SDLC Lab — System Planning

Course: Software Development Technologies

Format: Group work (2–3 students)

Lab Information

Group Members:

Student 1: [Wirane Mohamed]

Student 2: [Ye chengxin]

Student 3: [Yasser Idbouzkri]

System: [Student Management System]

Date: [06.02.2026]

Part A — Waterfall Planning
Task A1: Requirements and Stages

System Requirements :

Functional Requirements:

A student management system is a centralized software platform that allow to add , to add, update, and delete student records. It should also allow students to log in and view their personal information and grades. The system must provide a search function to find students quickly.

Non-Functional Requirements:

The system should be easy to use and have a simple, clear interface for both administrators and students. It must be secure so that only authorized users can access or modify data. The system should respond quickly to user actions and handle multiple users at the same time.

Design :

The system will use a simple client-server architecture where users access the application through a web interface. The main components will include a user interface for students and administrators, an application layer to handle the business logic, and a database to store student records. The design will focus on clarity and separation of responsibilities between components to make the system easy to maintain.

Implementation

Code Development

Write the code based on the design specifications

User Interface (UI)

Admin interface (manage students)
Student interface (view information and grades)

Backend / Business Logic

Student record management (add / update / delete)
Authentication (login)
Grade viewing

Database

Create tables (students, grades, accounts, etc.)
Connect the application to the database
Store and retrieve data reliably

4. Testing

Test Objectives

Make sure all features work correctly

Verify system reliability and stability

Test Types

Test student registration

Test login and authentication

Test grade viewing

Bug Handling

Identify errors and bugs

Fix issues before deployment

Validation

Confirm the system meets the requirements

Approve the system for deployment

5. Deployement

Rlease Preparation

Prepare the Student Management System for real use in the institution.

Check that all student, admin, and grade features work correctly.

Installation

Install the system on the school/university server

Set up the database for student records, accounts, and grades

User Access

Create accounts for administrators and students

Give users access to log in and use the system

Go Live & Monitoring

Make the system available for daily use

Monitor performance and fix any problems after launch

Task A2: System Diagram

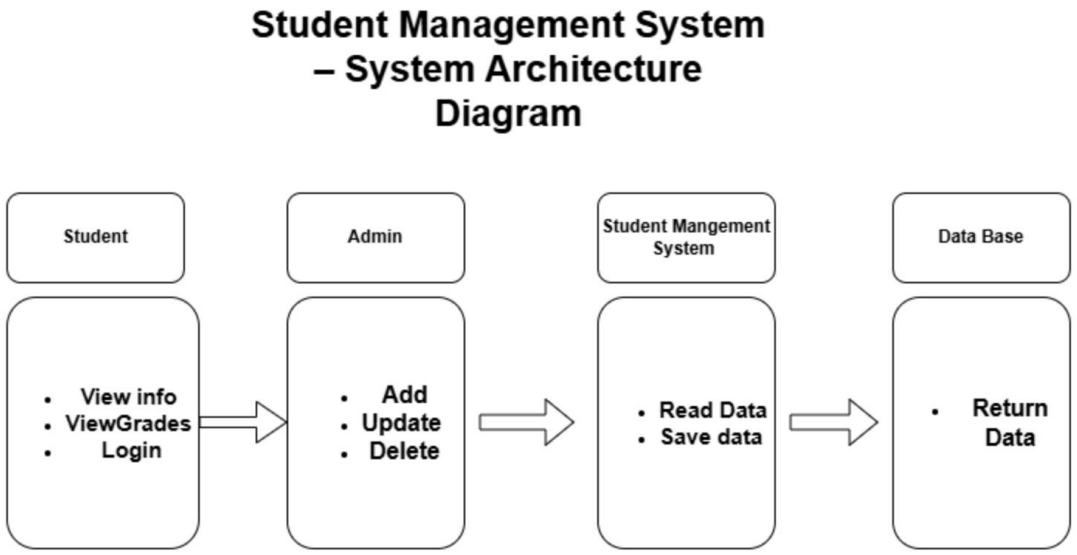

Figure1 : Diagram of Student Management System Architecture

Task B1: User Stories (Sprint 1):

Sprint 1 User Stories

User Story 1:

As a student,
I want to log in to the system,
so that I can access my personal information and grades.

Acceptance Criteria:

The student can enter a username and password to log in.

After logging in, the student can see their personal information and grades.

UserStory 2:

As an administrator,
I want to add new student records,
so that I can register students in the system.

Acceptance Criteria:

The admin can create a new student record with basic information.

The new student record is saved and appears in the student list.

User Story 3:

As an administrator
I want to update or delete student records,
so that I can keep student information accurate and up to date.

Acceptance Criteria:

The admin can edit existing student information.

The admin can delete a student record from the system.

Task B2: Handling Change

Sprint 2 (Backlog)

User Story 4 — New Requirement:

As a student,
I want to upload documents to the system,
so that I can submit required files online without going to the office.

Acceptance Criteria:

The student can upload a document from their account.

The uploaded document is saved and can be viewed by the administrator.

Part C — Reflection :

Question 1: Waterfall Rigidity

What problem occurred when the requirement was added in the Waterfall model?

Answer :

When the new requirement was added, it caused a problem because the requirements phase was already finished. In the Waterfall model, you cannot easily go back and change previous steps. This means the new feature could not be added without redoing earlier work. This shows that Waterfall is not flexible.

Question 2: Agile Adaptation

How did Agile handle the same situation differently?

Answer :

Agile handled the change by adding the new requirement as a new user story in the backlog. The work that was already done did not need to be changed. The team could plan the new feature for the next sprint. This shows that Agile is flexible and can accept changes easily.

Question 3: Model Suitability

Which model would be more suitable for rapidly changing systems, and why?

Answer:

Agile is better for systems that change a lot. It allows new requirements to be added at any time. The team can continue working without restarting the project. Waterfall is better only when requirements do not change.

Submission Date : 06/02/2026