

Darshan University

A Project Report on

**“ADMISSION MANAGEMENT”**

Under the subject

**Software Engineering (2301CS405)**

B. Tech, Semester – IV

Computer Science & Engineering Department

|  |  |
| --- | --- |
| Submitted By | |
| Student Name:Rushi Gambhava | Enrollment No.:23010101082 |
| 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”** 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.

(Rushi Gambhava)

Student’s Signature

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

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

**CERTIFICATE**

This is to certify that the SRS on **“Admission management” has** been satisfactorily prepared by Rushi Gambhava (23010101082) 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

**RUSHI GAMBHAVA**

**ABSTRACT**

Library management system is a system which aims in developing a computerized system to maintain all the daily work of library. This system will act as a tool to transfer traditional library into digital library. This project has many features which are generally not available in general library system like facility of user login, books search on one click etc. It also has a facility of admin login through which the admin can monitor the whole system.  This system will be designed with the basic features such as librarian can add/view/update/delete books and students' details in it. It has also a facility where student after logging in their accounts can see list of books issued and its issue date and return date and also they can request the librarian to add new books by filling the book request form. The librarian after logging into his account i.e. admin account can generate various reports such as student report , issue report, teacher report and book report.

Main purpose of this system is to reduce human efforts as much as possible.

**Table of Contents**

[List of Figures I](#_Toc190078015)

[List of Tables II](#_Toc190078016)

[1 Introduction 1](#_Toc190078017)

[1.1 Product perspective 1](#_Toc190078018)

[1.2 Product features 1](#_Toc190078019)

[1.2.1 There are three different users who will be using this product: 1](#_Toc190078020)

[1.2.2 The features that are required for the Administrator are: 1](#_Toc190078021)

[1.2.3 The features that are required for the Applicant are: 1](#_Toc190078022)

[1.3 Functional Requirement 2](#_Toc190078023)

[**1.3.1** **Student** 2](#_Toc190078024)

[**1.3.2** **Staff** 2](#_Toc190078025)

[**1.3.3** **Admin** 2](#_Toc190078026)

[1.4 Non-Functional Requirement 3](#_Toc190078027)

[1.4.1 Usability: 3](#_Toc190078028)

[1.4.2 Accuracy: 3](#_Toc190078029)

[1.4.3 Availability: 3](#_Toc190078030)

[1.4.4 Maintainability: 3](#_Toc190078031)

[2 Design and Implementation Constraints 4](#_Toc190078032)

[2.1 Use case diagram 4](#_Toc190078033)

[2.2 Activity diagram and Swimlane diagram 5](#_Toc190078034)

[2.3 Sequence diagram 9](#_Toc190078035)

[2.4 State diagram 10](#_Toc190078036)

[2.5 Class diagram 11](#_Toc190078037)

[2.6 Data flow diagram 12](#_Toc190078038)

[2.6.1 Context diagram (level-0) 12](#_Toc190078039)

[2.6.2 DFD Level-1 13](#_Toc190078040)

[3 External interface requirement (Screens) 14](#_Toc190078041)

[3.1 Screen-1: Registration Form 14](#_Toc190078042)

[3.2 Screen-2: View Profile Form 15](#_Toc190078043)

[3.3 Screen-3: View Student details book 16](#_Toc190078044)

[3.4 Screen-4: collect fees Page 17](#_Toc190078045)

[3.5 Screen-5: Add student Page 18](#_Toc190078046)

[4 Database design 19](#_Toc190078047)

[4.1 List of Tables 19](#_Toc190078048)

[5 Stories and Scenario 21](#_Toc190078049)

[5.1 Story-1: Add New Book in Library Catalogue 21](#_Toc190078050)

[5.1.1 Scenario# S1.1 21](#_Toc190078051)

[5.1.2 Scenario# S1.2 21](#_Toc190078052)

[5.1.3 Scenario# S1.3 21](#_Toc190078053)

[5.2 Story-2: Search Book **Error! Bookmark not defined.**](#_Toc190078054)

[5.3 Story-3: Manage due date for borrowed book **Error! Bookmark not defined.**](#_Toc190078055)

[5.4 Story-4: Renew book **Error! Bookmark not defined.**](#_Toc190078056)

[5.5 Story-5: Generate a report on book usage and availability **Error! Bookmark not defined.**](#_Toc190078057)

[6 Test cases 25](#_Toc190078058)

[7 References 33](#_Toc190078059)

# List of Figures

[Figure 2.1‑1 Use case diagram for library management system 4](#_Toc190078107)

[Figure 2.2‑1 Activity diagram of Admission Management(Student) 5](#_Toc190078108)

[Figure 2.2‑2 Activity Diagram of Admission Management(Staff) 6](#_Toc190078109)

[Figure 2.2‑2 Swimlane diagram for Admission Management 8](#_Toc190078110)

[Figure 2.4‑1 State diagram of student 10](#_Toc190078111)

[Figure 2.5‑1 Class diagram for Admission Management system 11](#_Toc190078112)

[Figure 2.6‑1 Context diagram for Admission management system 12](#_Toc190078113)

[Figure 2.6‑2 DFD level-1 for Admission management system 13](#_Toc190078114)

[Figure 3.1‑1 Screen-1: Registration Form 14](#_Toc190078115)

[Figure 3.2‑1 Screen-2: view formForm 15](#_Toc190078116)

[Figure 3.3‑1 Screen-3: View Student Details book 16](#_Toc190078117)

# List of Tables

[Table 3.1‑1 Screen element of Registration form 14](#_Toc146706861)

[Table 3.2‑1 Screen element of Login form 15](#_Toc146706862)

[Table 3.3‑1 Screen element of Add borrower book 16](#_Toc146706863)

[Table 4.1‑1 Table: Book 19](#_Toc146706864)

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

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

[Table 4.1‑4 Table: Student 20](#_Toc146706867)

# Introduction

## Product perspective

The Admission Management project is a web-based platform designed to simplify and improve the admissions process for schools and colleges. It offers a user-friendly portal for students to submit applications, upload documents, and track their status, and a secure dashboard for staff to review applications and communicate with applicants. The system integrates with existing student databases and payment services, provides automatic notifications, and generates reports to help schools analyse and enhance their admissions process. This project aims to save time, reduce errors, and make admissions easier and more efficient for everyone involved.

## Product features

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

* **Administrator** who will be managing admissions and student data.
* **Applicant** who will be applying for admission.
* **Guest** who will request information or submit an application.

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

* Manage the movement of student applications and avoid losing important documents.
* Search for a specific application based on applicant name, admission number, program, etc.
* Print admission-related documents such as offer letters or student IDs.
* Track which applicants have completed admission requirements and submitted necessary documents. Offers modules for Application Management, Student Records, and Queries.
* Approve or reject an application.
* View the list of applicants for each program or department.
* Accept admission confirmation or withdrawal from applicants.
* Add new programs or courses to the system.
* Edit existing applicant information or program details.
* View reports on the number of applications received, admissions approved, etc.
* Access all applicant accounts and update their application status.

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

* View different programs and courses available.
* View the status of their application.
* Create and manage an account in the admission system.
* View the list of documents they have submitted or need to submit.
* Request to change program or defer admission.
* Submit applications and upload required documents online.

## Functional Requirement

### **Student**

1. **Registration**: Students register using name, email ID, and mobile number.
2. **Login**: Students log in using email ID and password.
3. **View Profile**: Students can view their profile.
4. **Edit Profile**: Students can update their profile as needed.
5. **Fill Form**: Students fill out forms for any degree as per requirement.
6. **View Program Details**: Students view details of programs offered by the university.
7. **View Faculty Details**: Students view details about faculty members.
8. **Upload Documents**: Students upload documents like living certificates, proof of ID, and 12th mark sheets.
9. **View Fee Structure**: Students view fee structures for various programs.
10. **Fees Payment**: Students pay fees using cash, check, DD, or UPI.
11. **View Scholarship Details**: Students view scholarships provided by the government or university.
12. **Check Email**: Students check emails for updates.
13. **Collect Fee Receipt**: After payment, students collect a fee receipt.
14. **Apply for Scholarship**: Students apply for scholarships.
15. **View Education Loan Details**: Students view details about education loans.
16. **Apply for Transportation Service**: Students apply for transportation services, select routes and cities, and pay fees.
17. **Track Application Status**: Students can track the status of their admission applications.
18. **View Important Dates**: Students view critical dates related to admissions, payments, and other processes.
19. **Download Admission Form**: Students can download filled admission forms for their records.
20. **Provide Feedback**: Students submit feedback about the admission process.

### **Staff**

1. **Register**: Staff register using name, email ID, and password.
2. **Login**: Staff log in using email ID and password.
3. **View Student Details**: After students fill out forms, staff can view all details.
4. **Send Email**: Staff send emails to students about admission status.
5. **Collect Fees**: Staff collect student fees.
6. **Provide Fee Receipt**: Staff issue fee receipts after payment.
7. **Application Review and Approval**: Staff review and approve applications.
8. **Verify Documents**: Staff verify the uploaded documents submitted by students.
9. **Track Pending Applications**: Staff monitor and manage pending or incomplete applications.

### **Admin**

1. **Add Student**: This feature allows the admin to define and manage the requirements for student admissions
2. **Generate Reports**: Admin generates various reports (e.g., yearly admissions, scholarship applications, fee collection).
3. **Manage Staff Accounts**: Admin creates, updates, and deletes staff accounts.
4. **Audit Logs**: Admin monitors activity logs for transparency and security.
5. **Set Application Deadlines**: Admin defines and updates deadlines for applications, fee payments, and other key events.
6. **Monitor Program Popularity**: Admin analyzes the popularity of different programs based on student applications.
7. **Update Program Details**: Admin manages details of programs offered by the university.
8. **Handle Complaints**: Admin addresses complaints and issues raised by students or staff.
9. **Approve Scholarships**: Admin reviews and approves scholarship applications.
10. **Oversee Transportation Services**: Admin monitors transportation service usage and payments.

## 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

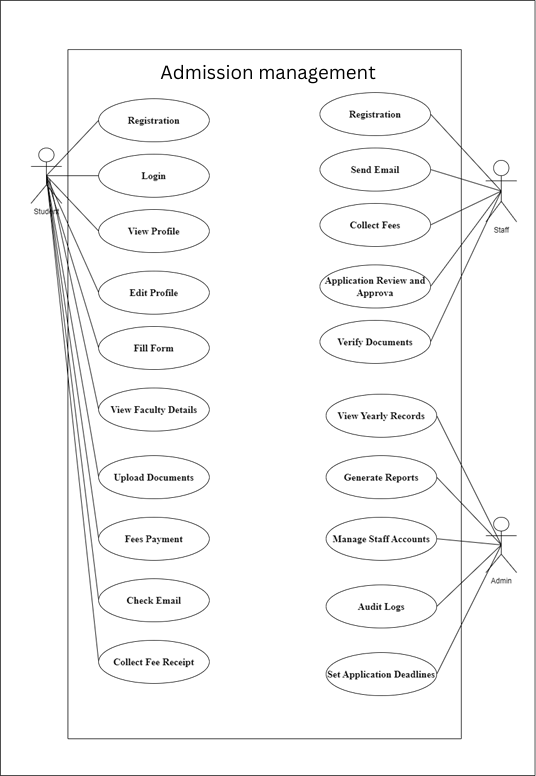


Figure 2.1‑1 Use case diagram for library management system

## Activity diagram and Swimlane diagram

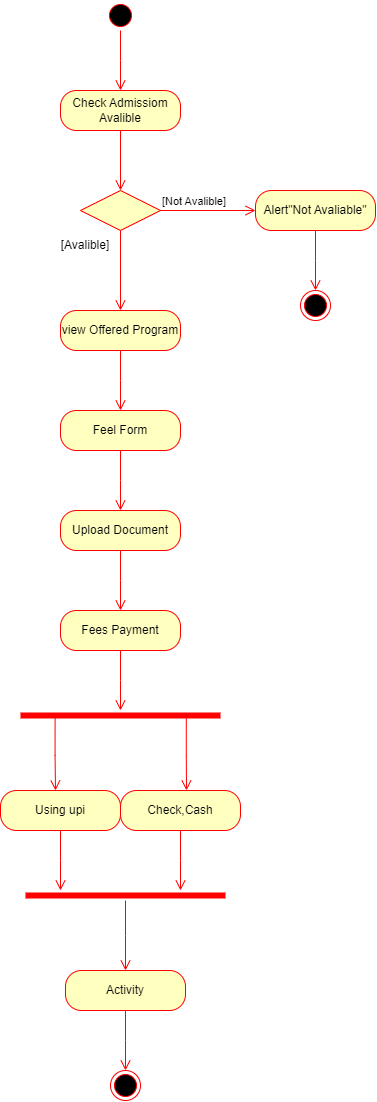


Figure 2.2‑1 Activity diagram of Admission Management(Student)

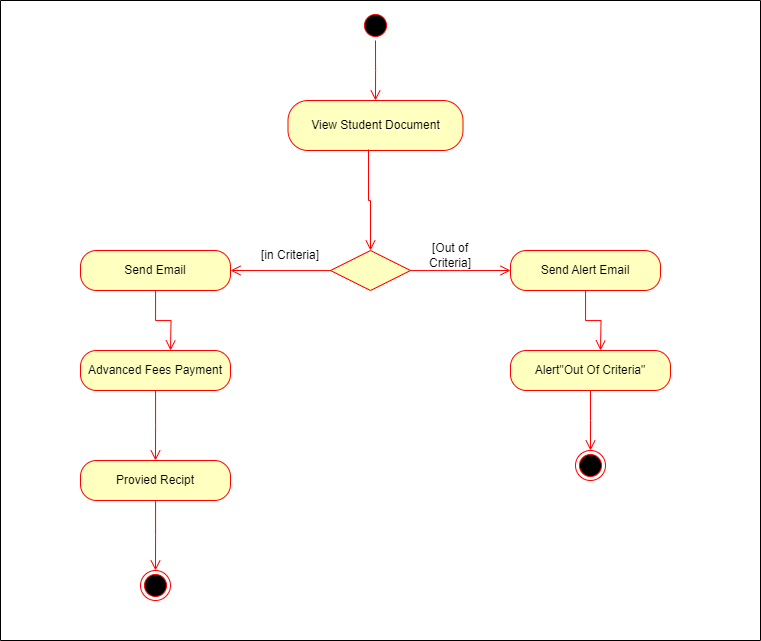


Figure 2.2‑2 Activity Diagram of Admission Management(Staff)

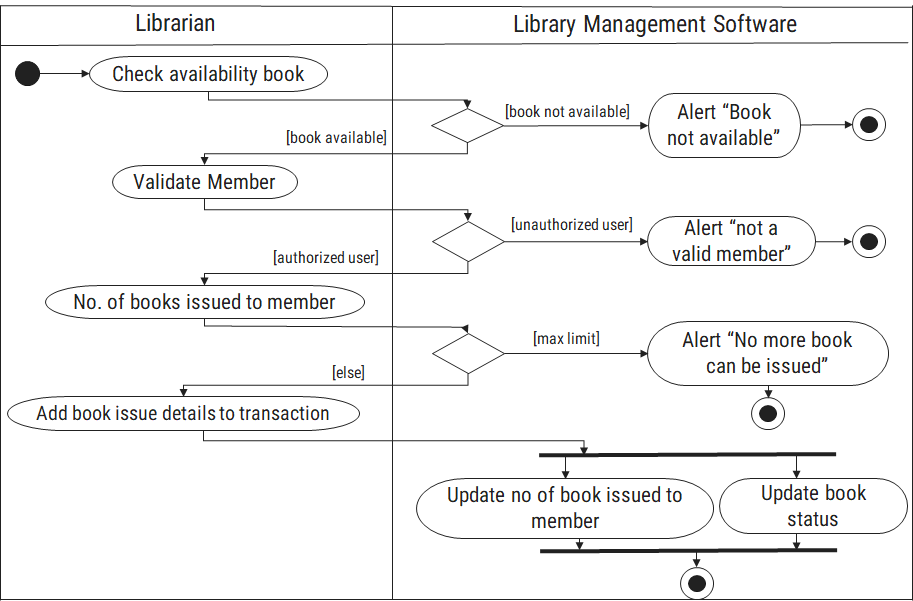
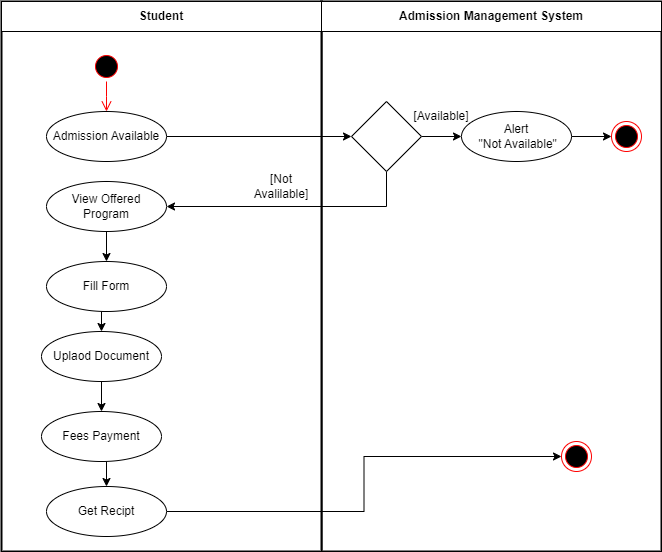
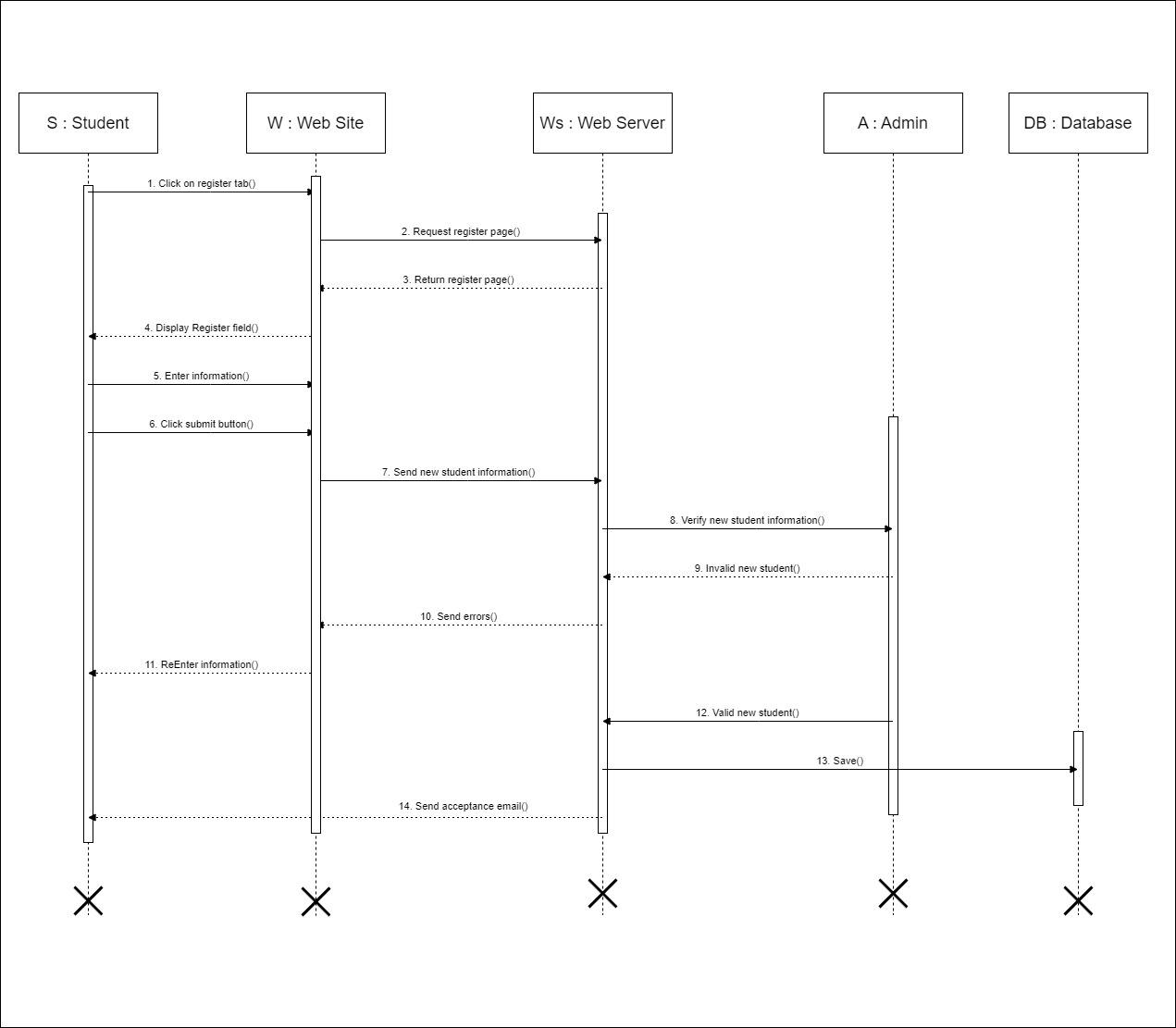
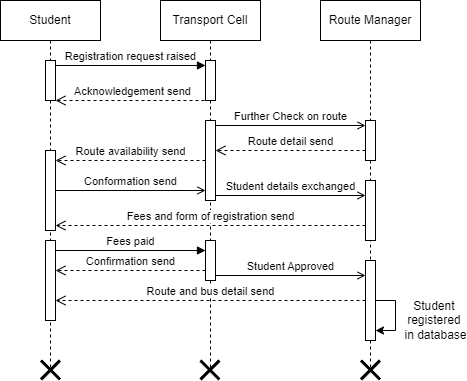


Figure 2.2‑3 Swimlane diagram for Admission Management

## Sequence diagram

 *Figure 2.3-1 Sequence diagram for Trasport Registration*



*Figure 2.3-2 Sequence diagram for Trasport Registration*

## State diagram

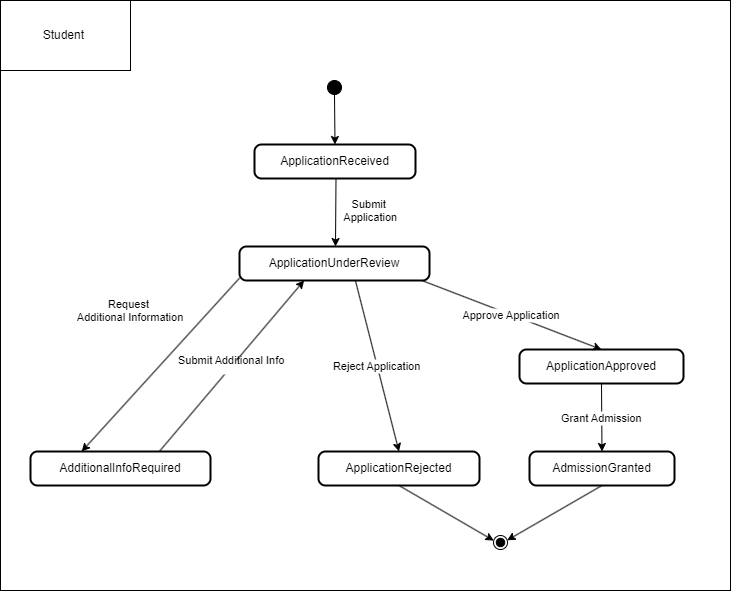


Figure 2.4‑1 State diagram of student

## Class diagram

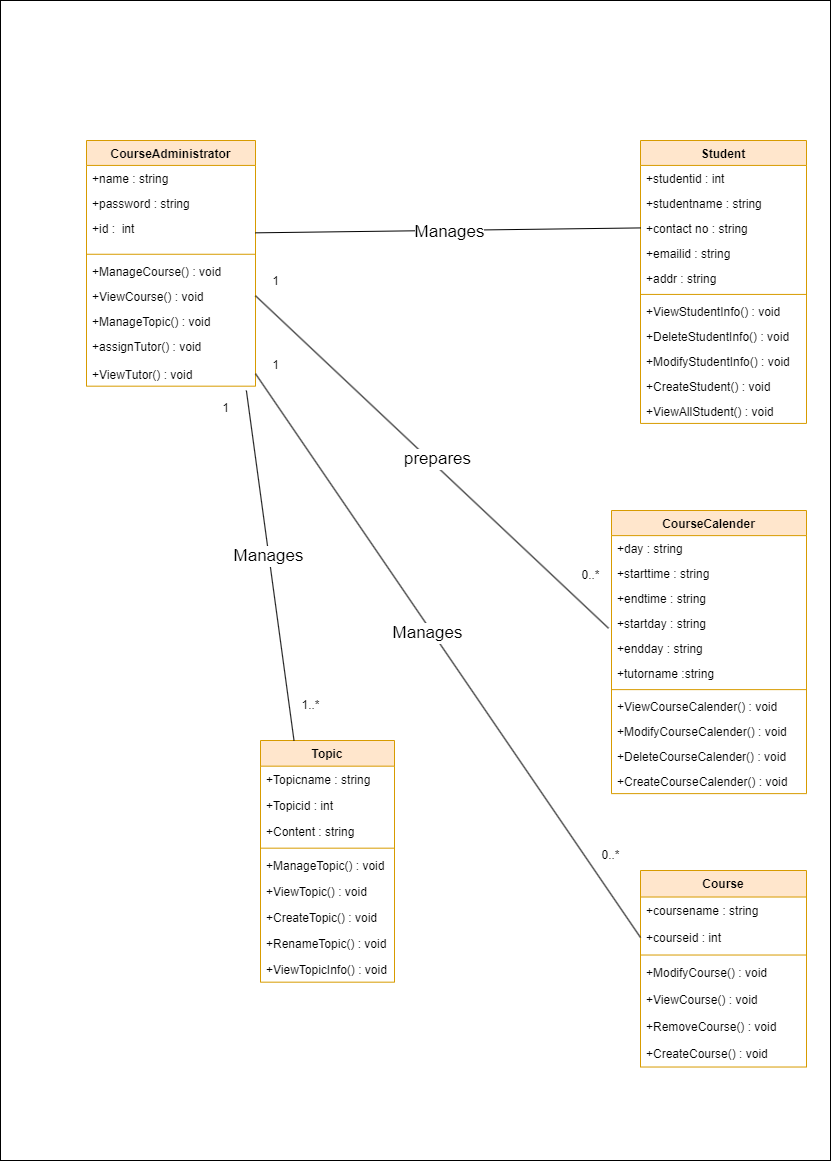


Figure 2.5‑1 Class diagram for Admission Management system

## Data flow diagram

### Context diagram (level-0)

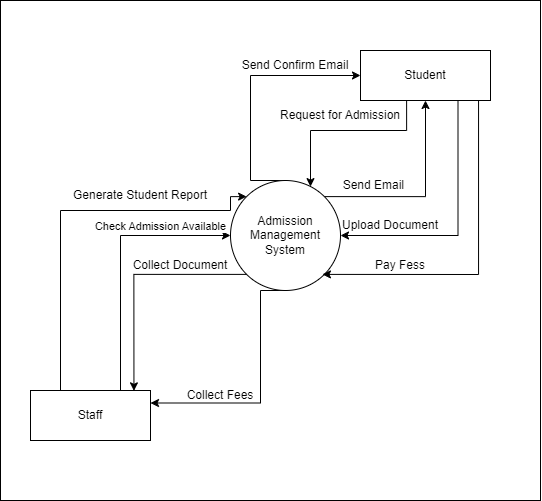


Figure 2.6‑1 Context diagram for Admission management system

### DFD Level-1

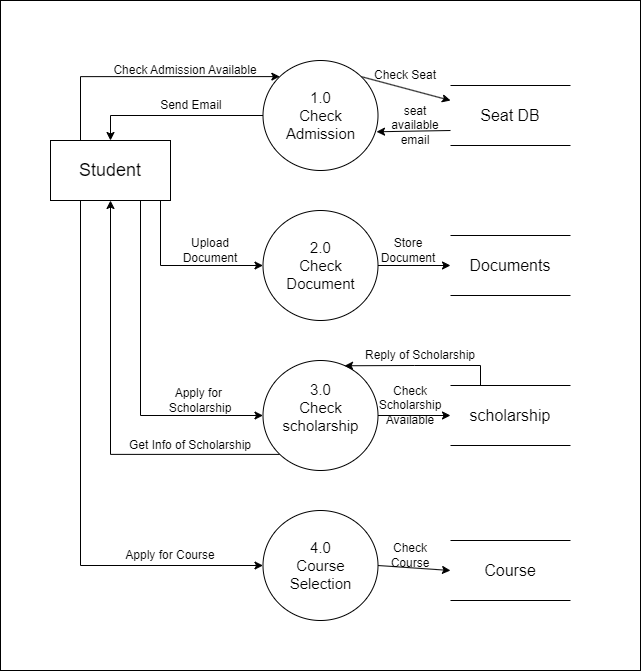


Figure 2.6‑2 DFD level-1 for Admission management system

# External interface requirement (Screens)

## Screen-1: Registration Form

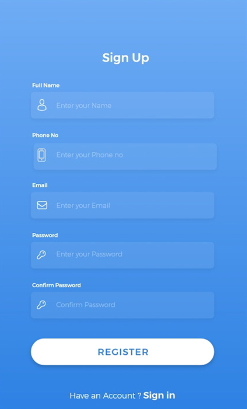


Figure 3.1‑1 Screen-1: Registration Form

**Purpose:** This form will allow the target end-users to register in the system. To register , the following information will be encoded in the system.

Table 3.1‑1 Screen element of Registration form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Full Name | Textbox | M | 1 | Full name field should be editable and accept the student full name. |
| 2 | Phone No | Textbox | M | 1 | Phone no field should be editable and accept student phone no. |
| 3 | Email | Textbox | M | 1 | Email field should be editable and accept the email with proper format. |
| 4 | Password | Password | M | 1 | Password field should be editable and accept the password and display as star or dot. |
| 5 | Confirm password | Password | M | 1 | Confirm Password field should be editable and accept the password and display as star or dot. |
| 6 | Register | Button | ------ | ------ | Register is a button for store the entered data into database. |
| 7 | Have a sign in | Link | ----- | ------ | If you are Successfully register then click in this link to redirect into the sign in or login page. |

## Screen-2: View Profile Form

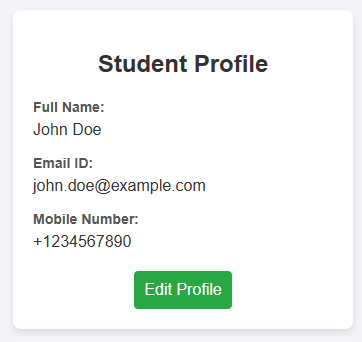


Figure 3.2‑1 Screen-2: view formForm

**Purpose:** This form will be used by the system’s users to access records and features of the system. The users will input the correct combination of their username and password to be able to login to the system.

Table 3.2‑1 Screen element of Login form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | Full Name | Textbox | M | 1 | Full name field should be editable and accept the student full name. |
| 2 | |  |  | | --- | --- | | |  | | --- | | Email ID | | | Textbox | M | 1 | Email field should be editable and accept a valid email in proper format |
| 3 | |  |  | | --- | --- | | |  | | --- | | Mobile Number | | | Textbox | M | 1 | Mobile number field should be editable and accept a valid phone number. |
| 4 | |  | | --- | |  |  |  | | --- | | Edit Profile | | Button | ------ | ------ | Edit Profile button should enable editing of the fields in the profile. |

## Screen-3: View Student details book

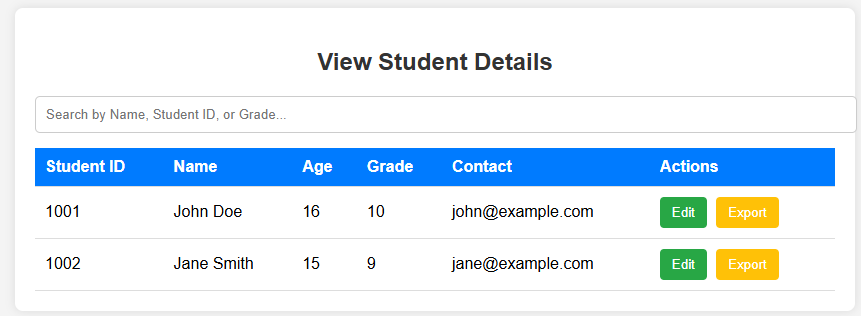


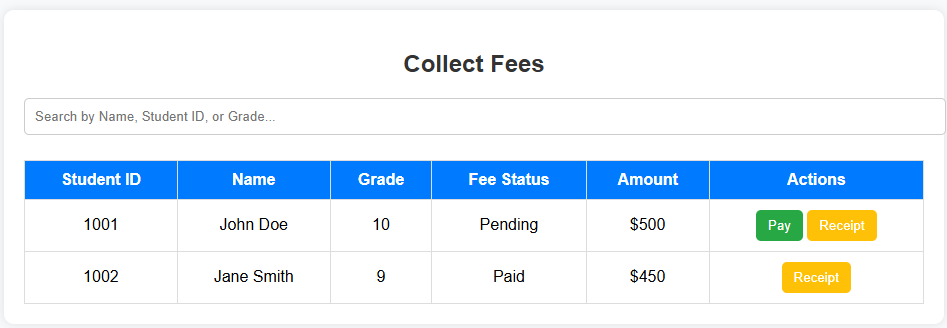
Figure 3.3‑1 Screen-3: View Student Details book

**Purpose:** This module will allow the system administrator to add, edit, update or delete borrowers of book. The admin can add borrower information and manage it.

Table 3.3‑1 Screen element of Add borrower book

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | **Search Box** | Textbox | O | N | Allows staff to search for students using Name, Student ID, or Grade. |
| 2 | **Student ID** | |  | | --- | |  |  |  | | --- | | Auto-generated | | M | 1 | Unique identifier for each student. |
| 3 | **Name** | Text | M | 1 | Displays the student's full name. |
| 4 | **Age** | Number | M | 1 | Displays the student’s age. |
| 5 | **Grade** | Dropdown | M | 1 | Displays the student’s grade level. |
| 6 | **Contact** | Email | M | 1 | |  | | --- | |  |  |  | | --- | | Shows the student’s registered email ID. | |
| 7 | **Edit Button** | Button (Green) | O | 1 | |  | | --- | |  |  |  | | --- | | Allows staff to edit student details. | |
| 8 | **Export Button** | Button (Yellow) | O | 1 | Allows staff to export student details. |

## Screen-4: collect fees Page



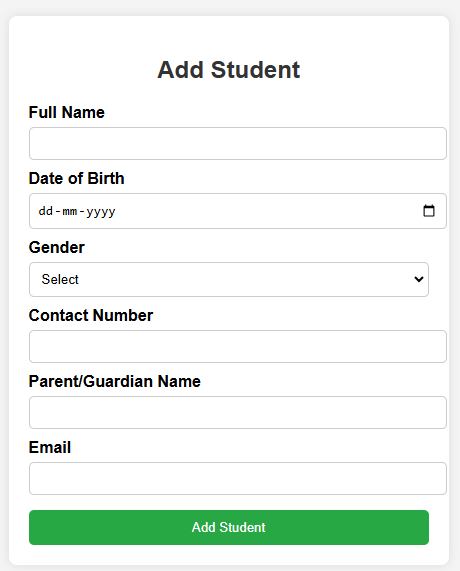
*Figure 3.4-1 Screen-4: collect fees page*

**Purpose:** This Page is Result of Student It Display student’s result of all exam they gave.

Table 3.4‑1 Screen element of Login form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.** | **Screen Element** | **Input Type** | **O/M** | **1/N** | **Description** |
| 1 | Search Bar | Textbox | O | 1 | Allows searching by Name, Student ID, or Grade. |
| 2 | Student ID | Label | M | N | Displays the unique ID of the student. |
| 3 | Name | Label | M | N | Displays the full name of the student. |
| 4 | Grade | Label | M | N | Displays the grade/class of the student. |
| 5 | Fee Status | Label | M | N | Displays whether the fee is Pending or Paid. |
| 6 | Amount | Label | M | N | Displays the fee amount for the student. |
| 7 | Pay Button | Button | O | 1 | Enables payment for students with a 'Pending' status. |
| 8 | Receipt Button | Button | M | 1 | Generates and displays the payment receipt. |

## Screen-5: Add student Page



*Figure 3.5-1 Screen-5: add student Page*

**Purpose:** This is Certificate request from that accept the which kind of certificate student and generate that certificate for student.

Table 3.5‑1 Screen element of Login form

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sr. | Screen Element | Input Type | O/M | 1/N | Description |
| 1 | |  | | --- | |  |  |  | | --- | | Full Name | | |  | | --- | | Textbox |  |  | | --- | |  | | M | 1 | |  | | --- | |  |  |  | | --- | | Full name field should be editable and accept the student's full name. | |
| 2 | Date of Birth | |  | | --- | |  |  |  |  | | --- | --- | | |  | | --- | | Date Picker | | | M | 1 | Allows selecting the student's date of birth in dd-mm-yyyy format. |
| 3 | Gender | Dropdown | M | 1 | Dropdown to select the student's gender (Male, Female, Other). |
| 4 | |  | | --- | |  |  |  | | --- | | Contact Number | | Textbox | M | 1 | Field should be editable and accept the student's contact number. |
| 5 | Parent/Guardian Name | Textbox | M | 1 | |  | | --- | |  |  |  | | --- | | Allows staff to search for students by name, student ID, or grade. | |
| 6 | Email | Textbox | M | 1 | Enables payment processing for students with pending fees. |
| 7 | |  | | --- | |  |  |  | | --- | | Add Student Button | | Button | ----- | ----- | Generates a payment receipt for the transaction. |

# Database design

## List of Tables

* Book
* Borrower
* Student
* Staff

Table 4.1‑1 Table: Book

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| BookID | int | NN | PK (Auto Increment) |  |
| BookName | varchar(100) | NN |  |  |
| ISBN | varchar(100) | AN |  |  |
| Publication\_year | int | AN |  |  |
| Language | varchar(50) | AN |  |  |

Table 4.1‑2 Table: Borrower

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| BorrowerID | int | NN | PK (Auto Increment) |  |
| BookID | varchar(100) | NN | FK | Reference of Book Table |
| BorrowedFromDate | DateTime | AN |  |  |
| BorrowedToDate | DateTime | AN |  |  |
| ActualReturnDate | DateTime | AN |  |  |
| IssuedBy | int | NN | FK | Reference of Student and Staff Table |

Table 4.1‑3 Table: Staff

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column | Data Type | Null | Keys & Constrains | Default Value & Description |
| StaffID | int | NN | PK (Auto Increment) |  |
| StaffName | varchar(100) | AN |  |  |
| IsAdmin | Boolean | AN |  |  |
| Designation | 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 |  |  |

# Stories and Scenario

## Story-1: Add New Book in Library Catalogue

|  |  |  |
| --- | --- | --- |
| *Story # S1* | : | As a Librarian,  I want to add a new book in library catalogue  So that everyone can easily find and borrow it. |
| Priority | **:** | High |
| Estimate | **:** | XL |
| Reason | **:** | The addition of a new book to the library catalogue is crucial for ensuring that the library's collection is up-to-date and accessible to everyone. |

### Scenario# S1.1

|  |  |  |
| --- | --- | --- |
| *Scenario# S1.1* | : | Adding a New Book with Valid Information |
| Prerequisite | **:** | Librarian is logged in to the Library management system. |
| Acceptance Criteria | **:** | **Given:**  The Librarian is navigated to the library catalog management page. Valid book information, including title, author, ISBN, and other relevant details is added.  **When:**  The librarian selects the "Add New Book" option  And The librarian enters valid book details  The librarian clicks the "Save" button to add the book to the catalog.  **Then t**he system successfully adds the book to the catalog and the librarian receives a confirmation message with the book's identification number. |

### Scenario# S1.2

|  |  |  |
| --- | --- | --- |
| *Scenario# S1.2* | : | Adding a New Book with Invalid Information. |
| Prerequisite | **:** | The librarian is logged into the library management system. |
| Acceptance Criteria | **:** | **Given:** The librarian is on the library catalogue management page  **When:** The librarian selects the "Add New Book" option and the librarian enters an incomplete or incorrect book details and librarian clicks the "Save" button to add the book to the catalogue.  **Then t**he system displays error messages for the incorrect or missing information and the book is not added to the catalogue. |

### Scenario# S1.3

|  |  |  |
| --- | --- | --- |
| *Scenario# S1.3* | : | Attempting to Add a Duplicate Book |
| Prerequisite | **:** | The librarian is logged into the library management system and the librarian is on the library catalogue management page |
| Acceptance Criteria | **:** | **Given**: The book information, including title, author, ISBN, and other relevant details, is available and the book with the same ISBN is already in the catalogue.  **When**: User Clicks on “Add book” button. Enter a number of copies with the same book detail mentioned in the field.  **Then**: Generate unique book id, barcode and spine label for various book of same title. |

## Story-2: View Profile Student in Admission Management

|  |  |  |
| --- | --- | --- |
| *Story # A2* | : | **As an** Admission Officer, **I want to** view a student's profile in the admission management system **So that** I can access their details and verify or update information as needed |
| Priority | **:** | High |
| Estimate | **:** | L |
| Reason | **:** | Accessing a student's profile is crucial for reviewing admission details, updating records, and ensuring accurate information is maintained. |

### Scenario# A2.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* A1.1 | : | Viewing a Student Profile with Valid Information |
| Prerequisite | **:** | The Admission Officer is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given**: The Admission Officer navigates to the student profile management page and searches for a student using a valid identifier (e.g., name, student ID).  **When**:  The Admission Officer enters the student's details in the search bar.  The system retrieves and displays the student's profile with complete information (e.g., name, age, grade, contact details).  **Then**: The Admission Officer successfully views the student's profile with all details displayed correctly. |

### Scenario# A2.2

|  |  |  |
| --- | --- | --- |
| *Scenario# A1.2* | : | Attempting to View a Non-Existent Student Profile |
| Prerequisite | **:** | The Admission Officer is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given**: The Admission Officer navigates to the student profile management page and searches for a student using an incorrect or non-existent identifier.  **When:**  The Admission Officer enters invalid student details in the search bar.  The system attempts to find the student but does not retrieve any records.  **Then**: The system displays an error message stating that the student profile does not exist, prompting the Admission Officer to check the entered details or add a new student |

## Story-3: Fees Payment in Admission Management

|  |  |  |
| --- | --- | --- |
| *Story # A3* | : | As an Admission Officer,  I want to process and manage student fee payment So that student payment records are updated, and they can proceed with the admission process. |
| Priority | **:** | High |
| Estimate | **:** | XL |
| Reason | **:** | Ensuring that student fees are correctly recorded is crucial for managing the admission process and preventing discrepancies in financial records. |

### Scenario# A3.1

|  |  |  |
| --- | --- | --- |
| *Scenario#* A1.1 | : | Processing Fee Payment Successfully |
| Prerequisite | **:** | Admission Officer is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The Admission Officer is on the fee payment page of the admission management system.  **When:**   * The Admission Officer selects the student and navigates to the fee payment section. * The Admission Officer enters the correct fee amount and payment details. * The Admission Officer clicks the "Submit Payment" button.   **Then:**   * The system successfully records the payment. * The Admission Officer receives a confirmation message with the transaction ID. * The student’s fee status is updated to "Paid." |

### Scenario# A3.2

|  |  |  |
| --- | --- | --- |
| *Scenario# A1.2* | : | Handling Payment Failure or Incomplete Payment |
| Prerequisite | **:** | Admission Officer is logged into the admission management system. |
| Acceptance Criteria | **:** | **Given:** The Admission Officer is on the fee payment page of the admission management system.  **When:**   * The Admission Officer attempts to process a payment but enters incomplete or incorrect payment details. * The Admission Officer clicks the "Submit Payment" button.   **Then:**   * The system displays an error message prompting the officer to check the details. * The payment is not processed, and the student’s fee status remains "Unpaid." * The Admission Officer can retry the payment after correcting the errors.   does not exist, prompting the Admission Officer to check the entered details or add a new student |

# Test cases

**Registration test case**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Name: | Admission mangement | Test Designed by: | R.C.Gambhava |
| Module Name: | **Login** | **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 | Verify accessibility of Admission Management System dashboard | Functional | Verify that the dashboard of the Admission Management System is accessible | TC\_001 |
| TC\_002 | Verify student application form accessibility and elements | GUI | Verify the availability and accessibility of all elements in the student form | TC\_002 |
| TC\_003 | Submit student application with valid data | Functional | Verify that the student application form can be submitted with valid data and that the system handles the submission correctly | TC\_003 |

|  |  |
| --- | --- |
| **Test Case Title** | Verify accessibility of Admission Management System dashboard |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | Web application should be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Access Admission Management System URL | The site should launch properly | Site launched successfully | Pass |  | https://admission.system.com/dashboard |  |
| 2 | Navigate to dashboard | User should be able to access the dashboard without login | Dashboard accessed without login | Pass |  | Username:  abc@gmail.com |  |
| 3 | Verify the loading time of the dashboard | Dashboard should load within 5 seconds | Dashboard loaded within 5 seconds | pass |  | Password: abcpassword |  |
| 4 | Verify that the dashboard is responsive on different devices (mobile, tablet, desktop) | Dashboard should be responsive and display correctly on all devices | Dashboard displayed correctly on all devices | Pass | Tested on mobile and tablet |  |  |
| 5 | Verify all dashboard elements (charts, menus, navigation links) are loaded correctly | All elements should load and be functional without errors | All elements loaded correctly | pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Verify student application form accessibility and elements |
| **Test Type** | GUI |
| **Test Priority** | HIGH |
| **Pre-condition** | The web application must be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Access the student application form from the menu | The form should open and be accessible | Form opened successfully | Pass |  |  |  |
| 2 | Verify all form fields (name, date of birth, address, etc.) are available | All fields should be displayed and editable | All fields displayed correctly | Pass |  |  |  |
| 3 | Verify "Submit" button exists | "Submit" button should be present | Submit button visible | Pass |  |  |  |
| 4 | Verify form field validation messages (if any) | Appropriate validation messages should appear for required fields if left empty | Validation messages appeared correctly | Pass | Tested by leaving fields empty |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Submit student application with valid data |
| **Test Type** | Functional |
| **Test Priority** | HIGH |
| **Pre-condition** | Web application and student form must be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Enter valid student data in all required fields (name, address, etc.) | Form should accept the valid data | Form accepted all inputs | Pass |  |  |  |
| 2 | Verify that the submitted data is saved correctly in the database | Data should be stored in the database accurately | Data saved correctly in the database | Pass | Verified by checking database entries |  |  |
| 3 | Verify that a confirmation email is sent to the applicant upon successful form submission | Applicant should receive a confirmation email | Confirmation email received successfully | Pass |  |  |  |
| 4 | Click on the "Submit" button | Data should be submitted, and confirmation message should appear | Data submitted successfully with confirmation message | Pass |  |  |  |

**View profile test case**

|  |  |  |  |
| --- | --- | --- | --- |
| Project Name: | Admission mangement | Test Designed by: | R.C.Gambhava |
| Module Name: | **View** **Profile** | **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 | Verify accessibility of View Student Profile page | Functional | Verify that the View Student Profile page is accessible. | TC\_001 |
| TC\_002 | Verify display of student profile details | GUI | Ensure that all student profile details (Name, ID, Grade, etc.) are visible and properly formatted. | TC\_002 |
| TC\_003 | Validate edit functionality in student profile | Functional | Check that users can edit student details where allowed and that changes are saved correctly. | TC\_003 |

|  |  |
| --- | --- |
| **Test Case Title** | Verify accessibility of Admission Management System dashboard |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | Web application should be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Access Student Profile URL | The site should launch properly | Site launched successfully | Pass |  | https://admission.system.com/profile |  |
| 2 | Navigate to student profile | User should be able to access the student profile page | Profile page accessed successfully | Pass |  | Username: abc@gmail.com |  |
| 3 | Verify the loading time of the profile page | Profile page should load within 5 seconds | Profile page loaded within 5 seconds | Pass |  | Password: abcpassword |  |
| 4 | Verify profile responsiveness on different devices (mobile, tablet, desktop) | Profile page should be responsive and display correctly on all devices | Profile displayed correctly on all devices | Pass | Tested on mobile and tablet |  |  |
| 5 | Verify that all student profile elements (photo, name, ID, grade, etc.) load correctly | All elements should load and be functional without errors | All elements loaded correctly | Pass |  |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Verify student application form accessibility and elements |
| **Test Type** | GUI |
| **Test Priority** | HIGH |
| **Pre-condition** | The web application must be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **Bug ID** |
| 1 | Access the student application form from the menu | The form should open and be accessible | Form opened successfully | Pass |  |  |  |
| 2 | Verify all form fields (name, date of birth, address, etc.) are available | All fields should be displayed and editable | All fields displayed correctly | Pass |  |  |  |
| 3 | Verify "Submit" button exists | "Submit" button should be present | Submit button visible | Pass |  |  |  |
| 4 | Verify form field validation messages (if any) | Appropriate validation messages should appear for required fields if left empty | Validation messages appeared correctly | Pass | Tested by leaving fields empty |  |  |

|  |  |
| --- | --- |
| **Test Case Title** | Verify accessibility of View Student Profile Page |
| **Test Type** | Functional |
| **Test Priority** | High |
| **Pre-condition** | Web application should be accessible |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Step** | **Test Case Description** | **Expected Result** | **Actual Result** | **Status** | **Comment** | **Data** | **BUG ID** |
| 1 | Access Student Profile URL | The site should launch properly | Site launched successfully | Pass |  | <https://admission.sy>  stem.com/profile |  |
| 2 | Navigate to student profile | User should be able to access the student profile page | Profile page accessed successfully | Pass |  | Username: abc@gmail.com |  |
| 3 | Verify the loading time of the profile page | Profile page should load within 5 seconds | Profile page loaded within 5 seconds | Pass |  | Password: abcpassword |  |
| 4 | Verify profile responsiveness on different devices (mobile, tablet, desktop) | Profile page should be responsive and display correctly on all devices | Profile displayed correctly on all devices | Pass | Tested on mobile and tablet |  |  |
| 5 | Verify that all student profile elements (photo, name, ID, grade, etc.) load correctly | All elements should load and be functional without errors | All elements loaded correctly | Pass |  |  |  |
| 6 | Validate edit functionality in student profile | Users should be able to edit student details where allowed and changes should be saved correctly | Changes saved successfully | Pass |  |  | TC\_003 |

# References

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