to provide these features. Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in run time environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

4. Bootstrap:

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS- and JavaScript-based design templates for typography, forms, buttons, navigation and other interface components.

5. MySQL:

MySQL is an open-source relational database management system. Its name is a combination of “My”, the name of co-founders Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Micro systems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

6. Java Server Pages (JSP):

Java Server Pages (JSP) is a technology for developing Webpages that supports dynamic content. This helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>. A Java Server Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands. Using JSP, you can collect input from users through Webpage forms, present records from a database or another source, and create Webpages dynamically. JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages, and sharing information between requests, pages etc.

7. Apache Tomcat:

A Java servlet is a Java software component that extends the capabilities of a server. Although servlets can respond to many types of requests, they most commonly implement web containers for hosting web applications on web servers and thus qualify as a server-side servlet web API. Catalina is Tomcat's servlet container. Catalina implements Sun Microsystems's specifications for servlet and Java Server Pages (JSP). In Tomcat, a Realm element represents a "database" of usernames, passwords, and roles (similar to Unix groups) assigned to those users. Different implementations of Realm allow Catalina to be integrated into environments where such authentication information is already being created and maintained, and then use that information to implement Container Managed Security as described in the servlet Specification.

4. SYSTEM DESIGN

4.1 Introduction to UML

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic, semantic and pragmatic rules. A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagrams, which is as follows:

1. User Model View:
   - This view represents the system from the users' perspective.
   - The analysis representation describes a usage scenario from the end-users' perspective.
2. Structural Model View:
   - In this model, the data and functionality are arrived from inside the system.
   - This model view models the static structures.

3. Behavioural Model View:
   - It represents the dynamic of behavioural as parts of the system, depicting he interactions of collection between various structural elements described in the user model and structural model view.

4. Implementation Model View:
   - In this view, the structural and behavioural as parts of the system are represented as they are to be built.

5. Environmental Model View:
   - In this view, the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

4.2 UML Diagrams

➢ Use case Diagram

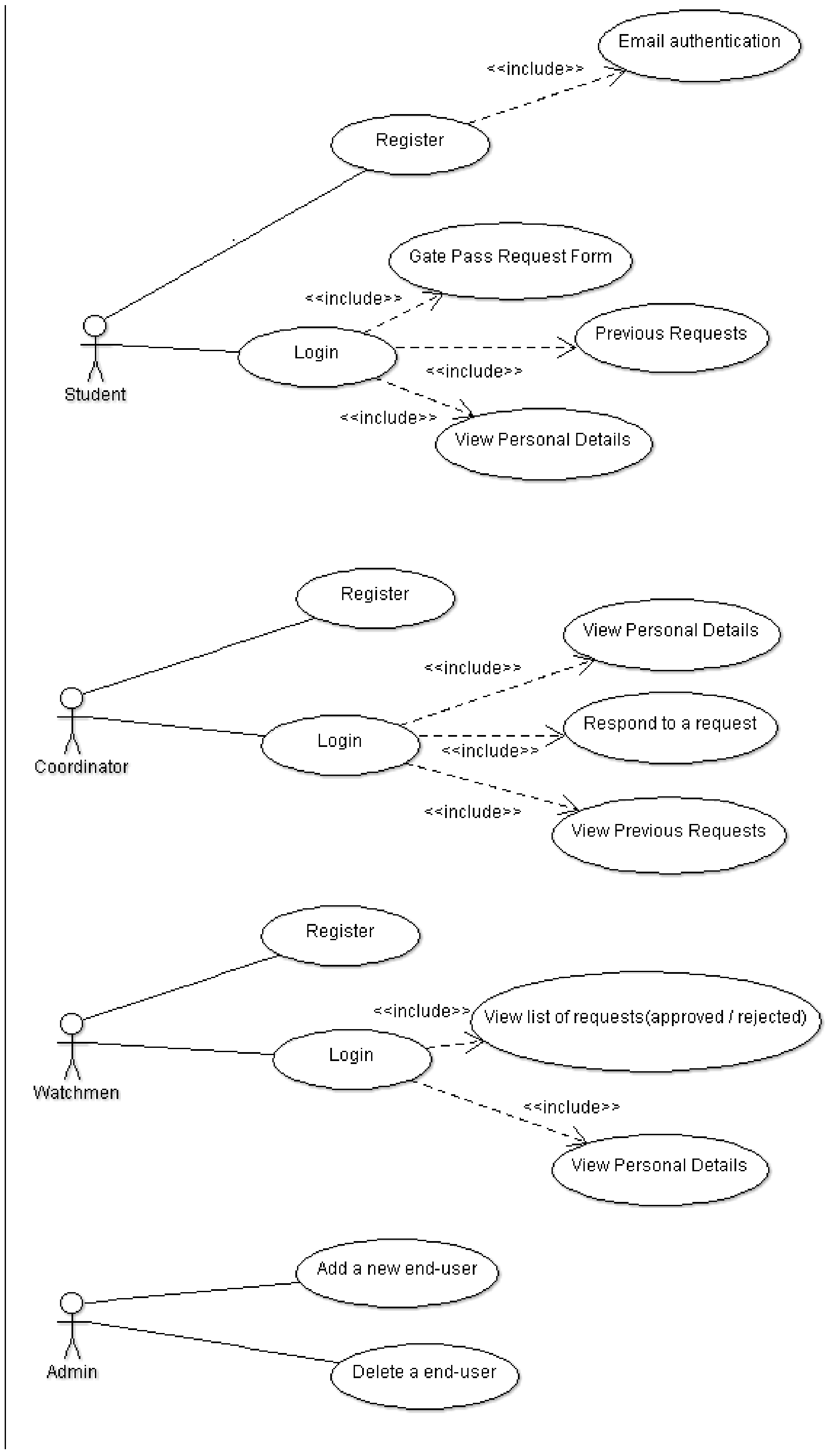
To model a system, the most important aspect is to capture the dynamic behaviour. To clarify a bit in detail, dynamic behaviour means the behaviour of the system when it is running /operating. So only static behaviour is not enough to model a system rather dynamic behaviour is more important than static behaviour. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Now as we must discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. So, use case diagrams are consisting of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a functionality of a system. So, to model the entire system numbers of use case diagrams are used.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So, when a system is analyzed to gather its functionalities use cases are prepared and actors are identified. In brief, the purposes of use case diagrams can be as follows:

- Used to gather requirements of a system.
- Used to get an outside view of a system.
- Identify external and internal factors influencing the system.
- Show the interacting among the requirements are actors

Use case diagram of our project:



➢ Sequence Diagram

. Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. A sequence diagram is a good way to visualize and validate various runtime scenarios. These can help to predict how a system will behave and to discover responsibilities a class may need to have in the process of modeling a new system. The aim of a sequence diagram is to define event sequences, which would have a desired outcome. The focus is more on the order in which messages occur than on the message per se. However, most sequence diagrams will communicate what messages are sent and the order in which they tend to occur.

Basic Sequence Diagram Notations

- **Class Roles or Participants**

  Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.

- **Activation or Execution Occurrence**

  Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.

- **Messages**

  Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

- **Lifelines**

  Lifelines are vertical dashed lines that indicate the object's presence over time.

- **Destroying Objects**

  Objects can be terminated early using an arrow labeled "<< destroy >>" that points to an X. This object is removed from memory.

  When that object's lifeline ends, you can place an X at the end of its lifeline to denote a destruction occurrence.

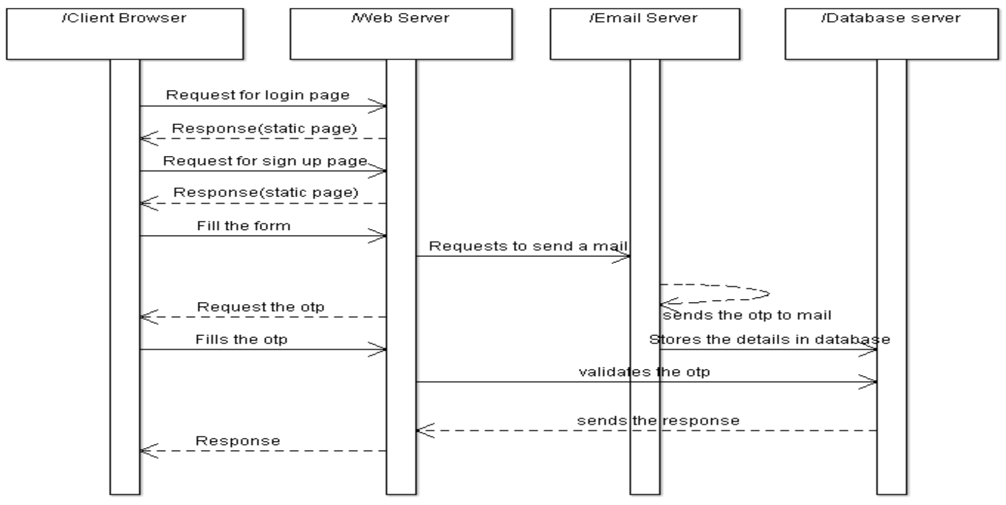
- **Loops**

  A repetition or loop within a sequence diagram is depicted as rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [].

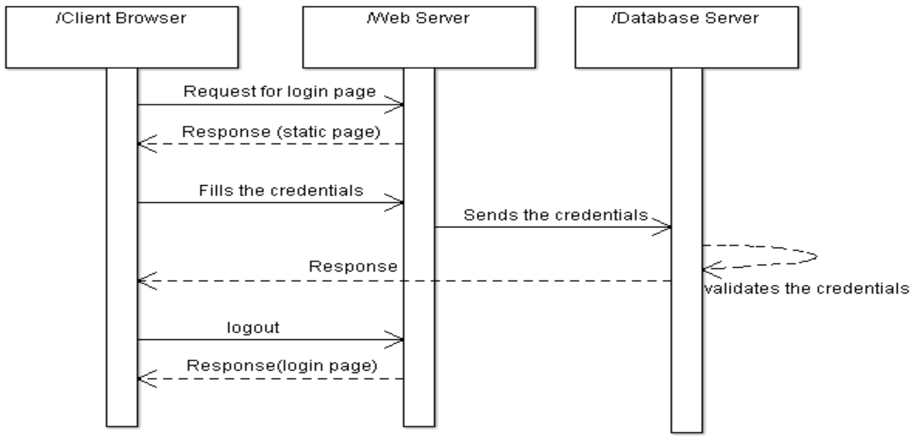
- **Guards**

  When modelling object interactions, there will be times when a condition must be met for a message to be sent to an object. Guards are conditions that need to be used throughout UML diagrams to control flow.

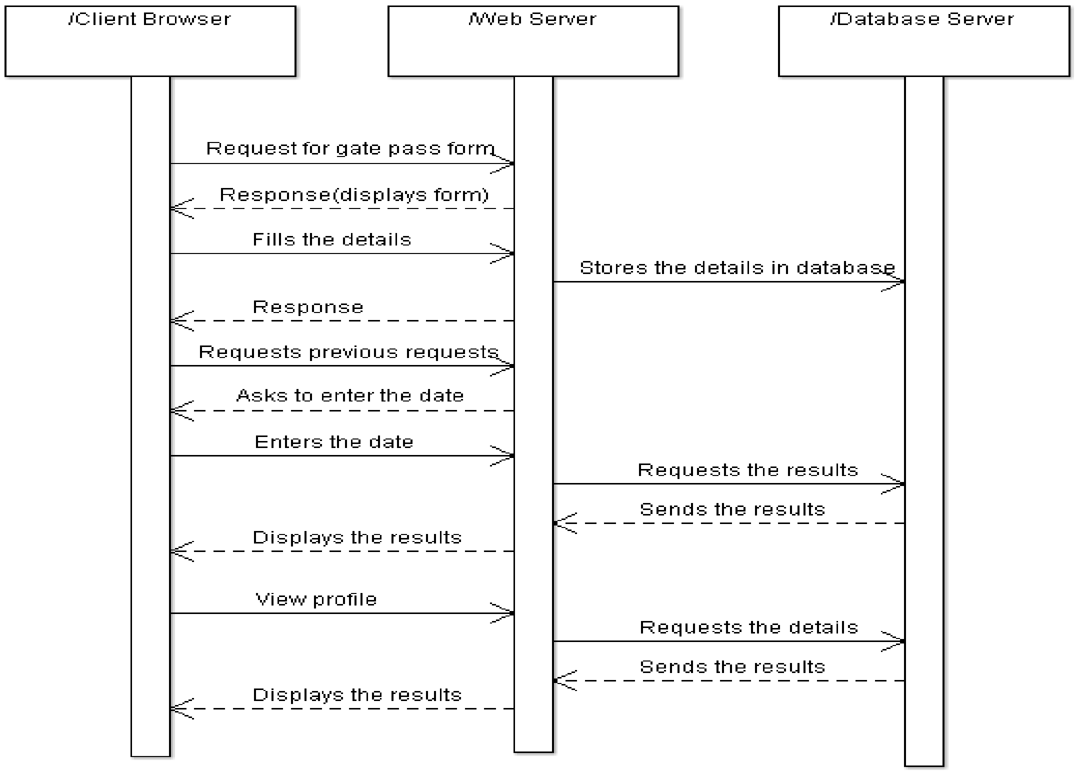
Sequence diagram for user registration:



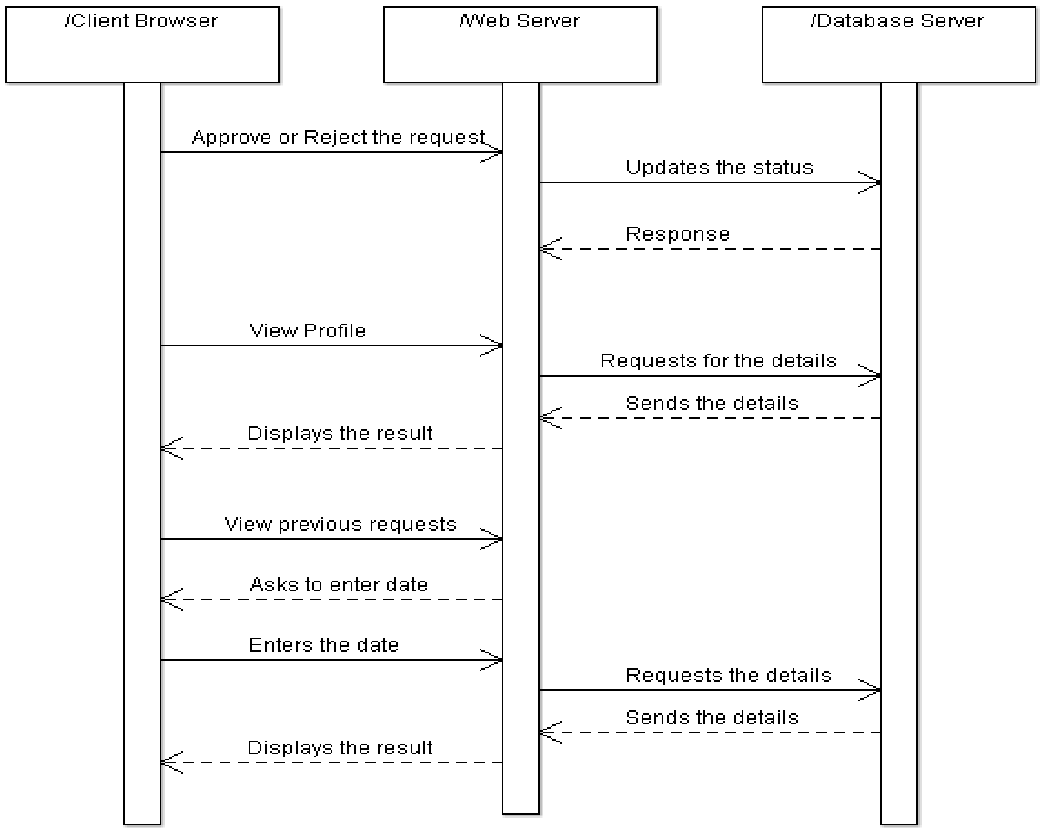
Sequence diagram for login and logout:



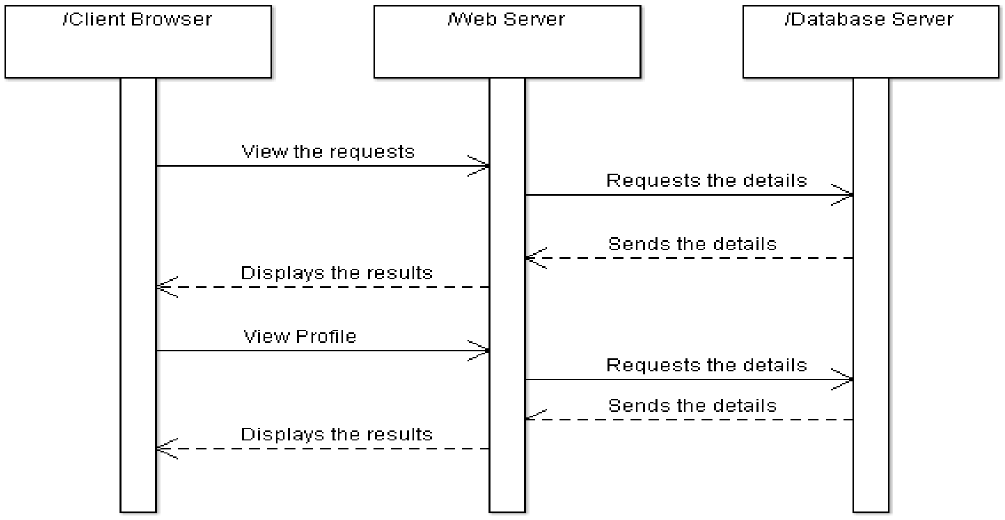
Sequence diagram for Student functionality:



Sequence diagram for Coordinator functionality:



Sequence diagram for Watch men functionality:



➢ Class Diagram

Class diagrams are the main building blocks of every object-oriented methods. The class diagram can be used to show the classes, relationships, interface, association, and collaboration. UML is standardized in class diagrams. Since classes are the building block of an application that is based on OOPs, so as the class diagram has appropriate structure to represent the classes, inheritance, relationships, and everything that OOPs have in its context. It describes various kinds of objects and the static relationship in between them.

The main purpose to use class diagrams are:

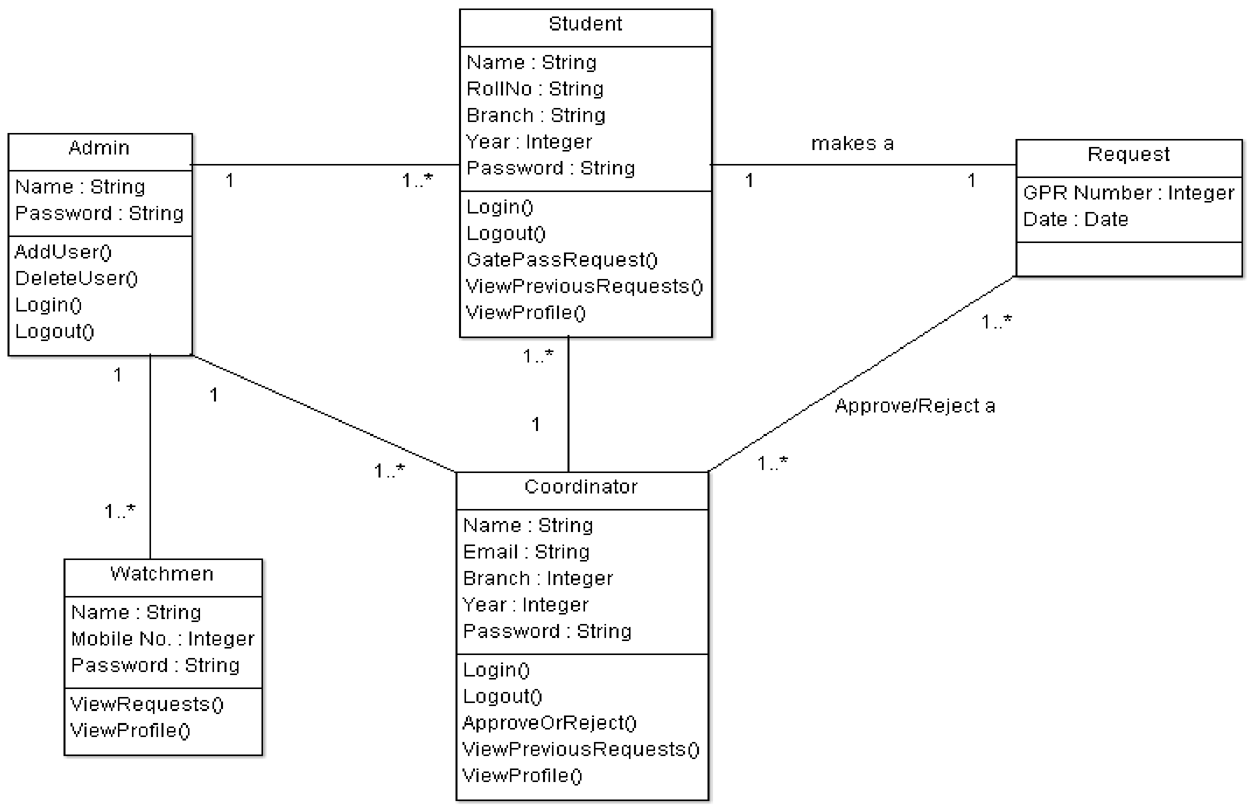
- This is the only UML which can appropriately depict various aspects of OOPs concept.
- Proper design and analysis of application can be faster and efficient.
- It is base for deployment and component diagram.

Each class is represented by a rectangle having a subdivision of three compartments name, attributes and operation.

There are three types of modifiers which are used to decide the visibility of attributes and operations.

- + is used for public visibility (for everyone)
- # is used for protected visibility (for friend and derived)
- – is used for private visibility (for only me)

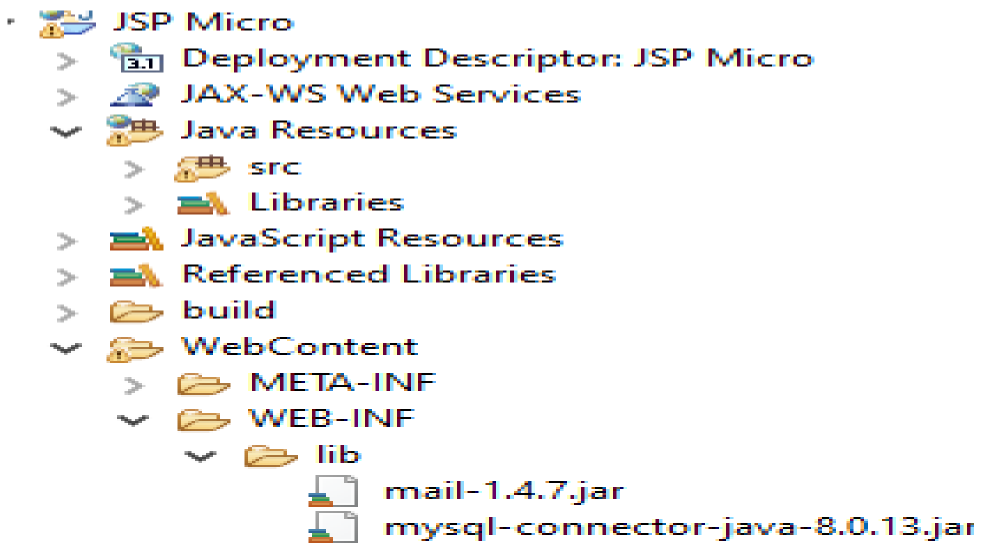
Class Diagram for our project:



5. IMPLEMENTATION

5.1 Implementation steps

- ✓ Eclipse installation (preferred oxygen version).
- ✓ Apache tomcat server (version 9.0)
- ✓ Set the java (1.8) path in the tomcat server.
- ✓ MySQL server installation.
- ✓ Create the required tables (Student, Coordinator, Watchmen, Requests) in the database instance.
- ✓ Create a Dynamic Web Project in eclipse. After creation the project directory structure should be as shown below.
- ✓ Place the mail.jar and mysql.jar file in the lib folder of web-content.

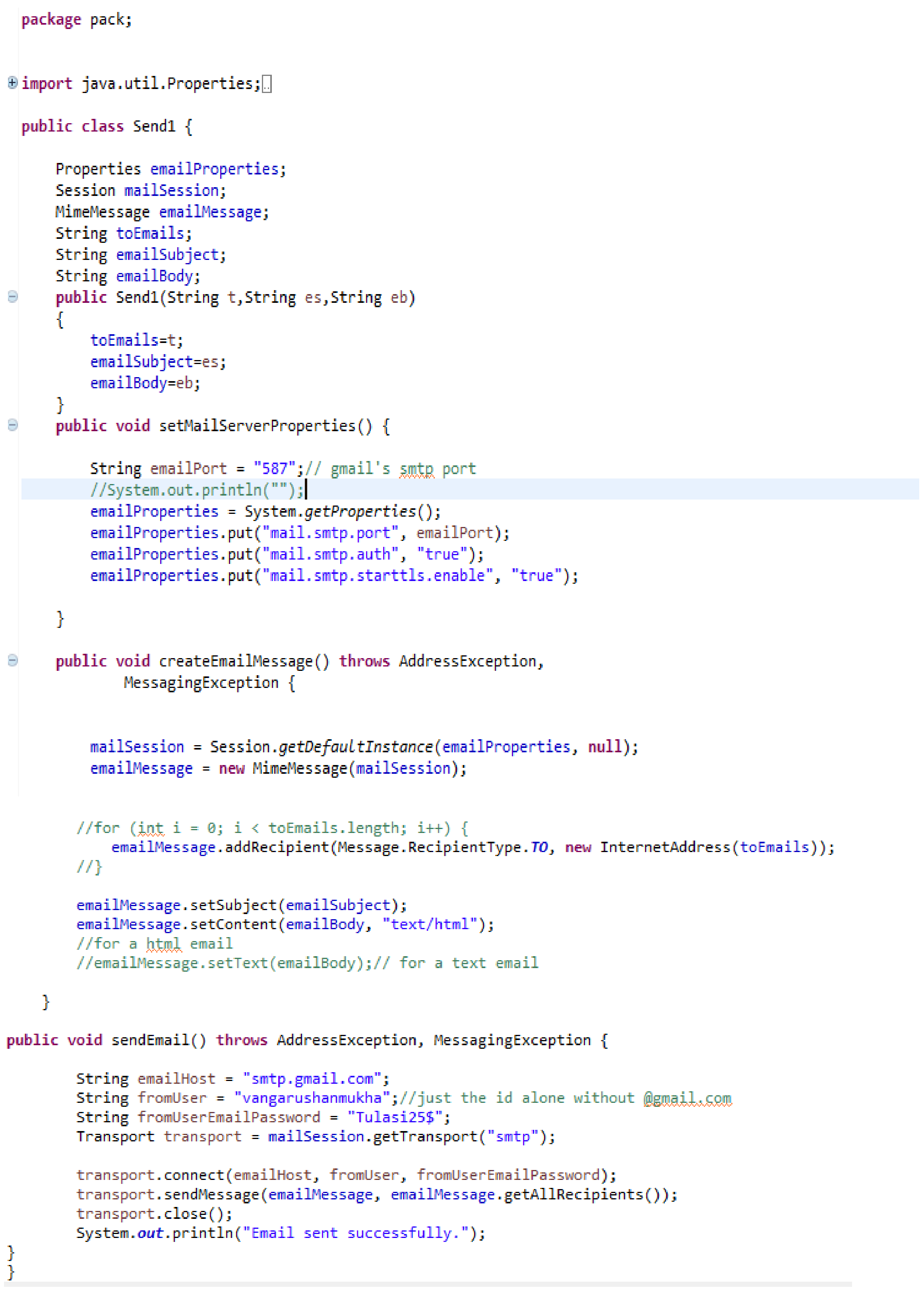


```
JSP Micro
  > Deployment Descriptor: JSP Micro
  > JAX-WS Web Services
  ˅ Java Resources
      > src
      > Libraries
  > JavaScript Resources
  > Referenced Libraries
  > build
  ˅ WebContent
      > META-INF
      ˅ WEB-INF
          ˅ lib
              mail-1.4.7.jar
              mysql-connector-java-8.0.13.jar
```

- ✓ Place all the .jsp pages in the WebContent folder and .java files in src folder.

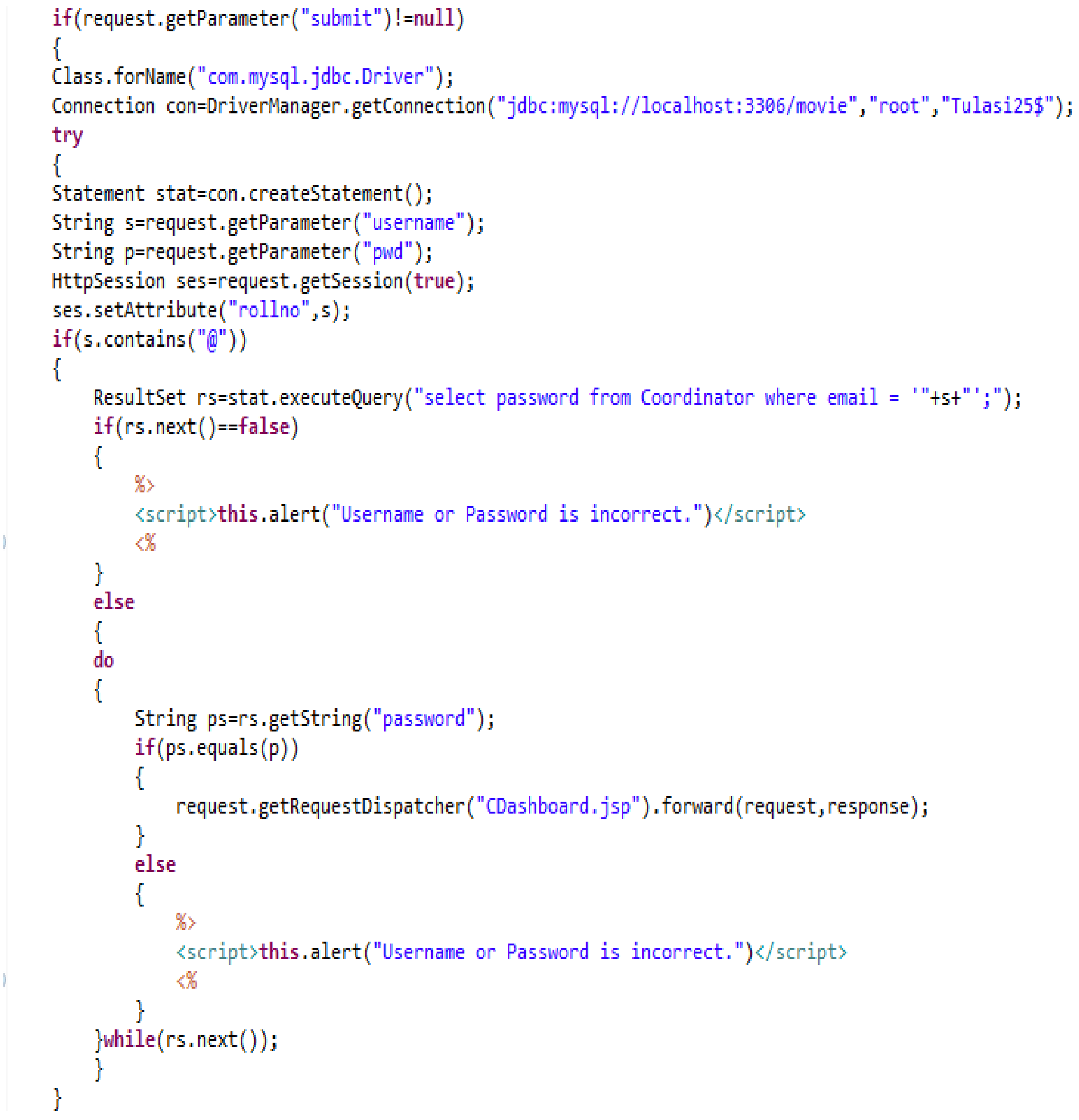
5.2 Code Snippets

- ✓ E-mail Authentication code:



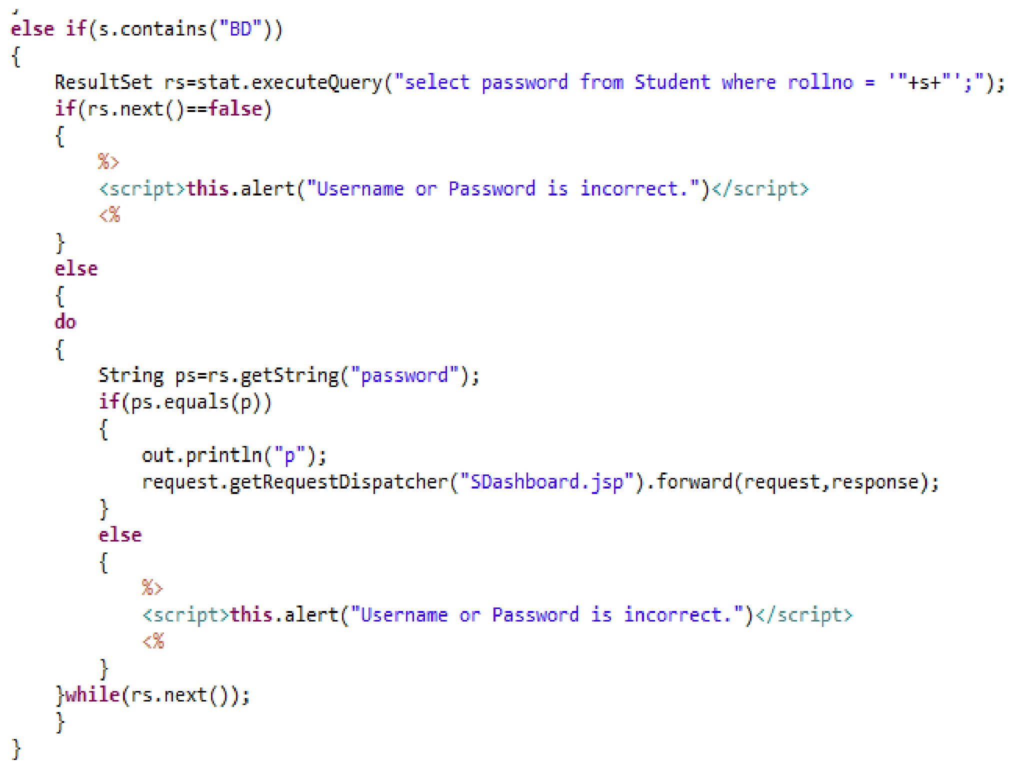
```
package pack;

import java.util.Properties;

public class Send1 {

    Properties emailProperties;
    Session mailSession;
    MimeMessage emailMessage;
    String toEmails;
    String emailSubject;
    String emailBody;
    public Send1(String t,String es,String eb)
    {
        toEmails=t;
        emailSubject=es;
        emailBody=eb;
    }
    public void setMailServerProperties() {

        String emailPort = "587";// gmail's smtp port
        //System.out.println("");
        emailProperties = System.getProperties();
        emailProperties.put("mail.smtp.port", emailPort);
        emailProperties.put("mail.smtp.auth", "true");
        emailProperties.put("mail.smtp.starttls.enable", "true");

    }

    public void createEmailMessage() throws AddressException,
            MessagingException {


        mailSession = Session.getDefaultInstance(emailProperties, null);
        emailMessage = new MimeMessage(mailSession);


        //for (int i = 0; i < toEmails.length; i++) {
            emailMessage.addRecipient(Message.RecipientType.TO, new InternetAddress(toEmails));
        //}

        emailMessage.setSubject(emailSubject);
        emailMessage.setContent(emailBody, "text/html");
        //for a html email
        //emailMessage.setText(emailBody);// for a text email

    }
public void sendEmail() throws AddressException, MessagingException {

        String emailHost = "smtp.gmail.com";
        String fromUser = "vangarushanmukha";//just the id alone without @gmail.com
        String fromUserEmailPassword = "Tulasi25$";
        Transport transport = mailSession.getTransport("smtp");

        transport.connect(emailHost, fromUser, fromUserEmailPassword);
        transport.sendMessage(emailMessage, emailMessage.getAllRecipients());
        transport.close();
        System.out.println("Email sent successfully.");
}
}
```

✓ Code Snippet for coordinator login:



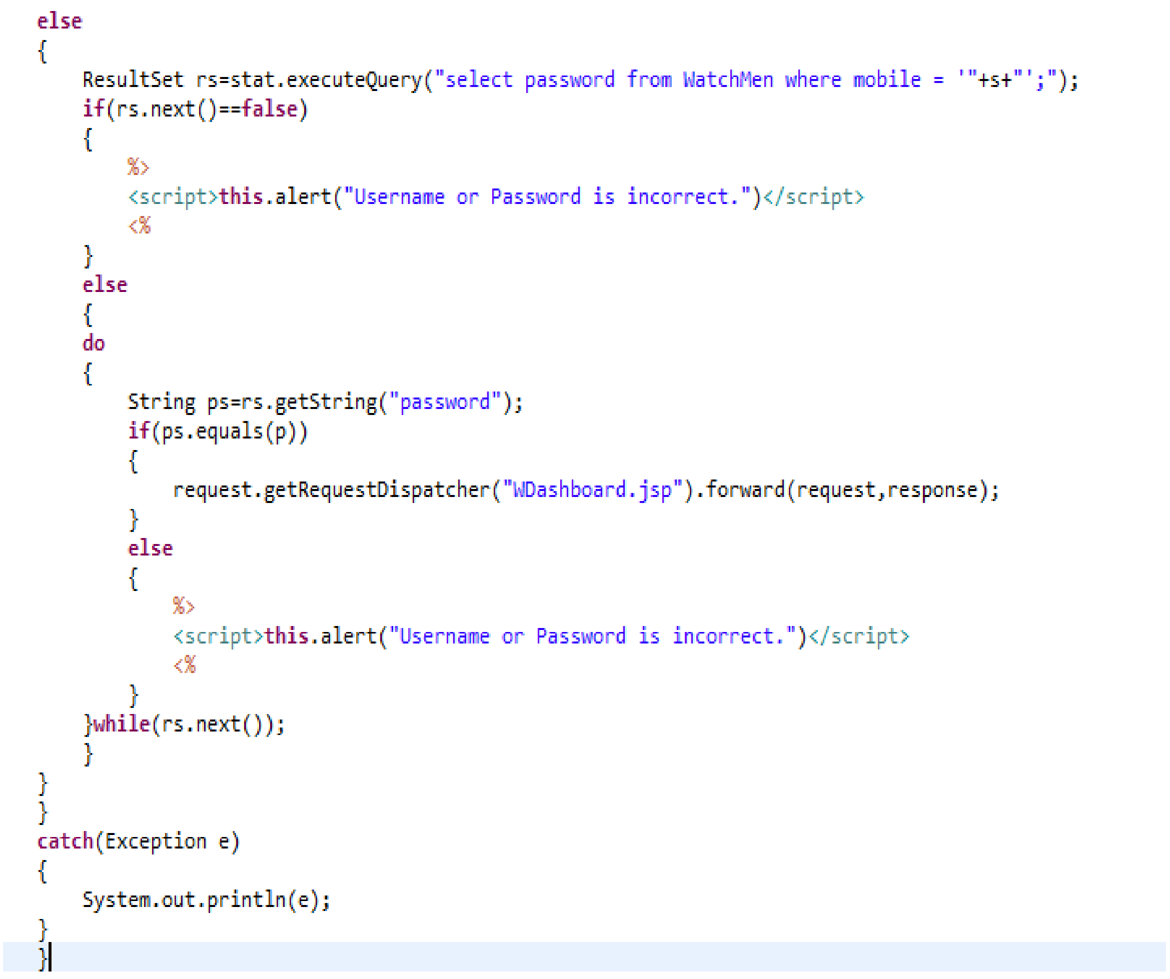
```
if(request.getParameter("submit")!=null)
{
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","Tulasi25$");
try
{
Statement stat=con.createStatement();
String s=request.getParameter("username");
String p=request.getParameter("pwd");
HttpSession ses=request.getSession(true);
ses.setAttribute("rollno",s);
if(s.contains("@"))
{
    ResultSet rs=stat.executeQuery("select password from Coordinator where email = '"+s+"';");
    if(rs.next()==false)
    {
        %>
        <script>this.alert("Username or Password is incorrect.")</script>
        <%
    }
    else
    {
    do
    {
        String ps=rs.getString("password");
        if(ps.equals(p))
        {
            request.getRequestDispatcher("CDashboard.jsp").forward(request,response);
        }
        else
        {
            %>
            <script>this.alert("Username or Password is incorrect.")</script>
            <%
        }
    }while(rs.next());
    }
}
```

✓ Code snippet for student login:



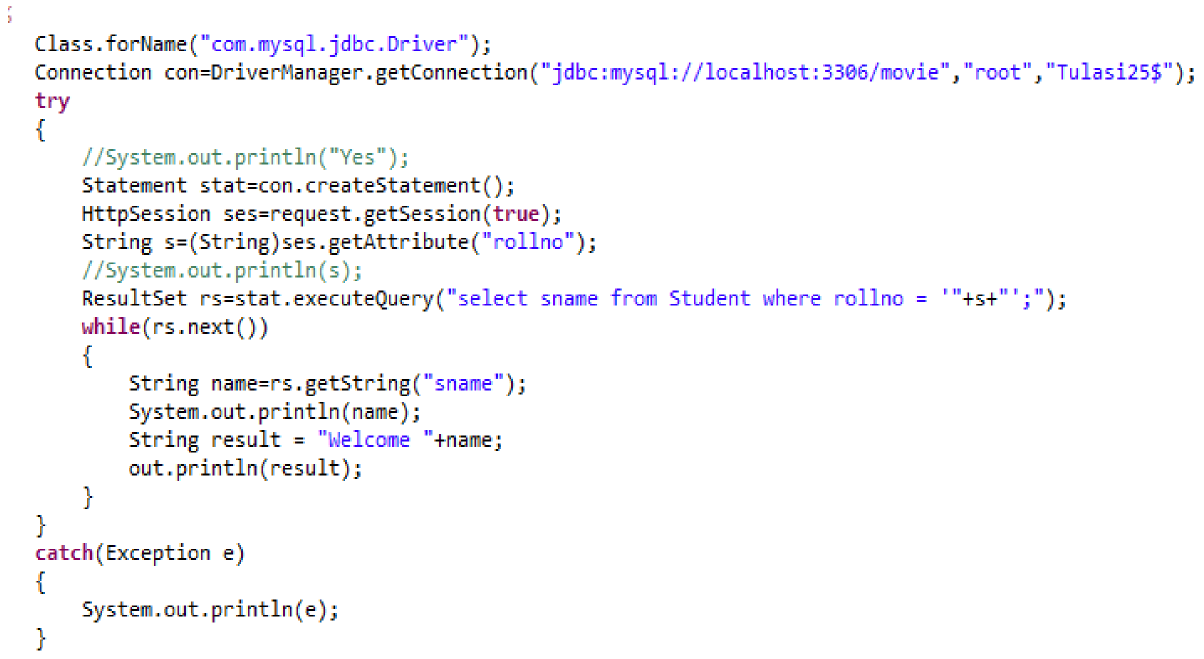
```
else if(s.contains("BD"))
{
    ResultSet rs=stat.executeQuery("select password from Student where rollno = '"+s+"';");
    if(rs.next()==false)
    {
        %>
        <script>this.alert("Username or Password is incorrect.")</script>
        <%
    }
    else
    {
    do
    {
        String ps=rs.getString("password");
        if(ps.equals(p))
        {
            out.println("p");
            request.getRequestDispatcher("SDashboard.jsp").forward(request,response);
        }
        else
        {
            %>
            <script>this.alert("Username or Password is incorrect.")</script>
            <%
        }
    }while(rs.next());
    }
}
```

✓ Code snippet for watchmen login:



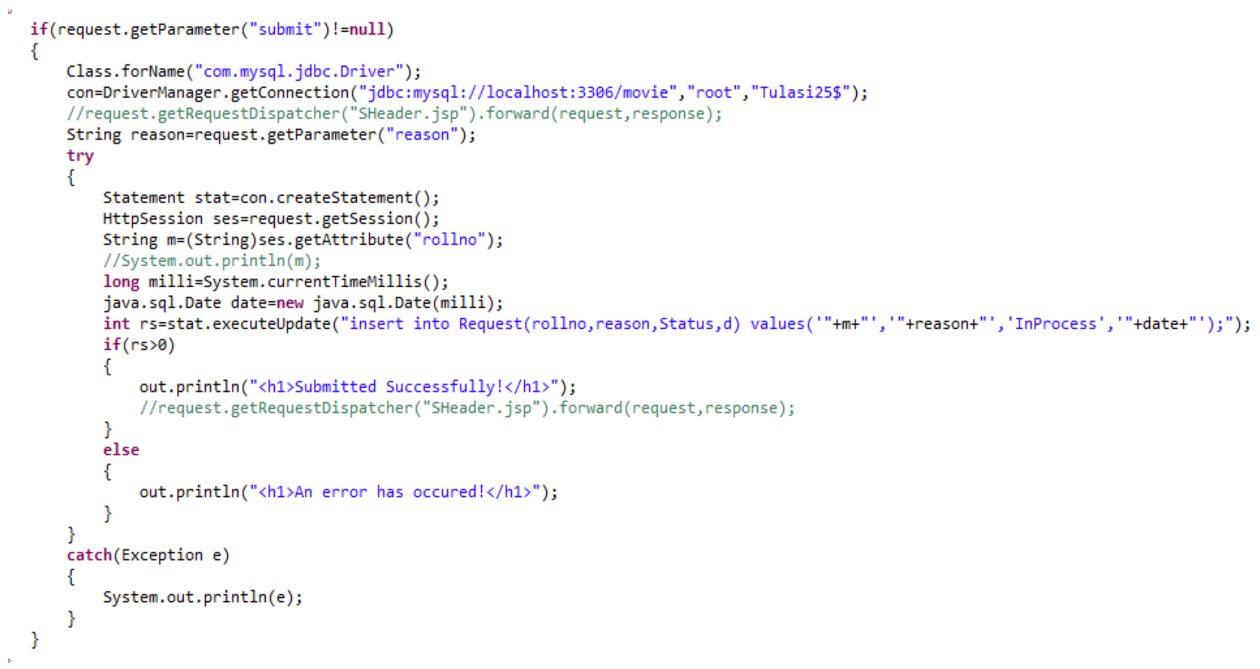
```
else
{
    ResultSet rs=stat.executeQuery("select password from WatchMen where mobile = '"+s+"';");
    if(rs.next()==false)
    {
        %>
        <script>this.alert("Username or Password is incorrect.")</script>
        <%
    }
    else
    {
    do
    {
        String ps=rs.getString("password");
        if(ps.equals(p))
        {
            request.getRequestDispatcher("WDashboard.jsp").forward(request,response);
        }
        else
        {
            %>
            <script>this.alert("Username or Password is incorrect.")</script>
            <%
        }
    }while(rs.next());
    }
}
}
catch(Exception e)
{
    System.out.println(e);
}
}
```

✓ Code snippet for Welcome (username) on top right corner of the page:



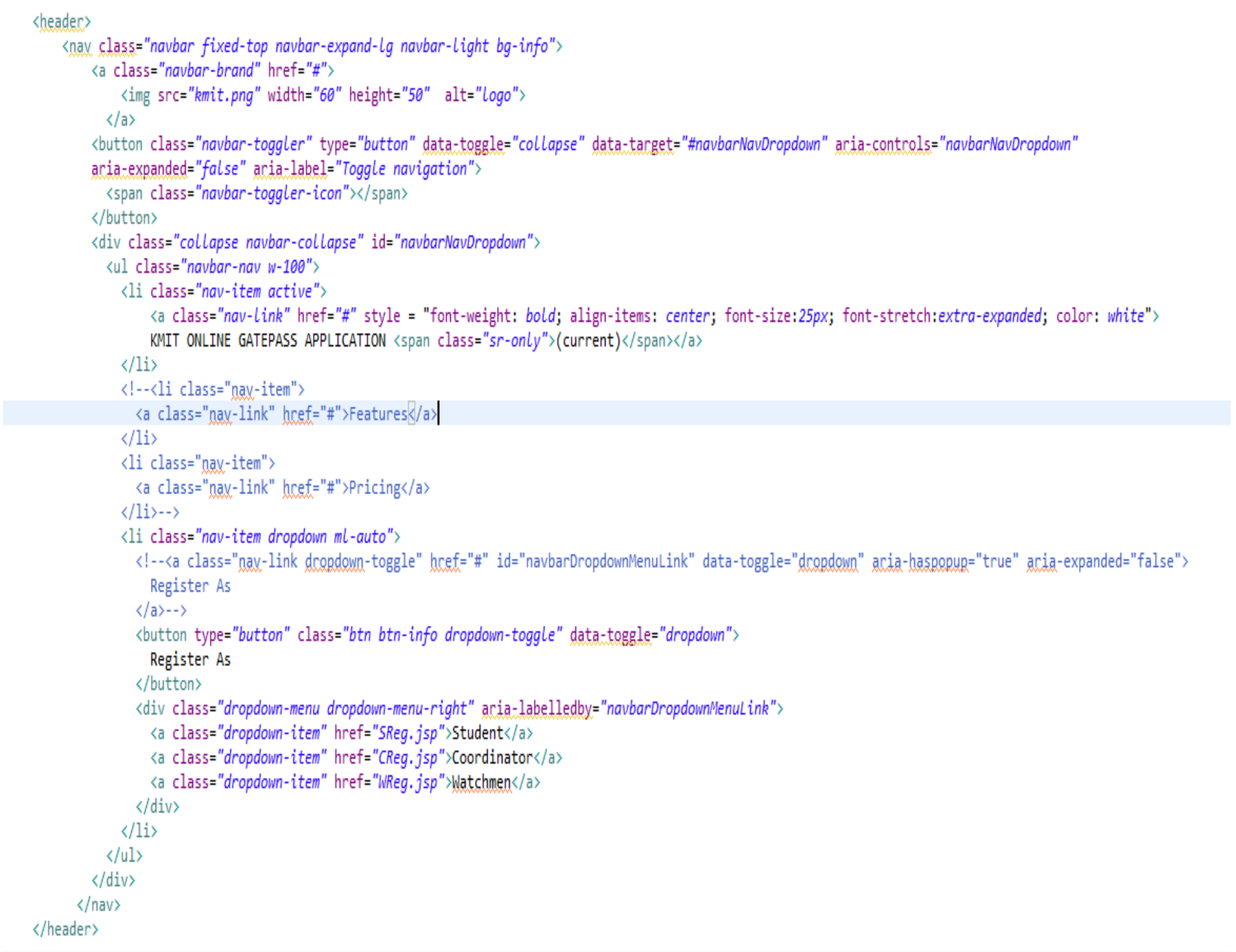
```
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","Tulasi25$");
try
{
    //System.out.println("Yes");
    Statement stat=con.createStatement();
    HttpSession ses=request.getSession(true);
    String s=(String)ses.getAttribute("rollno");
    //System.out.println(s);
    ResultSet rs=stat.executeQuery("select sname from Student where rollno = '"+s+"';");
    while(rs.next())
    {
        String name=rs.getString("sname");
        System.out.println(name);
        String result = "Welcome "+name;
        out.println(result);
    }
}
catch(Exception e)
{
    System.out.println(e);
}
```

✓ Code snippet for Gate pass request form:



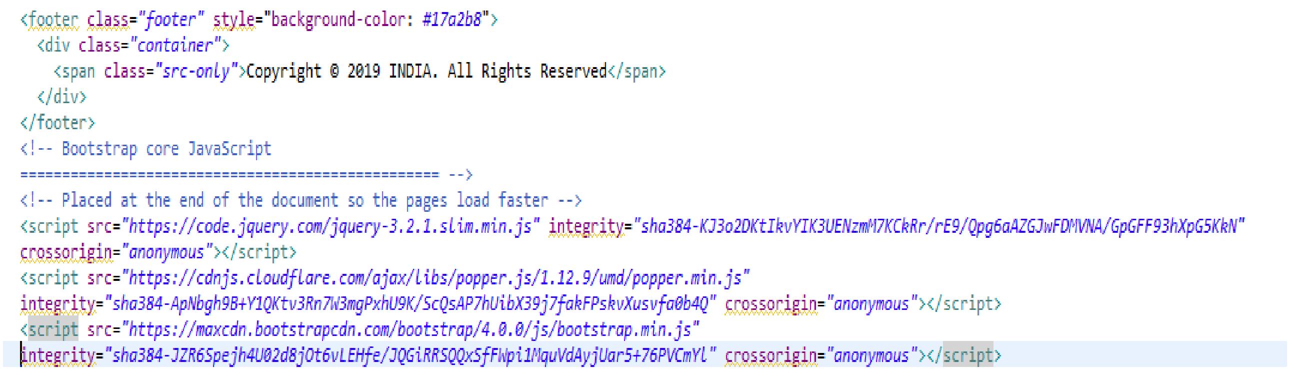
```
if(request.getParameter("submit")!=null)
{
    Class.forName("com.mysql.jdbc.Driver");
    con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","Tulasi25$");
    //request.getRequestDispatcher("SHeader.jsp").forward(request,response);
    String reason=request.getParameter("reason");
    try
    {
        Statement stat=con.createStatement();
        HttpSession ses=request.getSession();
        String m=(String)ses.getAttribute("rollno");
        //System.out.println(m);
        long milli=System.currentTimeMillis();
        java.sql.Date date=new java.sql.Date(milli);
        int rs=stat.executeUpdate("insert into Request(rollno,reason,Status,d) values('"+m+"','"+reason+"','InProcess','"+date+"');");
        if(rs>0)
        {
            out.println("<h1>Submitted Successfully!</h1>");
            //request.getRequestDispatcher("SHeader.jsp").forward(request,response);
        }
        else
        {
            out.println("<h1>An error has occured!</h1>");
        }
    }
    catch(Exception e)
    {
        System.out.println(e);
    }
}
```

✓ Code snippet of header for web page(html):



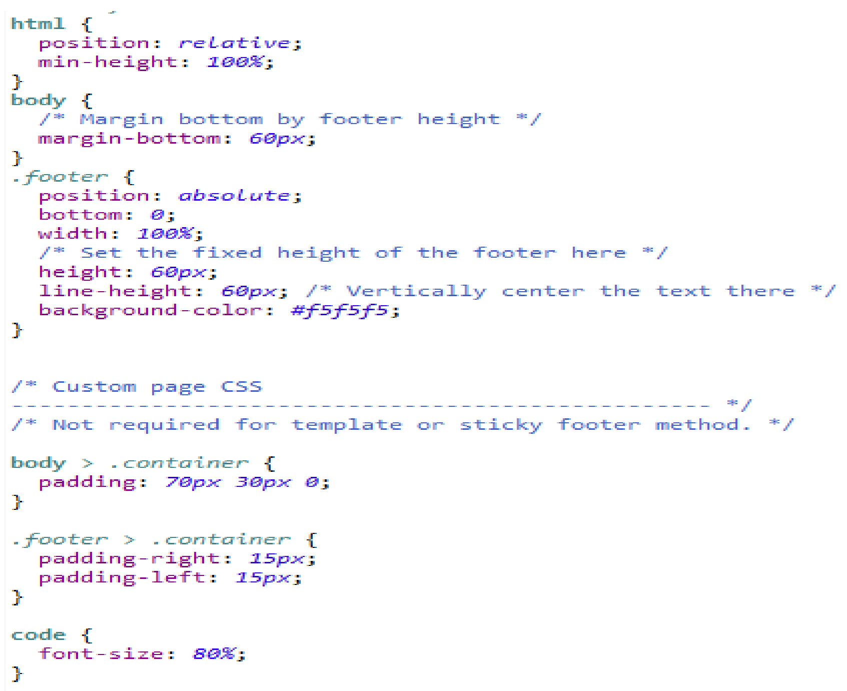
```html
<header>
    <nav class="navbar fixed-top navbar-expand-lg navbar-light bg-info">
        <a class="navbar-brand" href="#">
            <img src="kmit.png" width="60" height="50"  alt="logo">
          </a>
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown" aria-controls="navbarNavDropdown"
        aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNavDropdown">
          <ul class="navbar-nav w-100">
            <li class="nav-item active">
                <a class="nav-link" href="#" style = "font-weight: bold; align-items: center; font-size:25px; font-stretch:extra-expanded; color: white">
                KMIT ONLINE GATEPASS APPLICATION <span class="sr-only">(current)</span></a>
            </li>
            <!--<li class="nav-item">
              <a class="nav-link" href="#">Features</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="#">Pricing</a>
            </li>-->
            <li class="nav-item dropdown ml-auto">
              <!--<a class="nav-link dropdown-toggle" href="#" id="navbarDropdownMenuLink" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
                Register As
              </a>-->
              <button type="button" class="btn btn-info dropdown-toggle" data-toggle="dropdown">
                Register As
              </button>
              <div class="dropdown-menu dropdown-menu-right" aria-labelledby="navbarDropdownMenuLink">
                <a class="dropdown-item" href="SReg.jsp">Student</a>
                <a class="dropdown-item" href="CReg.jsp">Coordinator</a>
                <a class="dropdown-item" href="WReg.jsp">Watchmen</a>
              </div>
            </li>
          </ul>
        </div>
      </nav>
</header>
```

✓ Code snippet for footer of the web page (html):

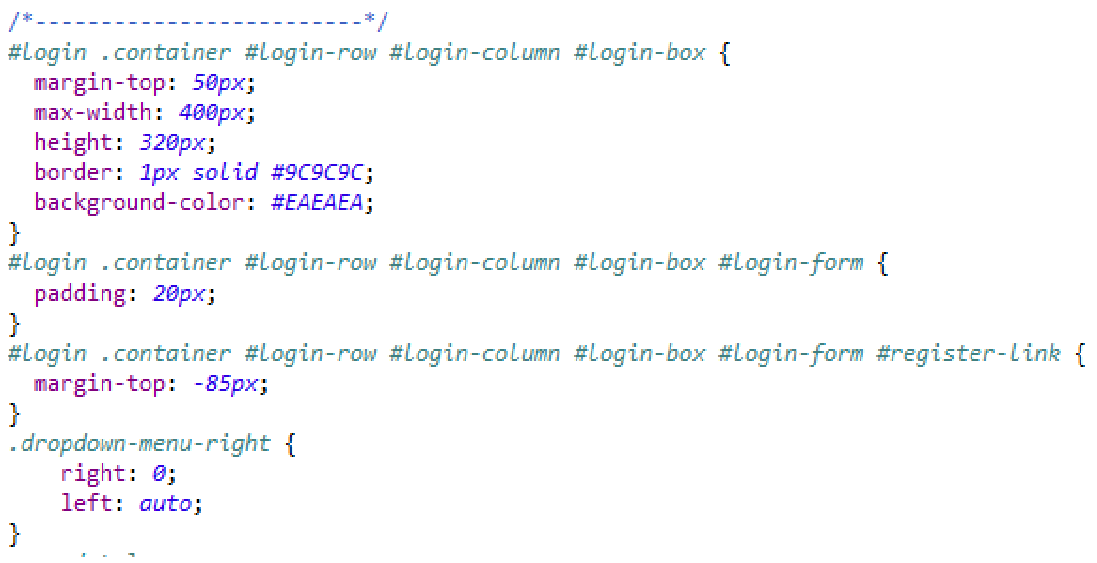


```html
<footer class="footer" style="background-color: #17a2b8">
  <div class="container">
    <span class="src-only">Copyright © 2019 INDIA. All Rights Reserved</span>
  </div>
</footer>
<!-- Bootstrap core JavaScript
================================================== -->
<!-- Placed at the end of the document so the pages load faster -->
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>
```

✓ Code snippet of .css file:



```
html {
  position: relative;
  min-height: 100%;
}
body {
  /* Margin bottom by footer height */
  margin-bottom: 60px;
}
.footer {
  position: absolute;
  bottom: 0;
  width: 100%;
  /* Set the fixed height of the footer here */
  height: 60px;
  line-height: 60px; /* Vertically center the text there */
  background-color: #f5f5f5;
}


/* Custom page CSS
-------------------------------------------------- */
/* Not required for template or sticky footer method. */

body > .container {
  padding: 70px 30px 0;
}

.footer > .container {
  padding-right: 15px;
  padding-left: 15px;
}

code {
  font-size: 80%;
}
```



```
/*------------------------*/
#login .container #login-row #login-column #login-box {
  margin-top: 50px;
  max-width: 400px;
  height: 320px;
  border: 1px solid #9C9C9C;
  background-color: #EAEAEA;
}
#login .container #login-row #login-column #login-box #login-form {
  padding: 20px;
}
#login .container #login-row #login-column #login-box #login-form #register-link {
  margin-top: -85px;
}
.dropdown-menu-right {
    right: 0;
    left: auto;
}
```

CONCLUSION

This web application (KMIT ONLINE GATEPASS APPLICATION) helps us to create a seamless environment through which students can request for gate pass and the in-charge person (coordinator) checks for a valid reason and responds to the request. This application is also an environment friendly as it involves in reduction of use of paper and reduces the amount of time spent on getting the permission. Most importantly it creates a chain system right from the in-charge (coordinator) through the student till the watchmen. It removes the cause of miss happenings as the response comes directly from the coordinator to the watchmen. It reduces the data entry in the registers as this work is taken care by the database server (MySQL).

REFERENCES

- Eclipse installation link: https://www.eclipse.org/downloads/packages/release/oxygen
- MySQL server link: https://www.mysql.com/downloads/
- Apache tomcat server link: https://tomcat.apache.org/download-90.cgi
- Email Authentication: https://www.journaldev.com/2532/javamail-example-send-mail-in-java-smtp
- Java Server Pages (JSP) tutorial: https://www.geeksforgeeks.org/introduction-to-jsp/
- Bootstrap tutorial: https://www.w3schools.com/bootstrap4/