**Abstract**

Campus Connect is a web-based platform designed to bridge the communication gap between the various roles of an organization.

Campus Connect enables teachers to share the various types of documents/resources and allows seamless access to resources by the students along with strong administrative functionalities.

Campus Connect offers the view/download option for the resources. Some of the most important features include centralized dashboards according to the role of the user, data compression, and secure data access.

Therefore, integrating feature advances such as role-based access control will improve operational efficiency for Campus Connect and create a digital ecosystem that is easily accessible to the academic community

**Keywords** – Resources, users, centralised platform, management, compression

|  |  |  |
| --- | --- | --- |
|  | **LIST OF CONTENTS** |  |
| **Chapter** | **Title** | **Page No** |
|  | ABSTRACT |  |
| **1** | INTRODUCTION | 1 |
| 1.1 | General overview of the problem | 1 |
| 1.2 | Aim and objective | 1 |
| 1.3 | Feasibility Study | 2 |
| 1.4 | Problem Definition | 3 |
| 1.5 | Analysis of the Problem | 3 |
| 1.6 | Solution Strategy | 4 |
| 1.7 | Software Requirement Specifications (SRS) | 4-7 |
| 1.7.1 | Functional Requirements | 4-7 |
| 1.7.2 | Non-Functional Requirements | 8 |
| **2** | **PROJECT PLANNING** | 9-12 |
| 2.1 | Hardware and Software Requirements | 9 |
| 2.2 | Team Structure | 10 |
| 2.3 | SLDC | 11 |
| 2.4 | Gantt–chart | 12 |
| **3** | **DESIGN STRATEGY FOR THE SOLUTION** |  |
| 3.1 | Data Flow Diagram | 13 |
| 3.2 | Entity Relationship Diagram | 14 |
| 3.3 | Relational Schema | 14 |
|  |  |  |
| **4** | **DETAILED TEST PLAN** | 15-17 |
| 4.1 | Test Unit and Methods | 16 |
| 4.2 | Approach for Testing | 17 |
| 4.3  4.4 | Test Deliveries  Schedules for Testing | 17  17 |
|  |  |  |
| **5** | **IMPLEMENTATION DETAILS** | 18-20 |
| 5.1 | Pseudocodes | 18-19 |
| 5.2 | Algorithms | 20 |
| **6** | **RESULT AND DISCUSSION** | 21-29 |
|  |  |  |
| **7** | **SUMMARY AND CONCLUSION** | 30-31 |
| 7.1 | Summary of Achievement | 30 |
| 7.2 | Difficulties Encountered During the Project | 30 |
| 7.3 | Limitations of the Project | 30 |
| 7.4 | Future scope of the Project | 30 |
| 7.5 | Conclusion | 31 |
|  | **REFERENCES AND BIBLIOGRAPHY** | 32 |
|  | **LIST OF FIGURES** |  |
| FIG. NO. | **FIGURE NAME** | PAGE NO. |
| 1 | **Team Structure** | 10 |
| 2 | **Scrum Process** | 11 |
| 3 | **Gantt Chart** | 12 |
| 4 | **Level 0 DFD** | 13 |
| 5 | **Level 1 DFD** | 13 |
| 6 | **Entity Relationship Diagram** | 14 |
| 7.1 | **Landing Page** | 21 |
| 7.2 | **Login Page** | 21 |
| 7.3 | **Error Handling for Invalid Credentials** | 22 |
| 7.4 | **Logout Confirmation Message** | 22 |
| 7.5 | **Admin’s Home Page** | 23 |
| 7.6 | **Admin Interaction: Managing Student Options via Sidebar** | 23 |
| 7.7 | **Admin Interaction: Managing Teacher Options via Sidebar** | 24 |
| 7.8 | **Admin Interaction: Managing HOD Options via Sidebar** | 24 |
| 7.9 | **Admin Interaction: Adding or Updating Users** | 25 |
| 7.10 | **Admin Interaction: Deleting Users** | 25 |
| 7.11 | **HOD’s Home Page** | 26 |
| 7.12 | **Add New Course** | 26 |
| 7.13 | **Manage Courses** | 27 |
| 7.14 | **Teacher’s Home Page** | 28 |
| 7.15 | **File Upload Demonstration** | 28 |
| 7.16 | **Student’s Page** | 29 |
| 7.17 | **File Access Demonstration** | 29 |

1. **Introduction**

Campus Connect is a digital solution designed to improve the management of academic resources which can exist in multiple forms such as PPTs, PDFs, Video Lectures and Audio Books. Campus Connect empowers administrators, faculty, and students to work collaboratively with enhanced efficiency and effectiveness. The core objective of Campus Connect is to provide adequate amount of resources to the students alongside allowing teachers to share the academic resources over a common platform. Campus Connect simplifies user interaction with the educational system, making it more intuitive and well-organized.

**1.1 General Overview of the Problem**

Lack of a single source for student materials leading to dispersed resources across different platforms which makes it hard for them to find what they need. Students often tend to refer to the resources from multiple platforms which may not be aligned to the actual course curriculum increasing the confusion among the students regarding the correctness of the resources. There are challenges in monitoring progress at educational intuitions since no incorporated tool is available for tracking course completion and progress. Teachers often face challenges in distributing resources across multiple platforms, making it harder to ensure students have easy access to the correct resources.

**1.2 Aims and Objectives**

**Aim:**

The overall objective of the Campus Connect project is to come up with a comprehensive centralized web-based platform that will enhance how modern academic institutions manage their resources.

**Objectives:**

* To facilitate Sharing of Resources.
* To ensure availability of correct resources to the students
* To implement the element of Role-Based Access Control.

**1.3 Feasibility Study**

1. Technical Feasibility:

Campus Connect will be a fully digital platform utilizing web-based technologies to ensure a user-friendly experience. Strong back-end development is essential for supporting features like role-based access, course management, and resource sharing. Scalable web and database servers will handle high user volumes and data. Compatibility with various devices and operating systems must meet the diverse needs of students, teachers, and administrators. Regular maintenance and updates will keep the platform secure and up to date with the latest technologies.

1. Operational Feasibility:

Campus Connect must be easy to use for all user categories, with a user-friendly design that minimizes the learning curve. Training and support resources will be provided. The platform should integrate smoothly with existing LMS, SIS, and other academic tools, enhancing rather than disrupting workflows. Institutions must have the necessary infrastructure and resources to maintain the platform post-deployment.

1. Scheduling Feasibility:

The project will follow an adaptive schedule, breaking the timeline into iterative sprints that deliver prioritized features. This approach allows for flexibility in responding to changing requirements and feedback. Resource allocation must be efficient, ensuring critical tasks are completed on time. Clear milestones and deliverables will be set at the end of each sprint, with regular reviews to keep the project on track and address any deviations from the timeline.

**1.4 Problem Definition**

1) Inefficient Course Management - Managing course content and assignments is difficult.

2) A Missing Centralized Hub of Resources - Resources are scattered across multiple platforms.

3) Chances arises that students may get mis-leaded or confused due to variety and availability of resources over the internet which might not fulfill the course’s syllabi.

4) Lack of ability to share external references and resources which might not exist within the institutions resource hub.

5) Continuous and rapidly increasing need of space to store the resources may make the system complex to manage and increase the operational charge.

**1.5 Analysis of the Problem**

The system that is currently existing has huge inefficiencies in course management, making it relatively challenging for instructors to get the room right for content and assignment posting. There is no centralized resource hub with materials being spread out across various places, creating a mess for students and faculties alike. The reason is that more chances exist for such students to rely on misleading or non-relevant online resources that have nothing to do with the syllabus of the course. For instance, external references cannot easily be shared since they are probably not included in the resource hub maintained by the institution, and hence, have limited access to possibly valuable materials. In addition, the cost of increasing storage requirements for additional growing resources could complicate the management of the system considerably and raise operational costs at a very higher level. Indeed, the issues and concerns above necessitate an improved streamlined and efficient approach in resource management.

**1.6 Solution Strategy**

1. Comprehensive Course Management System **-** Develop a course management module for efficient course creation, management, and progress tracking.
2. Centralized Repository of Resources **-** Create a unified platform for storing and sharing academic resources, enhancing collaboration and access to materials.
3. Only the resources approved by the teachers are shared with the students
4. Allow teachers to share external references and resources with the students.
5. Use compression technique, lossless, if possible, to Optimizely use the available storage and reduce the complexity and cost of the system.

**1.7 Software Requirement Specification**

SRS document or Software Requirement document provides a technical specification for the requirements of the software. The aim of software requirements definition becomes that of completely and consistently specifying the technical requirements for the software, accurately and without being vague.

**1.7.1 Functional Requirements**

**R1: Login Process**

**Description:** This requires the user to provide his valid login user id and password to enter his account.

**R1.1: Login for Administrator**

**Input:** Admin id and corresponding password

**Output:** Administrator will be redirected to the homepage of his account.

**Processing:** The username and the password entered by the user are verified against the database entry. If found valid then successful login with the username on top of the account will be displayed, else an error message will be generated “Login unsuccessful. Please try again”.

**R2: Administrator module**

**Description:** Administrator is given full control for site maintenance. He/she can add the user, update the user, delete the user, view total students, view total teachers, and view total HOD's.  **R2.1: Add user**

**Input:** Add a new user and assign role accordingly i.e. either student, teacher or HOD.

**Output:** A message is displayed “User Added Successfully”

**Processing:** The user details will be stored in the database.

**R2.2: Delete User**

**Input:** Delete an existing user from the drop-down list.

**Output:** A message is displayed “User successfully deleted”.

**Processing:** The user details will be deleted from the database.

**R2.3: Update User**

**Input:** Update the details of an existing user

**Output:** A message is displayed “User details updated successfully”.

**Processing:** The user details will be updated in the database.

**R3: HOD**

**Description:** The HOD has the ability to update department details, update faculty, approve requests, add course and manage course.

**R3.1: Update Department Details.**

**Input:** Select update department details option and press the button

**Output:** A message is displayed “Department Details Updated”.

**Processing:** The department details will be stored in database.

**R3.2: Update Faculty**

**Input:** Select update faculty option and press the button.

**Output:** A message is displayed “Faculty Updated”.

**Processing:** The faculty information will be stored in the database.

**R3.3: Approve Requests**

**Input:** Select approve requests option and press the button.

**Output:** A message is displayed “Request Approved”.

**R3.4: Add Course**

**Input:** Select add course option and press the button.

**Output:** A message is displayed “Course Added”.

**Processing:** The course information will be stored in the database.

**R3.2: Manage Course**

**Input:** Select manage course option and press the button.

**Output:** A message is displayed “Course Updated”.

**Processing:** The course information will be stored in the database.

**R4: Teacher**

**R4.1 Upload PDFs**

**Input:** Select a PDF and upload it**.**

**Output:** A message is displayed “Upload Successful”.

**Processing:** The PDF is stored in the database.

**R4.2 Upload PPTs**

**Input:** Select a PPT and upload it**.**

**Output:** A message is displayed “Upload Successful”.

**Processing:** The PPT is stored in the database.

**R4.3 Upload Audio and Reference Books**

**Input:** Select an Audio or Reference Book and upload it**.**

**Output:** A message is displayed “Upload Successful”.

**Processing:** The Audio or Reference Book is stored in the database.

**R4.4 Upload Video Tutorials**

**Input:** Select a Video and upload it.

**Output:** A message is displayed “Upload Successful”.

**Processing:** The Video is stored in the servers’ file system and the metadata of the video in RDMS.

**R5: Student**

**R5.1 Access PDFs**

**Input:** Department Name, Course Name, Semester and Subject.

**Output:** List of available PDFs along with view and download options.

**Processing:** The file attributes are fetched from the database and using those file attributes the correspondent file is fetched from the servers local file system.

**R5.2 Access PPTs**

**Input:** Department Name, Course Name, Semester and Subject.

**Output:** List of available PPTs along with view and download options.

**Processing:** The file attributes are fetched from the database and using those file attributes the correspondent file is fetched from the servers local file system.

**R5.3 Access Audio and Reference Books**

**Input:** Department Name, Course Name, Semester and Subject.

**Output:** List of available audio and reference along with view and download options.

**Processing:** The file attributes are fetched from the database and using those file attributes the correspondent file is fetched from the servers local file system.

**R5.4 Access Video Tutorials**

**Input:** Department Name, Course Name, Semester and Subject

**Output:** List of available videos along with view and download options

**Processing:** The video attributes are fetched from the database and using those file attributes the correspondent file is fetched from the servers local file system.

**1.7.2 Non-Functional Requirements**

* **Response Time**: Response time is an estimate for how long the operation will take considering a set of circumstances. For the software to work effectively, the response time needs to be on the lower side. There are not strict response time constraints in this case. Less response time means more traffic. Response to any of the requests should be quick.
* **Processing Time**: the main Processing Time involved will be downloading the different web pages with the data retrieve from the database. Static pages will be displayed quicker than the dynamic pages where the data needs to be retrieved from the database.
* **User Friendliness**: This web application has been developed keeping in mind the user friendliness. Any person with good knowledge over the internet and computers can be the user of the product.
* **Throughput**: Throughput describes what is expected to be done in a unit of time. To achieve efficiency, a system needs to maximize throughput.

**Project Planning**

* 1. **Hardware and Software Requirements**

**Hardware Requirements:**

* **CPU:** Intel v4 or equivalent, 8 cores (2.10 GHz).
* **Memory:** 32 GB
* **Storage:** 500 GB SSD (Primary), 2 TB HDD (Backup).
* **Network:** Gigabit Ethernet, 1 Gbps.
* **Client Machines:**
* **CPU:** Intel Pentium/AMD Athlon or newer
* **Memory:** 2 GB (min), 4 GB (recommended).
* **Network:** 1 Mbps internet speed.

**Software Requirements -Server:**

* Operating System: Linux /Windows Server.
* Deployment: VM & Kubernetes/Open Shift.
* Server Application:
  + Apache HTTPD or Nginx or Tomcat for WAR Packaging.
  + No need of any server application for JAR Packaging.
* Database Server: MySQL version 8.0.30
* JDK: 17
* Built Tool: Maven

**Software Requirements-Client:**

* Operating System: Linux /Windows/Macintosh/Android/Ios.
* Browser: Chrome/Edge/Firefox/Safari etc
  1. **Team Structure**

Project Guide

Dr. Samarjeet Borah

Subodh Adhikari

(202322020)

Manav Agarwal

(202322019)

**Fig 1: Team Structure**

* 1. **Software Development Life Cycle**

**Agile methodology**: Agile methodology represents an iterative approach towards the development of software, characterized by flexibility and customer collaboration. This methodology attaches great importance to continuous feedback, adaptive planning, and delivering functional software in short iterations, called sprints. It, hence, allows regular adjustments and improvements regarding changing requirements.

**Agile Framework**- ***Scrum***: Scrum is one of the agile frameworks that work on the management and control of iterative work at the project level. The Scrum technique includes a self-organizing, cross-functional team led by a Scrum Master who work in sprint fashion to deliver potentially shippable increments of the product, emphasizing communication, collaboration, and flexibility to adapt to emerging requirements.

1**. Product Backlog Creation**: The product owner prepares a priority list of features or requirements that the company wants to see in the product; it goes by the name of Product Backlog.

2. **Sprint Planning**: The Scrum Team, from the Product Backlog, selects a set of items for the upcoming Sprint and enforces the Sprint Goal.

3**. Sprint Execution**: Implementing the items selected with daily stand-ups to track progress and address impediments.

4. **Sprint Review**: This meeting takes place to present to the stakeholders the completed work for feedback and review.

5**. Sprint Retrospective**: The team looks back on the Sprint to understand what went well and what can be improved, all in time for the next Sprint.



**Fig 2: Scrum Process**

* 1. **Gantt-Chart**

**A chart with red and green squares

Description automatically generated**

**Fig 3: Gantt Chart**

1. **Design Strategy For The Solution**
   1. **Data Flow Diagram**

**A diagram of a response

Description automatically generated**

**Fig 4: Level 0 DFD**

**A diagram of a diagram

Description automatically generated**

**Fig 5: Level 1 DFD**

* 1. **Entity Relationship Diagram**

**A computer screen shot of a computer

Description automatically generated**

**Fig 6: ER Diagram**

* 1. **Relational Schema**

USERS(user\_id PRIMARY KEY, username UNIQUE NOT NULL, password NOT NULL, email UNIQUE NOT NULL, is\_active DEFAULT TRUE)

ROLES(role\_id PRIMARY KEY, role\_name NOT NULL, user\_id FOREIGN KEY REFERENCES USERS(user\_id) ON DELETE CASCADE)

DEPARTMENTS(department\_id PRIMARY KEY, department\_name NOT NULL, hod\_id FOREIGN KEY REFERENCES USERS(user\_id) ON DELETE SET NULL)

COURSES(course\_id PRIMARY KEY, course\_name NOT NULL, department\_id FOREIGN KEY REFERENCES DEPARTMENTS(department\_id) ON DELETE CASCADE)

SUBJECTS(subject\_id PRIMARY KEY, subject\_name NOT NULL, course\_id FOREIGN KEY REFERENCES COURSES(course\_id) ON DELETE CASCADE, teacher\_id FOREIGN KEY REFERENCES USERS(user\_id) ON DELETE SET NULL)

1. **DETAILED TEST PLAN**

Given that this is a project concerned with web applications, various sides of quality need to be addressed. These qualities are incorporated into a web application as the aftermath of good design. Furthermore, the whole test plan will be based on the review and analysis of these dimensions: -

**1.  Content:** this entails reviews at both the levels of syntax and semantics. At the level of syntax, spelling, punctuation and grammar will be evaluated in hardcoded tests appearing on the client screen of web pages. At the level of semantics, correctness of the information presented, consistency and lack of ambiguity are all reviewed.

**2.  Function:**all the functions described in the requirements specification document will be tested. Errors in the main function which will give criminal intent to the client should be dealt with some error messages. The three levels of testing were: unit testing; system testing; and integration testing.

**3. Usability:** it is tested in connecting each category of users with support from the interface and letting them learn and put into effect all terms of navigation syntax and semantics.

**4.  Navigability**: this is tested to ensure that all navigation syntax and semantics are exercised, uncovering navigation errors (e.g. dead links, improper links, etc.). Every link on every page and frames is tested.

**5.  Performance:** The performance is tested on various operating conditions, configurations, and loads such that the system is responsive to user interaction and can handle extreme loads without unacceptable degradation of operation.

* 1. **Test Units and Methods:**

The testing process essentially comprises three stages, namely unit, integration and system testing’s. A test case is a document that describes an input, action or event and an expected response to determine if a feature of the application is working properly. The three levels of testing are hence required to ensure correct behaviour of the application. The basic units that need to be tested are:

* + 1. **Unit Testing:**

In Unit Testing, we are required to identify the test units and test them separately after the development of each module. The units to be tested are as follows:

* Module 1 - Admin
* Module 2 - HOD

Module 3 -Teacher

* Module 4 - Student
  + 1. **Integration Testing:**

In Integration Testing, we are required to integrate the testing units from the unit-testing phase and then test then as a whole unit. The testing to be done is Unit 1 , Unit 2 ,Unit 3 and Unit 4 together after integration.

* + 1. **System Testing:**

In System testing we need to install the application on the server machine and then access it from the other machines connected via intranet.

* + 1. **Features to be Tested:**

All the functional features specified in the documents of requirements will be tested.

Functional features include the adding of user, updating the user profile, deleting any user, viewing total number of students, total number of teachers and total number of Heads of Departments, updating department details, updating faculty, approving requests, adding courses and managing them. As software is a web application, system configuration, for server and client, plays an important role. It should therefore undergo thorough testing along with the requirements of the application, so that the generality of the application is maintained. Not limited to this, even the platform on which the website will be hosted needs to be checked in accordance with the minimum requirements of this web application.

* 1. **Approach for Testing:**

As there are layers upon layers of modules in the system, the development of the system is to be carried out in a bottom-up fashion. This will facilitate integration testing and eventually testing of the whole system. For unit testing, structural testing would be done on the basis of branch coverage. System testing will be largely functional in nature. The focus will be on invalid and valid cases, boundary values, and special cases. Any special observation during the test will be noted as well as the reasons for that.

* 1. **Test Deliveries:**

Contrary of the testing document really needed besides test plan are as follows:

* Unit tests of tests;
* Information regarding integration tests; and
* Reports on system testing.

The unit tests, as well as information about integration tests, will be reviewed before system testing commences.

* 1. **Schedules for Testing:**

The entire Testing will start from 19th Nov and will be finished within 1st week of Dec. The schedule is consistent with overall schedule of the whole project

1. **IMPLEMENTATION DETAILS**
   1. **Pseudo Codes:**

Following are the pseudo codes of some of the important algorithms used in developing the application:

**5.1.1 Pseudo Code for Administrator login**

Begin

Admin\_login ()

{

getVal(login\_id);

getVal(password);

//Press login button to login successfully

If( the login\_ID or password entered is wrong) then

Display(“Login unsuccessful Try again”);

Else

//Redirect to home\_page

}

End

**5.1.2 Pseudo Code for HOD Login**

Begin

HOD\_login ()

{

getVal(login\_id);

getVal(password);

//Press login button to login successfully

If( the login\_ID or password entered is wrong) then

Display(“Login unsuccessful Try again”);

Else

//Redirect to home\_page

}

End

**5.1.3 Pseudo Code for Teacher Login**

Begin

Teacher\_login ()

{

getVal(login\_id);

getVal(password);

//Press login button to login successfully

If( the login\_ID or password entered is wrong) then

Display(“Login unsuccessful Try again”);

Else

//Redirect to home\_page

}

End

**5.1.4 Pseudo Code for Student Login**

Begin

Student\_login ()

{

getVal(login\_id);

getVal(password);

//Press login button to login successfully

If( the login\_ID or password entered is wrong) then

Display(“Login unsuccessful Try again”);

Else

//Redirect to home\_page

}

End

**5.2.1 Algorithm for Adding User in the Admin Model**

* Show a form to the admin with fields for username ,email, password and role.
* Admin enters data into the form fields.
* The admin submits the form, which sends the data to the server using the POST method.
* The server calls a service method to save the user data to the database.
* The service interacts with the repository to save the user data to the database.
* After successfully saving the user, redirect the user to a confirmation page or list of users.

**5.2.2 Algorithm for Updating User in the Admin Model**

* Show a form to the admin with fields for username ,email, password and role.
* Admin updates data into the form fields.
* The admin submits the form, which sends the data to the server using the POST method.
* The server calls a service method to save the user data to the database.
* The service interacts with the repository to save the user data to the database.
* After successfully saving the user, redirect the user to a confirmation page or list of users.

**5.2.3 Algorithm for Deleting User in the Admin Model**

* Show a dropdown to the user with a list of available users to select from.
* User selects a user from the dropdown menu.
* The form is submitted to admin/deleteuser.
* The server receives the submitted form with the selected user’s userId.
* The server deletes the user with the selected userId from the database.

After performing the appropriate action ,the server redirects the user to the respective page or shows a success message

1. **RESULT**

**A screenshot of a computer

Description automatically generated**

**Fig 7.1 Landing Page**

**A sign in form with text and numbers

Description automatically generated**

**Fig 7.2: Login Page**

**A sign in form with text and numbers

Description automatically generated**

**Fig 7.3: Error Handling for Invalid Credentials**

**A screen shot of a login form

Description automatically generated**

**Fig 7.4: Logout Confirmation Message**

**A screenshot of a computer

Description automatically generated**

**Fig 7.5: Admin’s Home Page**

**A screenshot of a computer

Description automatically generated**

**Fig 7.6: Admin Interaction: Managing Student Options via Sidebar**

**A screenshot of a computer

Description automatically generated**

**Fig 7.7: Admin Interaction: Managing Teacher Options via Sidebar**

**A screenshot of a computer

Description automatically generated**

**Fig 7.8: Admin Interaction: Managing HOD Options via Sidebar**

**A screenshot of a computer

Description automatically generated**

**Fig 7.9: Admin Interaction: Adding or Updating Users**

**A green card with white text

Description automatically generated**

**Fig 7.10: Admin Interaction: Deleting Users**

**A screenshot of a computer

Description automatically generated**

**Fig 7.11: HOD’s Home Page**

**A screenshot of a computer

Description automatically generated**

**Fig 7.12 Add New Course**

**A screenshot of a computer

Description automatically generated**

**Fig 7.13 Manage Courses**

**FA graph of different colored squares

Description automatically generated**

**Fig 7.14: Teacher’s Home Page**

**A screenshot of a computer

Description automatically generated**

**Fig 7.15: File Upload Demonstration**

**A screenshot of a computer dashboard

Description automatically generated**

**Fig 7.16: Student’s Page**

**A screenshot of a computer

Description automatically generated**

**Fig 7.17: File Access Demonstration**

**7. Summary and Conclusion**

**7.1 Summary of Achievement**

During the project, we successfully developed a comprehensive multi-role management system that contained distinct landing and home pages for different user roles. It contains a landing page acting as an entry point for all users, followed by a secure login mechanism, where it redirects users to their specific role-based dashboards according to their credentials. Admin users are provided with an administrative dashboard for system-wide tasks, while HODs have their own interface by which they can execute their own responsibilities at the departmental level. Teachers get a dedicated dashboard by which they can upload and manage resources. Student Home page can be made user-friendly to access the resource efficiently. This robust system demonstrates our ability to design and implement adapted user experiences that can meet many varied roles and privileges.

**7.2 Difficulties Encountered During the Project**

There were a number of challenges we had to overcome during the project that required thoughtful problem-solving and adaptation. The login mechanism with role-based redirection proved complex, especially ensuring proper validation and error handling for "Wrong Credentials." Designing and organizing role-specific dashboards while maintaining consistency and usability for different user roles demanded careful planning. It included the management of access control and permissions over resources so that users were only able to view or perform operations related to their roles. Other tasks involved the inclusion of resource upload functionality for teachers and providing students with smooth access by handling multiple file types and optimizing the system in terms of performance. There were more difficulties in debugging and testing across multiple roles to ensure a smooth and error-free user experience. But in the end, these challenges enhanced our problem-solving and technical skills.

**7.3 Limitations of the Project**

The project is functional and robust but has some limitations. The role-based access system relies heavily on predefined user credentials, which may become challenging to manage in larger systems without additional automation.  The user interface, no matter how consistent it may look, might need more tailoring to make it suitable for users with diverse skill levels in technology. Finally, the project lacks advanced features like real-time collaboration, notifications, or analytics that could further enrich the user experience.

**7.4 Future Scope of the Project**

The project has huge potential for future developments and scalability. Features like real-time notifications could be added in order to enhance user engagement as well as system monitoring. Integration with third-party tools, such as learning management systems or cloud storage, would further extend the project's utility and adaptability for modern educational

**7.5 Conclusion**

The Campus Connect project aims to improve academic management by developing a centralized platform that enhances communication, stores sharable academic resources and improves course management. The primary challenge was the lack of a unified system for managing academic and administrative interactions. This was addressed by implementing a centralized solution that simplifies access to essential tools and resources. The project comprises several modules i.e., Admin, Teacher, HOD, and Student and has functionalities according to their user roles. In conclusion Campus Connect efficiently covers all the scattered modes of sharing the resources under one user friendly platform.

**9.REFERENCES**

[1] .<https://docs.spring.io/spring-framework/reference/index.html> - SpringBoot

[2].<https://docs.spring.io/spring-framework/reference/data-access/orm/hibernate.html> - HibernateJPA

[3].<https://www.thymeleaf.org/documentation.html> - Thymeleaf

[4].<https://getbootstrap.com/> - Bootstrap

[5].<http://www.thymeleaf.org/extras/spring-security> - Thymeleaf Spring Secuirty

[6].<https://dev.mysql.com/doc/> - MySQL

[8].<https://maven.apache.org/guides/index.html–> - Maven Apache

[9].<https://docs.oracle.com/en/java/javase/17/> - JAVA