**Student Grading System**

Use Case (Requirement) Document

9th Jan 2025

Ver0.0Draft1

Document Identification

| Project Code |  |
| --- | --- |
| Project Name | Student Grading System |
| Document Id |  |
| Project Start Date | 09-01-2025 |
| Project End Date | 10-01-2025 |
| Project Type |  |
| Project Manager | Mr. Nilesh Kumar |
| Project Manager Cell No | +91-9015369467 |

Client Detail

| Client Name |  |
| --- | --- |
| Contact Person |  |
| Contact Person Email Id |  |
| Contact Person Cell No |  |

Version History

| Version | Date | Author(s) | Comment | Reviewed By |
| --- | --- | --- | --- | --- |
| Ver0.0Draft1 | 9th Jan, 2025 | Saday Samnotra | Initially Draft. |  |

Contents

[1 Introduction 4](#_Toc187353663)

[1.0 Purpose 4](#_Toc187353664)

[1.1 Scope 4](#_Toc187353665)

[1.2 Target Audience 4](#_Toc187353666)

[1.3 Document Management 4](#_Toc187353667)

[2 Overview 5](#_Toc187353668)

[2.0 Constraints 5](#_Toc187353669)

[2.1 Technologies 5](#_Toc187353670)

[2.2 Exclusions 5](#_Toc187353671)

[3 Use Cases 6](#_Toc187353672)

[3.0 High Level Use Case Association 6](#_Toc187353673)

[3.0.1 Super Admin 6](#_Toc187353674)

[3.0.2 Professor 7](#_Toc187353675)

[3.0.3 Student 7](#_Toc187353676)

[3.0.2 UC-002: Add User 10](#_Toc187353677)

[3.0.3 UC-003: Update User 12](#_Toc187353678)

[3.1.4 UC-004: Delete User 14](#_Toc187353679)

[4 Processing Details 16](#_Toc187353680)

[4.1 PD01: Employee Management System 16](#_Toc187353681)

[4.1.1 PD01A: Add User 16](#_Toc187353682)

[4.1.2 PD01A: Update User 16](#_Toc187353683)

# Introduction

## Purpose

The purpose of this document is to provide insight into the use cases related to the “Student Grading System”, which will be the baseline document for design, development and testing of the application.

## Scope

This document covers the business use case of “Student Grading System Phase-1.0” and assumption which is used at the time of designs & development of project. In “Student Grading System Phase-1.0”, following are new requirements/modules to be implemented:

1. Functionality to view a user details.
2. Functionality to add a user or multiple users.
3. Functionality to delete a user or multiple users.
4. Functionality to update a user or multiple users.

## Target Audience

The target audience for the Student Grading System includes educational institutions, schools, colleges, universities, and training centres seeking to simplify and enhance their grading processes, academic evaluations, and overall student performance management.

## Document Management

The requirements in this use case document shall be traced to the appropriate deliverables in the development and testing phases to ensure that all requirements are properly implemented and tested.

# Overview

The Objective of Student Grading System Phase-1.0 is:

**For Super Admin:**

1. Add, view, update, and delete users (Students and Professors).
2. Assign professors to students.
3. Create and manage other Super Admin accounts.

**For Professor:**

1. Register, login, and access the dashboard.
2. Update marks for assigned students and provide feedback.

**For Student:**

1. Self-register and log in.
2. View their own marks via the dashboard.

Assumptions

1. There will be only one admin.
2. Each user will have a unique identifier, that is a username.

## Constraints

* Java 17 will be used as the baseline version for development.
* This system does not support multi-language interface.

## Technologies

**Languages & Tool:** Node JS v16.13.2

**Operating Systems/Platform:** Windows 11, Google Chrome

**Device Databases:** PostgreSQL

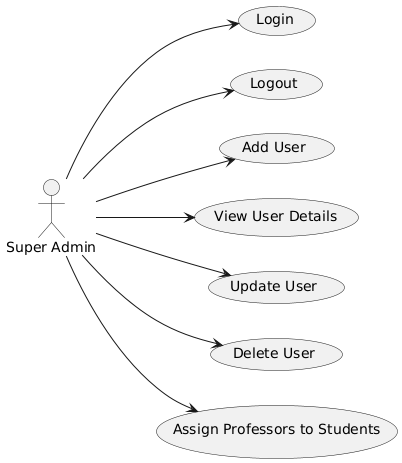
## Exclusions

* The execution of the project would be as stated in the above sections. Any other aspect or deviation introduced during the project development will be considered as Out of Scope or Change request and would call for impact analysis in terms of added timeframe and additional charges.

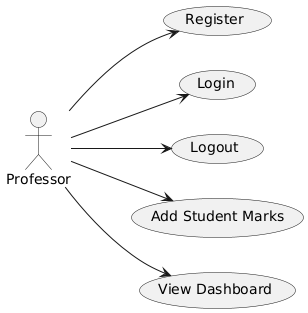
# Use Cases

## High Level Use Case Association

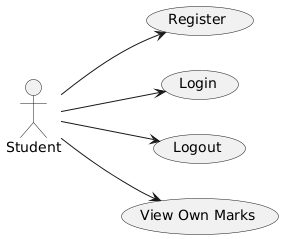
### Super Admin



### Professor



### Student



## High Level Use Case Association: Super Admin

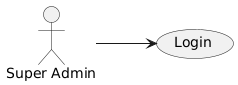
### UC-001: Login

**Purpose**

* + - This use case describes how the Super Admin can log in and access the dashboard of the Student Grading System.

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Student Grading System must be running and accessible.
    - The initial Super Admin must be setup and have valid login credentials.

**Post-Conditions**

* + - The Super Admin is successfully logged in and redirected to their dashboard.
    - The dashboard displays options for managing users and performing administrative tasks.

**Basic Flow**

* + - The Super Admin accesses the Student Grading System login page.
    - The system prompts the Super Admin to enter their username and password.
    - The Super Admin enters valid credentials and clicks the Login button.
    - The system authenticates the credentials:
    - If valid, the Super Admin is redirected to the dashboard.
    - The dashboard displays options to manage users, assign professors to students, and view system details.

**Failure Case**

* Invalid Credentials:

If the username or password is incorrect, the system displays an error message: *"Invalid username or password. Please try again."*

* Connection Issue:

If the system is down or inaccessible, an error message is displayed: *"Unable to connect to the system. Please try again later."*

* Account Locked:

If too many failed login attempts occur, the system locks the account and displays: *"Your account has been locked due to multiple failed attempts. Contact support."*

* Alternative Flow

If the Super Admin forgets their password, they can click the Forgot Password link to reset it.

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* employees\_master

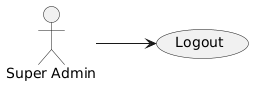
### UC-002: Logout

**Purpose**

* + - This use case describes how the Super Admin can log out of the Student Grading System to ensure session security.

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Super Admin must be logged in to the system.

**Post-Conditions**

* + - The Super Admin is successfully logged out of the system.
    - The system terminates the Super Admin’s session, and they are redirected to the login page.

**Basic Flow**

* + - The Super Admin is logged into the system and views the dashboard.
    - The Super Admin clicks the Logout button, usually located in the top-right corner of the dashboard.
    - The system verifies the request and terminates the session.
    - The system logs out the Super Admin and redirects them to the login page.
    - A confirmation message may be displayed: *"You have successfully logged out."*

**Failure Case**

* **Logout Failed**:
  + If there is an error during the logout process (e.g., session timeout, database issue), the system displays: *"An error occurred while logging out. Please try again later."*
* **Session Expired**:
  + If the session expires due to inactivity, the system automatically logs the Super Admin out and redirects them to the login page with a message: *"Your session has expired. Please log in again."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* employees\_master

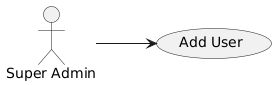
### UC-003: Add User

**Purpose**

* + - This use case describes how the Super Admin can add a new user (Student, Professor, or Super Admin) to the system.

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Super Admin must be logged in.

**Post-Conditions**

* + - A new user (Student, Professor, or Super Admin) is successfully added to the system.

**Basic Flow**

* + - Super Admin accesses the "Add User" page from the dashboard.
    - The system prompts the Super Admin to enter the user’s details, such as name, role (Student, Professor, or Super Admin), email, etc.
    - The Super Admin fills in the details and clicks the **Save** button.
    - The system validates the details and adds the user to the database.

**Failure Case**

* **Incomplete Information**: If required fields are left blank, the system displays a message: *"Please fill in all required fields."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA



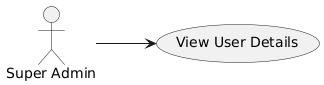
## UC-004: View User Details

**Purpose**

* + - This use case describes how the Super Admin can view the details of an existing user (Student, Professor, or Super Admin).

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Student Grading Sytem application should be running and accessible.
    - The Super Admin must be logged in.

**Post-Conditions**

* + - The Super Admin views the user’s details (name, role, email, etc.).

**Basic Flow**

* + - The Super Admin accesses the "View User Details" section from the dashboard.
    - The system displays a list of existing users with basic details.
    - The Super Admin selects a user from the list to view their complete details.
    - The system displays detailed information about the selected user (name, role, email, marks, etc.).

**Failure Case**

* **User Not Found**: If no user matches the search or filter criteria, the system displays: *"No users found matching the search criteria."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

## UC-005: Update User

**Purpose**

* + - This use case describes how the Super Admin can update the details of an existing user.

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Super Admin must be logged in.
    - The user to be updated must already exist in the system.

**Post-Conditions**

* + - The selected user’s details are successfully updated in the system.

**Basic Flow**

* + - The Super Admin selects the user to update from the "View User Details" section.
    - The system displays the current details of the user.
    - The Super Admin modifies the user’s details (e.g., name, role, marks).
    - The Super Admin clicks the Save button to apply the changes.
    - The system validates and updates the user’s details.

**Failure Case**

* **Invalid Input**: If the Super Admin enters invalid data, the system shows an error message: *"Please enter valid data in all fields.”*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

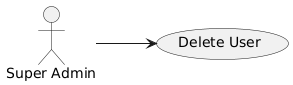
## UC-006: Delete User

**Purpose**

* + - This use case describes how the Super Admin can delete an existing user (Student, Professor, or Super Admin) from the system.

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Student Grading Sytem application should be running and accessible.
    - The Super Admin must be logged in.
    - The user to be deleted must already exist in the system.

**Post-Conditions**

* + - The selected user is deleted from the system.

**Basic Flow**

* + - **The Super Admin selects the user to delete from the "View User Details" section.**
    - **The system prompts for confirmation to delete the user.**
    - **The Super Admin confirms the deletion by clicking the Delete button.**
    - **The system removes the user’s data from the database.**

**Failure Case**

* **User Deletion Failed**:If there is an error during deletion, the system shows: *"An error occurred while deleting the user. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

## UC-007: Assign Professors to Students

**Purpose**

* + - This use case describes how the Super Admin can assign professors to students.

**Actors**

* + - The actor that interacts with this use case is **Super Admin**.



**Pre-Conditions**

* + - The Student Grading Sytem application should be running and accessible.
    - The Super Admin must be logged in.
    - Professors and students must already exist in the system.

**Post-Conditions**

* + - Professors are successfully assigned to their respective students.

**Basic Flow**

* + - **The Super Admin accesses the "Assign Professors to Students" page.**
    - **The system displays a list of students and available professors.**
    - **The Super Admin selects a student and assigns a professor from the available list.**
    - **The system updates the assignment in the database and confirms the changes.**

**Failure Case**

* **Assignment Failed**: If the assignment cannot be made (due to technical issues or missing data), the system displays: *"Unable to assign professor to student. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

## High Level Use Case Association: Professor

### UC-001: Register

**Purpose**

* + - This use case describes how the Professor can register an account in the system.

**Actors**

* + - The actor that interacts with this use case is **Professor**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Professor must not already have an account in the system.

**Post-Conditions**

* + - The Professor successfully registers an account.

**Basic Flow**

* + - **The Professor accesses the "Register" page from the login screen.**
    - **The system prompts the Professor to enter their details (name, email, password, etc.).**
    - **The Professor fills in the required details and clicks Register.**
    - **The system validates the details, and creates a new account.**

**Failure Case**

* **Duplicate Email**: If the email address already exists in the system, the system displays: *"This email is already registered. Please use another email."*
* **Incomplete Information**: If required fields are left blank, the system prompts: *"Please fill in all required fields."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

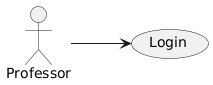
### UC-002: Login

**Purpose**

* + - This use case describes how the Professor can log in to the system to access their dashboard.

**Actors**

* + - The actor that interacts with this use case is **Professor**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Professor must be registered in the system.

**Post-Conditions**

* The Professor is successfully logged in and redirected to their dashboard, where they can view their tasks and student data.

**Basic Flow**

* + - **The Professor accesses the login page of the Student Grading System.**
    - **The system prompts the Professor to enter their username and password.**
    - **The Professor enters valid credentials and clicks the Login button.**
    - **The system validates the credentials and redirects the Professor to their dashboard.**

**Failure Case**

* **Duplicate Email**: If the email address already exists in the system, the system displays: *"This email is already registered. Please use another email."*
* **Incomplete Information**: If required fields are left blank, the system prompts: *"Please fill in all required fields."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

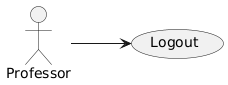
### UC-003: Logout

**Purpose**

* + - This use case describes how the Professor can log out from the system to ensure session security.

**Actors**

* + - The actor that interacts with this use case is **Professor**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Professor must be logged in to the system.

**Post-Conditions**

* The Professor is logged out, and the system terminates the session, redirecting them to the login page.

**Basic Flow**

* + - **The Professor accesses their dashboard.**
    - **The Professor clicks the Logout button located at the top-right corner of the dashboard.**
    - **The system terminates the session and redirects the Professor to the login page.**

**Failure Case**

* **Logout Failed**: If there is an issue with the logout process, the system displays: *"An error occurred while logging out. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

### UC-004: Add Student Marks

**Purpose**

* + - This use case describes how the Professor can add the marks for a student and add feedback in the system.

**Actors**

* + - The actor that interacts with this use case is **Professor**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Professor must be logged in to the system.
    - The Professor must have access to the relevant student’s details and course.

**Post-Conditions**

* The Professor is logged out, and the system terminates the session, redirecting them to the login page.

**Basic Flow**

* + - **The Professor accesses the "Add Student Marks" page from the dashboard.**
    - **The system displays a list of students assigned to the Professor, along with available fields for adding marks.**
    - **The Professor selects a student and enters the student's marks for the relevant subjects or assessments.**
    - **The Professor clicks the Save button to save the newly added marks.**
    - **The system validates the input and updates the student’s marks in the database.**
    - **A confirmation message is displayed: *"Marks successfully added."***

**Failure Case**

* **Invalid Marks**: If the marks entered are invalid (e.g., out of range), the system displays: *"Please enter valid marks."*
* **Connection Issue**: If the system cannot add the marks due to a connection issue, the system shows: *"Unable to add marks. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

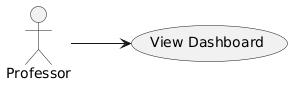
### UC-005: View Dashboard

**Purpose**

* + - This use case describes how the Professor can access and view their dashboard, which contains relevant data and tasks.

**Actors**

* + - The actor that interacts with this use case is **Professor**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Professor must be logged in to the system.
    - The Professor must be logged in to the system.

**Post-Conditions**

* The Professor sees a dashboard displaying their tasks, students, and relevant data.

**Basic Flow**

* + - **The Professor successfully logs in to the system.**
    - **The system redirects the Professor to the dashboard.**
    - **The Professor can view their students’ performance, assignments, and upcoming tasks.**

**Failure Case**

* **Dashboard Load Failure**: If the dashboard fails to load, the system displays: *"An error occurred while loading the dashboard. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

## High Level Use Case Association: Student

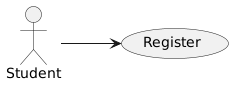
### UC-001: Register

**Purpose**

* + - This use case describes how a Student can register an account in the system.

**Actors**

* + - The actor that interacts with this use case is **Student**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Student must not already have an account in the system.

**Post-Conditions**

* The Student successfully registers an account

**Basic Flow**

* + - **The Student accesses the "Register" page from the login screen.**
    - **The system prompts the Student to enter their details (name, email, password, etc.).**
    - **The Student fills in the required details and clicks Register.**
    - **The system validates the details, creates a new account.**

**Failure Case**

* **Duplicate Email**: If the email address already exists in the system, the system displays: *"This email is already registered. Please use another email."*
* **Incomplete Information**: If required fields are left blank, the system prompts: *"Please fill in all required fields."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

### UC-002: Login

**Purpose**

* + - This use case describes how the Student can log in to the system to access their dashboard..

**Actors**

* + - The actor that interacts with this use case is **Student**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Student must be registered in the system.

**Post-Conditions**

* The Student is successfully logged in and redirected to their dashboard.

**Basic Flow**

* + - **The Student accesses the login page of the Student Grading System.**
    - **The system prompts the Student to enter their username and password.**
    - **The Student enters valid credentials and clicks the Login button.**
    - **The system validates the credentials and redirects the Student to their dashboard.**

**Failure Case**

* **Invalid Credentials**: If the username or password is incorrect, the system displays: *"Invalid username or password. Please try again."*
* **Connection Issue**: If the system is down or inaccessible, the system displays: *"Unable to connect to the system. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

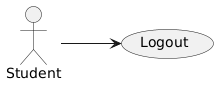
### UC-003: Logout

**Purpose**

* + - This use case describes how the Student can log out from the system to ensure session security.

**Actors**

* + - The actor that interacts with this use case is **Student**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Student must be logged in to the system.

**Post-Conditions**

* The Student is logged out, and the system terminates the session, redirecting them to the login page.

**Basic Flow**

* + - **The Student accesses their dashboard.**
    - **The Student clicks the Logout button located at the top-right corner of the dashboard.**
    - **The system terminates the session and redirects the Student to the login page.**
    - **A confirmation message may be displayed: *"You have successfully logged out."***

**Failure Case**

* **Logout Failed**: If there is an issue with the logout process, the system displays: *"An error occurred while logging out. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

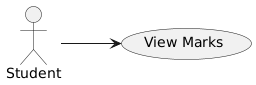
### UC-004: View Marks

**Purpose**

* + - This use case describes how the Student can view their marks in the system.

**Actors**

* + - The actor that interacts with this use case is **Student**.



**Pre-Conditions**

* + - The Student Grading System application should be running and accessible.
    - The Student must have marks recorded in the system.

**Post-Conditions**

* The Student can view their marks for different subjects or assessments.

**Basic Flow**

* + - **The Student accesses the "View Marks" page from their dashboard.**
    - **The system displays the list of subjects or assessments along with the Student’s marks for each.**
    - **The Student can view their marks for different subjects or assessments.**

**Failure Case**

* **Marks Not Available**: If no marks have been recorded yet, the system displays: *"No marks available. Please check back later."*
* **Connection Issue**: If there is a problem loading the marks, the system displays: *"Unable to retrieve marks. Please try again later."*

**Screen Design**

A screenshot of a computer

Description automatically generated

**Database Tables Involved**

* + - NA

# Processing Details

## PD01: Student Grading System

### PD01A: Add User

* + - Each field in the user row (name, team, etc.) must be checked for completeness.
    - Username must be unique across the system to avoid confusion and ensure proper identification.
    - Username must be a minimum of 5 characters and maximum of 28 characters.
    - Username must not contain prohibited characters or patterns, such as spaces, slashes, or any other special characters other than underscore and hyphen.

### PD01A: Update User

* + - Validate each updated field for correctness (e.g., name must not be empty, DOB must be a valid date, etc.).
    - Ensure any updated username adheres to the same rules as for adding a new user.
    - Username must be unique across the system to avoid confusion and ensure proper identification.
    - Username must be a minimum of 5 characters and maximum of 28 characters.
    - Username must not contain prohibited characters or patterns, such as spaces, slashes, or any other special characters other than underscore and hyphen.