Project Report: CPC Platform - Campus Project Connect Platform

Lead Developer: Brijesh Nishad
Roll No: 2101321550026
College: GNIOT - Greater Noida Institute of Technology
Batch: B.Tech CSE-IOT 2021-2025

Contents

1 Introduction

In today's fast-paced world, effective project management is crucial for the success of any organization. The Project Management System aims to streamline project management processes by providing a centralized platform for collaboration, communication, and task management. This report provides an in-depth analysis of the development process, from project initialization to deployment and maintenance.

2 Project Initialization

2.1 Development Environment Setup

The development environment was established using Eclipse IDE for Java EE Developers, ensuring a robust platform for building JSP Servlet applications. Apache Tomcat 9.0 server was chosen for deployment, offering seamless integration with the Java EE ecosystem.

2.2 Creating the Project

A new JSP Servlet project named "ProjectManagementSystem" was created within Eclipse IDE. This project served as the foundation for developing the project management system, providing a structured framework for organizing code and resources.

3 Database Design

3.1 Database Schema

The project management data was stored in a MySQL database, chosen for its reliability, scalability, and compatibility with Java applications. The database schema was carefully designed to accommodate the various entities involved in project management, including teams, members, organizations, faculties, projects, and project approvals.

4 User Interface Design

4.1 Wireframes/Mockups

Detailed wireframes and mockups were created to visualize the user interface of the project management system. These wireframes provided a clear representation of key functionalities such as team creation, member registration, login/authentication, and project approval request.

5 Authentication and Authorization

5.1 Mechanisms Implemented

Robust authentication and authorization mechanisms were implemented to ensure secure access to the project management system. Organizations, faculties, and students/members were provided with unique login credentials, and session management was utilized to track logged-in users and their roles.

6 Functionality Implementation

6.1 Servlets and JSP Pages

A set of servlets and JSP pages were developed to implement the core functionalities of the project management system. These included features such as team creation, member registration, login/authentication, project approval request, and project assignment. Each servlet was responsible for handling specific HTTP requests and responses, while JSP pages facilitated dynamic content generation and presentation.

6.2 Business Logic

The business logic of the project management system was implemented to manage project approval workflows, project assignment, and data validation. Input data was carefully validated to ensure accuracy and integrity, and business rules were enforced to maintain consistency throughout the system.

7 Database Interaction

7.1 Data Access Objects (DAOs)

A set of Data Access Object (DAO) classes were developed to interact with the MySQL database. These DAOs encapsulated the logic for performing CRUD (Create, Read, Update, Delete) operations on entities such as teams, members, projects, etc. This abstraction layer facilitated seamless database interaction and ensured data consistency and integrity.

8 Integration and Testing

8.1 Testing Approach

A rigorous testing approach was adopted to ensure the reliability and functionality of the project management system. Unit testing was conducted for individual servlets and DAO classes to validate their behavior under different scenarios. Integration testing was performed to verify the seamless interaction between frontend and backend components, ensuring that all features worked together seamlessly.

9 Deployment

9.1 Deployment Process

The project management system was deployed to an Apache Tomcat server, configured with the MySQL database connection. Server settings were optimized to ensure optimal performance and scalability, and the deployment process was carefully monitored to minimize downtime and disruptions.

10 Documentation and User Training

10.1 Documentation Provided

Comprehensive documentation was provided to assist users in understanding and utilizing the project management system effectively. A user manual and developer guide were created, offering step-by-step instructions for system navigation, feature utilization, and troubleshooting. Additionally, training sessions were conducted for end-users (organization, faculty, students/members) to familiarize them with the system's functionality and best practices.

11 Maintenance and Updates

11.1 Ongoing Maintenance

Plans were established for ongoing maintenance and updates to address bugs, implement new features, and improve system performance. User feedback was actively monitored, and necessary enhancements were made based on user experience and evolving business requirements. Regular software updates and patches were applied to ensure the security and stability of the system.

12 Project Structure

12.1 Source Directory

The source directory contains the core Java packages and classes of the project. These packages are organized based on functionality and responsibilities.

- `com.projectmanagement.servlets`: This package contains servlet classes responsible for handling HTTP requests and generating dynamic web content. Servlet files include:
  * `CreateTeamServlet.java`
  * `LoginServlet.java`
  * `RegisterServlet.java`
  * `ProjectApprovalServlet.java`
  * `AssignProjectServlet.java`
  * `DashboardServlet.java`
  * `LogoutServlet.java`
  * `UpdateTeamServlet.java`
  * `DeleteTeamServlet.java`
- `com.projectmanagement.model`: The model package contains classes representing data entities such as teams, members, projects, etc. Model files include:

    * `Team.java`
    * `Member.java`
    * `Organization.java`
    * `Faculty.java`
    * `Project.java`
    * `ProjectApproval.java`
    * `Task.java`
    * `Comment.java`

- `com.projectmanagement.dao`: DAO (Data Access Object) classes reside in this package. These classes are responsible for interacting with the database and performing CRUD (Create, Read, Update, Delete) operations. DAO files include:
    * `TeamDAO.java`
    * `MemberDAO.java`
    * `OrganizationDAO.java`
    * `FacultyDAO.java`
    * `ProjectDAO.java`
    * `ProjectApprovalDAO.java`
    * `TaskDAO.java`
    * `CommentDAO.java`
- `com.projectmanagement.util`: This package contains utility classes that provide helper functions and common functionalities used across the project. Utility files include:
    * `DBUtil.java`
    * `ValidationUtil.java`
    * `DateUtil.java`
    * `EmailUtil.java`

12.2 Web Directory

The web directory contains resources and files related to the web application's frontend.

- `WEB-INF`: This directory contains configuration files and views that are not directly accessible to the client. It includes the `web.xml` deployment descriptor and the `views` directory for JSP files.
    - `web.xml`: Deployment descriptor containing servlet mappings and configuration.
    - `views`: JSP files responsible for rendering dynamic web content and interacting with servlets. JSP files include:
        * `index.jsp`

    * `login.jsp`
    * `register.jsp`
    * `createTeam.jsp`
    * `projectApproval.jsp`
    * `assignProject.jsp`
    * `dashboard.jsp`
    * `updateTeam.jsp`
    * `deleteTeam.jsp`
    * `viewTask.jsp`
    * `addComment.jsp`

- `css`: Contains CSS (Cascading Style Sheets) files for styling the web application. CSS files include:
    * `styles.css`
    * `responsive.css`
- `js`: Contains JavaScript files for enhancing the interactivity and functionality of the web application. JavaScript files include:
    * `scripts.js`
    * `validation.js`
- `images`: Contains image files used in the web application. Image files include:
    * `logo.png`
    * `banner.jpg`

13 Database Schema

13.1 Tables

The MySQL database schema consists of several tables designed to store project management-related data.

Table 1: Database Tables and Attributes

Table	**Attributes**
Teams	team_id (PK), team_name, organization_id (FK)
Members	member_id (PK), member_name, team_id (FK), role, email, password
Organizations	organization_id (PK), organization_name, email, password
Faculties	faculty_id (PK), faculty_name, email, password
Projects	project_id (PK), project_name, description, status, organization_id (FK)
ProjectApprovals	approval_id (PK), project_id (FK), member_id (FK), status, date_requested

Each table is designed to capture specific information related to project management, such as teams, members, organizations, projects, and project approvals.

13.2 Sample Queries

Here are some sample SQL queries used to interact with the database:

- Create a new team:

```
INSERT INTO Teams (team_name, organization_id)
VALUES ('Team Alpha', 1);
```

- Retrieve all projects assigned to a specific member:

```
SELECT * FROM Projects WHERE organization_id = 1;
```

14 Code Listings

14.1 Servlet Example: `CreateTeamServlet.java`

```
package com.projectmanagement.servlets;

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.projectmanagement.dao.TeamDAO;
import com.projectmanagement.model.Team;

@WebServlet("/CreateTeamServlet")
public class CreateTeamServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String teamName = request.getParameter("teamName");
        String description = request.getParameter("description");

        Team team = new Team();
        team.setTeamName(teamName);
        team.setDescription(description);

        TeamDAO teamDAO = new TeamDAO();
        boolean isCreated = teamDAO.createTeam(team);

        if (isCreated) {
            response.sendRedirect("dashboard.jsp");
        } else {
            request.setAttribute("errorMessage", "Failed to create team. Try again.");
            request.getRequestDispatcher("createTeam.jsp").forward(request, response);
        }
    }
}
```

14.2 DAO Example: `TeamDAO.java`

```
package com.projectmanagement.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import com.projectmanagement.model.Team;
import com.projectmanagement.util.DBUtil;

public class TeamDAO {
```

```
private Connection connection;

public TeamDAO() {
    connection = DBUtil.getConnection();
}

public boolean createTeam(Team team) {
    try {
        String query = "INSERT INTO teams (team_name, description) VALUES (?, ?)";
        PreparedStatement pstmt = connection.prepareStatement(query);
        pstmt.setString(1, team.getTeamName());
        pstmt.setString(2, team.getDescription());
        int rowsAffected = pstmt.executeUpdate();
        return rowsAffected > 0;
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return false;
}

public List<Team> getAllTeams() {
    List<Team> teams = new ArrayList<>();
    try {
        String query = "SELECT * FROM teams";
        PreparedStatement pstmt = connection.prepareStatement(query);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
            Team team = new Team();
            team.setTeamId(rs.getInt("team_id"));
            team.setTeamName(rs.getString("team_name"));
            team.setDescription(rs.getString("description"));
            teams.add(team);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return teams;
}

public boolean updateTeam(Team team) {
    try {
        String query = "UPDATE teams SET team_name = ?, description = ? WHERE team_id = ?";
        PreparedStatement pstmt = connection.prepareStatement(query);
        pstmt.setString(1, team.getTeamName());
        pstmt.setString(2, team.getDescription());
```

```
            pstmt.setInt(3, team.getTeamId());
            int rowsAffected = pstmt.executeUpdate();
            return rowsAffected > 0;
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return false;
    }

    public boolean deleteTeam(int teamId) {
        try {
            String query = "DELETE FROM teams WHERE team_id = ?";
            PreparedStatement pstmt = connection.prepareStatement(query);
            pstmt.setInt(1, teamId);
            int rowsAffected = pstmt.executeUpdate();
            return rowsAffected > 0;
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return false;
    }

    public Team getTeamById(int teamId) {
        Team team = null;
        try {
            String query = "SELECT * FROM teams WHERE team_id = ?";
            PreparedStatement pstmt = connection.prepareStatement(query);
            pstmt.setInt(1, teamId);
            ResultSet rs = pstmt.executeQuery();
            if (rs.next()) {
                team = new Team();
                team.setTeamId(rs.getInt("team_id"));
                team.setTeamName(rs.getString("team_name"));
                team.setDescription(rs.getString("description"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return team;
    }
}
```

14.3 Utility Example: `DBUtil.java`

```
package com.projectmanagement.util;
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBUtil {
    private static Connection connection;

    static {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/projectdb",
                    "username", "password");
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        }
    }

    public static Connection getConnection() {
        return connection;
    }
}
```

15 User Manual

15.1 Login and Authentication

- **Step 1:** Navigate to the login page.
- **Step 2:** Enter your username and password.
- **Step 3:** Click on the ”Login” button.
- **Step 4:** If credentials are correct, you will be redirected to the dashboard.

15.2 Creating a Team

- **Step 1:** Navigate to the ”Create Team” page.
- **Step 2:** Enter the team name and select the organization from the dropdown.
- **Step 3:** Click on the ”Create Team” button.
- **Step 4:** If the team is created successfully, you will see a confirmation message.

16 Developer Guide

16.1 Project Setup

- **Step 1:** Download and install Eclipse IDE for Java EE Developers.

- **Step 2:** Install Apache Tomcat 9.0 server.
- **Step 3:** Clone the project repository from the version control system.
- **Step 4:** Import the project into Eclipse IDE.
- **Step 5:** Configure the Apache Tomcat server in Eclipse.
- **Step 6:** Set up the MySQL database and update the database connection details in `DBUtil.java`.
- **Step 7:** Run the project on the Apache Tomcat server.

16.2 Adding a New Feature

- **Step 1:** Identify the new feature requirement.
- **Step 2:** Create the necessary model classes in the `com.projectmanagement.model` package.
- **Step 3:** Develop the DAO classes to interact with the database.
- **Step 4:** Create servlets to handle HTTP requests related to the new feature.
- **Step 5:** Develop JSP pages to provide the user interface for the new feature.
- **Step 6:** Update the `web.xml` deployment descriptor if necessary.
- **Step 7:** Test the new feature thoroughly to ensure its proper functionality.

17 Training Sessions

17.1 Overview

Training sessions were conducted for different user groups to ensure they are proficient in using the Project Management System.

17.2 Organization Training

- **Session 1:** Introduction to the Project Management System
- **Session 2:** Managing Teams and Members
- **Session 3:** Overseeing Project Approvals and Assignments

17.3 Faculty Training

- **Session 1:** Using the Dashboard
- **Session 2:** Approving and Managing Projects
- **Session 3:** Viewing Team Progress and Feedback

17.4 Student/Member Training

- **Session 1:** Registration and Login
- **Session 2:** Creating and Joining Teams
- **Session 3:** Submitting and Tracking Project Requests

18 Conclusion

The Project Management System represents a significant milestone in enhancing project management capabilities within organizations. By leveraging cutting-edge technologies and best practices, the system provides a robust and user-friendly platform for managing projects efficiently. Moving forward, ongoing maintenance and updates will ensure that the system remains aligned with evolving business needs and technological advancements, delivering maximum value to stakeholders.