

Darshan University

A Project Report on

**“Admission Management System”**

Under the subject

**Software Engineering (2301CS405)**

B. Tech, Semester – IV

Computer Science & Engineering Department

|  |  |
| --- | --- |
| Submitted By | |
| Student Name: Bhoomi Tulsiyani | Enrolment No.: 23010101275 |
| 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 **“Admission Management System”** submitted in partial fulfilment for the Semester-4 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.

(Bhoomi Tulsiyani)

Student’s Signature

Date: 27/12/2024

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

**CERTIFICATE**

This is to certify that the SRS on **“Admission Management System” has** been satisfactorily prepared by **Bhoomi Tulsiyani** (**23010101275**) 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

**Bhoomi Tulsiyani**

**ABSTRACT**

The Admission Management System is a detailed solution designed to digitalize the admission process in educational institutions. This system aims to replace traditional, paper-based methods with an efficient, computerized platform that simplifies the management of admission-related tasks. This system provides features such as user login for students, faculty and admin, enabling secure access to information. Students can fill out admission forms online, upload required documents, and track the status of their application. The system allows admin to manage the entire admission process. Admins can add, view, update, and delete student records and faculty and monitor application progress. It also includes a facility for faculties to conduct tests, communicate with applicants, and address issues.

The primary goal of this system is to reduce manual workload, enhance accuracy, and improve the overall efficiency of the admission process, making it a seamless experience for both students and administrators.

**Table of Contents**

[List of Figures I](#_Toc190337041)

[List of Tables II](#_Toc190337042)

[1 Introduction 1](#_Toc190337043)

[1.1 Product perspective 1](#_Toc190337044)

[1.2 Product features 1](#_Toc190337045)

[1.2.1 There are three different users who will be using this product: 1](#_Toc190337046)

[1.2.2 The features that are required for the Admin are: 1](#_Toc190337047)

[1.2.3 The features that are required for the Faculty are: 1](#_Toc190337054)

[1.2.4 The features that are required for the Student are: 1](#_Toc190337055)

[1.3 Functional Requirement 1](#_Toc190337056)

[1.3.1 Admin 1](#_Toc190337057)

[1.3.2 Faculty 2](#_Toc190337058)

[1.4 Non-Functional Requirement 2](#_Toc190337066)

[1.4.1 Usability: 2](#_Toc190337067)

[1.4.2 Accuracy: 2](#_Toc190337068)

[1.4.3 Availability: 2](#_Toc190337069)

[1.4.4 Maintainability: 2](#_Toc190337070)

[2 Design and Implementation Constraints 3](#_Toc190337071)

[2.1 Use case diagram 3](#_Toc190337072)

[2.2 Activity diagram and Swimlane diagram 4](#_Toc190337073)

[2.3 Sequence diagram 8](#_Toc190337074)

[2.4 State diagram 10](#_Toc190337075)

[2.5 Class diagram 11](#_Toc190337076)

[2.6 Data flow diagram 12](#_Toc190337077)

[2.6.1 Context diagram (level-0) 12](#_Toc190337078)

[2.6.2 DFD Level-1 13](#_Toc190337079)

[2.6.3 DFD Level-2 14](#_Toc190337080)

[3 External interface requirement (Screens) 15](#_Toc190337081)

[3.1 Screen-1: Prepare Merit List 15](#_Toc190337082)

[3.2 Screen-2: Cancel Admission 16](#_Toc190337083)

[3.3 Screen-3: Fees Payment 17](#_Toc190337084)

[3.4 Screen-4: Track Application Status 18](#_Toc190337085)

[3.5 Screen-5: Check Eligibility of Applications 19](#_Toc190337086)

[4 Database design 20](#_Toc190337087)

[4.1 List of Tables 20](#_Toc190337088)

[5 Stories and Scenario 22](#_Toc190337089)

[5.1 Story-1: Add New Course for Admission 22](#_Toc190337090)

[5.1.1 Scenario# S1.1 22](#_Toc190337091)

[5.1.2 Scenario# S1.2 22](#_Toc190337092)

[5.1.3 Scenario# S1.3 22](#_Toc190337093)

[5.2 Story-2: Search Course 23](#_Toc190337094)

[5.2.1 Scenario# S2.1 23](#_Toc190337095)

[5.2.2 Scenario# S2.2 23](#_Toc190337096)

[5.3 Story-3: Pay fees 23](#_Toc190337097)

[5.3.1 Scenario# S3.1 23](#_Toc190337098)

[5.3.2 Scenario# S3.2 24](#_Toc190337099)

[6 Test cases 25](#_Toc190337100)

[7 References 28](#_Toc190337101)

# List of Figures

[Figure 2.1‑1 Use case diagram for Admission management system 3](#_Toc146706848)

[Figure 2.2‑1 Activity diagram for Apply for Admission 4](#_Toc146706849)

[Figure 2.2‑2 Activity diagram for Add Course 5](#_Toc146706849)

[Figure 2.2‑3 Swimlane diagram for Apply for Admission 6](#_Toc146706850)

[Figure 2.2‑4 Swimlane diagram for Add Course 7](#_Toc146706849)

[Figure 2.3‑1 Sequence diagram for Apply for Admission 8](#_Toc146706851)

[Figure 2.3‑2 Sequence diagram for Add Course 9](#_Toc146706851)

[Figure 2.4‑1 State diagram of Fees 10](#_Toc146706852)

[Figure 2.4‑2 State diagram for Admission form 10](#_Toc146706853)

[Figure 2.5‑1 Class diagram for Admisison management system 11](#_Toc146706854)

[Figure 2.6‑1 Context diagram for Admission management system 12](#_Toc146706855)

[Figure 2.6‑2 DFD level-1 for Admission management system 13](#_Toc146706856)

[Figure 2.6‑3 DFD level-2 for Admission management system 14](#_Toc146706857)

[Figure 3.1‑1 Screen-1: Prepare Merit List 15](#_Toc146706858)

[Figure 3.2‑1 Screen-2: Cancel Admission 1](#_Toc146706860)6

[Figure 3.3‑1 Screen-3: Fees Payment 1](#_Toc146706860)7

[Figure 3.4‑1 Screen-4: Track Application Status 1](#_Toc146706860)8

[Figure 3.5‑1 Screen-5: Check Eligibility of Application 1](#_Toc146706860)9

# List of Tables

[Table 3.1‑1 Screen element of Prepare Merit List 15](#_Toc146706861)

[Table 3.2‑1 Screen element of Cancel Admission 16](#_Toc146706862)

[Table 3.3‑1 Screen element of Fees Payment 17](#_Toc146706863)

[Table 3.4‑1 Screen element of Track Application Status 18](#_Toc146706863)

[Table 3.5‑1 Screen element of Check Eligibility of Application 1](#_Toc146706863)9

[Table 4.1‑1 Table: Course](#_Toc146706864) 20

[Table 4.1‑2 Table: Application](#_Toc146706865) 20

[Table 4.1‑3 Table: Fees 20](#_Toc146706866)

[Table 4.1‑4 Table: Student 20](#_Toc146706866)

[Table 4.1‑5 Table: Department](#_Toc146706867) 20

# Introduction

## Product perspective

This project focuses on transforming the traditional, manual admission process into an internet-based application to enhance efficiency and accessibility for users. The system allows applicants to manage their admission-related activities, such as submitting applications and tracking their status. It is a multi-user platform that supports both students and faculties. It efficiently handles fundamental functions such as student registration, document verification, application tracking, and fee management. Designed to meet the needs of educational institutions, this system provides a reliable solution to manage the entire admission process.

## Product features

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

* Admin who will be acting as the administrator.
* Faculty who will be reviewing student applications, managing student details, or providing recommendations.
* Students who will apply for admission

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

### Manage the admission process, including application deadlines and schedules.

### Add, edit, and delete student records in the system.

### View and process submitted admission applications.

### Manage user accounts for students and faculty.

### Handle queries or requests submitted by students.

### Add, edit or delete course offered for admission.

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

* Review and evaluate student applications assigned to them.
* Provide personalized recommendations for students.
* Access student details.
* Communicate with students regarding admission requirements or feedback.

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

* Register and create an account in the system.
* Fill out and submit admission application online.
* Upload required documents.
* View the status of their application and receive notifications or updates.
* Submit queries or requests related to admissions.

## Functional Requirement

### Admin

* Add Course: Admin can add new courses for admission.
* Delete Course: Admin can remove courses that are no longer offered.
* Get Details: Admin can view specific details about courses or users.
* Update Cut-off: Admin can update the admission cut-off criteria for courses.
* Get Fees Online: Admin manages the collection of fees online.
* Add Faculty: Admin can add new faculty members.
* Remove Faculty: Admin can remove faculty members.
* Respond to Requests: Admin handles and responds to queries or requests from students or faculty.
* Cancel Admission: Admin can cancel a student’s admission if needed.
* Prepare Merit List: Admin and Faculty generate a merit list based on eligibility and criteria.

### Faculty

## Give Personalized Suggestions: Faculty can provide personalized advice to students regarding admission.

## Contact Student: Faculty can communicate directly with students.

## Conduct Tests: Faculty can organize and manage admission tests.

## Report Issue: Faculty can report issues to the admin.

## Review Application: Faculty evaluates student applications.

## Check Eligibility of Applications: Faculty verifies if applications meet eligibility criteria.

### Student

* Login: Students can log in to access the system.
* Apply for Courses: Students can apply for any course.
* View Courses: Students can browse available courses.
* Update Profile: Students can edit and update their profile details.
* Fill Admission Form: Students submit the admission form for enrolment.
* Search for Eligibility: Students can check their eligibility for specific courses.
* Upload Documents: Students can upload required documents for verification.
* Pay Fees Online: Students can make fee payments online.
* Track Application Status: Students and faculty both can check their application status.

## 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 ‑ Use case diagram for admission management system

## Activity diagram and Swimlane diagram

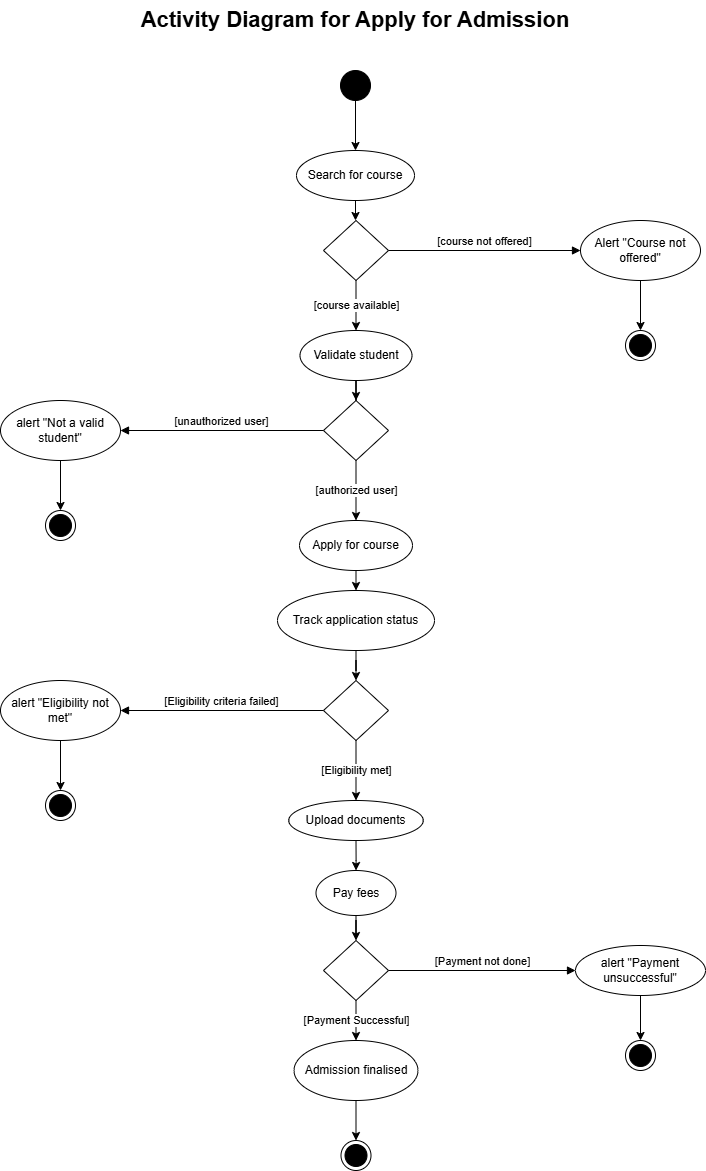


Figure ‑ Activity diagram for Apply for Admission

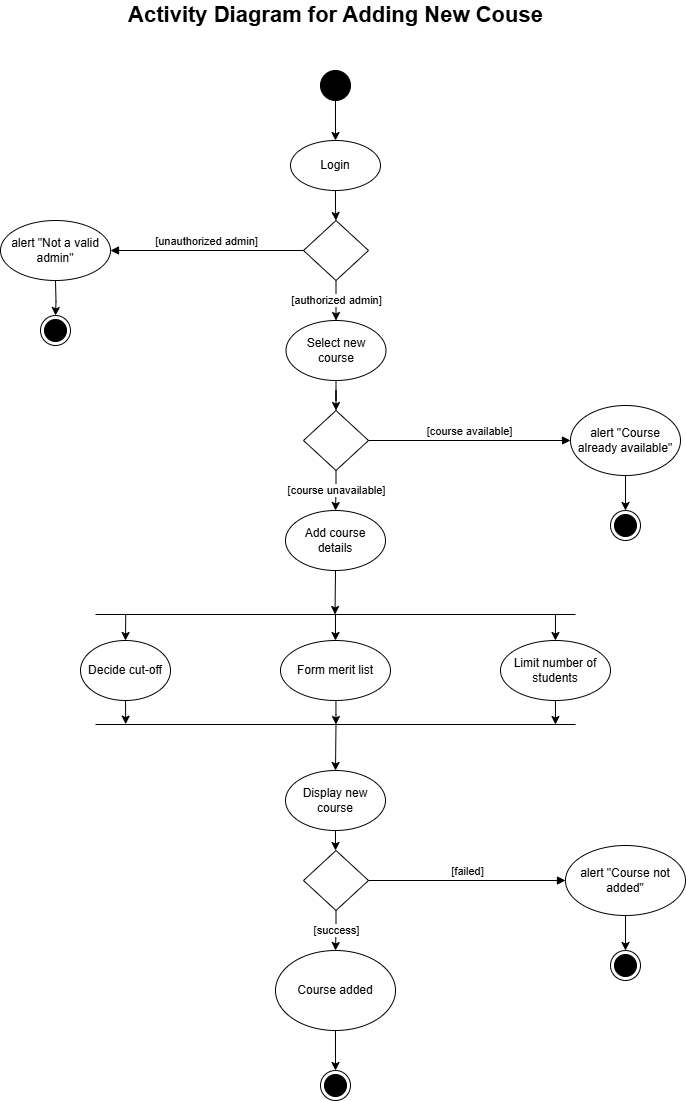


Figure ‑2 Activity diagram for Add Course

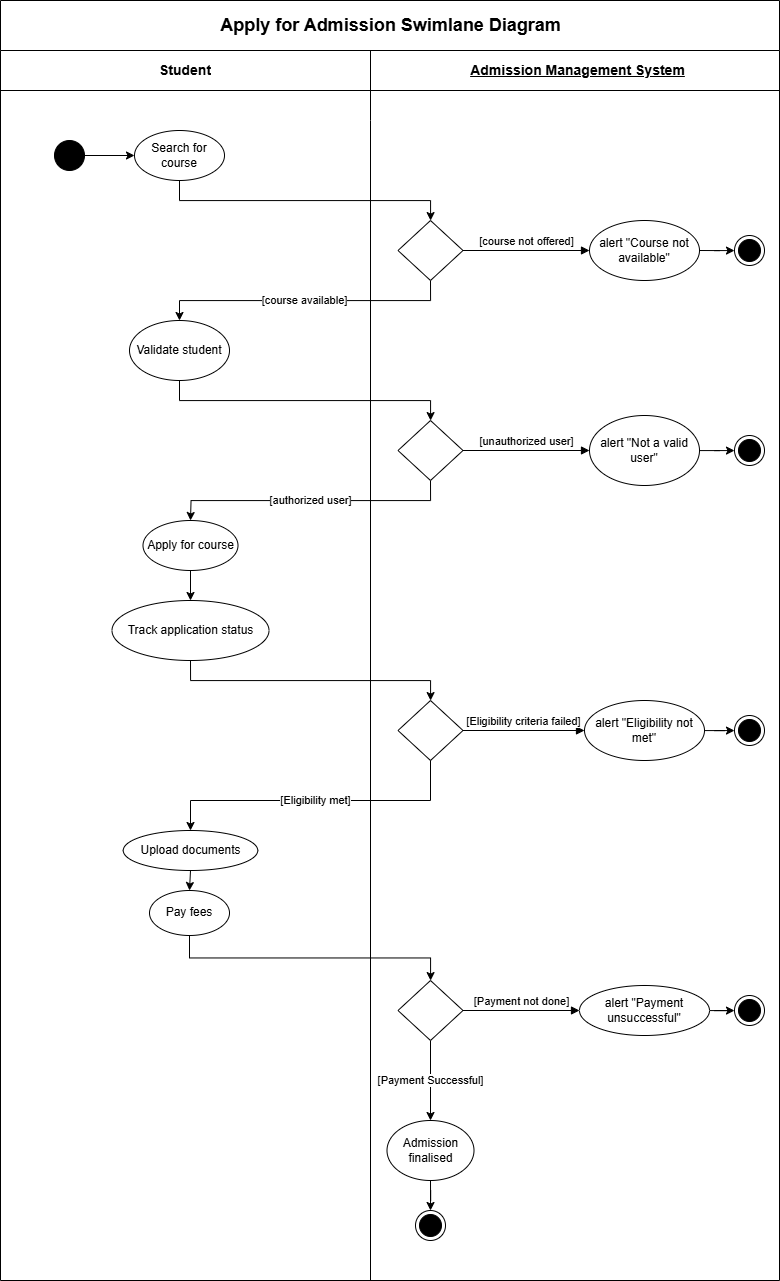


Figure ‑3 Swimlane diagram for Apply for Admission

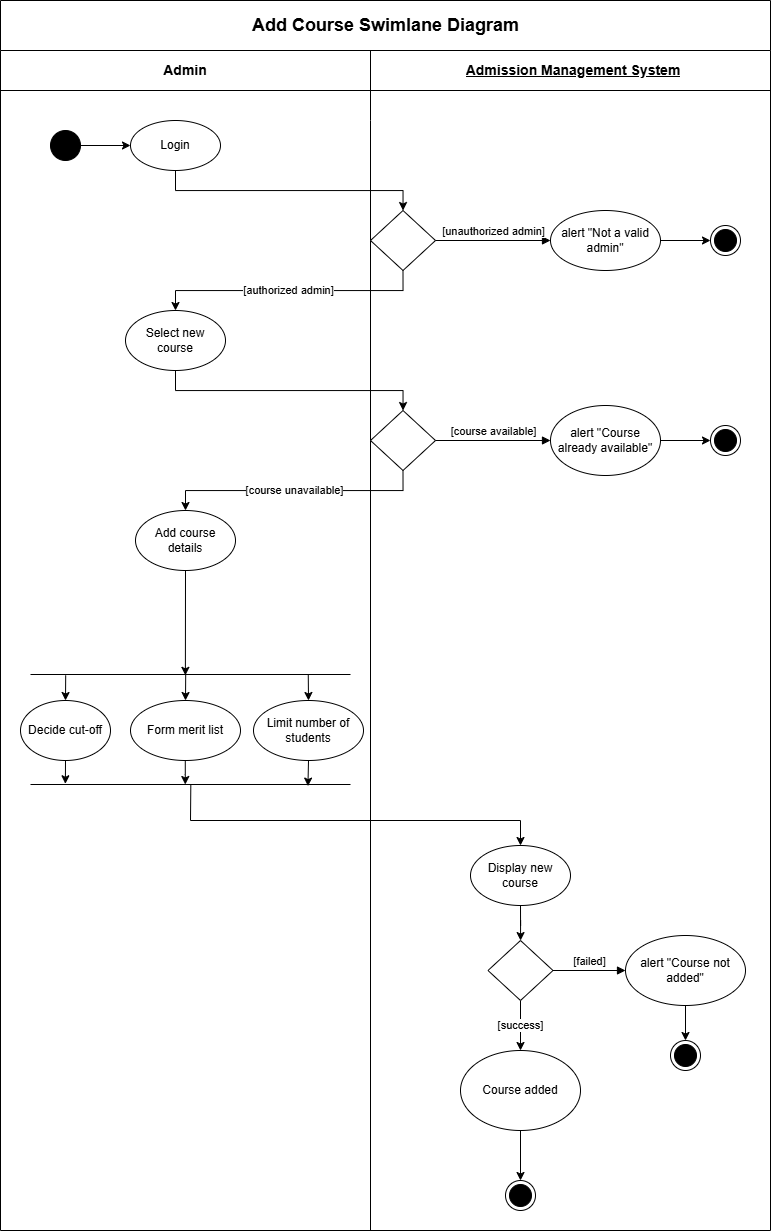


Figure ‑4 Swimlane diagram for Add Course

## Sequence diagram

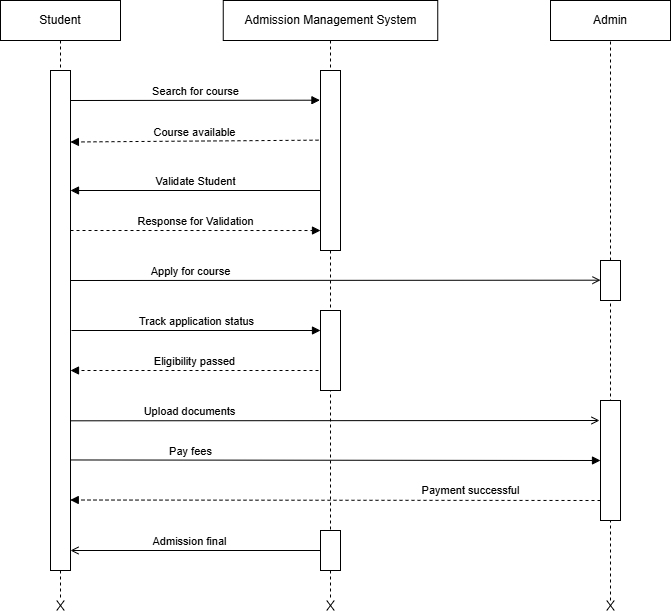


Figure ‑ Sequence diagram for Apply for Admission

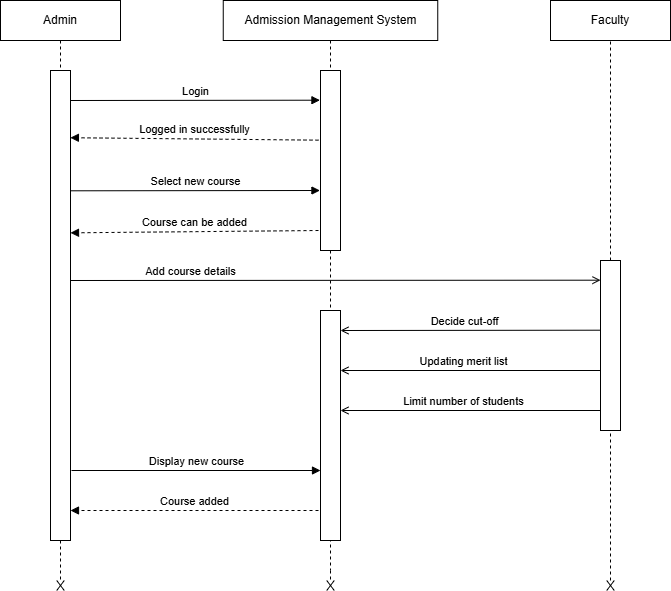


Figure ‑2 Sequence diagram for Add Course

## State diagram

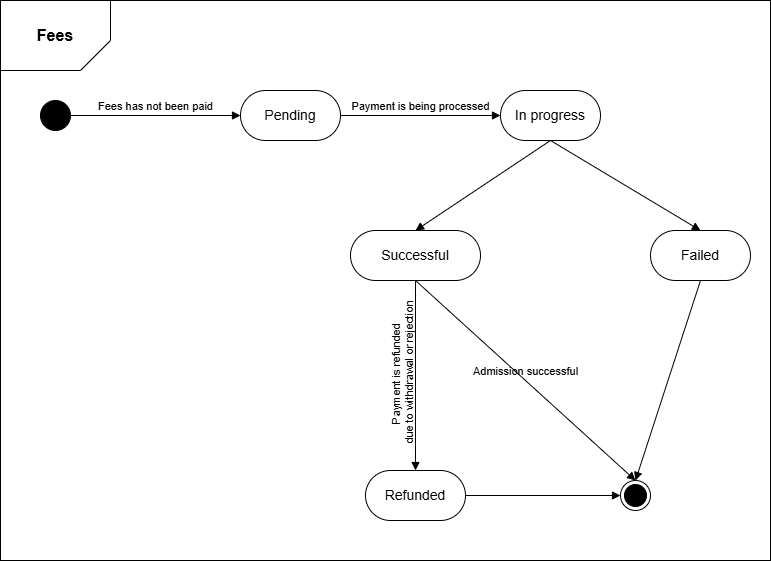


Figure ‑ State diagram of Fees

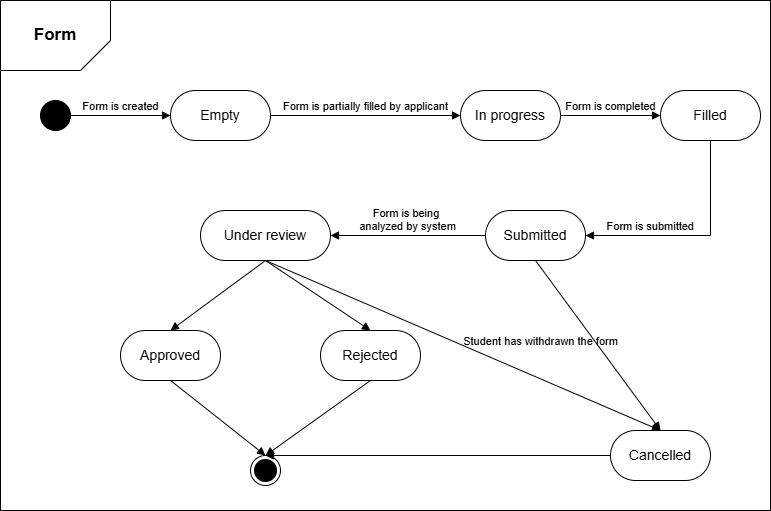


Figure ‑ State diagram for Admission Form

## Class diagram

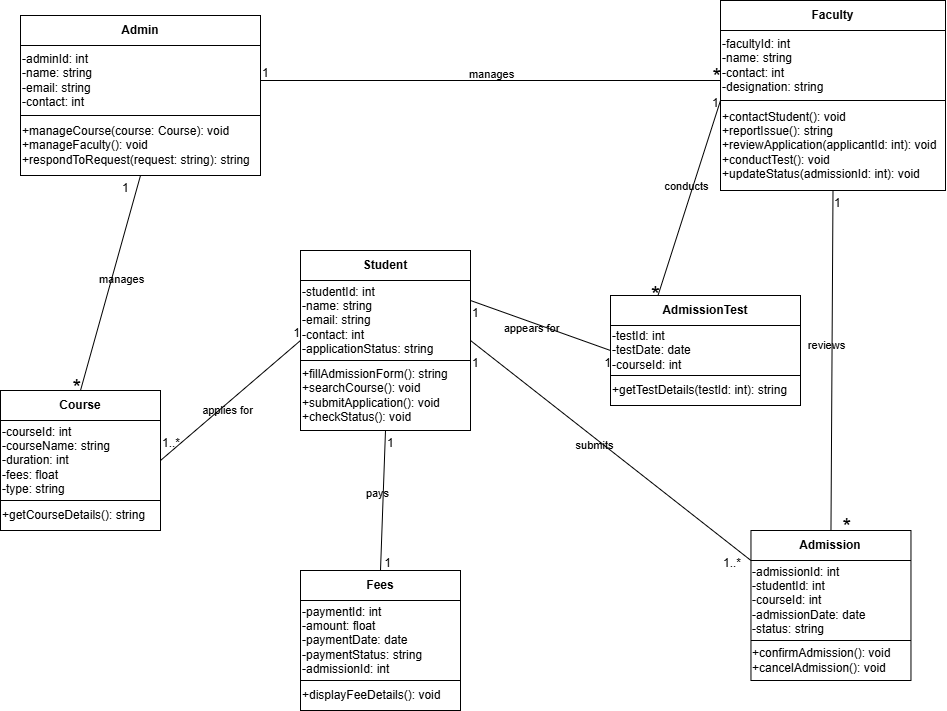


Figure ‑ Class diagram for Admission management system

## Data flow diagram

### Context diagram (level-0)

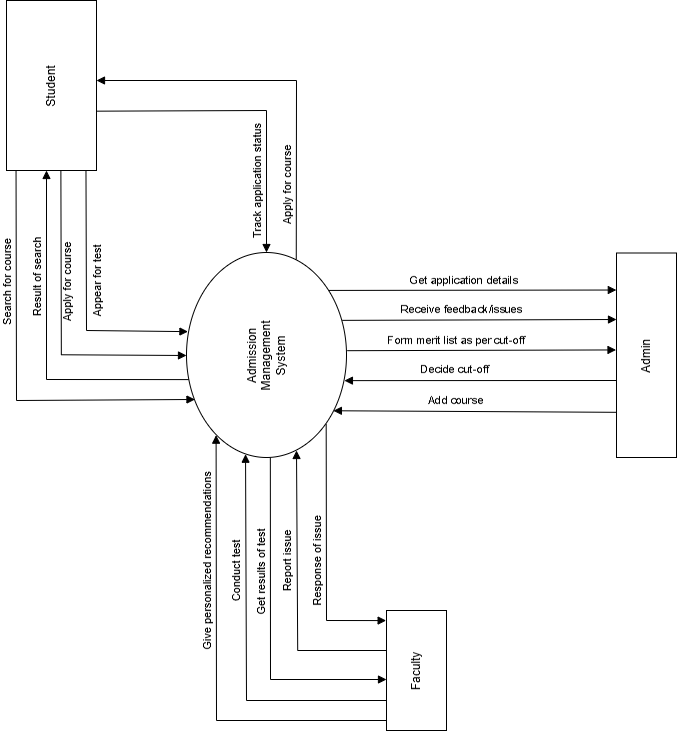


Figure ‑ Context diagram for Admission management system

### DFD Level-1

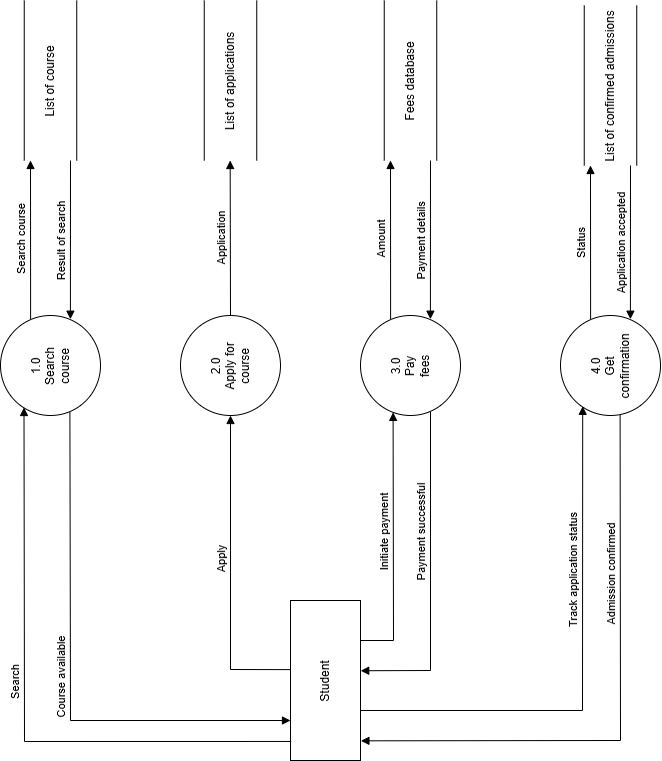


Figure ‑ DFD level-1 for Admission management system

### DFD Level-2

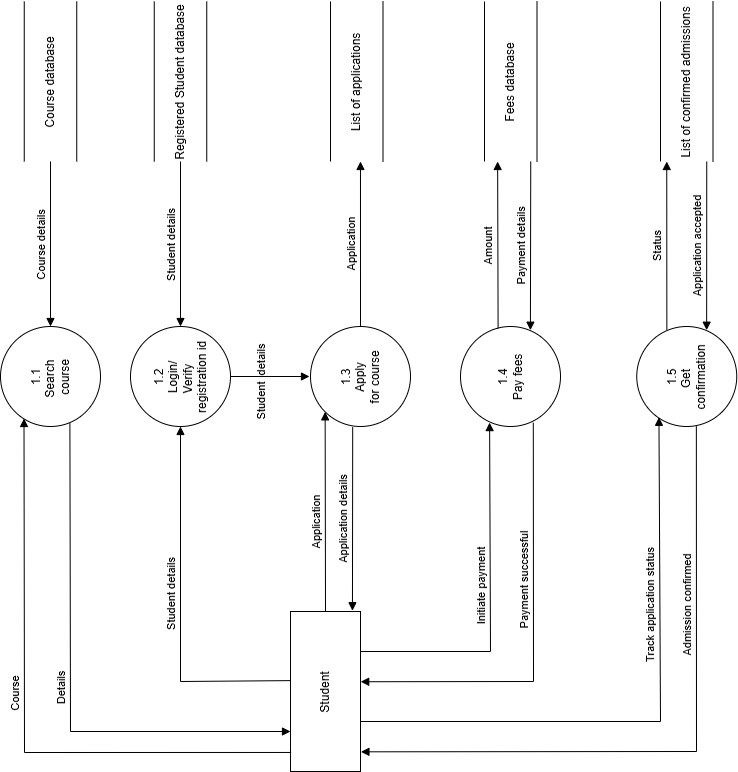


Figure ‑ DFD level-2 for Admission management system

# External interface requirement (Screens)

## Screen-1: Prepare Merit List

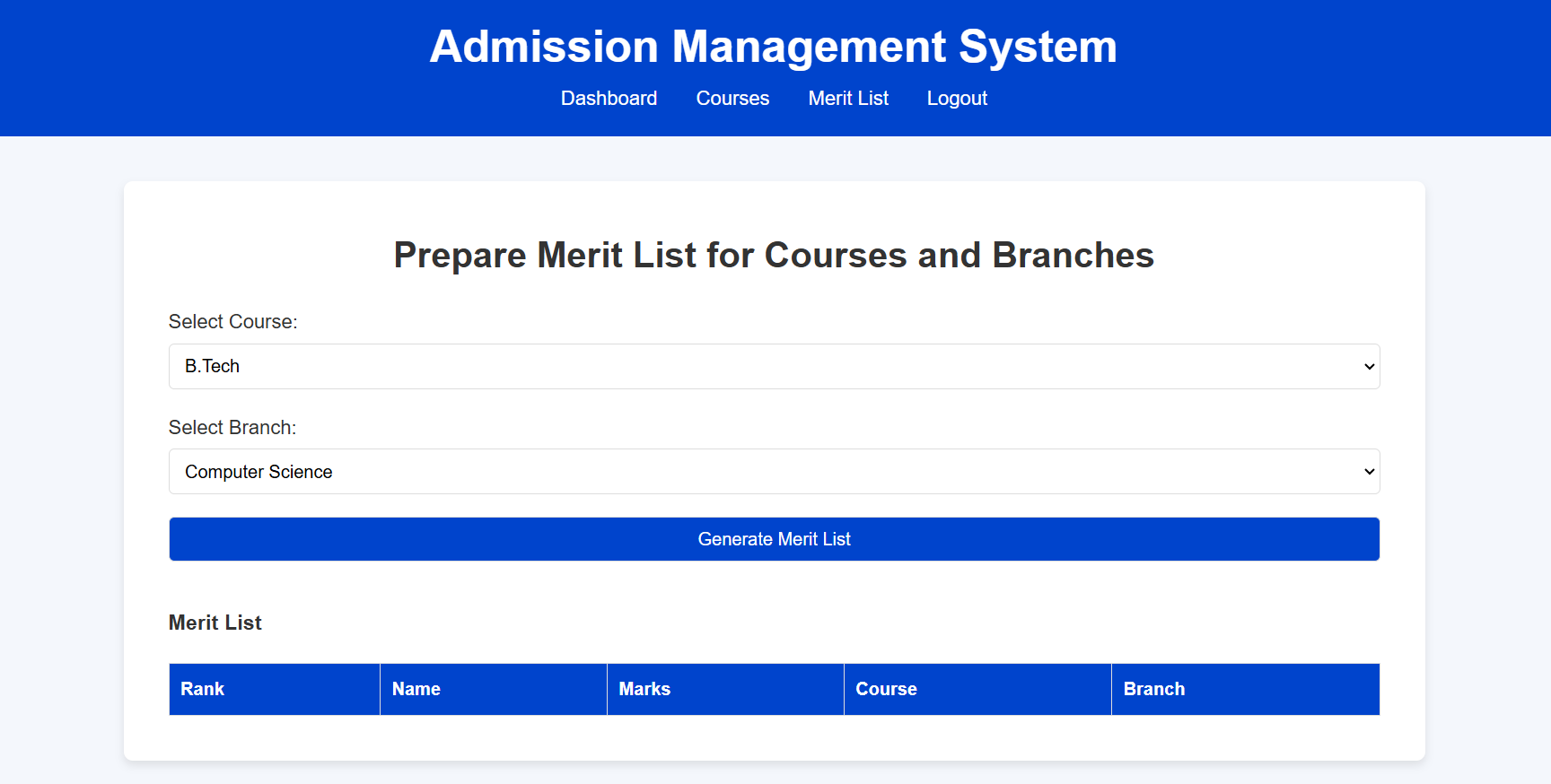


Figure ‑ Screen-1: Prepare Merit List

**Purpose:** This form will allow the admin to prepare merit list in the system for different courses and branches. To apply, the following information will be encoded in the system.

Table ‑ Screen element of Prepare Merit List

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Select Course | Dropdown | M | 1 | Course field should be selected from the dropdown. |
| 2 | Select Branch | Dropdown | M | 1 | Branch field should be selected from the dropdown. |
| 3 | Generate Merit list | Button | ------ | ------ | Generate Merit List is a button to store the criteria for merit list. |

## Screen-2: Cancel Admission

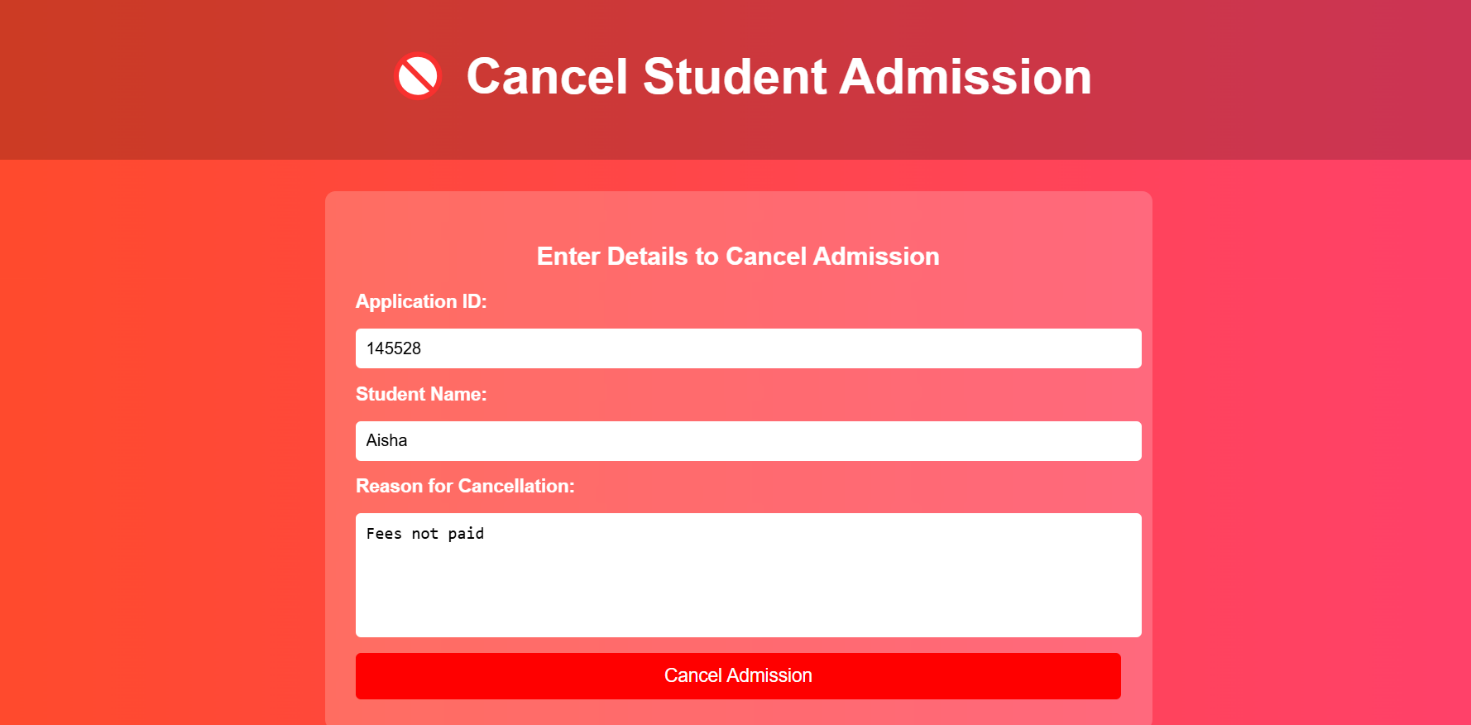


Figure ‑ Screen-2: Cancel Admission

**Purpose:** This form will be used by the admin to cancel admission of students under exceptional circumstances.

Table ‑ Screen element of Cancel Admission

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Application ID | Textbox | M | 1 | Application ID field should be editable and accept the Application ID. |
| 2 | Student Name | Textbox | M | 1 | Name field should be editable and accept the name with proper format. |
| 3 | Reason for Cancellation | Textbox | M | 1 | Reason for cancelling admission must be written. |
| 4 | Cancel Admission | Button | ------ | ------ | This button confirms the cancelling of admission by admin. |

## Screen-3: Fees Payment

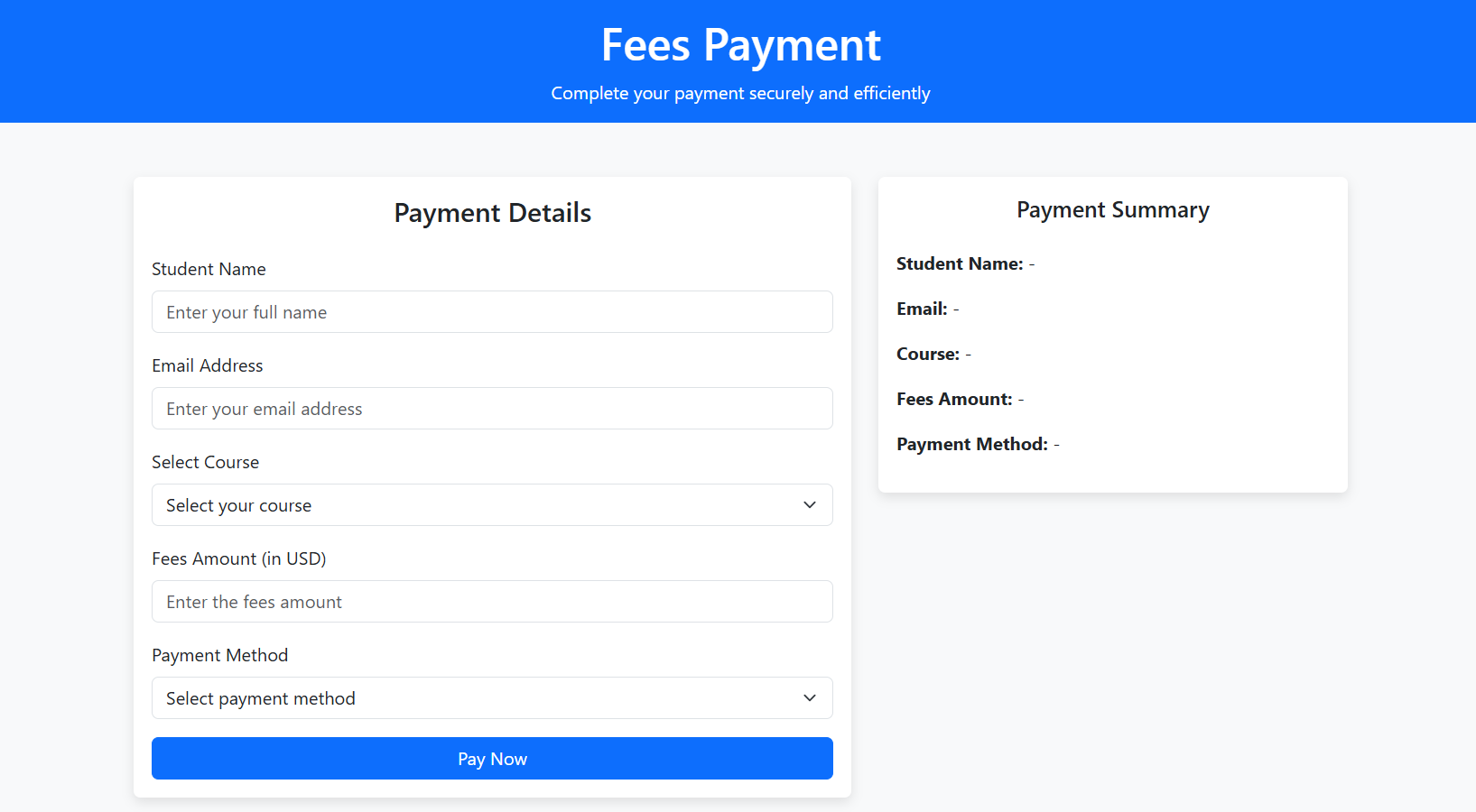


Figure ‑ Screen-2: Fees Payment

**Purpose:** This form will be used by the students to pay fees online securely.

Table ‑ Screen element of Fees Payment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Student Name | Textbox | M | 1 | Name field should be editable and accept the student name. |
| 2 | Email Address | Textbox | M | 1 | Email field should be editable and accept the email with proper format. |
| 3 | Select Course | Dropdown | M | 1 | Course can be selected from the dropdown showing all course options available. |
| 4 | Fees Amount | Textbox | M | 1 | Fees amount will automatically be filled when course is selected. |
| 5 | Payment Method | Dropdown | M | 1 | Payment method can be selected from the dropdown menu showing all options. |
| 6 | Pay Now | Button | ------ | ------ | This button confirms payment by user. |
| 7 | Payment Summary | ------ | ------ | ------ | This shows the details of payment summary by the customer. |

## Screen-4: Track Application Status

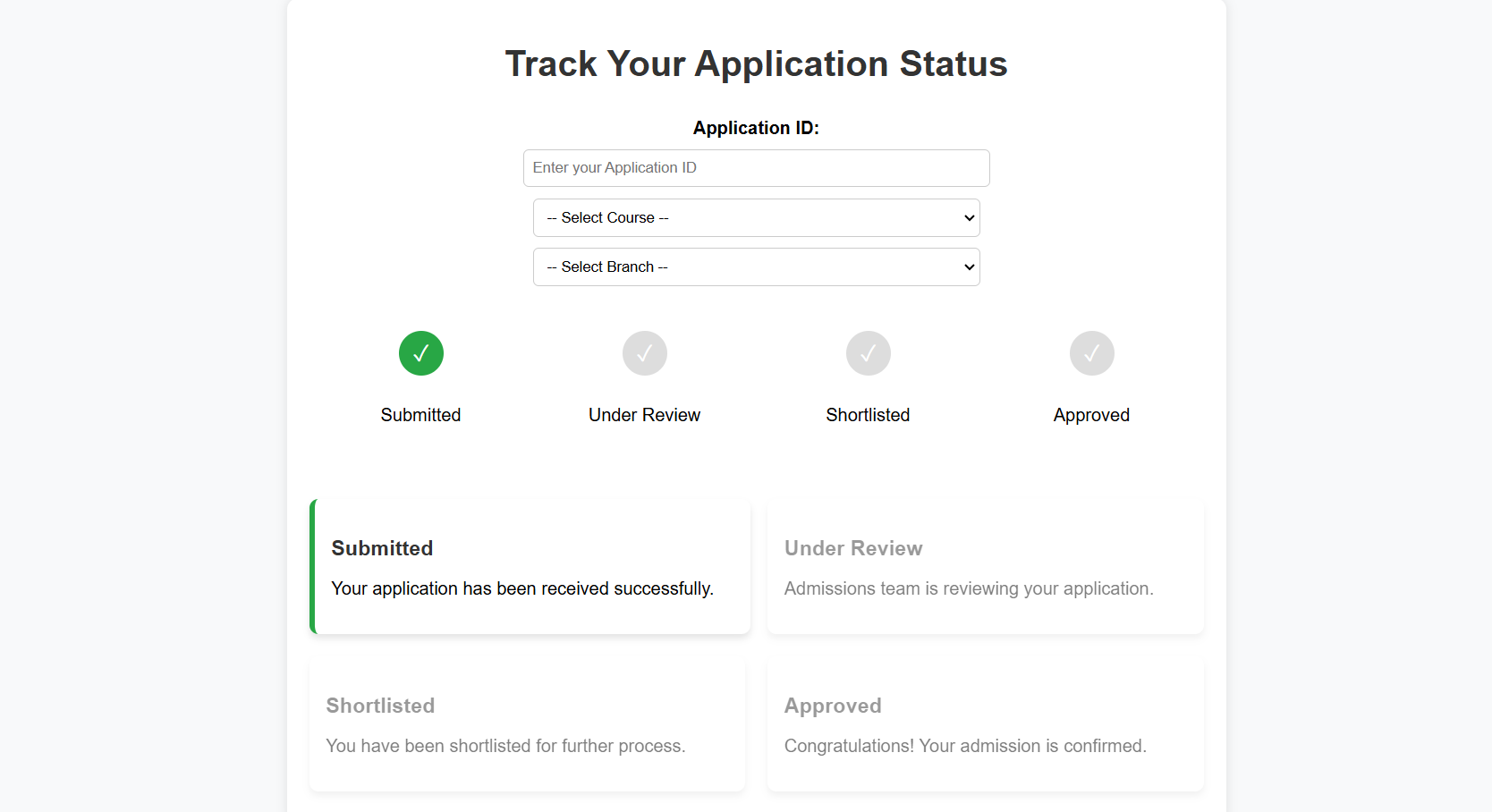


Figure ‑ Screen-3: Track Application Status

**Purpose:** This module will allow the student to track their application status.

Table ‑ Screen element of Track Application Status

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Application ID | Textbox | M | 1 | Application ID field should be editable. |
| 2 | Select Course | Dropdown | M | 1 | Course field should be selected from the dropdown. |
| 3 | Select Branch | Dropdown | M | 1 | Branch field should be selected from the dropdown. |

## Screen-5: Check Eligibility of Applications

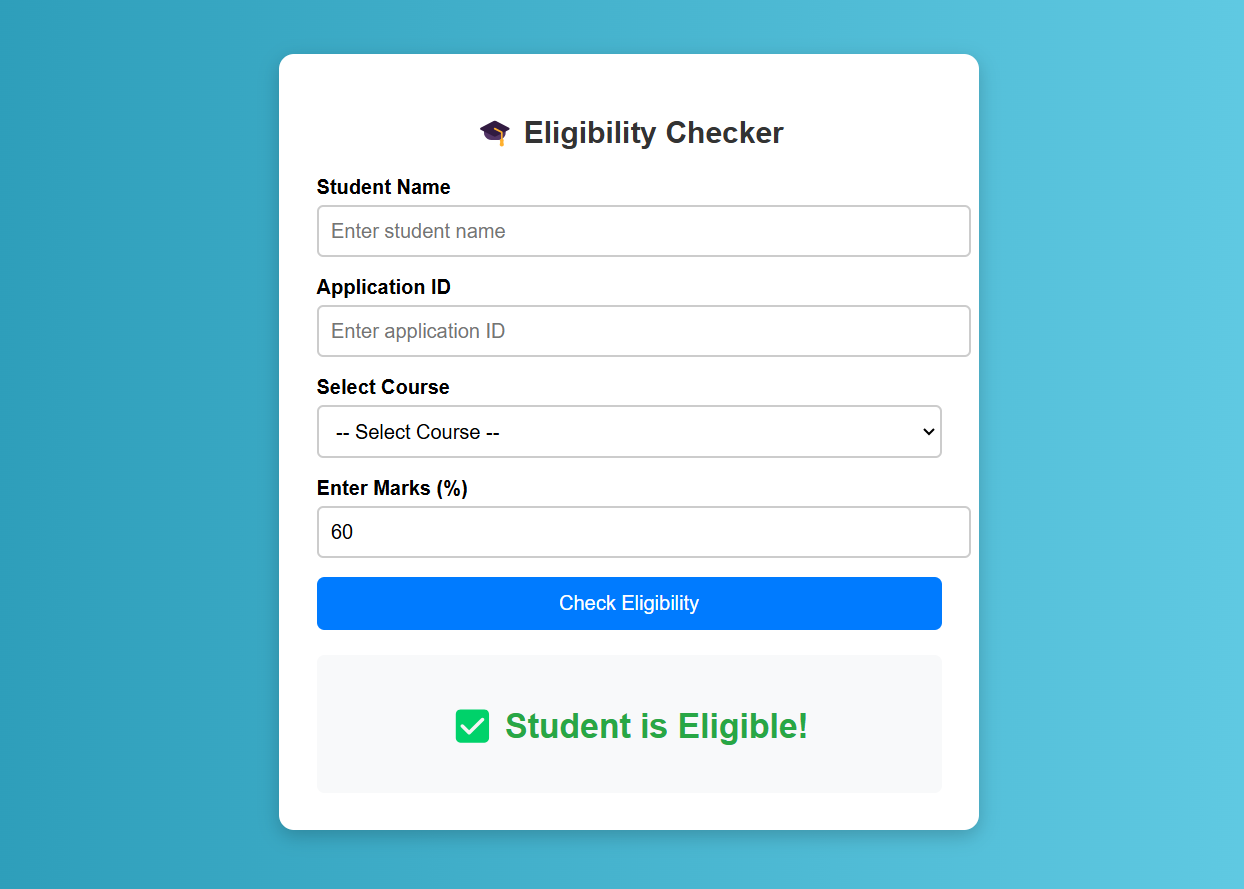


Figure ‑ Screen-3: Check Eligibility of Application

**Purpose:** This module will allow faculty to check eligibility of application of students.

Table ‑ Screen element of Check Eligibility of Application

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Student Name | Textbox | M | 1 | Name field should be editable. |
| 2 | Application ID | Textbox | M | 1 | Application ID field should be editable. |
| 3 | Select Course | Dropdown | M | 1 | Course field should be selected from the dropdown. |
| 4 | Enter Marks | Textbox | M | 1 | Marks of student should be entered here to check eligibility. |
| 5 | Check Eligibility | Button | ----- | ----- | This button checks the marks entered as per eligibility criteria of that course and displays result below. |

# Database design

## List of Tables

* Course
* Application
* Fees
* Student
* Department

Table ‑ Table: Course

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| CourseID | int | NN | PK (Auto Increment) |  |
| CourseName | varchar(100) | AN |  |  |
| DepartmentID | varchar(100) | NN | FK | Reference of Department Table |
| Credits | int | AN |  |  |
| StudentID | int | NN | FK | Reference of Student Table |

Table ‑ Table: Application

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| ApplicationID | int | NN | PK (Auto Increment) |  |
| StudentID | int | NN | FK | Reference of Student Table |
| CourseID | int | NN | FK | Reference of Course Table |
| ApplicationDate | DateTime | AN |  |  |

Table ‑ Table: Fees

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| FeesID | int | NN | PK (Auto Increment) |  |
| StudentID | int | NN | FK | Reference of Student Table |
| CourseID | int | NN | FK | Reference of Course Table |
| FeesStatus | varchar(100) | AN |  |  |
| TransactionDate | DateTime | NN |  |  |

Table ‑ Table: Student

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| StudentID | int | NN | PK (Auto Increment) |  |
| StudentName | varchar(100) | AN |  |  |
| Phone | int | AN |  |  |
| DOB | DateTime | AN |  |  |
| CourseID | int | NN | FK | Reference of Course Table |

Table ‑5 Table: Department

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| DepartmentID | int | NN | PK (Auto Increment) |  |
| DepartmentName | varchar(100) | AN |  |  |
| CourseList | varchar(100) | NN | FK | Reference of Course Table |
| AllotedSeats | int | AN |  |  |

# Stories and Scenario

## Story-1: Add New Course for Admission

|  |  |  |
| --- | --- | --- |
| *Story # S1* | : | As an Admin,  I want to add a new course in college for admission  So that students can easily search and apply for it. |
| Priority | **:** | High |
| Estimate | **:** | XL |
| Reason | **:** | The addition of a new course to the college is crucial for ensuring that the courses catalogue is up-to-date and students can apply for the same. |

### Scenario# S1.1

|  |  |  |
| --- | --- | --- |
| *Scenario# S1.1* | : | Adding a New Course with Valid Information |
| Prerequisite | **:** | Admin is logged in to the admission management system. |
| Acceptance Criteria | **:** | **Given:**  The Admin is navigated to the courses list management page. Valid course information, including name, department, credits, and other relevant details is added.  **When:**  The admin selects the "Add New Course" option  And The admin enters valid course details  The admin clicks the "Save" button to add the course to the list of courses.  **Then** the system successfully adds the course to the list and the admin receives a confirmation message with the course's id number. |

### Scenario# S1.2

|  |  |  |
| --- | --- | --- |
| *Scenario# S1.2* | : | Adding a New Course with Invalid Information. |
| Prerequisite | **:** | The admin is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The admin is on the list of courses management page  **When:** The admin selects the "Add New Course" option and the admin enters an incomplete or incorrect course details and admin clicks the "Save" button to add the course.  **Then** the system displays error messages for the incorrect or missing information and the course is not added to the list. |

### Scenario# S1.3

|  |  |  |
| --- | --- | --- |
| *Scenario# S1.3* | : | Attempting to Add a Duplicate Course |
| Prerequisite | **:** | The admin is logged into the admission management system and the admin is on the list of courses management page |
| Acceptance Criteria | **:** | **Given**: The course information, including name, department, credits, and other relevant details, is available and the course with the same name and department is already in the list.  **When**: User Clicks on “Add course” button. Enter a number of copies with the same course detail mentioned in the field.  **Then**: Generate unique course id for various course of same name and department. |

## Story-2: Search Course

|  |  |  |
| --- | --- | --- |
| *Story # S2* | : | As a student,  I want to search for courses by name, department, or college,  So that I can quickly find courses offered as per my preference. |
| Priority | **:** | High |
| Estimate | **:** | M |
| Reason | **:** | Implementing a search functionality is essential for enhancing the user experience, as it allows student to efficiently discover and access the course details. |

### Scenario# S2.1

|  |  |  |
| --- | --- | --- |
| *Scenario# S2.1* | : | Searching for a course that is offered for admission. |
| Prerequisite | **:** | The student is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The student is on the list of courses page  **When:** The student types the course or its details in the search bar and clicks the "Search" button to search for the course that is offered by college.  **Then** the system displays the course details for admission. |

### Scenario# S2.2

|  |  |  |
| --- | --- | --- |
| *Scenario# S2.2* | : | Searching for a course that is not offered for admission. |
| Prerequisite | **:** | The student is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The student is on the list of courses page  **When:** The student types the course or its details in the search bar and clicks the "Search" button to search for the course that is not offered by college.  **Then** the system displays the message that “No Course Found”. |

## Story-3: Pay fees

|  |  |  |
| --- | --- | --- |
| *Story # S3* | : | As a student,  I want to pay fees  So that I can ensure my admission has been confirmed and my seat has been reserved. |
| Priority | **:** | High |
| Estimate | **:** | L |
| Reason | **:** | Paying fees on time is crucial because it confirms my seat in the course and I don’t have to pay the penalty. |

### Scenario# S3.1

|  |  |  |
| --- | --- | --- |
| *Scenario# S3.1* | : | Paying fees with valid details. |
| Prerequisite | **:** | The student is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The student is navigated to Payment section.  **When:**  The student selects the "Pay Fees" option  And The student enters valid fees details along with payment methods.  The student clicks the "Pay" button to pay fees  **Then** the system shows the confirmation message showing “Payment Successful, Admission Confirmed”. |

### Scenario# S3.2

|  |  |  |
| --- | --- | --- |
| *Scenario# S3.2* | : | Paying fees with invalid details. |
| Prerequisite | **:** | The student is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The student is navigated to Payment section.  **When:**  The student selects the "Pay Fees" option  And The student enters invalid fees details along with payment methods.  The student clicks the "Pay" button to pay fees  **Then** the system shows the failure message showing “Payment Unsuccessful”. |

# Test cases

|  |  |  |  |
| --- | --- | --- | --- |
| Project Name: | Admission Management System | Test Designed by: | Bhoomi Tulsiyani |
| Module Name: | **Fees Payment** | **Test Designed date:** | 01-01-2025 |
| Release Version: | **1.0** | **Test Executed by:** | **R. B. Gondaliya** |
|  |  | **Test Execution date:** | 20-02-2023 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pre-condition: The student must be registered and have pending admission fees. | | | | |
| Test Case ID | **Test Title** | **Test Type** | **Description** | **Test Case ID** |
| TC\_001 | Successful fee payment | Functional | Students pay their admission fees online successfully. Payment is processed securely, and a confirmation receipt is generated. | TC\_001 |
| TC\_002 | Failed Fee Payment Due to Insufficient Balance | Functional | Students try to pay admission fees online but the transaction fails due to insufficient balance or some other technical issue. | TC\_002 |

|  |  |
| --- | --- |
| **Test Case Title** | Successful Fee Payment |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | The student must be registered and have pending admission fees. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Navigate to the fee payment page | The page should load properly | Page loaded successfully | Pass |  | <http://localhost:3000/students/payment> |  |
| 2 | Select a payment method (Credit Card, UPI, Net Banking) | Payment method should be selected successfully | Payment method selected | Pass |  | Payment Method: Credit Card |  |
| 3 | Enter valid payment details and submit | Payment should be processed, and confirmation should appear. | Payment successful, receipt generated | pass |  | Transaction ID: PAY12345 |  |
| 4 | Verify that the payment is reflected in the student’s profile | Payment status should change to "Paid" | Payment status updated to "Paid" | Pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Failed Fee Payment Due to Insufficient Balance |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | The student must be registered, have pending admission fees, and use a payment method with insufficient balance. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Navigate to the fee payment page | The page should load properly | Page loaded successfully | Pass |  | http:localhost  :3000//students/  payment |  |
| 2 | Select a payment method (Credit Card, UPI, Net Banking) | Payment method should be selected successfully | Payment method selected | Pass |  | Payment Method: Credit Card |  |
| 3 | Enter valid payment details but with insufficient balance | Payment should be declined, and an error message should appear | Error message: "Insufficient balance, transaction failed." | Pass |  | Transaction ID: PAY67890 |  |
| 4 | Verify that the payment status remains "Pending" in the student’s profile | Payment status should not change to "Paid" | Payment status remains "Pending" | Pass |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Project Name: | Admission Management System | Test Designed by: | Bhoomi Tulsiyani |
| Module Name: | **Course Enrollment** | **Test Designed date:** | 01-01-2025 |
| Release Version: | **1.0** | **Test Executed by:** | **R. B. Gondaliya** |
|  |  | **Test Execution date:** | 20-02-2023 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pre-condition: The student must be registered and logged into the system. | | | | |
| Test Case ID | **Test Title** | **Test Type** | **Description** | **Test Case ID** |
| TC\_001 | Enroll in a Course Successfully | Functional | Students enroll in available courses after completing registration. Students select a course, enroll, and view the confirmation. | TC\_001 |
| TC\_002 | Attempt to Enroll in a Full Course | Functional | Students enroll in a course that is already full after completing registration. Students select a course, enroll, and see the message saying the course is already full. | TC\_002 |

|  |  |
| --- | --- |
| **Test Case Title** | Enroll in a Course Successfully |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | The student must be registered and logged into the system. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Navigate to the course enrollment page | The page should load properly | Page loaded successfully | Pass |  | <http://localhost:3000/students/courses> |  |
| 2 | Select a course from the list | The selected course should be highlighted | Course selected successfully | Pass |  | Course: B.Sc. Computer Science |  |
| 3 | Click the "Enroll" button | Confirmation message should appear | Enrollment confirmation displayed | pass |  | Transaction ID: PAY12345 |  |
| 4 | |  | | --- | |  |  |  | | --- | |  |  |  |  |  | | --- | --- | --- | | |  | | --- | |  |  |  | | --- | | Verify that  the enrolled  course  appears in  the  student’s  profile | | | The enrolled course should be listed under "My Courses" | Course displayed in student’s profile | Pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Attempt to Enroll in a Full Course |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | The student must be registered and attempting to enroll in a course that has reached maximum capacity. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Navigate to the course enrollment page | The page should load properly | Page loaded successfully | Pass |  | http:localhost  :3000/students  /courses |  |
| 2 | |  | | --- | |  |  |  | | --- | | Select a course that has reached maximum capacity | | System should notify that the course is full | Error message displayed | Pass |  | Course: B.Tech Artificial Intelligence (Full) |  |
| 3 | Click the "Enroll" button | Enrollment should be blocked, and an error message should appear | Error message: "Course is full, enrollment not possible." | Pass |  |  |  |

# References

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