

Darshan University

A Project Report on

**“College Management System”**

Under the subject

**Software Engineering (2301CS405)**

B. Tech, Semester – IV

Computer Science & Engineering Department

|  |  |
| --- | --- |
| Submitted By | |
| Student Name: Harshil Solanki | Enrollment No : 23010101256 |
| Academic Year  (2024-2025) | |
| Internal Guide  Prof. R. B. Gondaliya  Darshan University | Dean-DIET  Dr. Gopi Sanghani  Darshan University |

|  |  |
| --- | --- |
|  | **Computer Science & Engineering Department**  **Darshan University** |

**DECLARATION**

We hereby declare that the SRS, submitted along with the **Software Engineering** **(2301CS405)** for entitled **“College Management System”** submitted in partial fulfilment for the Semester-5 of **Bachelor Technology (B. Tech)** in **Computer Science and Engineering (CSE)** Departmentto Darshan University, Rajkot, is a record of the work carried out at **Darshan University, Rajkot** under the supervision of R. B. Gondaliya and that no part of any of report has been directly copied from any students’ reports, without providing due reference.

Harshil Solanki

(Student’s Signature)

Date: \_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
|  | **Computer Science & Engineering Department**  **Darshan University** |

**CERTIFICATE**

This is to certify that the SRS on **“College Management System” has** been satisfactorily prepared by **Solanki Harshil** (**23010101256**) under my guidance in the fulfillment of the course **Software Engineering (2301CS405)** work during the academic year 2024-2025.

|  |  |  |
| --- | --- | --- |
| Internal Guide  Prof. R. B. Gondaliya  Darshan University |  | Dean-DIET  Dr. Gopi Sanghani  Darshan University |

**Acknowledgement**

I wish to express my sincere gratitude to my project guide Prof. R. B. Gondaliyaand all the faculty members for helping me through my project by giving me the necessary suggestions and advices along with their valuable co- ordination in completing this work.

I also thank my parents, friends and all the members of the family for their precious support and encouragement which they had provided in completion of my work. In addition to that, I would also like to mention the Darshan University personals who gave me the permission to use and experience the valuable resources required for the project from the University premises.

Thus, in conclusion to the above said, I once again thank the faculties and members of **Darshan University** for their valuable support in completion of the project.

Thanking You

**Solanki Harshil**

**ABSTRACT**

The College Management System is designed to streamline and automate various administrative and academic processes within an educational institution. It aims to enhance efficiency, accuracy, and accessibility of information for students, faculty, and administrative staff. This system offers a comprehensive solution for managing student admissions, course registrations, examination schedules, grading, and feedback. It also facilitates faculty management, including assigning roles, tracking performance, and updating course materials. For administrative staff, the system provides tools for managing finances, generating reports, and maintaining secure records. By integrating these functions into a single platform, the College Management System reduces manual workload, minimizes errors, and improves overall communication and collaboration within the institution.

This abstract provides a clear and concise overview of the system, highlighting its main features and benefits. If you need further details or adjustments, feel free to let me know!

**Table of Contents**

–

[List of Figures I](#_Toc185960543)

[List of Tables II](#_Toc185960544)

[1 Introduction 1](#_Toc185960545)

[1.1 Product perspective 1](#_Toc185960546)

[1.2 Product features 1](#_Toc185960547)

[1.2.1 There are three different users who will be using this product: 1](#_Toc185960548)

[1.2.2 The features that are required for the Admin are: 1](#_Toc185960549)

[1.2.3 The features that are required for the Student are: 1](#_Toc185960550)

[1.2.4 The features that are required for the Faculty are: 1](#_Toc185960551)

[1.3 Functional Requirement 2](#_Toc185960552)

[1.3.1 Admin 2](#_Toc185960553)

[1.3.2 Student 2](#_Toc185960554)

[1.3.3 Faculty 2](#_Toc185960555)

[1.4 Non-Functional Requirement 2](#_Toc185960556)

[1.4.1 Usability: 2](#_Toc185960557)

[1.4.2 Accuracy: 2](#_Toc185960558)

[1.4.3 Availability: 2](#_Toc185960559)

[1.4.4 Maintainability: 2](#_Toc185960560)

[2 Design and Implementation Constraints 3](#_Toc185960561)

[2.1 Use case diagram 3](#_Toc185960562)

[2.2 Activity diagram and Swimlane diagram 4](#_Toc185960563)

[2.3 Sequence diagram 7](#_Toc185960564)

[2.4 State diagram 8](#_Toc185960565)

[2.5 Class diagram 9](#_Toc185960566)

[2.6 Data flow diagram 10](#_Toc185960567)

[2.6.1 Context diagram (level-0) 10](#_Toc185960568)

[2.6.2 DFD Level-1 10](#_Toc185960569)

[2.6.3 DFD Level-2 11](#_Toc185960570)

[3 External interface requirement (Screens) 12](#_Toc185960571)

[3.1 Screen-1: Registration Form 12](#_Toc185960572)

[3.2 Screen-2: Login Form 13](#_Toc185960573)

[3.3 Screen-3: Add borrower book 14](#_Toc185960574)

[4 Database design 15](#_Toc185960575)

[4.1 List of Tables 15](#_Toc185960576)

[5 Stories and Scenario 16](#_Toc185960577)

[5.1 Story-1: Add New Book in Library Catalogue 16](#_Toc185960578)

[5.1.1 Scenario# S1.1 16](#_Toc185960579)

[5.1.2 Scenario# S1.2 16](#_Toc185960580)

[5.1.3 Scenario# S1.3 16](#_Toc185960581)

[5.2 Story-2: Search Book 17](#_Toc185960582)

[5.3 Story-3: Manage due date for borrowed book 17](#_Toc185960583)

[5.4 Story-4: Renew book 17](#_Toc185960584)

[5.5 Story-5: Generate a report on book usage and availability 17](#_Toc185960585)

[6 Test cases 18](#_Toc185960586)

[7 References 22](#_Toc185960587)

# List of Figures

[Figure 2.1‑1 Use case diagram for College Management System 3](#_Toc185796388)

[Figure 2.2‑1 Activity diagram for Next Semester Registration 4](#_Toc185796389)

[Figure 2.2‑2 Activity diagram for Examination 5](#_Toc185796390)

[Figure 2.2‑3 Swimlane diagram for Next Semester Registration 6](#_Toc185796391)

[Figure 2.3‑1 Sequence diagram for Book Issue 7](#_Toc185796392)

[Figure 2.4‑1 State diagram of Book 8](#_Toc185796393)

[Figure 2.4‑2 State diagram for Librarian 8](#_Toc185796394)

[Figure 2.5‑1 Class diagram for Library management system 9](#_Toc185796395)

[Figure 2.6‑1 Context diagram for Library management system 10](#_Toc185796396)

[Figure 2.6‑2 DFD level-1 for Library management system 10](#_Toc185796397)

[Figure 2.6‑3 DFD level-2 for Issue book 11](#_Toc185796398)

[Figure 3.1‑1 Screen-1: Registration Form 12](#_Toc185796399)

[Figure 3.2‑1 Screen-2: Login Form 13](#_Toc185796400)

[Figure 3.3‑1 Screen-3: Add borrower book 14](#_Toc185796401)

# List of Tables

[Table 3.1‑1 Screen element of Registration form 12](#_Toc146706861)

[Table 3.2‑1 Screen element of Login form 13](#_Toc146706862)

[Table 3.3‑1 Screen element of Add borrower book 14](#_Toc146706863)

[Table 4.1‑1 Table: Book 15](#_Toc146706864)

[Table 4.1‑2 Table: Borrower 15](#_Toc146706865)

[Table 4.1‑3 Table: Staff 15](#_Toc146706866)

[Table 4.1‑4 Table: Student 15](#_Toc146706867)

# Introduction

## Product perspective

This project transforms the college management system into an online platform, making it easy to manage various tasks. Admins can handle branches, students, and faculty data, schedule exams, and manage attendance. Students can view grades, update profiles, check class schedules, upload assignments, give feedback, download study materials, and pay fees online. Faculty members can view records, upload content, check salaries, manage assignments, give grades, and respond to feedback. This system simplifies administrative and academic tasks, making everything more efficient and accessible.

## Product features

### There are three different users who will be using this product:

* Admin who will be handle the software.
* Student Accesses academic resources and personal profiles.
* Faculty Manages course content and student assignments.

### The features that are required for the Admin are:

* New branch will be entered in database
* Any changes in branch should be updated in database.
* If University wants to remove branch for their upcoming year then removed from system
* New student data can be add in the database
* if New Faculty join then faculty data can be add in the database
* Admina can schedule exams
* Admin can Upload the feedback for Faculty, course and teaching assistance
* Admin is able to view the Attendance of Students and Staff
* Can Change the Attendance of any Student and Staff
* Assign courses to faculty members

### The features that are required for the Student are:

* can see their grade of the exam
* can view their profile
* can update their profile
* can see their class for the upcoming lectures
* can upload assignment given by faculty
* can give feedback on Faculty, Teaching assistance and Course
* can download the study material uploaded by the faculty
* can Pay their fee using upi or netbanking.

### The features that are required for the Faculty are:

* can view their records
* can upload subject related content so that student can have benefit of it
* can check salary of till the join date
* login in their account
* can view the assignment upload by the student
* can give grade on student assignment
* can upload ppt and pdf
* can respond to feedback if necessary
* can delete their profiles

## Functional Requirement

### Admin

* Add Branch: New branch will be entered in database
* Update Branch: Any changes in branch should be updated in database.
* Remove Branch: If University wants to remove branch for their upcoming year then removed from system
* Add Student : New student data can be add in the database
* Add Faculty : if New Faculty join then faculty data can be add in the database
* Schedule Exams: Admina can schedule exams
* Upload Feedback: Admin can Upload the feedback for Faculty, course and teaching assistance
* View Attendance : Admin is able to view the Attendance of Students and Staff
* Update Attendance: Can Change the Attendance of any Student and Staff
* Course Assignment: Assign courses to faculty members

### Student

* View Grade: Student can see their grade of the exam
* View Student Profile : Student Can view their profile
* Update Profile: Student can update their profile
* View Class : Student can see their class for the upcoming lectures
* Assignment Upload : Student can upload assignment given by faculty
* Give Feedback : Student can give feedback on Faculty, Teaching assistance and Course
* Download Contant : Student can download the study material uploaded by the faculty
* Fee Payment : Student can Pay their fee using upi or netbanking

### Faculty

* Faculty Record : Faculty can view their records
* Upload Video : Faculty can upload subject related content so that student can have benefit of it
* Check Salary : Faculty can check salary of till the join date
* Login : login in their account
* Assignment view : Faculty can view the assignment upload by the student
* Give Grade : Faculty can give grade on student assignment
* Upload study material : Faculty can upload ppt and pdf
* Response to Feedback : Faculty and admin can respond to feedback if necessary
* Profile Deletion: Faculty members can delete their profile

## Non-Functional Requirement

### Usability:

* The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.

### Accuracy:

* The data stored about the books and the fines calculated should be correct, consistent, and reliable.

### Availability:

* The System should be available for the duration when the library operates and must be recovered within an hour or less if it fails. The system should respond to the requests within two seconds or less.

### Maintainability:

* The software should be easily maintainable and adding new features and making changes to the software must be as simple as possible. In addition to this, the software must also be portable.

# Design and Implementation Constraints

## Use case diagram

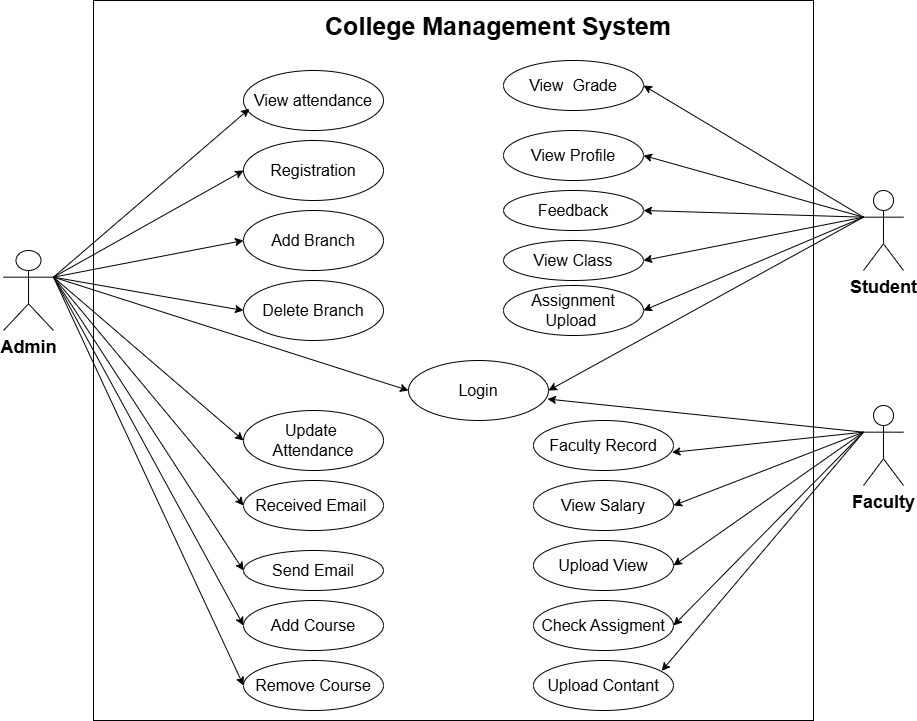


Figure 2.1‑1 Use case diagram for College Management System

## Activity diagram and Swimlane diagram

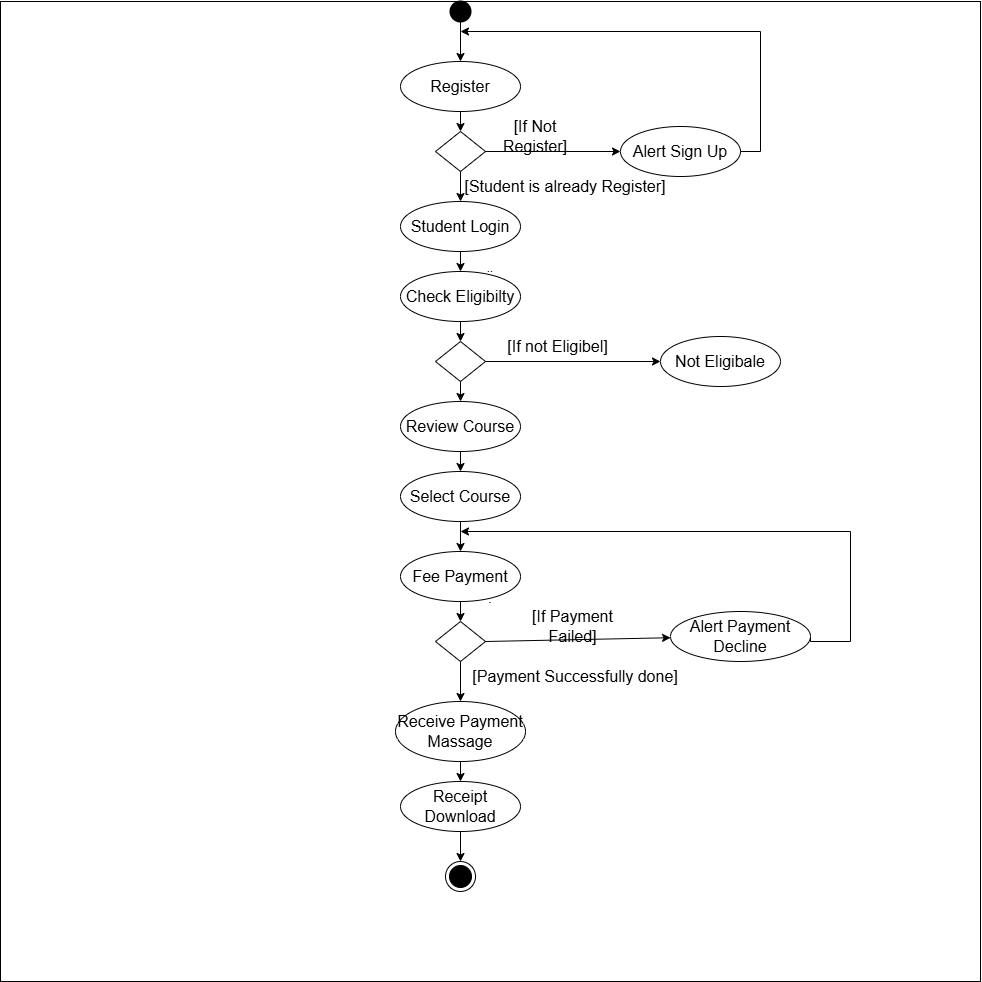


Figure 2.2‑1 Activity diagram for Next Semester Registration

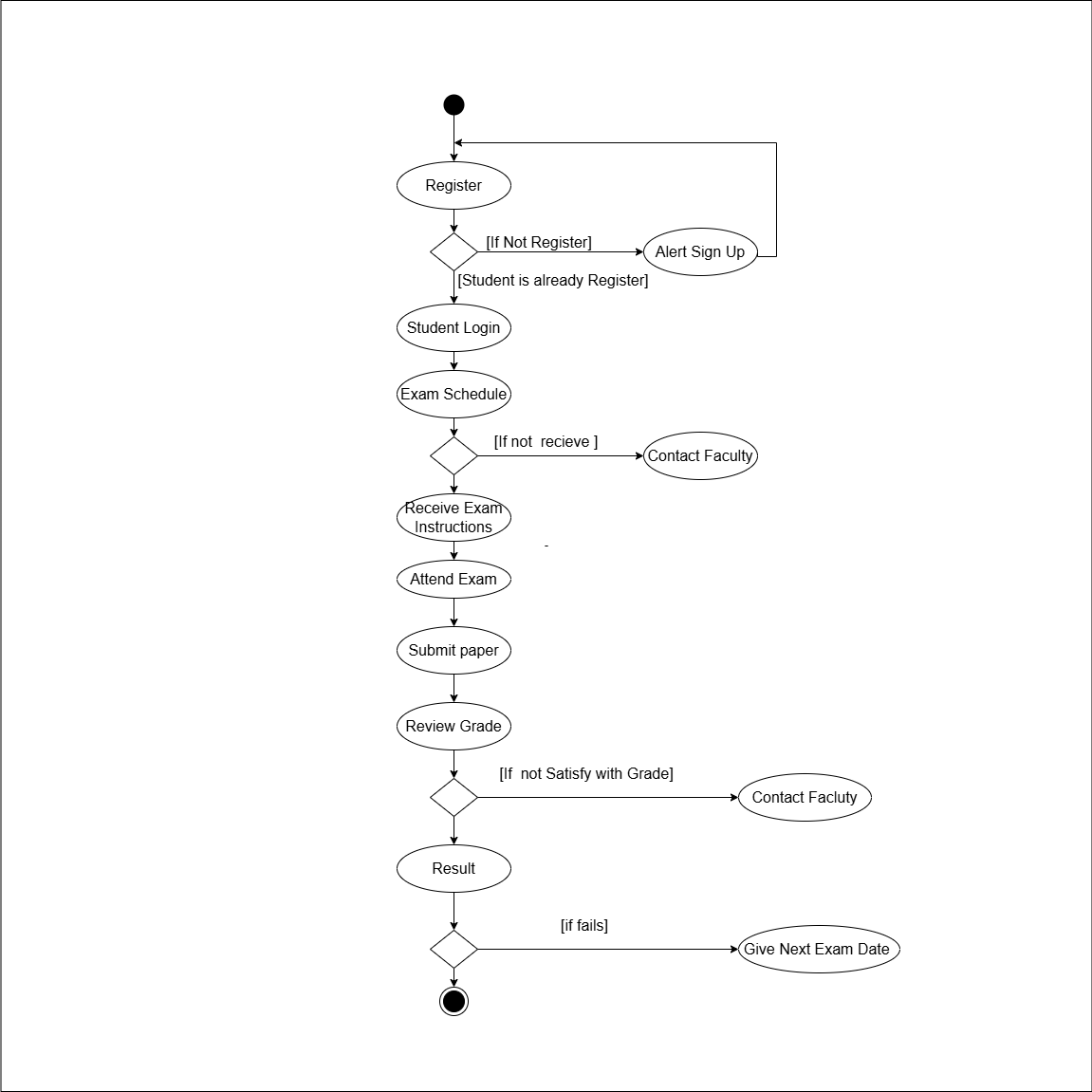


Figure 2.2‑2 Activity diagram for Examination

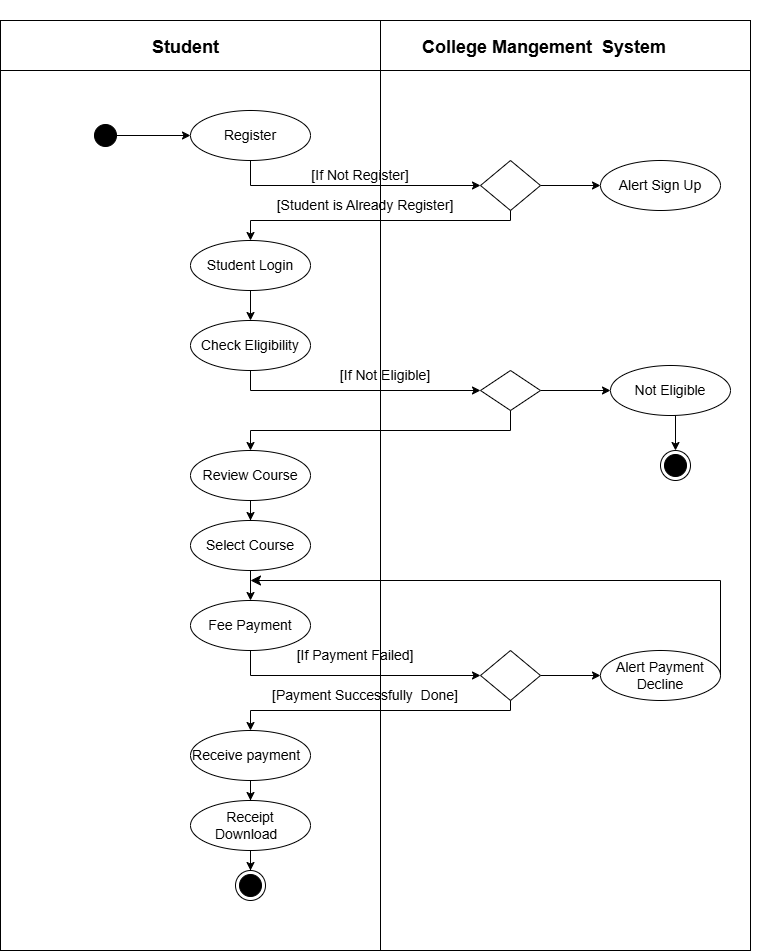
****

Figure 2.2‑3 Swimlane diagram for Next Semester Registration

## Sequence diagram



Figure 2.3‑1 Sequence diagram for Book Issue

## State diagram



Figure 2.4‑1 State diagram of Book



Figure 2.4‑2 State diagram for Librarian

## Class diagram



Figure 2.5‑1 Class diagram for Library management system

## Data flow diagram

### Context diagram (level-0)

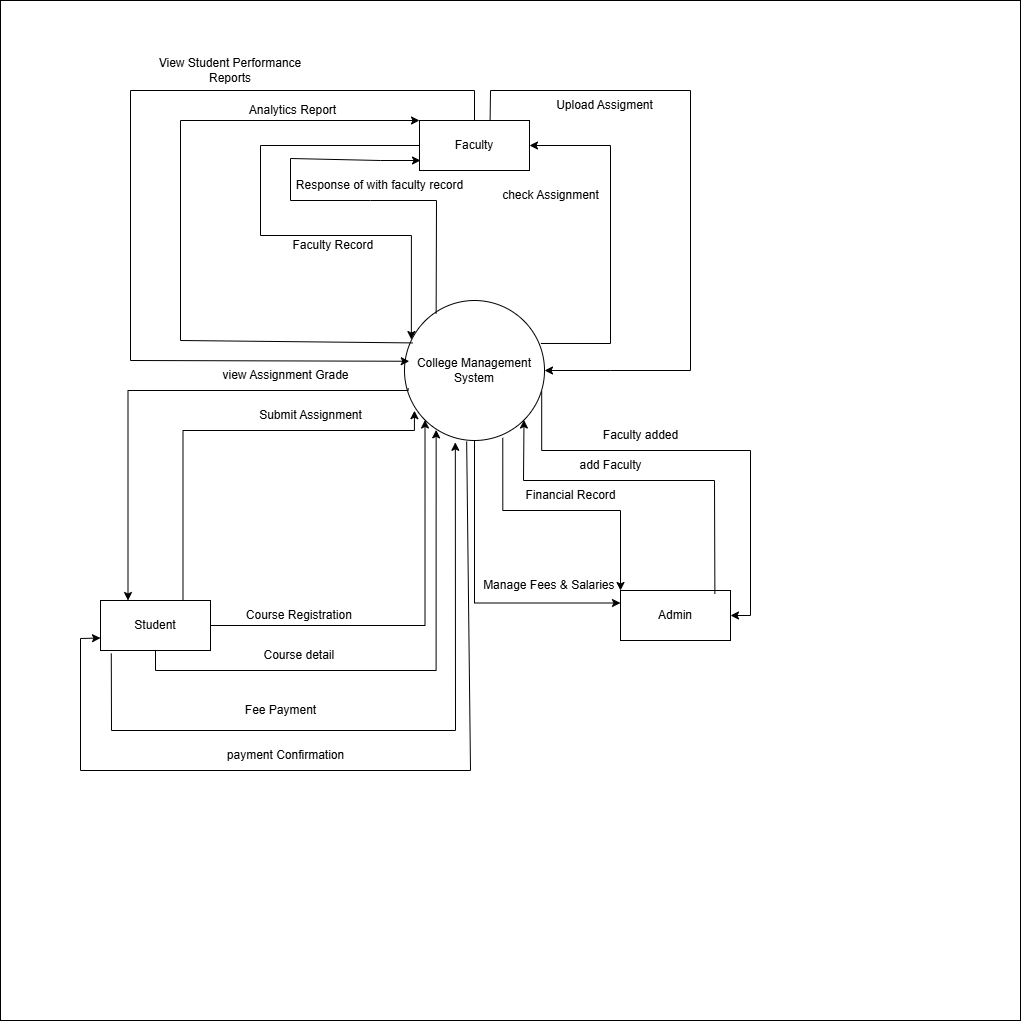


Figure 2.6‑1 Context diagram for College management system

### DFD Level-1

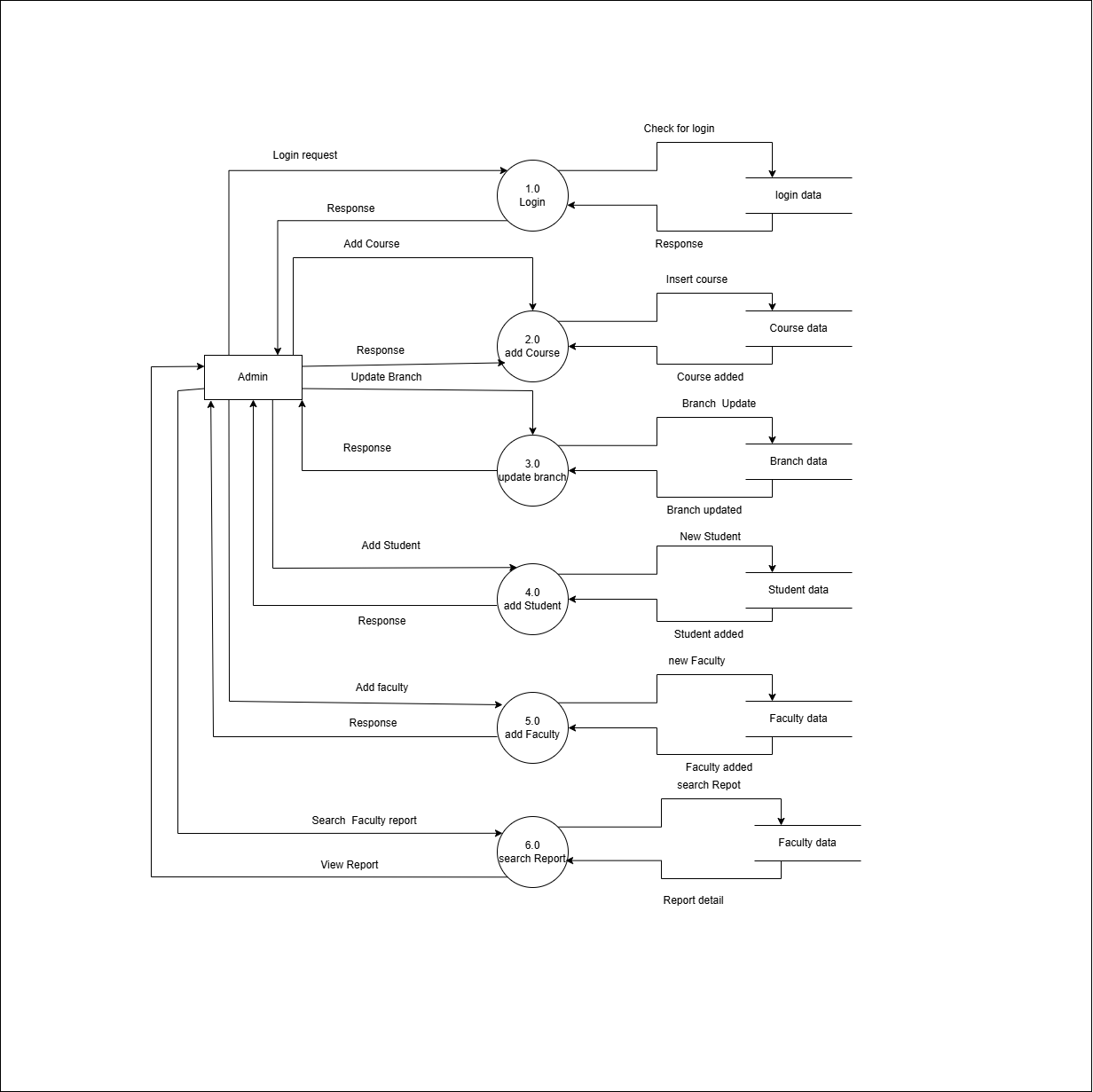


Figure 2.6‑2 DFD level-1 for College management system

### DFD Level-2

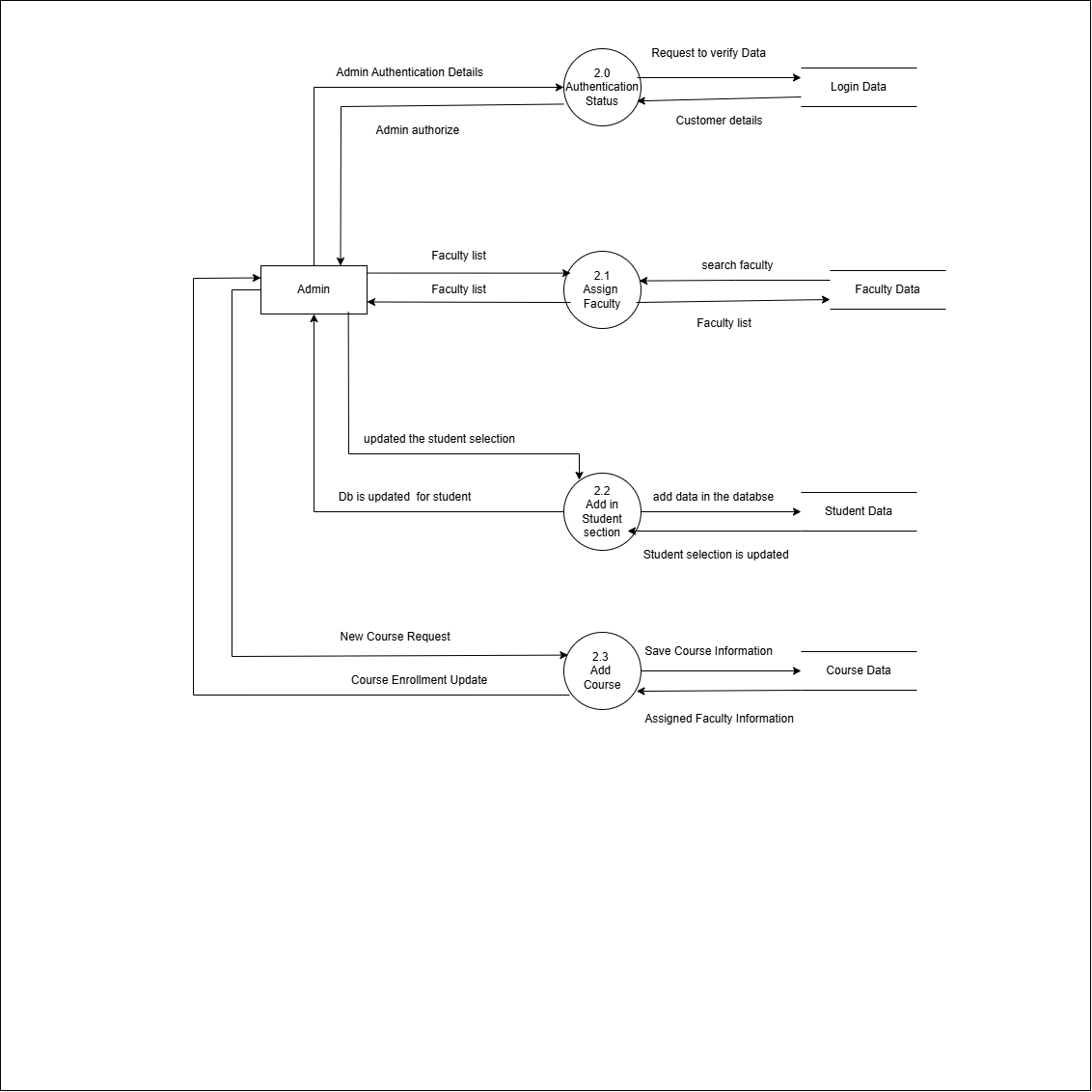


Figure 2.6‑3 DFD level-2 for Add Course

# External interface requirement (Screens)

## Screen-1: Registration Form

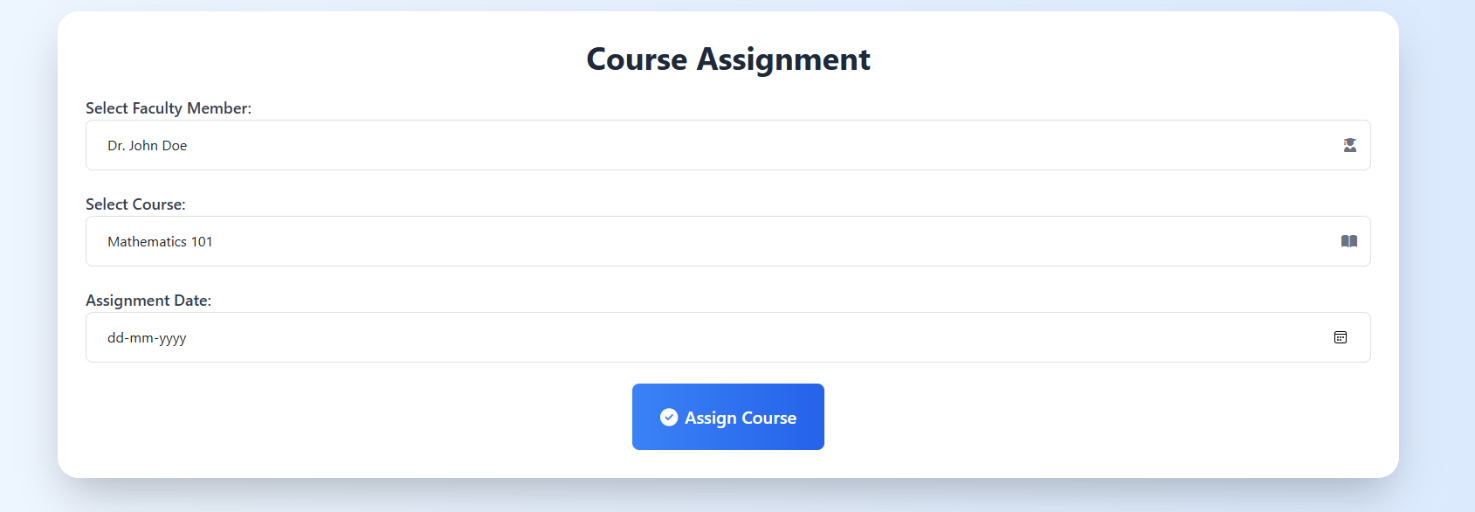


Figure 3.1‑1 Screen-1: Course Assignment

**Purpose:** This module allows faculty to create, assign, and manage course assignments for students. It enables students to view and submit assignments while providing faculty with tools to evaluate and provide feedback, ensuring a smooth assignment management process.

Table 3.1‑1 Screen element of Course Assignment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Select Faculty Member | Dropdown | M | 1 | Allows selection of a faculty member from the available list. |
| 2 | Select Course | Dropdown | M | 1 | Allows selection of a course to assign to the faculty member. |
| 3 | Assignment Date | Date Picker | M | 1 | Allows selection of the date for course assignment. |
| 4 | Assign Course | Button | --- | --- | Submits the course assignment details for storage in the database. |

## Screen-2: Login Form

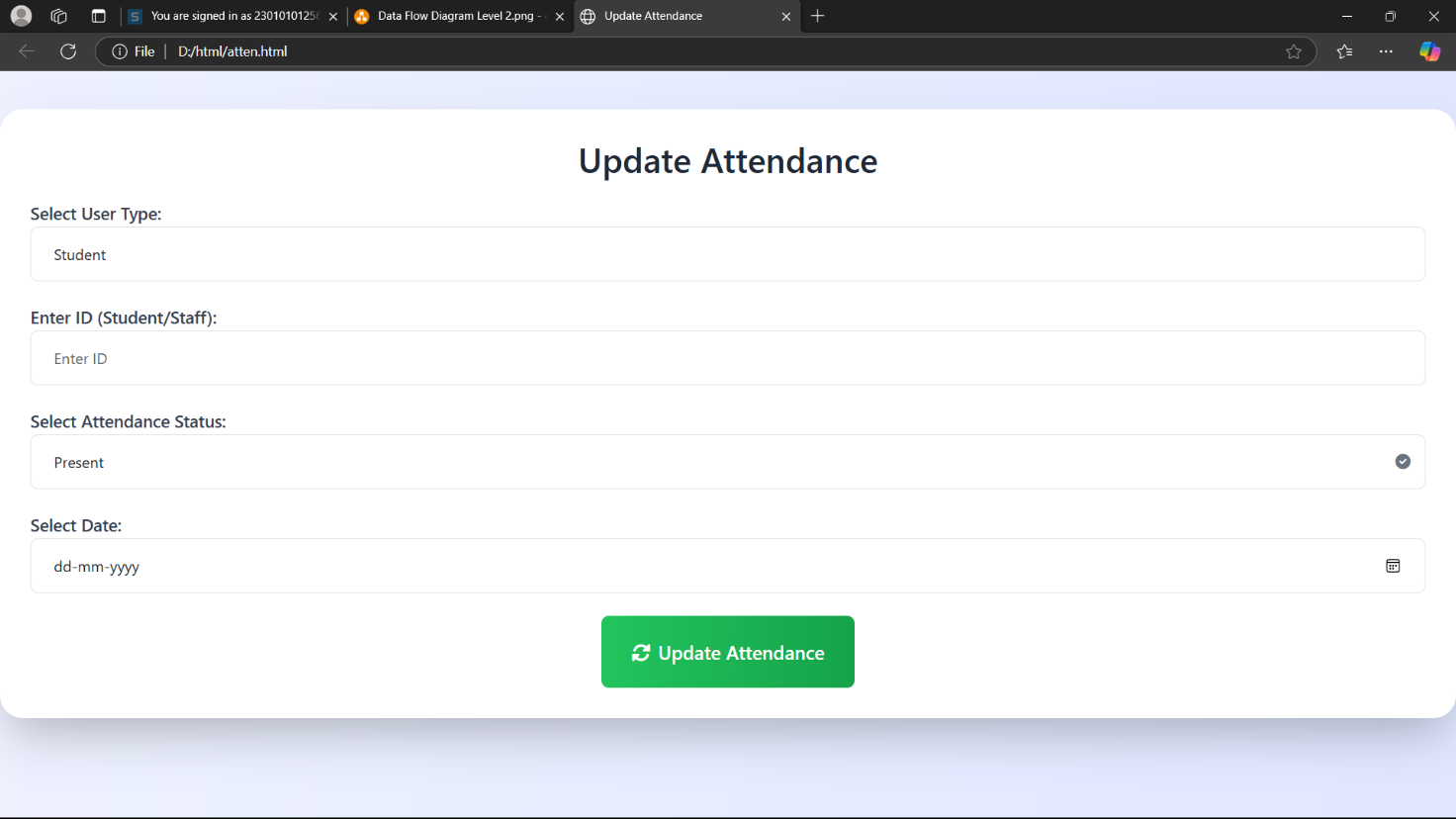


Figure 3.2‑1 Screen-2: Update Attendance

**Purpose:** This module enables administrators and faculty to update and manage student attendance records efficiently. It allows for marking attendance, making corrections, and keeping track of attendance status, ensuring accurate record-keeping and monitoring of student participation.

Table 3.2‑1 Screen element of Update Attendance

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Select User Type | Dropdown | M | 1 | Allows selection of user type (Student/Staff). |
| 2 | Enter ID (Student/Staff) | Textbox | M | 1 | ID field should be editable and accept a valid Student or Staff ID. |
| 3 | Select Attendance Status | Dropdown | M | 1 | Allows selection of attendance status (Present/Absent). |
| 4 | Select Date | Date Picker | M | 1 | Allows selection of the date for attendance update. |
| 5 | Update Attendance | Button | --- | --- | Submits the attendance details for update in the database. |

## Screen-3: Add borrower book

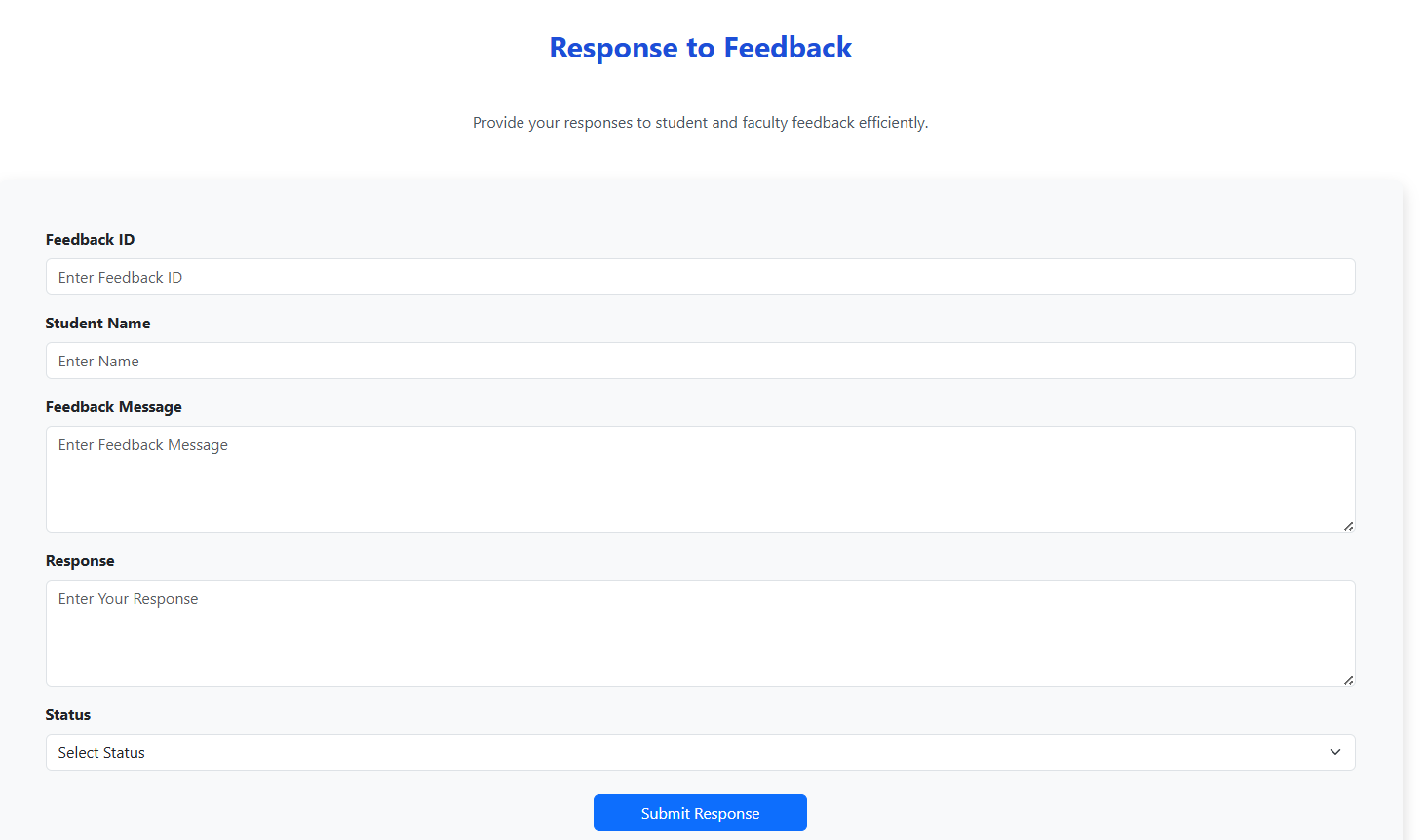


Figure 3.3‑1 Screen-4: Response to Feedback

**Purpose:** This module allows administrators or faculty to efficiently respond to feedback from students and faculty members. It provides a structured way to address concerns, suggestions, and inquiries, ensuring effective communication and resolution of issues.

Table 3.3‑1 Screen element Response to Feedback

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Feedback ID | Textbox | M | 1 | Field to enter or display the feedback ID. |
| 2 | Student Name | Textbox | M | 1 | Field to enter or display the student's name. |
| 3 | Feedback Message | Textarea | M | 1 | Field to enter or display the feedback message. |
| 4 | Response | Textarea | M | 1 | Field to input the response to the feedback. |
| 5 | Status | Dropdown | M | 1 | Allows selection of feedback status (e.g., pending, resolved). |
| 6 | Submit Response | Button | --- | --- | Submits the response and updates the feedback status. |

## Screen-3: Add borrower book

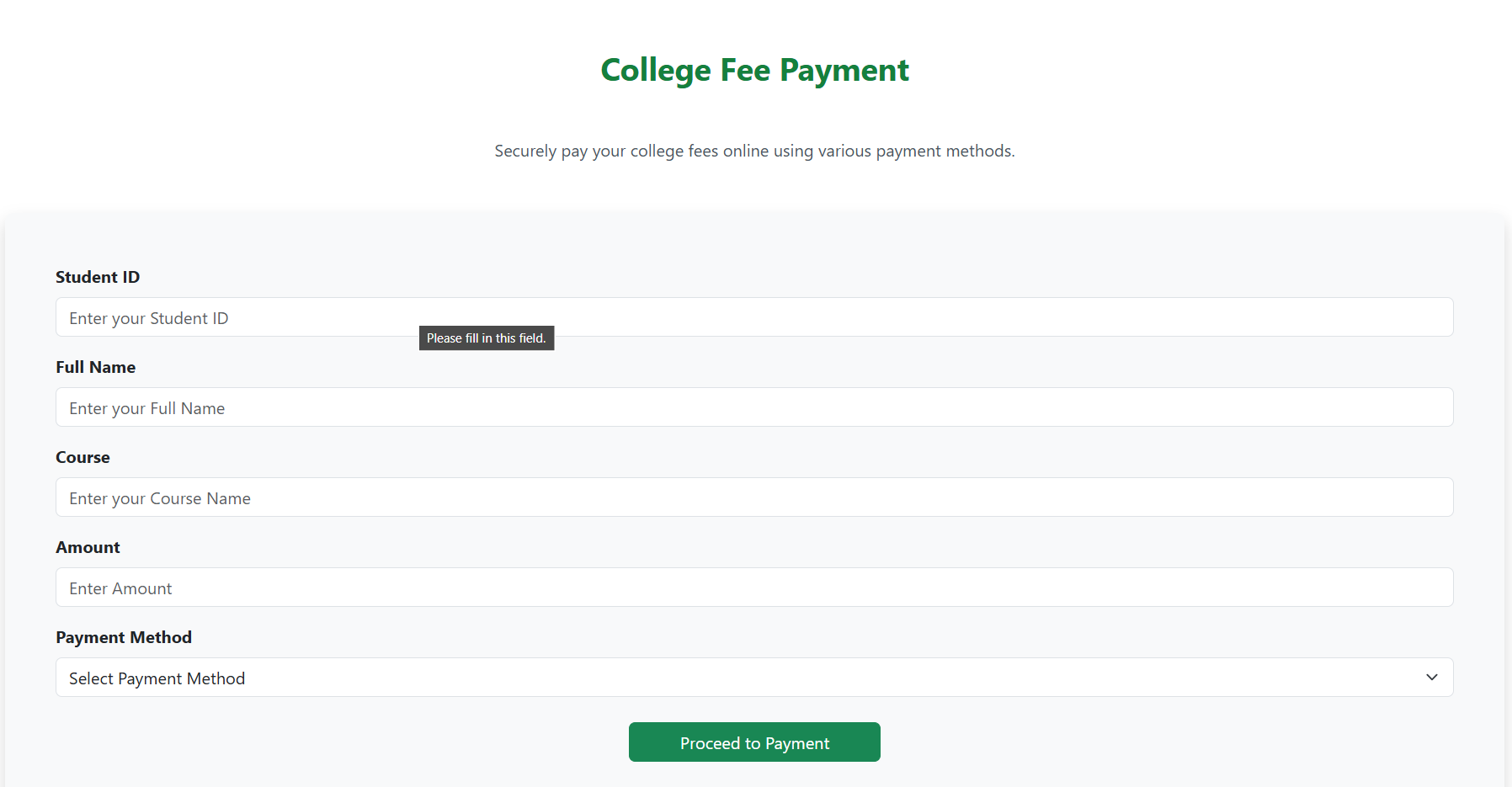


Figure 3.3‑1 Screen-3: College Fee Payment

**Purpose:** This module enables students to securely pay their college fees online using various payment methods. It ensures a convenient, efficient, and transparent fee payment process, reducing manual workload and minimizing errors.

Table 3.3‑1 Screen element of College Fee Payment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Student ID | Textbox | M | 1 | Field to enter the unique Student ID. |
| 2 | full Name | Textbox | M | 1 | Field to enter the full name of the student. |
| 3 | Course | Textarea | M | 1 | Field to enter the course name. |
| 4 | Amount | Textarea | M | 1 | Field to enter the fee amount to be paid. |
| 5 | Payment Method | Dropdown | M | 1 | Dropdown to select the payment method. |
| 6 | Proceed to Payment | Button | --- | --- | Button to proceed with the payment process. |

## Screen-3: Add borrower book

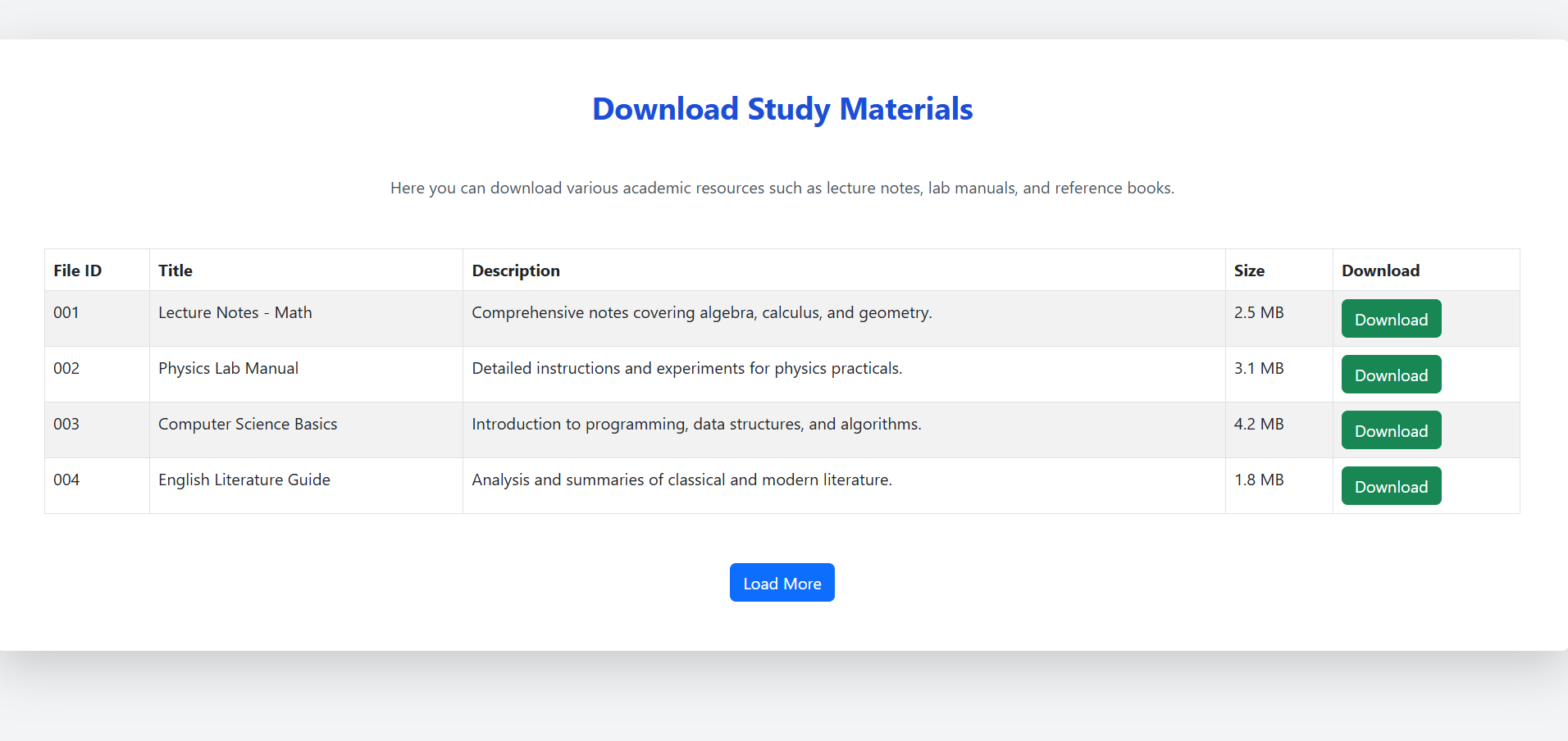


Figure 3.3‑1 Screen-5: Download Study Materials

**Purpose:** This module allows students to download various academic resources such as lecture notes, lab manuals, and reference books uploaded by faculty. It provides an organized way to access study materials efficiently, enhancing the learning experience.

Table 3.3‑1 Screen element of Download Study Materials

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | File ID | Label | M | 1 | Unique identifier for the study material file. |
| 2 | Title | Label | M | 1 | Name of the study material. |
| 3 | Description | Label | M | 1 | Brief details about the content of the study material. |
| 4 | Size | Label | M | 1 | File size of the material in MB. |
| 5 | Download Button | Button | M | 1 | Button to download the corresponding file. |
| 6 | Load More Button | Button | O | 1 | Button to load more study materials if available. |

# Database design

## List of Tables

* Admin
* Course
* Faculty
* Student
* Department

*Table 4.1-1 Table: Admin*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **AdminID** | int | NN | PK (Auto Increment) |  |
| **Name** | varchar(100) | NN |  |  |
| **Email** | varchar(100) | AN |  |  |
| **Phone\_no** | int | NN |  |  |
| **Role** | varchar(50) | AN |  |  |
| **FacultyID** | int | NN | FK | Reference of Faculty table |

*Table 4.1-2 Table: Course*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **CourseID** | int | NN | PK (Auto Increment) |  |
| **CourseCode** | int | NN |  |  |
| **DepartmentId** | int | NN | FK | Reference of Department table |
| **CourseName** | Varchar(100) | NN |  |  |
| **description** | Varchar(1000) | AN |  |  |

*Table 4.1-3 Table:* Faculty

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **FacultyID** | int | NN | PK (Auto Increment) |  |
| **FacultyName** | varchar(100) | NN |  |  |
| **Email** | varchar(100) | AN |  |  |
| **Office** | Varchar(100) | AN |  |  |

*Table 4.1-4 Table: Student*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **StudentID** | int | NN | PK (Auto Increment) |  |
| **StudentName** | varchar(100) | AN |  |  |
| **Gender** | varchar(100) | AN |  |  |
| **DOB** | DateTime | AN |  |  |
| **Department** | varchar(100) | AN |  |  |
| **Contact** | number(10,0) | AN |  |  |
| **DepartmentID** | int | NN | FK | Reference of Department table |

*Table 4.1-5Table: Department*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Data Type** | **Null** | **Keys & Constrains** | **Default Value & Description** |
| **DepartmentID** | int | NN | PK (Auto Increment) |  |
| **DepartmentName** | varchar(100) | AN |  |  |
| **HeadOfDepartment** | varchar(100) | NN | FK | Reference of the Faculty table |

# Stories and Scenario

## Story-1: ****View Exam Grades and Performance Tracking****

|  |  |  |
| --- | --- | --- |
| *Story #* ***S1*** | **:** | As a **Student**, I want to **view my grade for the exam** So that **I can keep track of my performance**. |
| **Priority** | **:** | Medium |
| **Estimate** | **:** | L |
| **Reason** | **:** | Students need to easily access their grades for exam performance tracking. |

### Scenario# S1.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.1*** | **:** | Viewing Exam Grade |
| **Prerequisite** | **:** | The student is logged into the student portal of the system. |
| **Acceptance**  **Criteria** | **:** | **Given**: The student is on the student dashboard and has completed at least one exam.  **When**: The student selects the "View Grades" option from the dashboard. The system retrieves and displays the student’s exam grades.  **Then**: The system successfully displays the student’s exam grades along with relevant details, such as the exam name, grade, and any feedback from the instructor. The student is able to see their performance in a clear and understandable format. |

### Scenario# S1.2

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.2*** | **:** | **Viewing Exam Grade with Error in Data** |
| **Prerequisite** | **:** | : The student is logged into the student portal of the system. |
| **Acceptance**  **Criteria** | **:** | **Given**: The student is on the student dashboard and their exam grade has been entered into the system, but an error occurred while processing the data  **When**: The student selects the "View Grades" option from the dashboard. The system attempts to retrieve and display the student's exam grade but encounters a data issue.  **Then**: The system displays a message indicating that there was an error retrieving the exam grade. The student is informed to try again later or contact support for assistance. |

### Scenario# S1.3

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.3*** | **:** | **Viewing Exam Grade After Re-evaluation** |
| **Prerequisite** | **:** | The student is logged into the student portal of the system, and their exam has been re-evaluated. |
| **Acceptance**  **Criteria** | **:** | **Given**: The student is on the student dashboard, and their exam grade has been re-evaluated after an initial grading.  **When**: The student selects the "View Grades" option from the dashboard. The system retrieves and displays the updated exam grade and any associated feedback from the instructor.  **Then**: The system successfully displays the updated exam grade along with the re-evaluation details, including any feedback or changes from the instructor. |

## Story-2: View Faculty Member's Professional Records

|  |  |  |
| --- | --- | --- |
| *Story #* ***S2*** | : | As a **Faculty Member**, I want to **view my records** So that **I can keep track of my professional details**. **Priority**: High |
| **Priority** | **:** | High |
| **Estimate** | **:** | M |
| **Reason** | **:** | Faculty members need to have easy access to their records for verification and updating purposes. |

### Scenario# S1.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.1*** | **:** | Viewing Faculty Member's Professional Records |
| **Prerequisite** | **:** | The faculty member is logged into the faculty portal of the system. |
| **Acceptance**  **Criteria** | **:** | **Given**: The faculty member is on the faculty dashboard, but some of their professional records are incomplete or missing.  **When**: The faculty member selects the "View Professional Records" option from the dashboard. The system attempts to retrieve the faculty member’s professional records but encounters missing or incomplete data.  **Then**: The system displays a message indicating that certain records are incomplete or missing. The faculty member is prompted to update their records or contact the relevant department to complete the information. |

### Scenario# S1.2

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.2*** | **:** | **Viewing Faculty Member's Professional Records After Update** |
| **Prerequisite** | **:** | : The faculty member is logged into the faculty portal of the system, and their professional records have been updated . |
| **Acceptance**  **Criteria** | **:** | **Given**: The faculty member is on the faculty dashboard, and their professional records have been updated since the last view.  **When**: The faculty member selects the "View Professional Records" option from the dashboard. The system retrieves and displays the updated professional records and details.  **Then**: The system successfully displays the updated professional records, including new information such as recent publications, updated qualifications, or job responsibilities. The faculty member is able to review the most recent version of their professional details. |

### Scenario# S1.3

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.3*** | **:** | **Viewing Faculty Member's Professional Records with Error in Data** |
| **Prerequisite** | **:** | The faculty member is logged into the faculty portal of the system, and their professional records have been entered into the system, but there is an error . |
| **Acceptance**  **Criteria** | **:** | **Given**: The faculty member is on the faculty dashboard, and their professional records have been stored in the system.  **When**: the faculty member clicks on "View Professional Records," but the system encounters a problem with the data,  **Then**: the system shows an error message and asks the faculty member to contact support or try again later. They are also given an option to report the issue for further investigation. |

## Story-3: ****Manage University Data and Records****

|  |  |  |
| --- | --- | --- |
| *Story #* ***S3*** | **:** | **As an Admin**, I want to **manage the university's data** So that **I can maintain accurate records and ensure smooth operations**. |
| **Priority** | **:** | High |
| **Estimate** | **:** | M |
| **Reason** | **:** | The admin needs to efficiently manage all aspects of the university data, including branches, students, faculty, exams, attendance, feedback, and assignments. |

### Scenario# S1.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.1*** | **:** | **Adding a New Branch to the Database** |
| **Prerequisite** | **:** | The admin is logged into the university management system. |
| **Acceptance**  **Criteria** | **:** | **Given**: The admin is on the branch management page of the system.  **When**: The admin selects the "Add New Branch" option. The admin enters the required details for the new branch .The admin clicks the "Save" button.  **Then**: The system successfully adds the new branch to the database. The admin receives a confirmation message that the branch has been successfully added. |

### Scenario# S1.2

|  |  |  |
| --- | --- | --- |
| *Scenario#* ***S1.2*** | **:** | **Updating Branch Information in the Database** |
| **Prerequisite** | **:** | The admin is logged into the university management system, and a branch already exists in the database. |
| **Acceptance**  **Criteria** | **:** | **Given**: The admin is on the branch management page, and the details of an existing branch need to be updated.  **When**: The admin selects the branch to be updated. The admin modifies the branch. The admin clicks the "Update" button to save the changes.  **Then**: The system successfully updates the branch information in the database. The admin receives a confirmation message that the branch has been updated. |

# Test cases

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Name:** | **College Management System** | **Test Designed by:** | **P. U. Jadeja** |
| **Module Name:** | **Course Assignment** | **Test Designed date:** | 01-10-2023 |
| **Release Version:** | **1.0** | **Test Executed by:** | **R. B. Gondaliya** |
|  |  | **Test Execution date:** | 15-01-2023 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pre-condition**: **Web application should be accessible** | | | |  |
| **Test Case ID** | **Test Title** | **Test Type** | **Description** | **Test Case ID** |
| **TC\_001** | |  | | --- | |  |  |  | | --- | | Login to Course Assignment web application with valid credential | | Functional | Login to Course Assignment system web application through valid credentials | TC\_001 |
| **TC\_002** | Access the Course Dashboard page after login | Functional | Verify that user can access the Course Dashboard page after login | TC\_002 |
| **TC\_003** | Enroll in a course | Functional | Verify that user can successfully enroll in a course | TC\_003 |

|  |  |
| --- | --- |
| **Test Case Title** | Login to Course Assignment web application with valid credentials |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | Web application should be accessible and the user should have valid credentials (username and password). |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Access Course Assignment web application URL | The site should launch properly | Site launched successfully | Pass |  | |  | | --- | |  |  |  | | --- | | courseassignment.example.com/login | |  |
| 2 | Enter valid Username  in username  field | Username field should be editable and accept the  Username | Username input accepted | Pass |  | Username: Rbgondaliya@ gmail.com |  |
| 3 | Enter valid Password in Password  field | Password field should be editable and accept the password and display as star or dot | Password input displayed in dot and  accepted | pass |  | Password: rbgondaliya |  |
| 4 | Enter valid captcha  code in  captch field | Captcha field should be editable and accept captcha; captcha is case-sensitive | Captcha input accepted | Pass | Step required when human action validation: get captcha from image near the captcha field | get captcha from image which is near by captcha  field |  |
| 5 | Click on  login button | User should log in and be navigated to the dashboa | User navigated to dashboard and username displayed on top right | pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Enroll in a course |
| **Test Type** | Functional |
| **Test Priority** | Medium |
| **Pre-condition** | 1. Web application should be accessible.  2. User should be logged into the Course Assignment system.  3. User should have access to the course enrollment page. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Access Course Enrollment page | Course enrollment page should be accessible | Page loaded successfully | Pass |  | Courseassignment  .example.com/enroll |  |
| 2 | Browse and select a course | User should be able to browse available courses and select one | Course selected successfully | Pass |  | Course: Introduction to Programming |  |
| 3 | Click on "Enroll" button | Enroll button should be visible and clickable | Enroll button clicked successfully | Pass |  |  |  |
| 4 | |  | | --- | |  |  |  | | --- | | Verify enrollment confirmation message | | A confirmation message should appear indicating successful enrollment | Enrollment successful message displayed | Pass |  |  | Bug\_002 |
| 5 | |  | | --- | |  |  |  | | --- | | Check if the enrolled course appears in "My Courses" section | | The enrolled course should appear in the "My Courses" list | Course listed in "My Courses" | Pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Access the Course Dashboard page after login |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | 1. Web application should be accessible.  2. User should be logged into the Course Assignment system.  3. User should be redirected to the dashboard after successful login. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case**  **Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug**  **ID** |
| 1 | |  | | --- | |  |  |  | | --- | | Access Course Assignment web application URL | | The site should launch successfully | Site launched successfully | Pass |  | |  | | --- | |  |  |  | | --- | | courseassignment  .example.com/login | |  |
| 2 | Log in with valid credentials (Username and Password) | User should be able to enter valid credentials and click the login button | Login successful, redirected to dashboard | Pass |  | |  | | --- | |  |  |  | | --- | | Username: validUser@example.com  Password: valid123 | |  |
| 3 | Verify redirection to the Course Dashboard page | User should be redirected to the Course Dashboard page | Redirected to dashboard successfully | Pass |  |  |  |
| 4 | Verify that the dashboard displays the username at the top-right corner | Username should be displayed at the top-right corner of the dashboard | Username displayed at the top-right | Pass |  |  |  |
| 5 | Check if all dashboard components are loaded correctly | All sections and elements on the dashboard should load properly | All dashboard elements loaded correctly | Pass |  |  |  |

# 6.1 Test cases

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Name:** | **College Management System** | **Test Designed by:** | **Harshil Solanki** |
| **Module Name:** | **Fee Payment** | **Test Designed date:** | 20-2-2025 |
| **Release Version:** | **1.0** | **Test Executed by:** | **R. B. Gondaliya** |
|  |  | **Test Execution date:** | 15-01-2023 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pre-condition**: **Web application should be accessible** | | | |  |
| **Test Case ID** | **Test Title** | **Test Type** | **Description** | **Test Case ID** |
| **TC\_001** | |  | | --- | |  |  |  | | --- | | Fee Payment: Student can Pay using UPI | | Functional | Verify that the student can pay the fee using UPI | TC\_001 |
| **TC\_002** | Fee Payment: Student can Pay using Net Banking | Functional | Verify that the student can pay the fee using Net Banking | TC\_002 |
| **TC\_003** | Fee Payment: Verify successful payment confirmation | Functional | Verify that a confirmation message appears after successful payment | TC\_003 |

|  |  |
| --- | --- |
| **Test Case Title** | Student can Pay using UPI |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | Web application should be accessible.  User should be logged into the Course Assignment system.  User should have access to the fee payment page.  User should have a valid UPI ID to make the payment. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Access Course Assignment web application URL | The site should launch properly | Site launched successfully | Pass |  | |  | | --- | |  |  |  | | --- | | courseassignment.example.com/login | |  |
| 2 | Enter valid Username  in username  field | Username field should be editable and accept the  Username | Username input accepted | Pass |  | Username: Rbgondaliya@ gmail.com |  |
| 3 | Enter valid Password in Password  field | Password field should be editable and accept the password and display as star or dot | Password input displayed in dot and  accepted | pass |  | Password: rbgondaliya |  |
| 4 | Enter valid captcha  code in  captch field | Captcha field should be editable and accept captcha; captcha is case-sensitive | Captcha input accepted | Pass | Step required when human action validation: get captcha from image near the captcha field | get captcha from image which is near by captcha  field |  |
| 5 | Click on  login button | User should log in and be navigated to the dashboa | User navigated to dashboard and username displayed on top right | pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Enroll in a course |
| **Test Type** | Functional |
| **Test Priority** | Medium |
| **Pre-condition** | 1. Web application should be accessible.  2. User should be logged into the Course Assignment system.  3. User should have access to the course enrollment page. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Access Course Enrollment page | Course enrollment page should be accessible | Page loaded successfully | Pass |  | Courseassignment  .example.com/enroll |  |
| 2 | Browse and select a course | User should be able to browse available courses and select one | Course selected successfully | Pass |  | Course: Introduction to Programming |  |
| 3 | Click on "Enroll" button | Enroll button should be visible and clickable | Enroll button clicked successfully | Pass |  |  |  |
| 4 | |  | | --- | |  |  |  | | --- | | Verify enrollment confirmation message | | A confirmation message should appear indicating successful enrollment | Enrollment successful message displayed | Pass |  |  | Bug\_002 |
| 5 | |  | | --- | |  |  |  | | --- | | Check if the enrolled course appears in "My Courses" section | | The enrolled course should appear in the "My Courses" list | Course listed in "My Courses" | Pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Access the Course Dashboard page after login |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | 1. Web application should be accessible.  2. User should be logged into the Course Assignment system.  3. User should be redirected to the dashboard after successful login. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case**  **Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug**  **ID** |
| 1 | |  | | --- | |  |  |  | | --- | | Access Course Assignment web application URL | | The site should launch successfully | Site launched successfully | Pass |  | |  | | --- | |  |  |  | | --- | | courseassignment  .example.com/login | |  |
| 2 | Log in with valid credentials (Username and Password) | User should be able to enter valid credentials and click the login button | Login successful, redirected to dashboard | Pass |  | |  | | --- | |  |  |  | | --- | | Username: validUser@example.com  Password: valid123 | |  |
| 3 | Verify redirection to the Course Dashboard page | User should be redirected to the Course Dashboard page | Redirected to dashboard successfully | Pass |  |  |  |
| 4 | Verify that the dashboard displays the username at the top-right corner | Username should be displayed at the top-right corner of the dashboard | Username displayed at the top-right | Pass |  |  |  |
| 5 | Check if all dashboard components are loaded correctly | All sections and elements on the dashboard should load properly | All dashboard elements loaded correctly | Pass |  |  |  |

# References

* http://www.w3schools.com/html/html\_intro.asp
* https://www.w3schools.com/php/default.asp
* https://www.javatpoint.com/uml