# Software Requirement Specifications

## Flex Student Portal Project

**Version:** 1.0

**Project Code:** —

**Supervisor:** Ms. Fizza Mansoor

**Co-Supervisor:** —

**Project Team:** 22k5024 Hadi  
 22k5018 Aheed  
 22k5144 Lucky

**Submission Date:** 4/28/2025

# Document History

| **Version** | **Name of Person** | **Date** | **Description of change** |
| --- | --- | --- | --- |
| 1.0 | Hadi | 12.5.2024 | Document Created |

# Distribution List

| **Name** | **Role** |
| --- | --- |
| Ms.Fizza Mansoor | Supervisor |
| Co-Supervisor | Co-Supervisor |

# Document Sign-Off

| **Version** | **Sign-off Authority** | **Sign-off Date** |
| --- | --- | --- |
| 1.0 | Supervisor | — |

# Table of Contents

1. Introduction  
    1.1 Purpose of Document  
    1.2 Intended Audience  
    1.3 Abbreviations  
    1.4 Document Convention
2. Overall System Description  
    2.1 Project Background  
    2.2 Project Scope  
    2.3 Not In Scope  
    2.4 Project Objectives  
    2.5 Stakeholders  
    2.6 Operating Environment  
    2.7 System Constraints  
    2.8 Assumptions & Dependencies
3. External Interface Requirements  
    3.1 Hardware Interfaces  
    3.2 Software Interfaces  
    3.3 Communications Interfaces
4. Functional Requirements  
    4.1 Functional Hierarchy  
    4.2 Use Cases
5. Non-functional Requirements  
    5.1 Performance Requirements  
    5.2 Safety Requirements  
    5.3 Security Requirements  
    5.4 User Documentation
6. References
7. Appendices

# 1. Introduction

## 1.1 Purpose of Document

The purpose of this document is to outline the functional and nonfunctional requirements of the Student Portal Website. It serves as a guide for developers, stakeholders, and project managers to ensure the product meets its intended goals.

## 1.2 Intended Audience

* Developers: To understand technical and functional requirements.
* Project Managers: For planning and resource allocation.
* Users (Students and Teachers): To comprehend the intended functionality.
* Documentation Writers: For user manuals and guides.

## 1.3 Abbreviations

* API: Application Programming Interface
* CRUD: Create, Read, Update, Delete
* DBMS: Database Management System
* HTTP: HyperText Transfer Protocol
* SQL: Structured Query Language
* GDPR: General Data Protection Regulation
* FERPA: Family Educational Rights and Privacy Act

## 1.4 Document Convention

* Headings: Arial, Bold, 14 pt
* Body Text: Cambria, Regular, 12 pt

# 2. Overall System Description

## 2.1 Project Background

The Student Portal Website is a new, self-contained product designed to support academic interactions within an institution. It provides attendance tracking, academic marks distribution, and timetable management.

## 2.2 Project Scope

The portal provides login modules for students and teachers. It includes managing student profiles, attendance, marks, transcripts, feedback, challan generation, and timetable management.

## 2.3 Not In Scope

Administrative functionalities are not included in the current version; they are reserved for a future release.

## 2.4 Project Objectives

* Improve accessibility to academic resources.
* Provide secure data management.
* Enhance student performance tracking.
* Support digital education goals of the institution.

## 2.5 Stakeholders

* Students
* Teachers
* Developers
* Project Managers
* Institutional IT Department

## 2.6 Operating Environment

* Client: Windows 10+, macOS 11+, Android 10+, iOS 13+
* Server: Node.js with Express.js
* Database: PostgreSQL
* Browser Compatibility: Chrome 90+, Firefox 80+, Edge 91+, Safari 14+

## 2.7 System Constraints

* Must use ReactJS for frontend and Node.js with Express for backend.
* PostgreSQL as the database.
* Integration with institutional services via APIs.

## 2.8 Assumptions & Dependencies

* Assumes reliable internet access.
* Depends on availability of ReactJS, Node.js libraries, and institutional IT infrastructure.

# 

# 3. External Interface Requirements

## 3.1 Hardware Interfaces

* Desktops, laptops, tablets, and smartphones with a minimum resolution of 1024x768 pixels.
* Input devices: Standard keyboards, touchscreens, and pointing devices like mice.

## 3.2 Software Interfaces

* Frontend: ReactJS
* Backend: Node.js with Express.js
* Database: PostgreSQL (pgAdmin 14)
* Libraries: Redux, bcrypt, multer

## 3.3 Communications Interfaces

* Communication via HTTPS protocol (SSL/TLS encryption).
* Local Host Communication: Using GET and POST methods for database interaction.

# 

# 

# 

# 

# 4. Functional Requirements

## 4.1 Functional Hierarchy

* Student & Teacher Login
* View and Manage Attendance
* View and Manage Marks
* View Transcript
* Course Registration and Management
* Challan Management
* Feedback Collection
* Timetable Management
* Change Password
* Profile Management

## 

## 4.2 Use Cases

## Use Case: Login

**Use Case ID:** UC-01  
 **Actors:** Student, Teacher, Backend  
 **Feature:** User Authentication

**Pre-condition:** User credentials must be registered in the system.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | User enters username and password | System verifies credentials against the database |
| 2 | Credentials are valid | User is redirected to their respective dashboard |

**Alternate Scenarios**

* 1a. Invalid credentials → System shows an error message "Invalid login credentials."

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | User gains access to personalized functionalities |

**Use Case Cross Referenced:** Change Password, View Attendance, View Marks, Withdraw from Course

## Use Case: Change Password

**Use Case ID:** UC-02  
 **Actors:** Student  
 **Feature:** Password Management

**Pre-condition:** User must be logged into the portal.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Select "Change Password" option | System prompts for current and new passwords |
| 2 | Enter current and new password | System verifies current password and updates the new one |

**Alternate Scenarios**

* 2a. Incorrect current password → System denies password update.

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Password is updated successfully |

**Use Case Cross Referenced:** Login

## Use Case: View Attendance

**Use Case ID:** UC-03  
 **Actors:** Student  
 **Feature:** Attendance Management

**Pre-condition:** Attendance data must exist for the student.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Select "View Attendance" option | System fetches and displays attendance records |

**Alternate Scenarios**

* 1a. No attendance data found → System displays "No attendance records available."

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Attendance details are displayed |

**Use Case Cross Referenced:** Login

## Use Case: View Marks

**Use Case ID:** UC-04  
 **Actors:** Student  
 **Feature:** Marks Distribution

**Pre-condition:** Teachers must have uploaded marks to the system.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Select "View Marks" option | System retrieves and displays marks for courses |

**Alternate Scenarios**

* 1a. Marks not uploaded yet → System displays "Marks not available."

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Marks are displayed to the student |

**Use Case Cross Referenced:** Login

## Use Case: Withdraw from Course

**Use Case ID:** UC-05  
 **Actors:** Student  
 **Feature:** Course Management

**Pre-condition:** Students must be enrolled in courses.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Select course to withdraw | System prompts for confirmation |
| 2 | Confirm withdrawal | System updates student’s course enrollment status |

**Alternate Scenarios**

* 2a. Withdrawal deadline passed → System denies withdrawal request.

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Student is unenrolled from the course |

**Use Case Cross Referenced:** Course Registration

## Use Case: View Challan

**Use Case ID:** UC-06  
 **Actors:** Student  
 **Feature:** Challan Management

**Pre-condition:** Fee challan details must be generated.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Navigate to "View Challan" | System retrieves and displays fee challan details |

**Alternate Scenarios**

* 1a. No challan available → System shows "No challan generated."

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Challan details are displayed and available for download |

**Use Case Cross Referenced:** None

## Use Case: Manage Students

**Use Case ID:** UC-07  
 **Actors:** Teacher  
 **Feature:** Student Management

**Pre-condition:** Teacher must have access to course data.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Navigate to "Manage Students" | System displays student list |
| 2 | Add, update, or remove student details | System updates database accordingly |

**Alternate Scenarios**

* 2a. Unauthorized action → System denies access.

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Student information is updated |

**Use Case Cross Referenced:** Manage Attendance

## Use Case: Manage Attendance

**Use Case ID:** UC-08  
 **Actors:** Teacher  
 **Feature:** Attendance Management

**Pre-condition:** Teacher must have assigned classes.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Navigate to "Manage Attendance" | System displays list of students |
| 2 | Update attendance records | System saves updated attendance |

**Alternate Scenarios**

* 2a. Connection lost → System retries saving attendance once connection is restored.

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Attendance records are saved and updated |

**Use Case Cross Referenced:** Manage Students

## Use Case: Generate Marks Report

**Use Case ID:** UC-09  
 **Actors:** Teacher  
 **Feature:** Marks Distribution

**Pre-condition:** Marks must already be entered.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Select "Generate Marks Report" | System compiles marks data into a report |

**Alternate Scenarios**

* 1a. No marks entered → System shows error "No marks available to generate report."

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Marks report is generated and downloadable |

**Use Case Cross Referenced:** View Marks

## Use Case: Register Student

**Use Case ID:** UC-10  
 **Actors:** Backend  
 **Feature:** Student Registration

**Pre-condition:** Student details must be provided.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Enter student details | System assigns ID and creates credentials |

**Alternate Scenarios**

* 1a. Duplicate ID detected → System rejects registration.

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Student is registered successfully |

**Use Case Cross Referenced:** Login

## Use Case: Manage Feedback

**Use Case ID:** UC-11  
 **Actors:** Student, Teacher  
 **Feature:** Feedback Management

**Pre-condition:** Feedback functionality must be active.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Submit feedback (student) | System saves feedback |
| 2 | Review feedback (teacher) | System displays feedback received |

**Alternate Scenarios**

* 1a. Feedback submission fails → System prompts to retry.

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Feedback is stored and accessible for analysis |

**Use Case Cross Referenced:** None

## Use Case: Show Timetable

**Use Case ID:** UC-12  
 **Actors:** Student  
 **Feature:** Timetable Management

**Pre-condition:** Timetable data must exist.

**Scenarios**

| **Step#** | **Action** | **Software Reaction** |
| --- | --- | --- |
| 1 | Select "Show Timetable" | System retrieves and displays personalized timetable |

**Alternate Scenarios**

* 1a. No timetable data available → System shows "No timetable found."

**Post Conditions**

| **Step#** | **Description** |
| --- | --- |
| 1 | Timetable is displayed successfully |

**Use Case Cross Referenced:** None

# 

# 5. Non-functional Requirements

## 5.1 Performance Requirements

* Response time: ≤ 2 seconds under normal load.
* Database to handle up to 10,000 concurrent users.
* Uptime: 99.9% over a rolling 30-day period.

## 5.2 Safety Requirements

* Regular database backups every 12 hours.
* Prevention of invalid data submissions.
* Secure handling of sensitive data.

## 5.3 Security Requirements

* All communication must use SSL/TLS.
* Passwords hashed using bcrypt.
* Role-based access control (RBAC) implemented.
* User session timeout after 1 hour of inactivity.

## 5.4 User Documentation

* Video tutorials embedded in the portal.
* Help page for FAQs and troubleshooting.

# 6. References

* ReactJS, Node.js, PostgreSQL official documentation.

# 7. Appendices

## Error Handling Terms:

1. Error 500 (Server Error): Indicates an internal server issue.
2. Error 400 (Bad Request): Indicates an issue with the client request, such as missing or incorrect data.
3. ON CONFLICT: A PostgreSQL clause to handle duplicate key violations during INSERT queries.
4. EXCLUDED: A PostgreSQL keyword used in ON CONFLICT clauses to reference the conflicting rows.

## Entity Relationship Diagram:

## Use Case Diagram:

