LEAVE AND ON DUTY FORM MANAGEMENT SYSTEM

A MINI PROJECT REPORT

***Submitted* by**

NANDHAGOPAL R (1413061)

NAVEEN R (1413063)

PRAVEENRAJ M S (1413075)

MADHUVANTHI M (1413054)

in partial fulfillment for the requirement of award of the degree

of

BACHELOR OF ENGINEERING

in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

K.S.R. COLLEGE OF ENGINEERING

(An Autonomous Institution, Affiliated to Anna University Chennai and Approved by AICTE)

TIRUCHENGODE-637 215

NOVEMBER 2017

K.S.R. COLLEGE OF ENGINEERING

TIRUCHENGODE – 637215

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Mini Project report **"LEAVE AND ON DUTY FORM MANAGEMENT SYSTEM"** is the bonafide work of **"NANDHAGOPAL R (1413061), NAVEEN R (1413063), PRAVEENRAJ M S (1413075) and MADHUVANTHI M (1413054)"** carried out the project work under my supervision.

Signature

Dr.A.RAJIVKANNAN M.E., Ph.D.,

HEAD OF THE DEPARTMENT

Professor

Department of CSE

K.S.R College of Engineering,

(Autonomous)

Tiruchengode - 637215.

Signature

Mr.K.DINESH KUMAR M.E.,

SUPERVISOR

Assistant Professor

Department of CSE

K.S.R College of Engineering,

(Autonomous)

Tiruchengode - 637215.

Submitted for Project viva-voce held on ……

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

We feel highly honored to extend our sincere gratitude to our beloved Founder cum Chairman **Lion Dr.K.S.RANGASAMY MJF** K.S.R Educational Institutions and our Chairman cum Managing Trustee **Mr.R.SRINIVASAN BBM, MISTE** for providing all facilities to complete this project work.

We would like to acknowledge the constant and kind support provided by our Principal **Dr.K.KALIANNAN M.Tech., M.S., MBA., Ph.D.,** who supported us in all the endeavors and been responsible for inculcating us all through our career.

We act sense of gratitude to **Dr.A.MAHABUBBASHA M.E., Ph.D.,** Director of Computer Science and Engineering department for offering suggestions at the right time.

We feel highly elated to thank our respectable Head of the Department **Dr.A.RAJIVKANNAN M.E., Ph.D., MISTE and MCSI,** who guided us and was a pillar of support for the successful completion of the project.

We are great indebted to our Project Coordinator **Dr.T.POONGOTHAI M.E., Ph.D.,** of our department for her valuable suggestions and guidance to our project.

We are the most fortunate in having the opportunity to work under the internal guide **Mr.K.DINESH KUMAR M.E.,** express our sincere thanks to him. This project has brought out the hidden talents within us it is a pleasure to express our gratefulness to our beloved parents for providing their support and confidence to us for the completion of the project and our heartfelt thanks to our entire department faculty and members, beloved friends, directly and indirectly who helped us during the tenure of the project.

ABSTRACT

The main objective of the proposed system is to automate the leave and on duty management and to decrease the paper work and easier record maintenance by having a particular website for leave and on duty maintenance. This approach basically deals with the record of leave and on duties taken by students in the organization. This system also approach's to reduce the formalities and time delay faced by the students. The students apply for the leave, faculty checks whether the reason for the leave or on duty is valid or not and if so the reason is valid and the leave is accepted by the faculty otherwise it will be rejected. If the leave is rejected by the faculty, the student has to continue the process from the beginning to apply the leave. In this paper we review the various computerized system which is being developed by using different techniques. Based on this review the Leave and On Duty Management System simplifies the leave and on duty process, makes it more maintainable, gives better, clearer and more frequent information to end users, standardizes the processing of different types of leave and lowers the amount of data entry and verification activities.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACRONYM	ABBREVIATION
CSS	Cascading Style Sheet
GB	Giga Bytes
HD	Hard Disk
HTML	Hyper Text Markup Language
OS	Operating System
RAM	Random Access Memory
J2EE	Java Enterprise Edition
JDBC	Java Database Connectivity
ODBC	Open Database Connectivity
EJB	Enterprise Java Bean
SQL	Structured Query Language
API	Application Program Interface

CHAPTER 1

INTRODUCTION

The project entitled **LEAVE AND ONDUTY MANAGEMENT SYSTEM** is a wed based application development system using integrated development Environment. We planned to develop this application to decrease the paper work and easier record maintenance by having a particular website for leaves maintenance. This approach basically deals with the record of leaves and on duty taken by the student in the institution. This system also approaches to reduce the formalities and time delay facing by students. This module is a single leave and on duty form management system that is a record of virtual information regarding working hours and tasks. Faculty will have permissions to look after data of every students of their respected faculty can approve leave through this application and can view leave information of every individual. This application can be used to automate the workflow of the leave applications and their approvals. This project's main idea is to develop a centralized application connected to database which will maintain student leave and on duties. Leave and on duty form management application will reduce paperwork and maintain record in a more efficient & systematic way.

1.1 FUNCTIONAL REQUIREMENTS

Admin

- Create, update and delete user details after login
- Can change login password
- Logout from the system
- View student status.

Faculty

- To approve leave and on duty forms applied by student.
- To view the student records.

Student

- To apply leave and on duty forms.
- To view the applied records.

1.2 NON –FUNCTIONAL REQUIREMENTS

- Secure access of confidential data.
- 24x7 availability.
- Browse testing and support for Mozilla Firefox and Google Chrome.

CHAPTER 2

PROBLEM DESCRIPTION

2.1 EXISTING SYSTEM

- In most of the organizations, each student has been provided with leave or on duty form at the time of appointment.
- In the existing system, the manual process, receiving data from students and getting approval are done through manual process.
- Students have to suffer lots of problems and formalities for the approval of leave and on duty.
- These records are entered in manual process will take long time, separate workers need to maintaining the databases. All the details are stored via separate databases.

2.2 DRAWBACKS OF EXISTING SYSTEM

- Heavy work load because maintained in the form of files or records.
- Time consumption is high.
- Difficult to data maintenance and search.
- It taken more memory space.
- Data can be loss.

To avoid all these limitation and make the working more accurately the system need to be computerized.

2.3 PROPOSED SYSTEM

- The main objective of the existing system is to provide a user-friendly interface.
- Due to inconvenience in managing the data using leave or on duty forms, it is required to have a computer based system where a student can login and apply for the leave.
- Once the details are fed into the computer there is no need for various persons to deal with separate sections.
- Only a single person is enough to maintain all the reports.

2.4 ADVANTAGES OF PROPOSED SYSTEM

- Large volume of data can be stored with case.
- Maintenance of leave is flexible.
- Records stored are updated now and then.
- Stored data and procedures can be easily edited.
- Reports can be generated with case.
- Accurate calculations are made and less man power required.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 HARDWARE SPECIFICATION

PROCESSOR	:	Dual core and above
PROCESSOR SPEED	:	2.80 GHz
MAIN STORAGE	:	512 MB RAM
RAM	:	2 GB or more
HARD DISK	:	60 GB or more
KEYBOARD	:	104 Keys

3.2 SOFTWARE SPECIFICATION

TARGETED OS	:	Windows 7/8/8.1/10
FRONT END	:	HTML, CSS, JS
BACK END	:	J2EE, SQL

CHAPTER 4

SOFTWARE SPECIFICATION

4.1 FRONT END

4.1.1 HTML

HTML is the standard markup language for creating Web pages.

- HTML stands for Hyper Text Markup Language
- HTML describes the structure of Web pages using markup
- HTML elements are the building blocks of HTML pages
- HTML elements are represented by tags
- HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
- Browsers do not display the HTML tags, but use them to render the content of the page

Hyper Text: Hyper Text simply means "Text within Text". A text has a link within it, is a hypertext. Every time when you click on a word which brings you to a new webpage, you have clicked on a hypertext.

Markup language: A markup language is a programming language that is used make text more interactive and dynamic. It can turn a text into images, tables, links etc.
An HTML document is made of many HTML tags and each HTML tag contains different content.

4.1.2 CSS

- **CSS** stands for **C**ascading **S**tyle **S**heets
- CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
- CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
- External style sheets are stored in **CSS files**

Cascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs and variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

4.1.3 JAVASCRIPT

AS a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming style. It has an API for working with text, array, dates, regular expression, and basic manipulation of the DOM, but does not include any I/O such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

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 database, and non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop application, including desktop widgets.

4.2 BACK END

4.2.1 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.
- At the client tier, J2EE supports pure HTML, as well as Java applets or applications. It relies on Java Server Pages and servlet code to create HTML or other formatted data for the client.
- Enterprise JavaBeans (EJBs) provide another layer where the platform's logic is stored. An EJB server provides functions such as threading, concurrency, security and memory management. These services are transparent to the author.
- Java Database Connectivity (JDBC), which is the Java equivalent to ODBC, is the standard interface for Java databases.
- The Java servlet API enhances consistency for developers without requiring a graphical user interface.

4.2.2 SQL

SQL consists of a data definition language, data manipulation language and data control language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data control. Although SQL is often described and to a great extent is a declarative language.

SQL is a standard language for storing, manipulating and retrieving data in databases.

CHAPTER 5

SYSTEM ANALYSIS

5.1 REQUIREMENT ANALYSIS

Since the design of a new or revised system cannot begin until the analyst fully understands the existing system, this stage cannot be omitted in software development. Hence we are going to start with the analysis of the existing manual system under the following heading: data collection, data storage, data communication and manipulation and system cost.

5.2 DATA COMMUNICATION

The process like – sorting, comparing, analysis and calculation are to be performed on the records which are stored on files. Difficulties arise when it actually comes to manipulation of data. Processing data manually consumes a lot of time; for instance, sorting of folders into years of admission. Hence the access time or process time of the present system is extremely high and results in staffs spending more time than is necessary.

Now, with the availability of high speed modern electronic information processing machines, this need for improved speed of transaction processing could easily be accomplished.

CHAPTER 6

MODULE DESCRIPTION

6.1 ADMIN

In Admin module, users logged into the system are capable of doing the following operations. Here the admin is responsible for analyse, insert, update, delete and view the details of the students and faculties. He/she can change the details of the students and faculties.

6.1.1 Add new user

Admin can add new student or faculty by giving separate id and name as input. The default password on creation is “KSRCE” for all created new staff. Here, the new student and faculty is added. The admin only have rights to do this for a security purpose.

6.1.2 Update existing user

Admin can update existing student or faculty details by giving their id. Using this module admin can update the basic details. Admin can also change the user role of faculty.

6.1.3 Delete existing user

Admin can delete student or faculty name or details from the system if they are not need. All details of student or faculty are deleted permanently from the system. i.e., if anyone has relieved their details will be erased.

6.1.4 Change Password

Admin can change login password using this module. So that if the password is forgot by the student or faculty can be changed by admin.

6.1.5 Logout

Admin should logout securely from the system.

6.2 FACULTY

In Faculty module, users logged into the system are capable of doing the following operations. Here the faculty is responsible for approve and view the details of the leave and on duty applied by the students.

6.2.1 Approve record

Faculty can approve the records of leave and on duty applied by the student by giving separate id or name as input. The approval can be accept or reject.

6.2.2 View record

Faculty can view all records of leave and on duty applied by the student by giving separate id or name as input or the records can be searched through other user details.

6.2.3 Logout

Faculty should logout securely from the system.

6.3 STUDENT

In Student module, users logged into the system are capable of doing the following operations. Here the student can apply leave or on duty and can view the details of the applied leave and on duty.

6.3.1 Apply leave or on duty

Students can apply leave and on duty through designed forms by giving the proper details.

6.3.2 View record

Students can view the applied leave and on duty details. This is also used know whether the applied leave or on duty is accepted or rejected by the faculty.

6.3.3 Logout

Student should logout securely from the system.

CHAPTER 7

DESIGN

7.1 USE CASE DIAGRAM

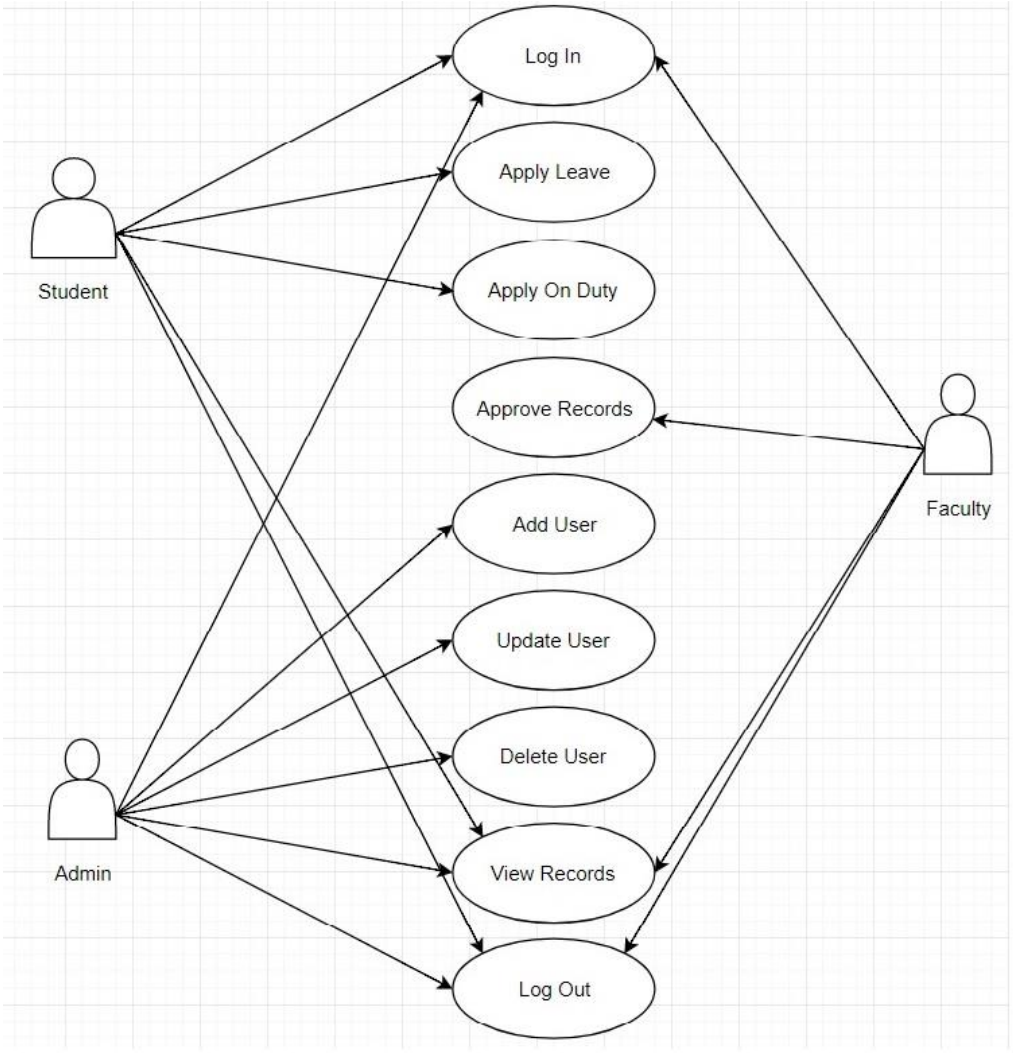


Figure 7.1 Use Case Diagram

In this management system, the actors are student, faculty and administrators. The detailed description of use case is represented in the use case diagram.

7.2 MODULE DIAGRAM

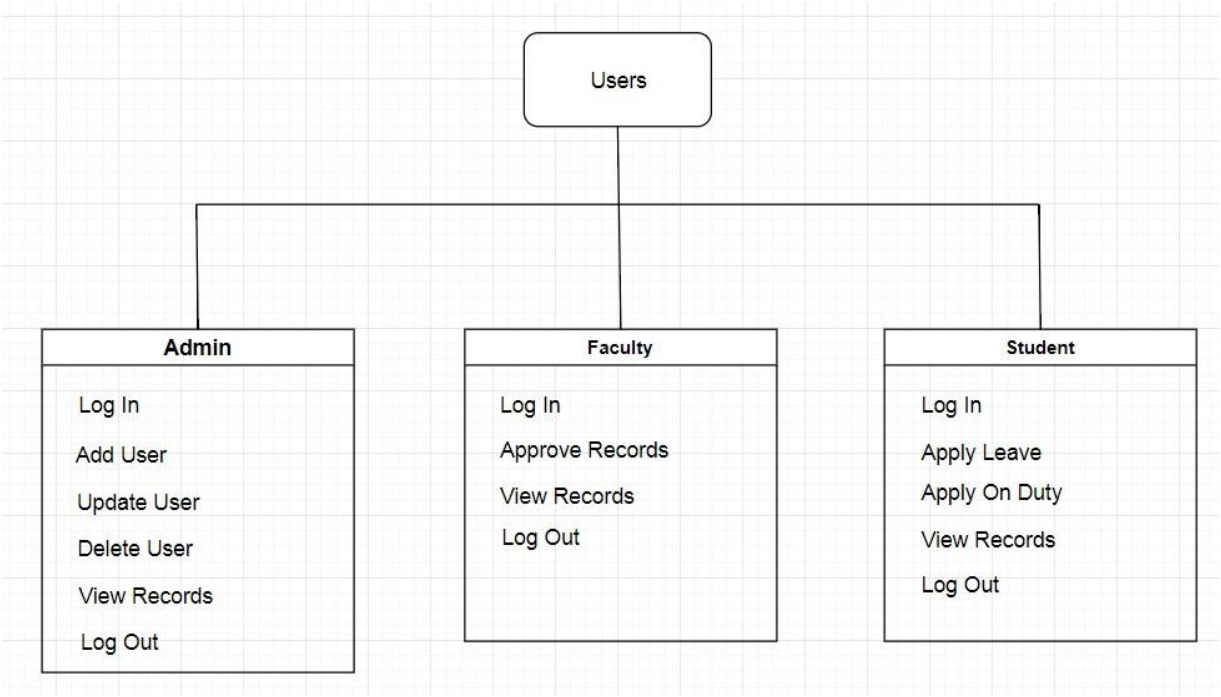


Figure 7.2 Module Diagram

CHAPTER 8

TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. There are various types of test. Each test type addresses a specific testing requirement.

8.1 UNIT TESTING

In unit testing, we have to test the programs making up the system. For this reason, Unit testing sometimes called as Program testing. Unit testing on the modules are independently of one another, to locate errors. This enables to detect errors in coding and logic in the module. The testing was carried out during programming stage itself.

8.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.3 SYSTEM TESTING

System testing ensures that the entire integrated software system meets the requirements. It tests a configuration to ensure known and predictable results. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. In this testing it is based on the coding to assign or performs the function by using the methods and data for the program to be run.

8.3.1 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner coding, structure and language of the software.

8.3.2 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box must be written from a definitive source document, such as specification or requirements document. The test provides inputs and responds to outputs without considering how the software works.

8.4 SYSTEM IMPLEMENTATION

When the initial design was done for the system, the client was consulted for the acceptance of the design so that further proceedings of the system development can be carried on. The aim of the system illustration was to identify any malfunction of the system. After the management of the system was approved the system implemented in the concern, initially the system was run parallel with existing manual system. The system has been tested with live data and has proved to be error free and user friendly.

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENTS

9.1 CONCLUSION

The project titled as “Leave and On Duty Form Management System” is a web based application. This software provides facility for create, update and delete staff details after login. The software is developed with modular approach. All modules in the system have been tested with valid data and invalid data and everything work successfully. Thus, the system has fulfilled all the objectives identified and is able to replace the existing system.

The system is very flexible and versatile. This software has a user-friendly screen that enables the user to use without any inconvenience. Validation checks induced have greatly reduced errors. Provisions have been made to upgrade the software.

9.2 FUTURE ENHANCEMENTS

In future, we can use photo reorganization instead of using heterogeneous database more over high speed, accuracy and non-redundant data are the main advantages of the system. Data entry errors can be minimized through validity checks. After the verification only, the data are placed in the permanent database.

CHAPTER 10

APPENDIX

10.1 SOURCE CODE

LOGIN PAGE

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>Login Form</title>
        <link rel="stylesheet" href="assets/demo.css">
        <link rel="stylesheet" href="assets/form-login.css">
</head>
<header>
        <h1>Student Leave and OnDuty Forms</h1>
</header>
<body>
        <div class="main-content">
                <form class="form-login" method="post" name="formlogin" action="LoginServlet" >
                        <div class="form-log-in-with-email">
                        <div class="form-white-background">
                        <div class="form-title-row">
                                <h1>Student Log in</h1>
                        </div>
                        <div class="form-row">
```

```
                    <label>
                        <span>Reg No.</span>
                        <input type="text" name="regno">
                    </label>
                </div>
                <div class="form-row">
                    <label>
                        <span>Password</span>
                        <input type="password" name="password">
                    </label>
                </div>
                <div class="form-row">
                    <button type="submit">Log in</button>
                </div>
            </div>
            <a href="faculty-login.html" class="form-forgotten-password" style="font-size:20px;">Staff Login &middot;</a>
            </div>
        </form>
    </div>
</body>
</html>
```

Login Servlet

```
@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet
{
	LoginDao loginDaoObj;
	public void init(ServletConfig config) throws ServletException
	{
		loginDaoObj = new LoginDao();
	}
	public void doPost(HttpServletRequest request, HttpServletResponse response) throws
	ServletException, IOException
	{
		String getRegno = request.getParameter("regno");
		String getPass =request.getParameter("password");
		PrintWriter out = response.getWriter();
		if(getRegno.length()==0)
			out.println("<html><body><script> alert(' Please Enter Regno. !!!');
			</script></body></html>");
		else if(getPass.length()==0)
			out.println("<html><body><script> alert(' Please Enter Password
			!!!');</script></body></html>");
		else if(loginDaoObj.verifyUser(getRegno,getPass))
		{
			HttpSession session=request.getSession();
			session.setAttribute("Regno",getRegno);
			session.setAttribute("Name",loginDaoObj.getStudentName());
			response.sendRedirect("/LeaveOndutyForm/LeaveForm.jsp");
		}
		else
			out.println("<html><body><script> alert(' Wrong Username or Password
			!!!');</script></body></html>");
	}
}
```

Student Bean

```
public class StudentBean
{
	int studentId,year;
	String name,regno,section;
	public int getStudentId() {
		return studentId;
	}
	public void setStudentId(int studentId) {
		this.studentId = studentId;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getRegno() {
		return regno;
	}
	public void setRegno(String regno) {
		this.regno = regno;
	}
	public int getYear() {
		return year;
	}
	public void setYear(inti) {
		this.year = i;
	}
	public String getSection() {
		return section;
	}
	public void setSection(String section) {
		this.section = section;
	}
}
```

Data Access Object

```
public class LeaveAppliedFacultyDao {

        LinkedList<Integer> studentId = new LinkedList<Integer>();
        LinkedList<Integer> leaveFormId =new LinkedList<Integer>();
        LinkedList<LeaveFormBean> leaveFormBeanObjectList =new LinkedList<LeaveFormBean>();
        LinkedList<StudentBean> studentBeanObjectList = new LinkedList<StudentBean>();

        static String searchBy;
        int update1=0, update2=0;
        Connection con;

        LeaveFormBean leaveFormBeanObject;
        StudentBean studentBeanObject;
        SearchByBean sb =new SearchByBean();

        public LeaveAppliedFacultyDao()
         {
                CreateConnectionDao conObj=new CreateConnectionDao();
                con=conObj.dbConnection();
         }

        public LinkedList<LeaveFormBean> leaveApplied(String searchBy)
        {
                LeaveAppliedFacultyDao.searchBy=searchBy;
                PreparedStatement preparedStatement;
                PreparedStatement preparedStatement1;
                ResultSet resultSet1;
                try {
                        preparedStatement = con.prepareStatement("select studentid,leaveformid from
                        leaveapplied where acceptreject=? ");


                        preparedStatement.setString(1,searchBy);
                }
```

```
                resultSet1 = preparedStatement.executeQuery();


                while(resultSet1.next())
                {
                        studentId.add(resultSet1.getInt(1));
                leaveFormId.add(resultSet1.getInt(2));
                }


                Iterator<Integer> leaveFormIdIterator = leaveFormId.iterator();
                while(leaveFormIdIterator.hasNext())
                {
                        preparedStatement1 =con.prepareStatement("select * from leaveform where
                        leaveformid=?");
                        preparedStatement1.setInt(1,(int)leaveFormIdIterator.next());
                        ResultSet resultSet2 = preparedStatement1.executeQuery();
                        leaveFormBeanObject = new LeaveFormBean();
                        while(resultSet2.next())
                        {
                                leaveFormBeanObject.setDateOfApply(resultSet2.getString(3));
                                leaveFormBeanObject.setDateOfLeaveFrom(resultSet2.getString(4));
                                leaveFormBeanObject.setDateOfLeaveTo(resultSet2.getString(5));
                                leaveFormBeanObject.setReason(resultSet2.getString(6));
                                leaveFormBeanObjectList.add(leaveFormBeanObject);
                        }
                }
        }
        catch (SQLException e) {
                System.out.println("LeaveAppliedFacultyDao : Problem in getting Data");
        }
        sb.setLeaveFormId(leaveFormId);
        sb.setLeaveFormList(leaveFormBeanObjectList);
        return leaveFormBeanObjectList;
}
```

```
public LinkedList<StudentBean> leaveAppliedStudents()
{
	try
	{
		Iterator<Integer> studentIdIterator = studentId.iterator();
		while(studentIdIterator.hasNext())
		{
			PreparedStatement preparedStatement = con
					.prepareStatement("select studentname,regno,year,section
					from student where studentid=?");
			preparedStatement.setInt(1,(int)studentIdIterator.next());
			ResultSet resultSet1 = preparedStatement.executeQuery();
			studentBeanObject = new StudentBean();
			while(resultSet1.next())
			{
				studentBeanObject.setName(resultSet1.getString(1));
				studentBeanObject.setRegno(resultSet1.getString(2));
				studentBeanObject.setYear(resultSet1.getInt(3));
				studentBeanObject.setSection(resultSet1.getString(4));
				studentBeanObjectList.add(studentBeanObject);
			}
		}
	} catch (SQLException e) {
		System.out.println("LeaveAppliedFacultyDao : Problem in leave applied Students");
	}
	sb.setStudentList(studentBeanObjectList);
	return studentBeanObjectList;
}
public LinkedList<Integer> getLeaveFormId()
{
	LinkedList<Integer> leaveFormId =new LinkedList<Integer>();
	try {
		PreparedStatement preparedStatement = con
```

```
                                        .prepareStatement("select leaveformid from leaveapplied where
                                        acceptreject=? ");

                        preparedStatement.setString(1,LeaveAppliedFacultyDao.searchBy);


                        ResultSet resultSet1 = preparedStatement.executeQuery();

                        while(resultSet1.next())

                        {

                                leaveFormId.add(resultSet1.getInt(1));

                        }

                        preparedStatement.close();

                }

                catch(Exception e)

                {

                        System.out.println("Problem in leave form id method "+e.getMessage());

                }


                return leaveFormId;

        }

}
```

JSP Page

```
<%@page import="com.project.leaveod.beans.SearchByBean"%>
<%@page import="com.project.leaveod.beans.StudentBean"%>
<%@page import="com.project.leaveod.beans.LeaveFormBean"%>
<%@page import="java.util.*"%>
<%@page import="com.project.leaveod.dao.LeaveAppliedFacultyDao"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1" %>
<%
	String nameFromSession=(String)session.getAttribute("UserName");
	if(nameFromSession==null)
		response.sendRedirect("/LeaveOndutyForm/form-login.html");
	String searchBy = (String)request.getParameter("searchby1");
	LeaveAppliedFacultyDao obj= new LeaveAppliedFacultyDao();

	LinkedList<LeaveFormBean> leaveFormList= obj.leaveApplied(searchBy);
	LinkedList<StudentBean> studentList = obj.leaveAppliedStudents();
  %>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta charset="utf-8">
	<meta http-equiv="X-UA-Compatible" content="IE=edge">
	<meta name="viewport" content="width=device-width, initial-scale=1">
	<title>Leave Form</title>
	<link rel="stylesheet" href="assets/demo.css">
	<link rel="stylesheet" href="assets/form-basic.css">
</head>
	<header>
		<h1>Leave Form</h1>
<a href="LogoutServlet">Logout</a>
</header>
```

```
<body>
	<form action="SearchByServlet" method="post">
		<select name="searchby1">
			<option value="pending" selected="selected">Pending</option>
			<option value="accept">Accept</option>
			<option value="reject">Reject</option>
		</select>
		<input type="submit" value="Go" >
	</form>

	<form action="SearchBy2Servlet" method="post">
		<select name="searchby2">
			<option value="name" selected="selected">Name</option>
			<option value="regno">Regno</option>
			<option value="dateofapply">Date of Apply</option>
			<option value="leavefrom">Leave from Date</option>
			<option value="leaveto">Leave to Date</option>
		</select>
		<input type="text" name="searchvalue">
		<input type="submit" value="Search" >
	</form>

	<form action="UpdateFromFacultyServlet" method="post">
		<input type="submit" value="Submit All" >
		<table border="1" width="100%" style=”text-align:center;”>
			<tr bgcolor= #4CAF50;>
			<th>Serial Number</th>
			<th>Name</th>
			<th>Regno</th>
			<th>Year</th>
			<th>Section</th>
			<th>Date Of Apply</th>
```

```
                    <th>Date of Leave From</th>
                    <th>Date of Leave To</th>
                    <th>Reason</th>
                    <th>Accept/Reject</th>
                    </tr>

<%      Iterator<StudentBean> studentListIterator = studentList.iterator();
        Iterator<LeaveFormBean> listIterator = leaveFormList.iterator();
        int i=1;
        while(studentListIterator.hasNext() &&listIterator.hasNext())
        {
                StudentBean studentBeanObj = studentListIterator.next();
 %>

<tr>
<td><% out.print(i++); %>
<td><% out.print(studentBeanObj.getName()); %></td>
<td><% out.print(studentBeanObj.getRegno()); %></td>
<td><% out.print(studentBeanObj.getYear()); %></td>
<td><% out.print(studentBeanObj.getSection()); %></td>
<%
        LeaveFormBean leaveFormBeanObj = listIterator.next();
 %>
<td><% out.print(leaveFormBeanObj.getDateOfApply()); %></td>
<td><% out.print(leaveFormBeanObj.getDateOfLeaveFrom()); %></td>
<td><% out.print(leaveFormBeanObj.getDateOfLeaveTo());%></td>
<td><% out.print(leaveFormBeanObj.getReason()); %></td>
<td>
<select name="facultyselection" style=”width:100%”>
        <option value="pending">Pending</option>
        <option value="accept">Accept</option>
        <option value="reject">Reject</option>
</select>
```

```
            </td>
</tr>
<% } %>
</table>
</form>
</body>
</html>
```

10.2 DATABASE TABLES

FIELD	TYPE	SIZE	CONSTRAINT
studentid	int	11	Primary key, Auto Increment
studentname	varchar	30	Not null
regno	varchar	7	Not null, Unique
year	int	11	Not null
section	char	1	Not null
password	varchar	20	Not null

Table 10.1 Student Table

FIELD	TYPE	SIZE	CONSTRAINT
facultyid	int	11	Primary key, Auto increment
facultyname	varchar	30	Not null
advisorforyear	int	11	Not null
advisorforsection	char	1	Not null
role	varchar	20	Not null
username	varchar	30	Not null, Unique
password	varchar	30	Not null

Table 10.2 Faculty Table

FIELD	TYPE	SIZE	CONSTRAINT
leaveformid	int	11	Primary key, Auto increment
studentid	int	11	Foreign key (Student Table)
dateofapply	date		Not null
dateofleavefrom	varchar	10	Not null
dateofleaveto	varchar	10	Not null
reason	varchar	50	Not null

Table 10.3 Leave Form Table

FIELD	TYPE	SIZE	CONSTRAINT
odformid	int	11	Primary key, Auto increment
studentid	int	11	Foreign key (Student Table)
dateofapply	date		Not null
permissionfrom	varchar	20	Not null
permissionto	varchar	20	Not null
reason	varchar	50	Not null
workassignedby	varchar	30	Not null

Table 10.4 On Duty Form Table

FIELD	TYPE	SIZE	CONSTRAINT
leaveappliedid	int	11	Primary key, Auto increment
studentid	int	11	Foreign key (Student Table)
leaveformid	int	11	Foreign key (Leave Form Table)
acceptreject	varchar	10	Not null

Table 10.5 Leave Applied Table

FIELD	TYPE	SIZE	CONSTRAINT
odappliedid	int	11	Primary key, Auto increment
studentid	int	11	Foreign key (Student Table)
odformid	int	11	Foreign key (On Duty Form Table)
acceptreject	varchar	10	Not null

Table 10.6 On Duty Applied Table

10.3 SCREENSHOTS

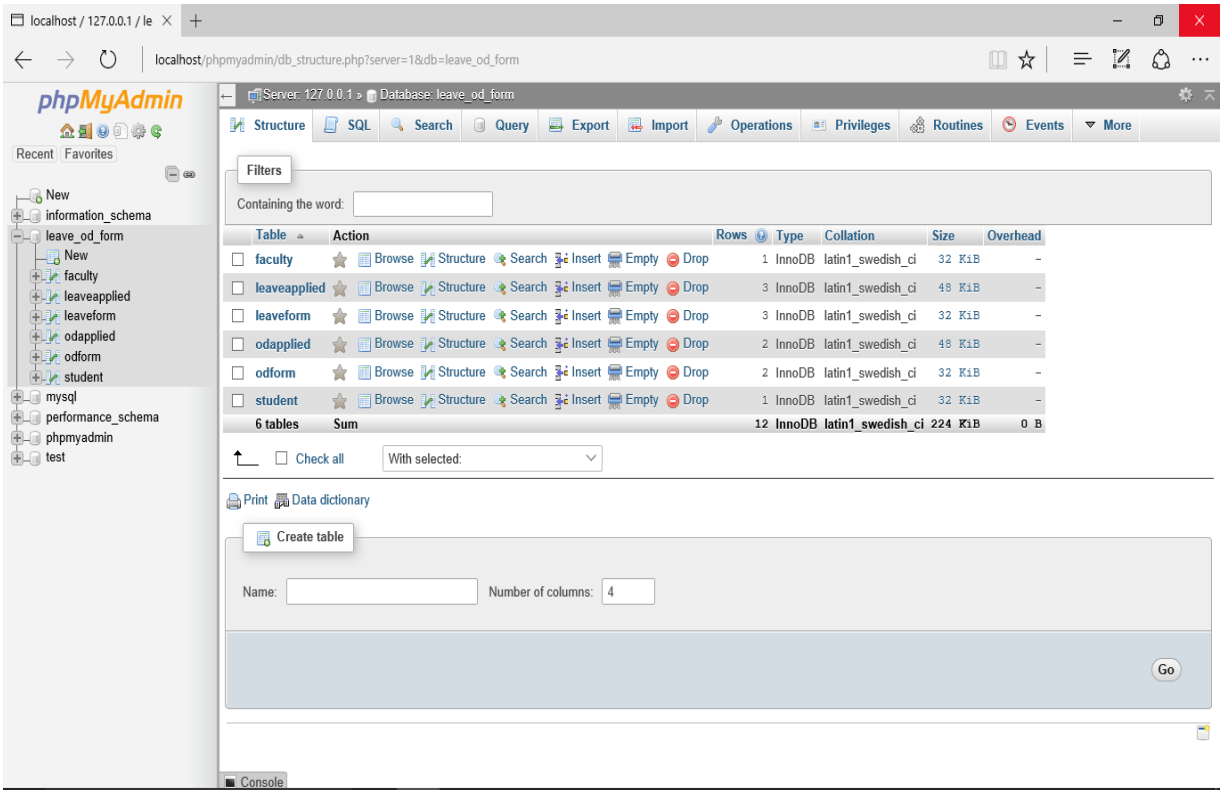


Figure 10.1 Admin Home

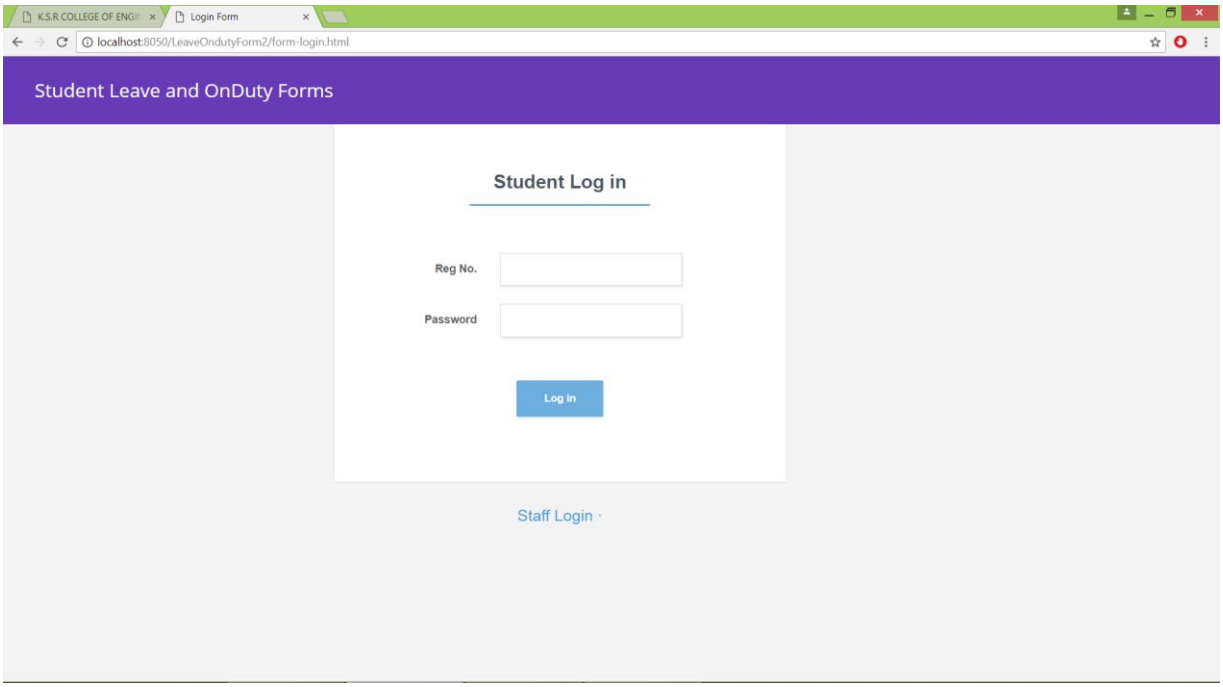


Figure 10.2 Student Login

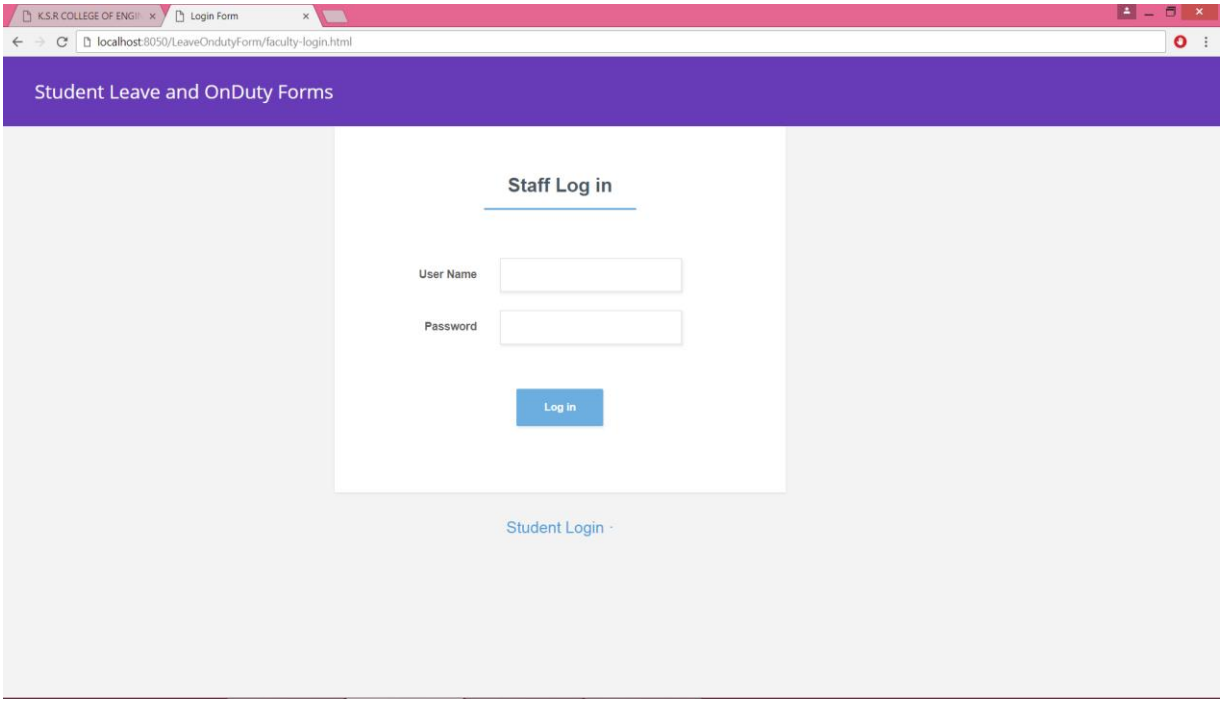


Figure 10.3 Faculty Login

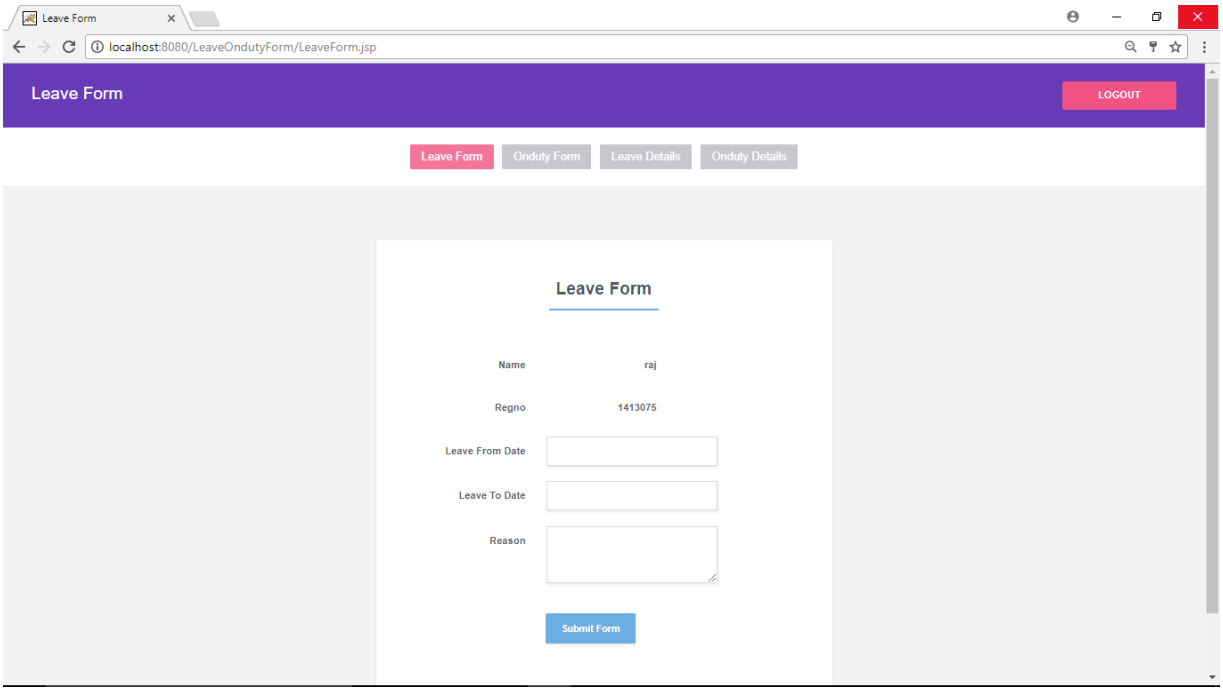


Figure 10.4 Leave Form

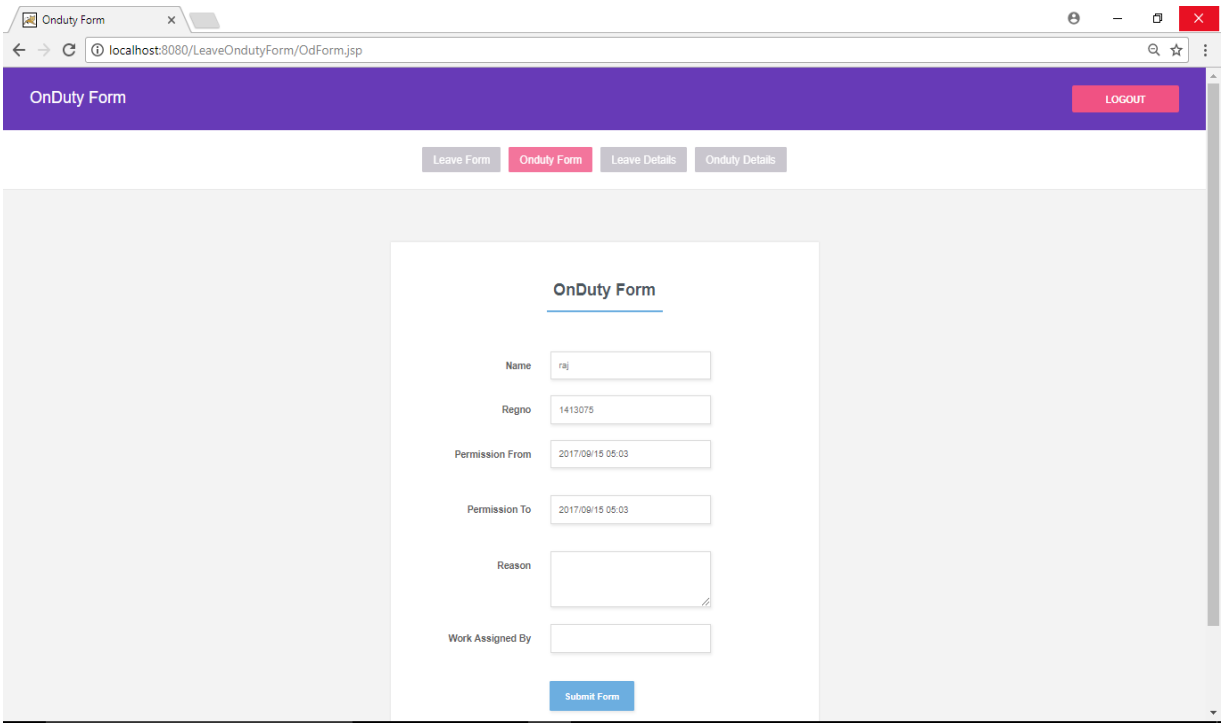
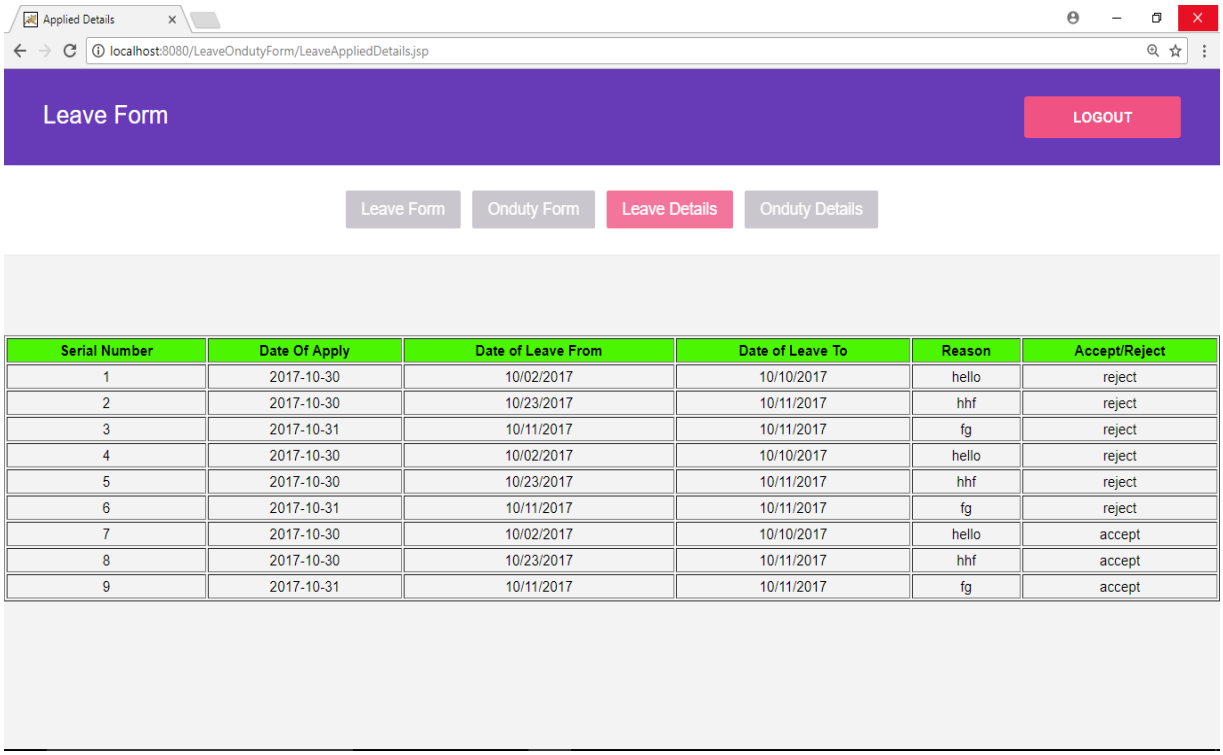
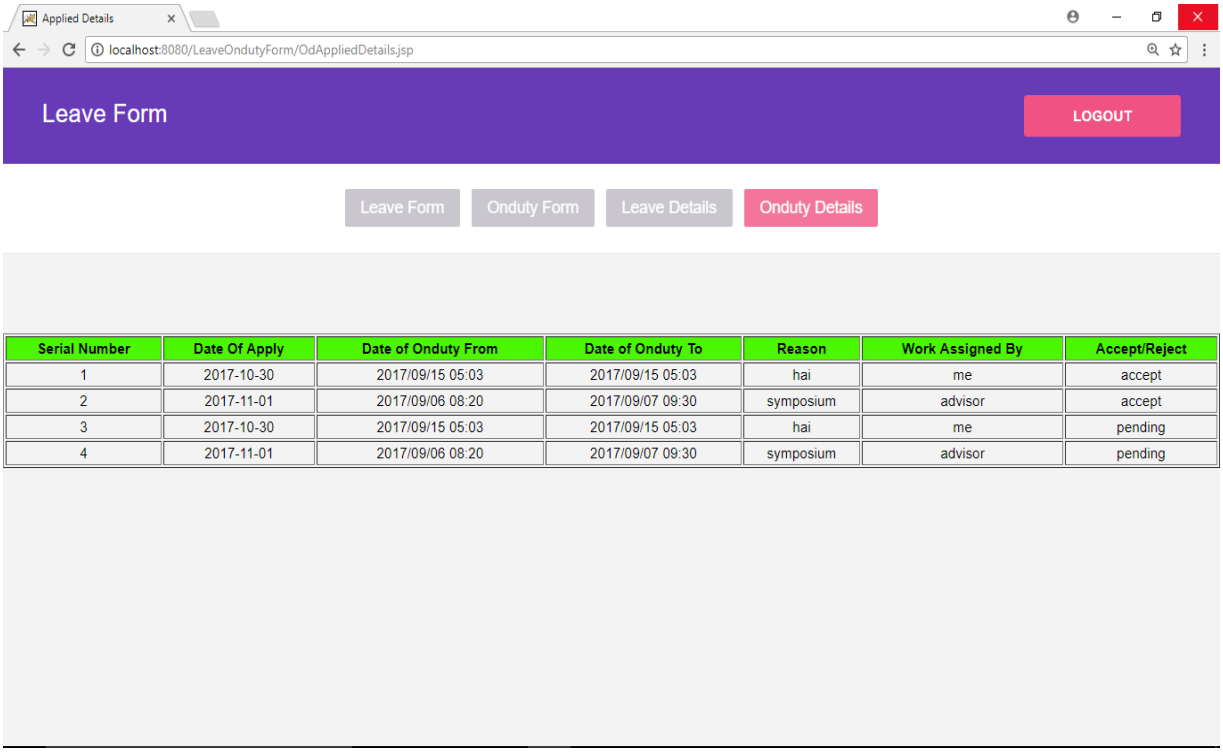


Figure 10.5 On Duty Form



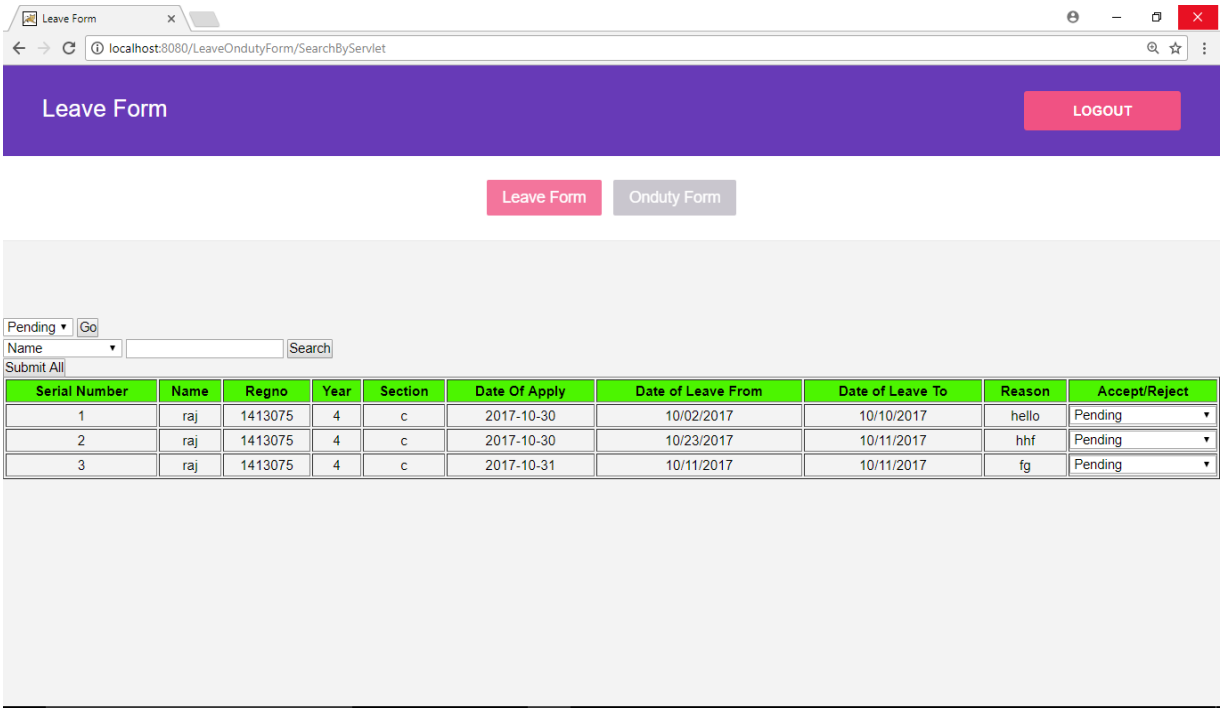
Serial Number	Date Of Apply	Date of Leave From	Date of Leave To	Reason	Accept/Reject
1	2017-10-30	10/02/2017	10/10/2017	hello	reject
2	2017-10-30	10/23/2017	10/11/2017	hhf	reject
3	2017-10-31	10/11/2017	10/11/2017	fg	reject
4	2017-10-30	10/02/2017	10/10/2017	hello	reject
5	2017-10-30	10/23/2017	10/11/2017	hhf	reject
6	2017-10-31	10/11/2017	10/11/2017	fg	reject
7	2017-10-30	10/02/2017	10/10/2017	hello	accept
8	2017-10-30	10/23/2017	10/11/2017	hhf	accept
9	2017-10-31	10/11/2017	10/11/2017	fg	accept

Figure 10.6 Applied Leave Details



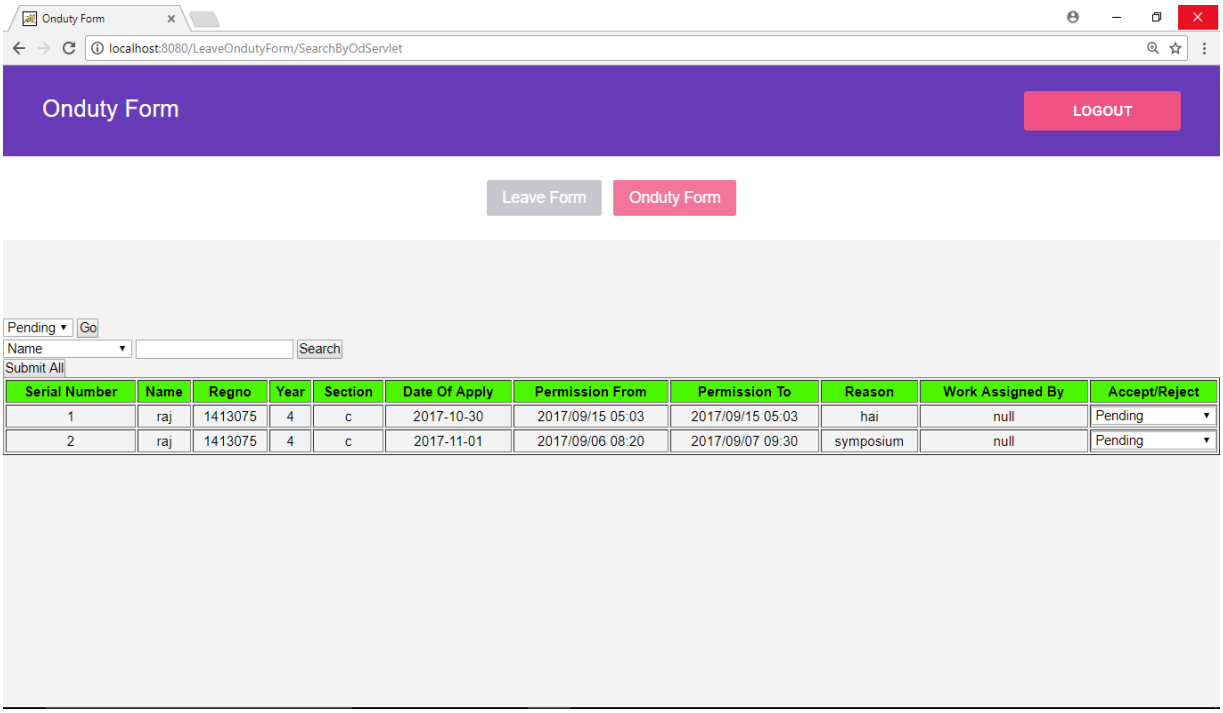
Serial Number	Date Of Apply	Date of Onduty From	Date of Onduty To	Reason	Work Assigned By	Accept/Reject
1	2017-10-30	2017/09/15 05:03	2017/09/15 05:03	hai	me	accept
2	2017-11-01	2017/09/06 08:20	2017/09/07 09:30	symposium	advisor	accept
3	2017-10-30	2017/09/15 05:03	2017/09/15 05:03	hai	me	pending
4	2017-11-01	2017/09/06 08:20	2017/09/07 09:30	symposium	advisor	pending

Figure 10.7 Applied On Duty Details



Serial Number	Name	Regno	Year	Section	Date Of Apply	Date of Leave From	Date of Leave To	Reason	Accept/Reject
1	raj	1413075	4	c	2017-10-30	10/02/2017	10/10/2017	hello	Pending
2	raj	1413075	4	c	2017-10-30	10/23/2017	10/11/2017	hhf	Pending
3	raj	1413075	4	c	2017-10-31	10/11/2017	10/11/2017	fg	Pending

Figure 10.8 Leave Records



Serial Number	Name	Regno	Year	Section	Date Of Apply	Permission From	Permission To	Reason	Work Assigned By	Accept/Reject
1	raj	1413075	4	c	2017-10-30	2017/09/15 05:03	2017/09/15 05:03	hai	null	Pending
2	raj	1413075	4	c	2017-11-01	2017/09/06 08:20	2017/09/07 09:30	symposium	null	Pending

Figure 10.9 On Duty Records

REFERENCES

1. Front End : http://www.w3schools.com
2. J2EE : http://www.javatpoint.com/java-tutorial
3. Java : http://www.tutorialspoint.com/java
4. JDBC : https://www.javatpoint.com/java-jdbc
5. Servlet : http://www.servlets.com
6. Queries : http://www.stackoverflow.com
7. Database : https://dev.mysql.com
8. Book : Java - Complete Reference by Herbert Schildt