Course Registration System Specification

I.System Introduction

The Course Registration System is developed to facilitate online course enrollment for students, grade management for lecturers, and data administration for system administrators.

II. Supplementary Specification

Supportability

None

III. Functional Requirements

3.1 User Role

3.1.1 Login

- Users are allowed to verify their identity to access the system. After successful login, the system will redirect the user to the homepage corresponding to their role (Student, Lecturer, or Administrator).
- Condition: The user already has an account assigned to them in the system.
    + If the login is successful, the user is granted a working instance and accesses internal functions.
    +If it fails, the system remains on the login page and does not grant access.
- When a user accesses the login page, the system will display a form requesting their ID and password. After the user enters the information and clicks the sign-in button, the system will verify the data is validity. Once the user's role is successfully authenticated, the system will redirect the user to the corresponding Dashboard page:
    + If a Student -> Student Homepage.
    + If a Lecturer -> Lecturer Homepage.
    + If an Admin -> Admin Page.
- If the user enters the wrong ID or password, the system will display an error message and delete the ID and password so the user can re-enter them.

3.1.2 Logout

- This allows users to securely end their current session. The system will revoke access rights and delete session data to ensure that no one can access the account once the user leaves.

- Condition: The user has successfully logged into the system.
    + If successful, the user will be redirected to the login page.
- When the user clicks the logout button, the system sends a session cancellation request to the server. The server then deletes the session and any temporarily stored information, and the system redirects the user to the login page.

3.1.3 Password Recovery

- Allows users to reset their password if they forget their old one. The system authenticates the user's identity by sending a one-time verification code (OTP) to the email address registered in the system.
- Condition: Users must have a valid account in the system, and their account must have an accurate email address that has been stored in the database.
    + If successful: The old password is revoked, and the new password is updated in the database. The user can log in with the new password.
    + If unsuccessful: The old password remains unchanged.
- On the Login screen, the user clicks the "Forgot password" link. The system displays a verification form; the user enters the information and clicks "Send OTP code," prompting the user to enter a new password.

    + If the information matches the data: The system generates an OTP code (e.g., 6 digits), stores this code in temporary memory with an expiration time (e.g., 5 minutes).

    + The system sends this OTP code to the user's email address.

    + The user enters the generated OTP code and the new password, confirms the new password, and then clicks "Save changes."
- If the password or OTP code is incorrect, the system will delete the OTP code and password and prompt the user to re-enter them.

3.2 Student Role

3.2.1 View Available Courses

- This allows students to view a list of courses/modules offered in the current semester or a selected semester. Students can see detailed information such as the number of credits, instructors, course duration, and prerequisites for registration.

- Condition: Students have successfully logged into the system, and the semester's timetable list has been updated by the Admin.

+ Students receive information about the course modules, and the system displays a list according to the school's criteria so that students can register for courses appropriately.

- Students will see the Student Dashboard. The system displays the default interface with:

+ Semester selection bar: Defaults to the current semester.

+ Course list: Displays all open courses. This includes the course number, course code, course name, credits, required credits, and registration requirements (The registration requirements are divided into 3 parts: a, b, and c. Priority must be given to taking credit (a) first, then b and c).

3.2.2 Course Registration

- Students can select and register for open courses (Status: Open). The system will check the requirements (credits, prerequisites, schedule, class size) before recording the registration. The course status on the interface will change immediately after successful registration.

- Condition:

+ The student is logged in and is currently on the Student Dashboard page.

+ Course registration is currently underway (Registration is open).

+ Students must not be subject to any registration restrictions (due to outstanding tuition fees, disciplinary actions, etc.).

- If successful:

+ New registration data is saved to Collection registrations.

+ The current class size increases by 1 (e.g., from 11/40 to 12/40).

+ The function button changes from "Register" to "Registered" (grayed out).

+ The system displays a success message (Toast message).

- If unsuccessful:

+Data remains unchanged, and the system reports an error with a specific reason.

+ Students observe the list of course modules on the control panel.

+ Unregistered courses have the status "Open" (blue) and a "Register" button.

+ Registered courses have the status "Registered" (green) and the button is disabled.

+ The system performs logical checks (Backend Validation):

+ Prerequisites: Checks if the student has passed the required courses.

+ Class size: Checks if there are still available slots (e.g., 45/50 -> Available).

+ Schedule conflict: Compares the schedule of this course with courses already registered.

+ Credits: Checks if the total credits after adding this course exceed 24.

+ The system determines the registration type (New Course / Retake / Improvement Course) based on grade history.

3.2.3 Drop/Cancel Course

- Students are allowed to remove a previously registered course from their course list. Course cancellation is only permitted within the specified time frame. After successful cancellation, the class size will be reduced, and that spot will be made available for another student to register.

- Condition: The student has successfully logged in. The course cancellation period is still valid. The student has successfully registered for the course they wish to cancel.

-If successful:

+ The corresponding record in Collection registrations is permanently deleted (or marked cancelled depending on the business process).

+ The current class size in Collection courses is reduced by 1.

+ The total number of credits for the student in the semester is reduced accordingly.

+The course status on the interface returns to "Open" (allows re-registration).

- If unsuccessful: The data remains unchanged.

3.2.4 View Registered Courses

- This allows students to review the list of courses they have successfully registered for during the semester. This interface typically serves as a student's timetable, displaying details about classrooms, class times, and the total number of credits registered.
- Condition:

+ The student has successfully logged into the system.

+ The student has registered for at least one course (if not registered, the list is empty).

3.3 Lecturer Role

3.3.1 View Class List

- This allows instructors to view a list of assigned courses for the semester. When a specific course is selected, the system displays a detailed list of students who have successfully registered for that course, including basic personal information for class management purposes.

- Condition: The instructor has successfully logged into the system. The instructor has been assigned to at least one course in the Collection courses.

- The instructor can see the list of students in their class. The instructor can export the list to a file (Excel/PDF) if needed.

3.3.2 Grade Management

- This system allows instructors to enter, update, and save grades for students in their assigned courses. The system supports direct grade entry via a web interface. Saved grades will be displayed immediately on the student's screen.
- Condition: The instructor has successfully logged in. The course has either finished or is within the time frame for entering grades. The instructor is the one assigned to teach that class.
- If successful: The grade column in Collection registrations is updated with the new value.
- If unsuccessful: The old grade data is retained, and an error message is displayed.

3.4 Admin Role

3.4.1 Course Management

- This allows administrators to add, update, or delete course modules in the system. This is a fundamental function for setting up data for each semester before opening the registration portal for students.

- Condition: The admin has successfully logged in, and the list of instructors is now available in the system.

- Add: A new document is created in Collection courses.

- Update: Course information is changed and immediately visible to students and instructors.

- Delete: The course is removed from the system.

3.4.2 Section Management

- This function allows administrators to manage specific course modules for each subject. Administrators can set and adjust the maximum class size for each class to match classroom capacity or registration demand. Additionally, administrators can monitor class status (full or with available spaces) to decide whether to open or cancel classes.

- Condition: Admin has successfully logged into the system. The course has been created in the system. The course section to be edited is currently active.

- If successful:

+ The maximum number of slots for the course in Collection courses is updated to the new value.

+ The system allows (or blocks) additional students from registering based on the newly updated number of slots.

- If unsuccessful:

+The number of slots remains unchanged, and no changes are saved to the database.

3.4.3 Student Management

- This allows administrators to manage student profiles within the system. Operations include: adding new students, updating personal information, and deleting/locking student accounts. The system ensures the uniqueness of student ID numbers and email addresses.

- Condition:Admin has successfully logged in. The student list file is prepared in the correct format.

- Add new: A student account is created in the Collection users with the role of student. The student can log in using the default password.

- Update: Student information changes immediately.

- Delete: The student can no longer log in to the system.

3.4.4 Open Registration Portal

- This allows the administrator to activate the course registration period for a specific semester. The administrator will set the start and end times. During this period, student registration/cancellation functions will be unlocked.
- Condition: Admin has successfully logged in. The course data for that semester has been prepared.
- If successful:
    + The system status changes to "OPEN".
    + The start and end times are saved to the system configuration.
    + Students accessing the system will see the "Register" button highlighted.
- If unsuccessful:

+ The system status remains "CLOSED".

3.4.5 Close Registration Portal

- This allows the administrator to disable the course registration feature system-wide. When the portal closes, students cannot add, edit, or delete registered courses. The system also has a mechanism to automatically close when the actual time exceeds the configured "End Time".
- Condition: The registration portal is currently in "OPEN" status. Admin has successfully logged in.
- If successful:
    + The system status in Collection settings changes to "CLOSED".
    + The "Register/Cancel" button on the student interface is disabled or hidden.
    + Registration data is finalized.
- If unsuccessful:
    + The system status remains "OPEN".

3.4.6 Lecturer Management

- This allows the administrator to manage faculty profiles within the system. Operations include: adding new faculty members, updating personal information, and deleting faculty accounts.
- Condition: Admin has successfully logged in. The list of Departments/Institutes has been set up in the system.
- Add new: A user account with the role of lecturer has been created. Lecturers can use this account to log in and view classes.
- Update: The lecturer's information displayed on the timetable has been refreshed.
- Delete: The lecturer has been removed from the list and can no longer be assigned to teach.

IV. Non-Functional Requirements

4.1 Security

- Authorization: The system must ensure appropriate access rights. Students may only register for courses and view their own grades; Lecturers may only input grades for the classes they teach; Admins have full system administration privileges.

4.2 Performance and Speed

- Response Time: Basic operations (such as clicking the Register button or searching for courses) must receive a response within a maximum of 2 seconds.
- Load Capacity: The system must be able to handle at least 500 concurrent users during peak registration periods without crashing or data loss.

4.3 Stability and Reliability

- Availability: The system must ensure 24/7 operation throughout the course registration period (achieving 99.9% uptime).
- Data Integrity: In the event of a network error during registration, the system must ensure data consistency (i.e., the registration is either fully successful or not executed at all, adhering to SQL Transaction properties).

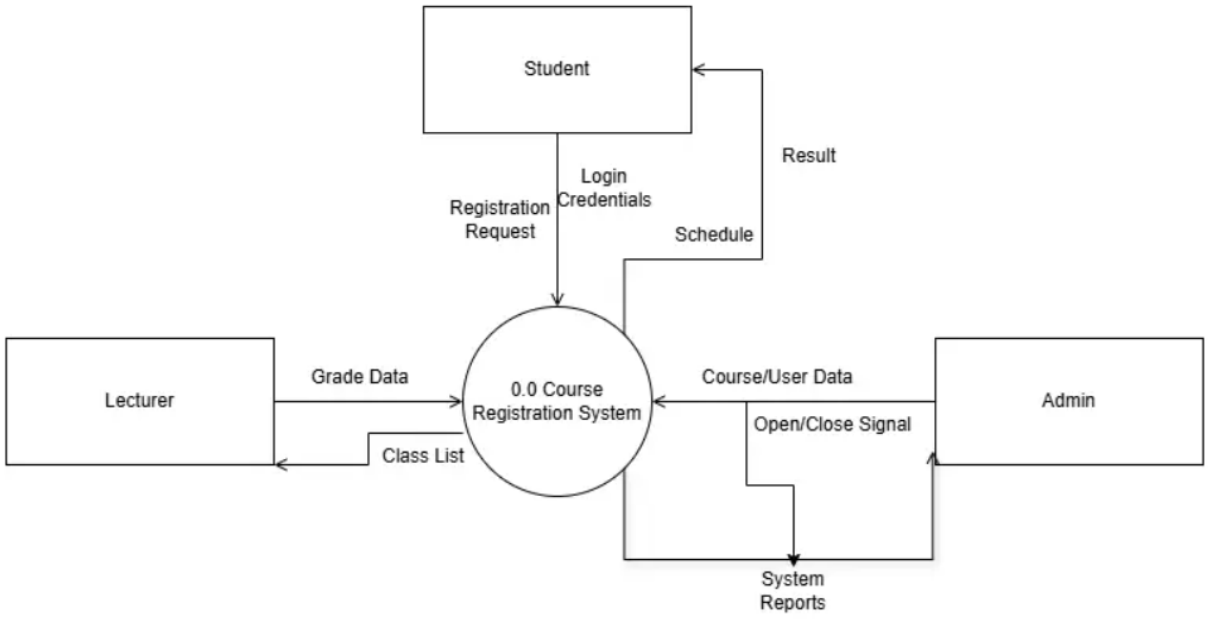
4.4 Usability

- Intuitive Interface: The interface should be simple with a clear layout so that students can perform registration independently without needing complex instruction manuals.
- Error Messages: Error messages (such as "Invalid ID or Password!") must be displayed in clear.

V. Data Flow Diagram

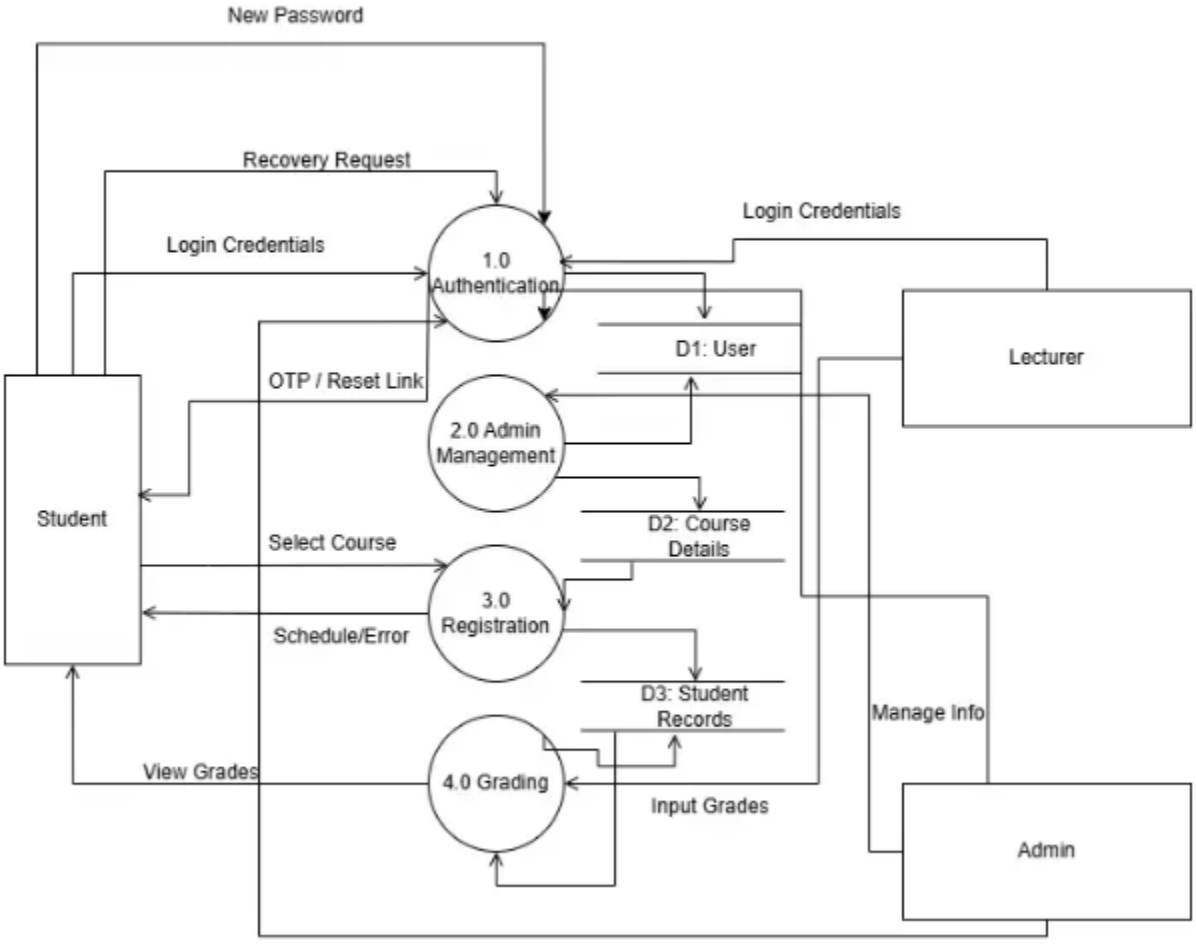
- This section describes in detail how data moves and transforms within the system using Data Flow Diagrams (DFD).

5.1 DFD Level 0



- **Description:** The Course Registration System acts as a central processing unit, interacting with three main actors:
- **Student:** Submits login information and course registration requests. The system returns registration results and timetables.
- **Lecturer:** Submits grade data and receives a list of students in the class.
- **Admin:** Submits course settings, signals to open/close registration periods, and receives summary reports (System Reports).

5.2.DFD Level 1



Description of Processing Steps:

1.0 Authentication:

- Receive login information from Students, Lecturers, and Admins.
- Research data in repository **D1:User to verify identity**.
- Support password recovery via OTP/Reset Link.

2.0 Admin Management:

Allow Admins to add/edit/delete course information and save it to repository D2:Course Details.

Manage user information and update it in repository D1:User.

3.0 Registration:

- This is the most important process. Receive "Select Course" requests from students.
- Check course information from repository **D2:Course Details** (check class size, schedule).
- If valid, save the registration result to repository **D3:Student Records** and return a confirmation message.

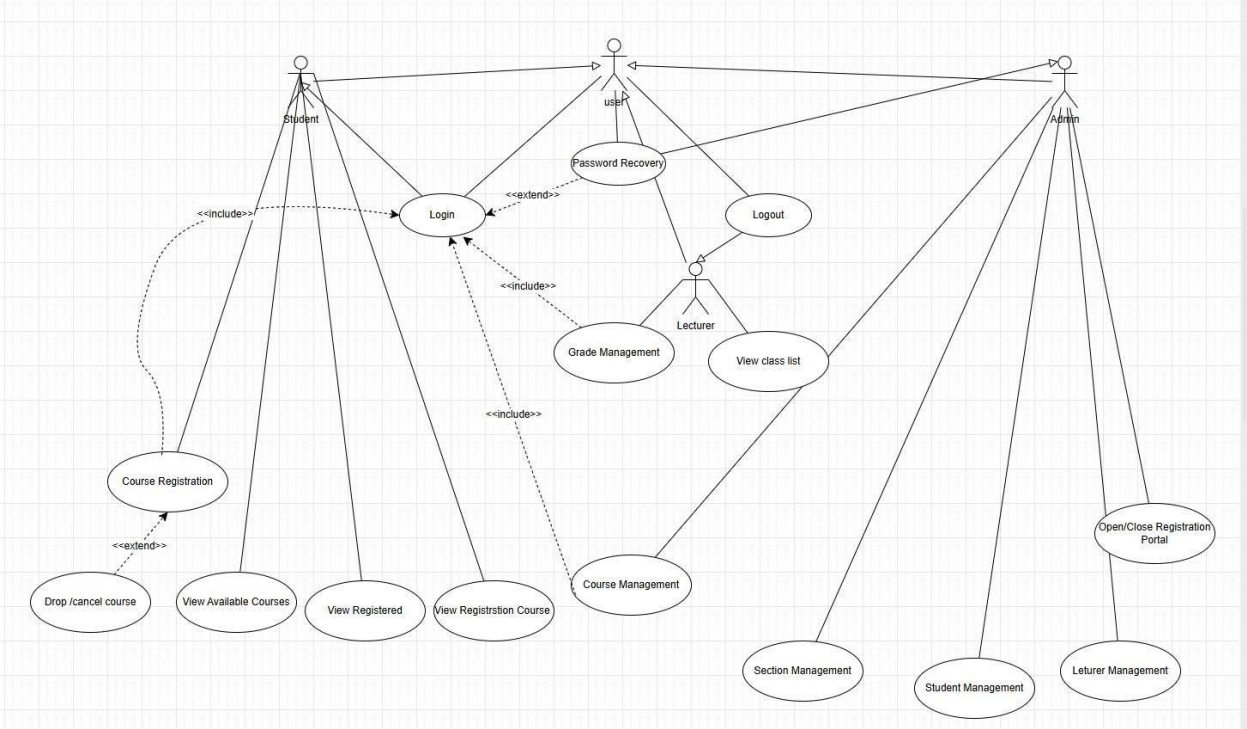
4.0 Grading (Score Processing):

- Receive grade data from instructors (Input Grades).
- Update grades in student records in the D3: Student Records database.
- Data Stores: The system uses MongoDB to organize the following data stores:
- **D1: User**: Stores user accounts, passwords (hash), and user roles.
- **D2: Course Details**: Stores course and section information.
- **D3: Student Records**: Stores registration history and grade reports.

VI. Use case Diagram

- The Course Registration System supports online course registration for students, grade management for lecturers, and academic administration for admins.

6.1 Diagram Overview



6.2 Actors

- Student: Registers and manages courses.

- Lecturer: Manages class lists and grades.

- Admin: Manages subjects and course classes.

6.3 Use Case Specifications

6.3.1 Login

- Actors: Student, Lecturer, Admin

- Description: Authenticate users.

- Pre-condition: User has a valid account.

- Post-condition: User accesses system or receives error.

- Main Flow: Enter credentials → system validates → access granted.

6.3.2 Logout

- Actors: Student, Lecturer, Admin

- Description: End session.

- Pre-condition: User logged in.

- Post-condition: User logged out.

6.3.3 Forgot Password

- Actors: Student, Lecturer, Admin

- Description: Recover password via OTP.

- Pre-condition: Login failed.

- Post-condition: Password reset.

6.3.4 View Available Courses

- Actor: Student

- Description: View open courses.

6.3.5 Register Course

- Actor: Student

- Description: Register for a course.

- Pre-condition: Registration period open.

6.3.6 Cancel Registration

- Actor: Student

- Description: Cancel registered course.

6.3.7 View Registered Schedule

Actor: Student

Description: View registered courses.

6.3.8 View Class List

- Actor: Lecturer

- Description: View student list.

6.3.9 Update Grades

- Actor: Lecturer

- Description: Update student grades.

6.3.10 Manage Subjects

- Actor: Admin

- Description: Add, edit, delete subjects.

6.3.11 Manage Course Classes

- Actor: Admin

- Description: Manage course classes and capacity.

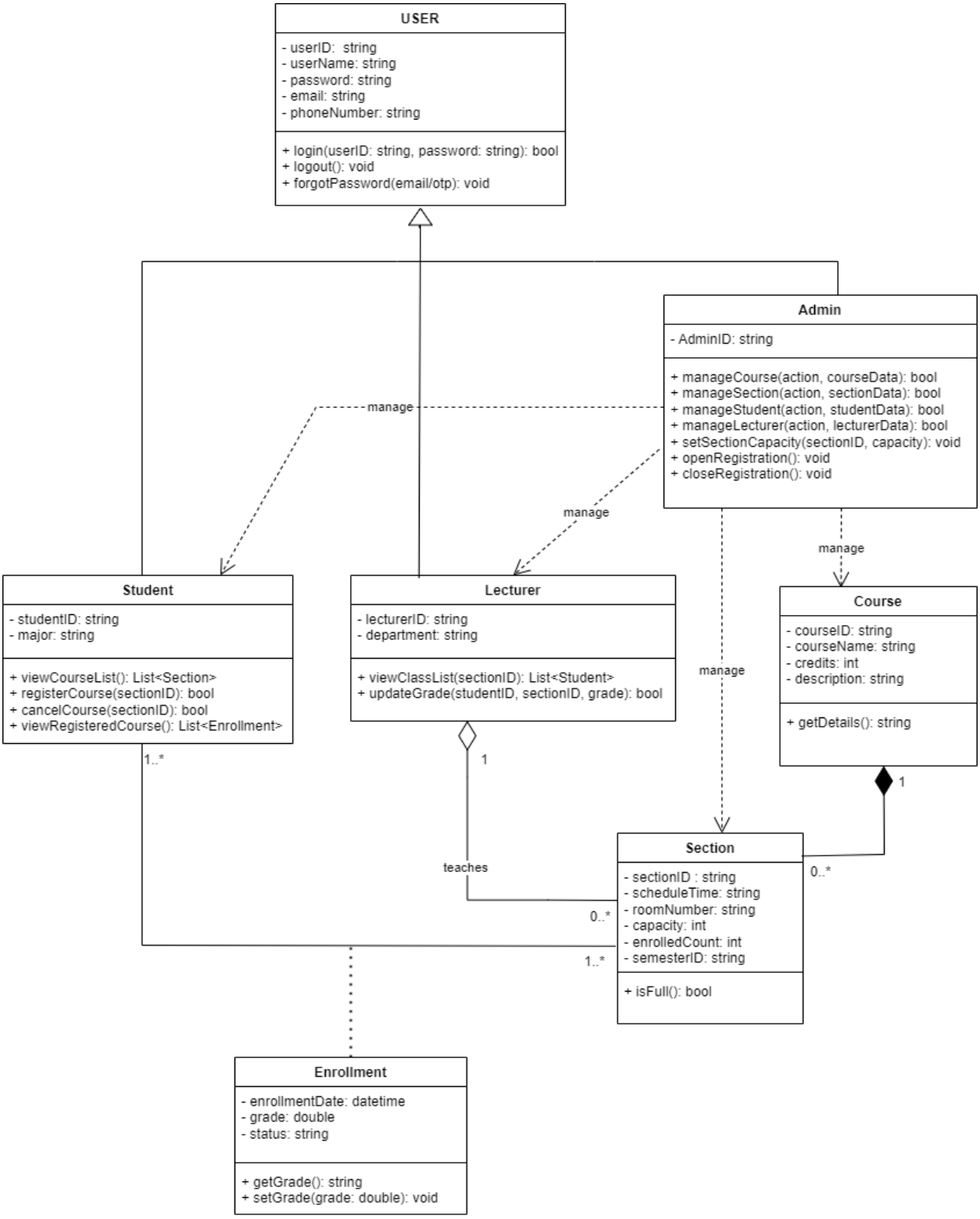
6.4 Conclusion

- This document describes functional requirements of the Course Registration System.

VII. Class Diagram

7.1. Overview

The Class Diagram describes the static logical structure of the Course Registration System, illustrating the main classes, their attributes, methods, and relationships. The system supports three primary user roles: **Student**, **Lecturer**, and **Admin**, each inheriting from a common abstract class **User**. The design follows object-oriented principles to ensure modularity, reusability, and maintainability.



7.2. User Class Hierarchy

7.2.1. User (Abstract Class)

- The User class is an abstract superclass representing common information and behaviors shared by all system users.

Attributes:

- userId: String – Unique identifier for the user.
- userName: String – Full name of the user.
- email: String – Email address used for communication.
- password: String – Hashed user password.

Methods:

- login(userId, password): Boolean – Authenticates the user.
- logout(): Void – Logs the user out of the system.
- forgotPassword(): Void – Supports password recovery via Email or OTP.

7.2.2. Student Class

- Inherits from User, representing students who use the system to register for courses.

Attributes:

- studentID: String – Unique student identifier.
- major: String – Student's academic major.

Methods:

- viewCourseList(): List<Section> – Displays all open course sections.
- registerCourse(sectionID): Boolean – Registers the student for a section.
- dropCourse(sectionID): Boolean – Cancels a registered course.
- viewRegisteredCourses(): List<Enrollment> – Displays current timetable.

7.2.3. Lecturer Class

- Inherits from User, representing faculty members responsible for teaching and grading.

Attributes:

- lecturerID: String – Unique lecturer identifier.
- department: String – Academic department.

Methods:

- viewClassList(sectionID): List<Student> – Views students enrolled in a class.
- inputGrade(studentID, sectionID, grade): Boolean – Inputs or updates grades.

7.2.4. Admin Class Inherits from User, manages system configuration.

Attributes:

- adminId: String – Unique identifier.

Methods:

- manageCourse(), manageUsers(): Boolean – CRUD operations for entities.
- openRegistration(), closeRegistration(): Void – Controls the registration period.

7.3. Course and Section Management

7.3.1. Course Class

- Stores general information about academic courses (e.g., "Introduction to Programming").

- **Attributes:** courseId (String), courseName (String), credits (Integer).
- **Methods:** getDetails(): String.

7.3.2. Section Class

- Represents a specific instance of a course (e.g., "Intro to Programming - Monday Morning").

  - **Attributes:** sectionID, scheduleTime, roomNumber, maxCapacity (Integer), currentEnrolled (Integer).

  - **Methods:** isFull(): Boolean – Checks if capacity is reached.

7.4. Enrollment Class (Association Class) Records the Many-to-Many relationship between Student and Section.

Attributes:

- enrollmentDate: DateTime
- grade: Double
- status: String (Registered/Dropped)

Methods:

- updateGrade(newGrade): Void.

7.5. Class Relationships Summary

- **Inheritance:** Student, Lecturer, Admin inherit from User.

- **Composition:** A Course is composed of multiple Section instances (1 course has 0..* sections).

- **Association:** Admin manages Student, Lecturer, Course. Lecturer teaches Sections.

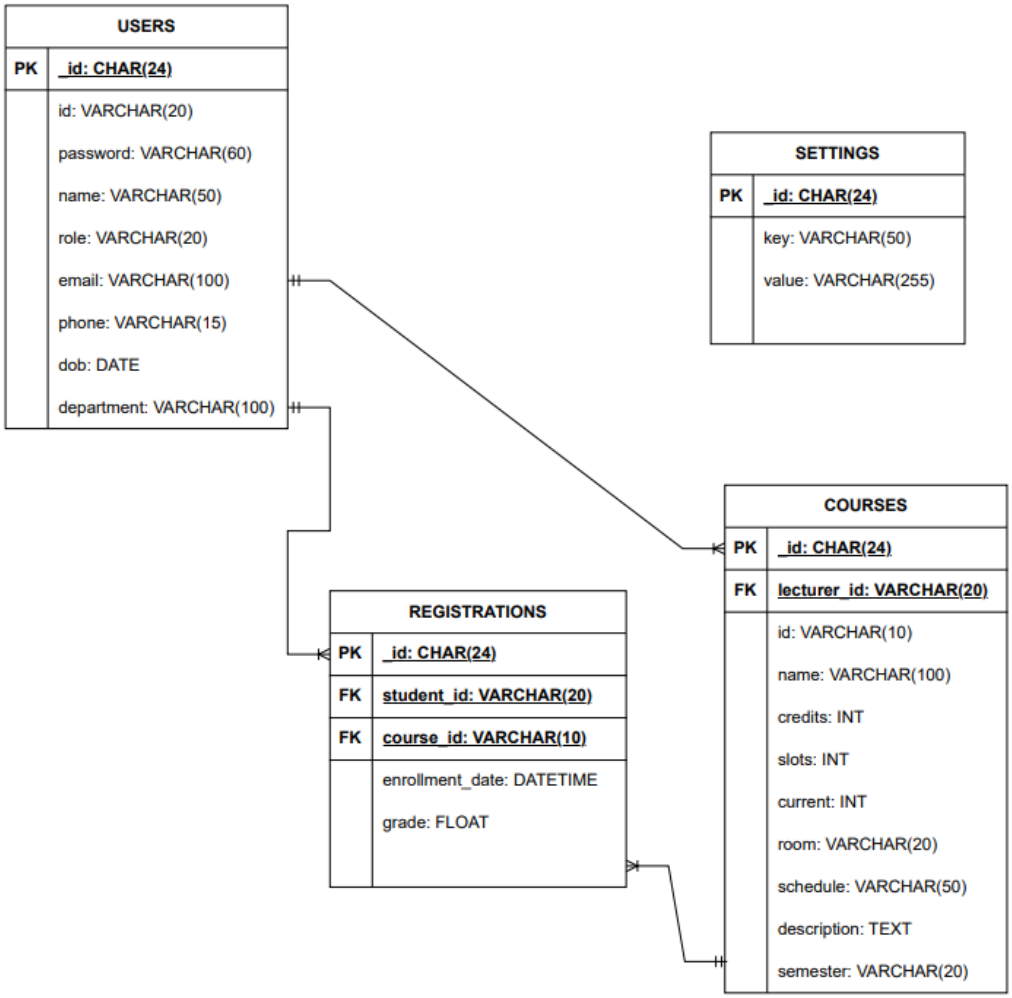
- **Association Class:** The relationship between Student and Section is managed by the Enrollment class to store specific data like Grades.

7.6. Conclusion

- The Class Diagram accurately models the functional requirements, effectively separating concerns between User Management, Course Management, and

Academic Records. It serves as the blueprint for the backend logic implementation in Python.

VIII. Data Model



- The system utilizes MongoDB as the primary database management system. The data is organized into four main collections: users, courses, registrations, and settings. Below is the detailed schema design represented in JSON format with data constraints.

8.1 Collection: users

- This collection stores profile and authentication data for all system actors (Students, Lecturers, and Admins). The role field determines the access permissions.

{

"_id": ObjectId("659d1234..."), // Primary Key

"id": "sv2021001", // String (Unique, Max: 20 chars)

"password": "hashed_password_123", // String (Min: 60 chars)

"name": "Nguyen Van A", // String (Max: 50 chars)

"role": "student", // String (Enum: "student", "lecturer", "admin")

"email": "sv2021001@uni.edu.vn", // String (Max: 100 chars)

"phone": "0909123456", // String (Max: 15 chars)

"dob": "2003-05-20", // String (Format: YYYY-MM-DD)

"department": "Software Engineering" // String (Max: 100 chars)

}

8.2 Collection: courses

- This collection manages course sections. Each document represents a specific class section available for registration, including schedule and capacity details.

{

"_id": ObjectId("659d5678..."), // Primary Key

"id": "CS101", // String

"name": "C++ Programming", // String (Max: 100 chars)

"credits": 3, // Integer

"slots": 50, // Integer

"current": 45, // Integer (Min: 0)

"lecturer_id": "gv001", /// String (Foreign Key -> Ref users.id)

"room": "C201", // String (Max: 20 chars)

"schedule": "Mon 07:30 - 11:30", // String (Max: 50 chars)

"description": "Introduction to OOP with C++", // String (Text/Max: 500 chars)

"semester": "Spring 2026" // String (Max: 20 chars)

}

8.3 Collection: registrations

- This collection functions as an associative entity to record student enrollments. It links a specific Student (users) to a specific Course (courses) and stores the academic result.

{

"_id": ObjectId("659d9999..."), // Primary Key

"student_id": "sv2021001", // String (Foreign Key -> Ref users.id)

"course_id": "CS101", // String (Foreign Key -> Ref courses.id)

"enrollment_date": "2026-01-06", // String (Format: YYYY-MM-DD HH:mm:ss)

"grade": 8.5 // Float (Min: 0.0, Max: 10.0) or Null

}

8.4 Collection: settings

- This collection stores system-wide configurations, used to control features like opening or closing the registration portal.

{

"_id": ObjectId("659d0000..."), // Primary Key

"key": "registration_open", // String (Unique, Max: 50 chars)

"value": "1" // String (Max: 255 chars)

}

IX. Interface Design Description

- This section provides a detailed visualization of the user interface for the Course Registration System. The design adheres to the usability requirements specified in Section 4.4, ensuring an intuitive, clean, and responsive experience for Students, Lecturers, and Admins.

9.1 Authentication Interfaces

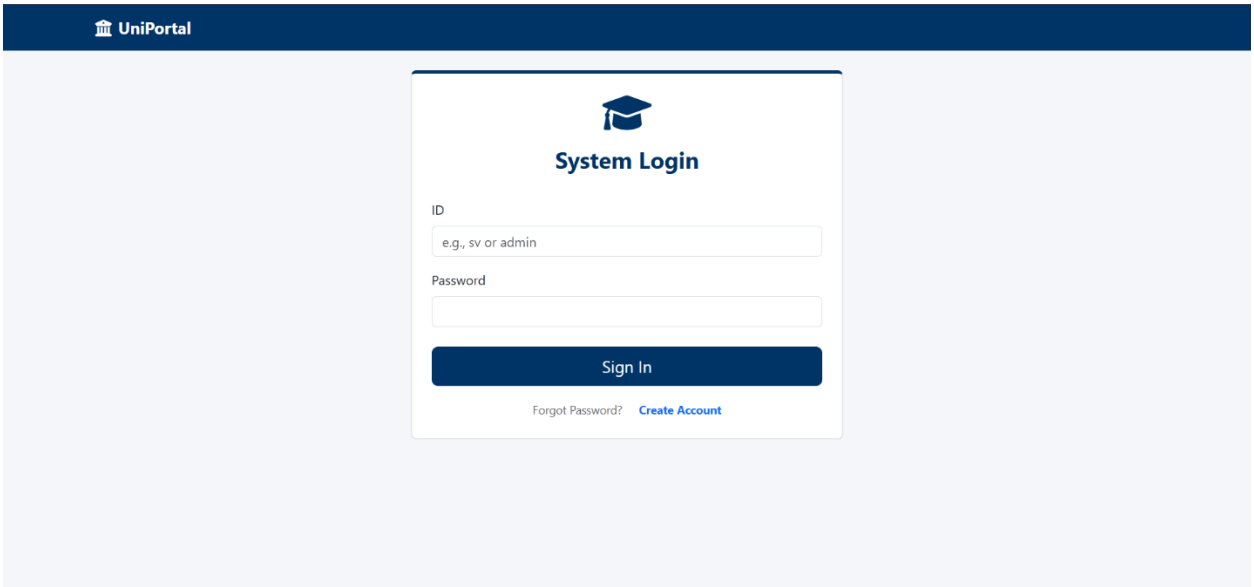
- These interfaces are common to all users and serve as the entry point to the system.

9.1.1 System Login

- The login screen is designed with a minimalist layout to minimize distraction. It requires users to authenticate using their unique ID and Password.

Key Elements:

- Input Fields: User ID (e.g., student ID or admin username) and Password.
- Action Button: "Sign In" triggers the authentication process (Functional Requirement 1.0).
- Navigation Links: Quick access to "Forgot Password?" and "Create Account" for account management.



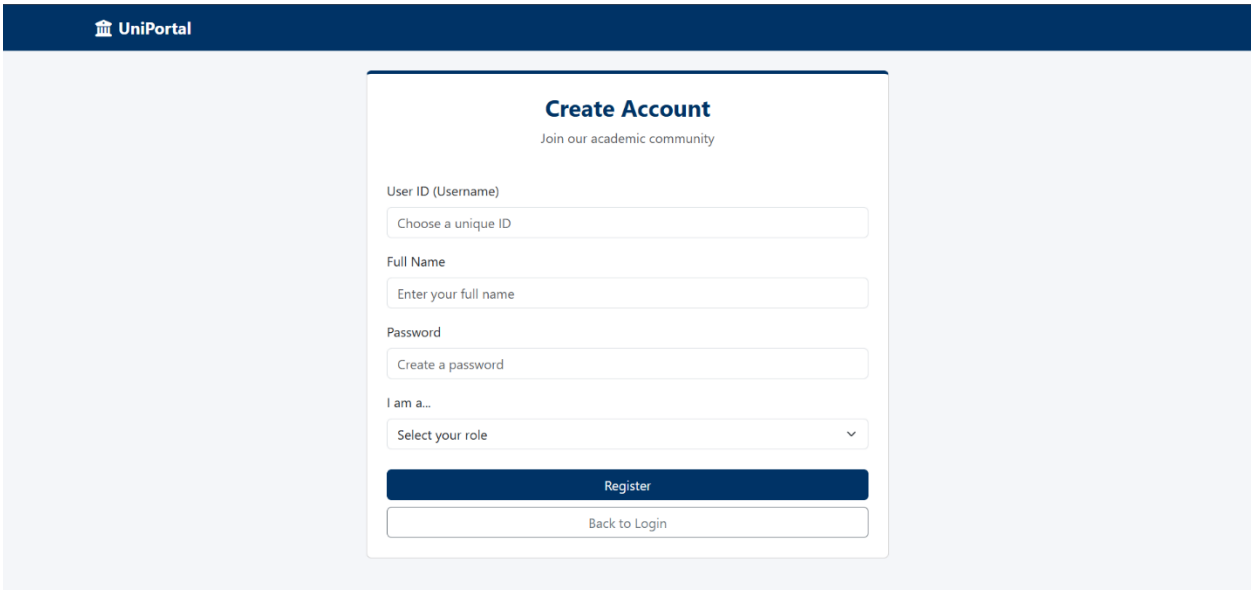
9.1.2 Create Account

- This interface serves a critical role in system testing and data integrity. It acts as a validation checkpoint to ensure that any new entity entering the system (Student, Lecturer, or Admin) possesses all mandatory attributes defined in the Class Diagram (userId, userName, password, role, email, phoneNumber).

Purpose: To verify that the system correctly captures and instantiates a new User object with full information before committing data to the database.

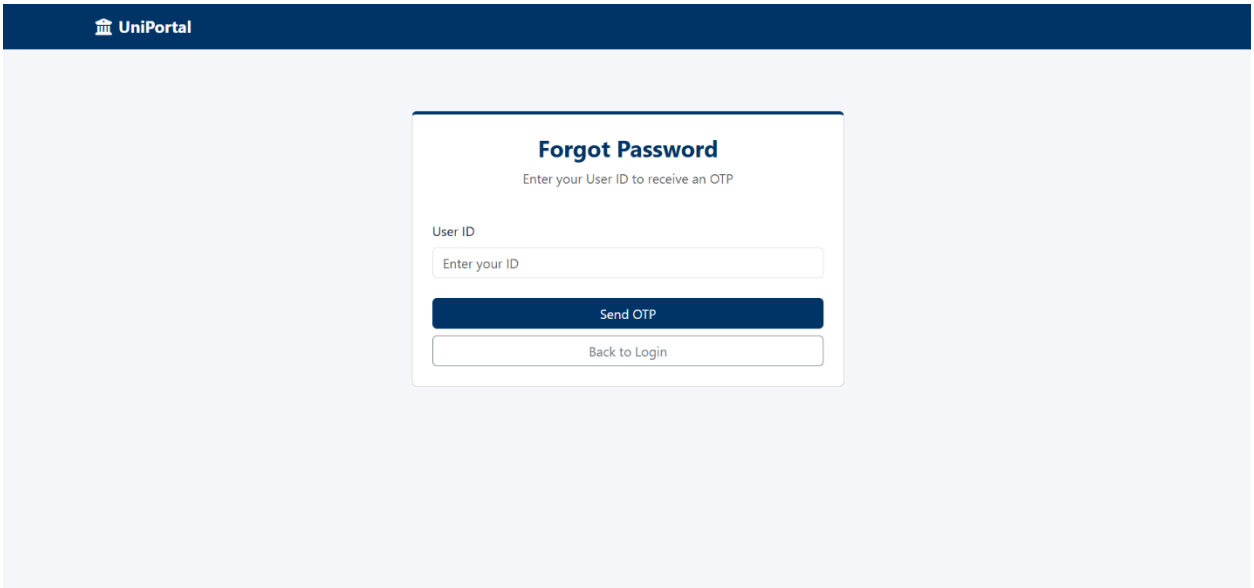
Key Fields:

- User ID: Must be unique.
- Full Name: Display name for the dashboard.
- Password: Input for creating secure credentials.
- Email: User email address
- PhoneNumber: User phone number
- Role Selection: A dropdown menu to strictly categorize the user (Student/Lecturer/Admin), ensuring correct authorization levels upon login.



9.1.3 Forgot Password

- Designed to handle account recovery securely via OTP, reducing administrative overhead.
- Workflow: The user enters their User ID. The system validates the ID and sends an OTP to the registered email/phone.
- Key Elements: Clear instruction text ("Enter your User ID to receive an OTP") and a "Send OTP" button.



9.2. Student Interface (Dashboard and Registration)

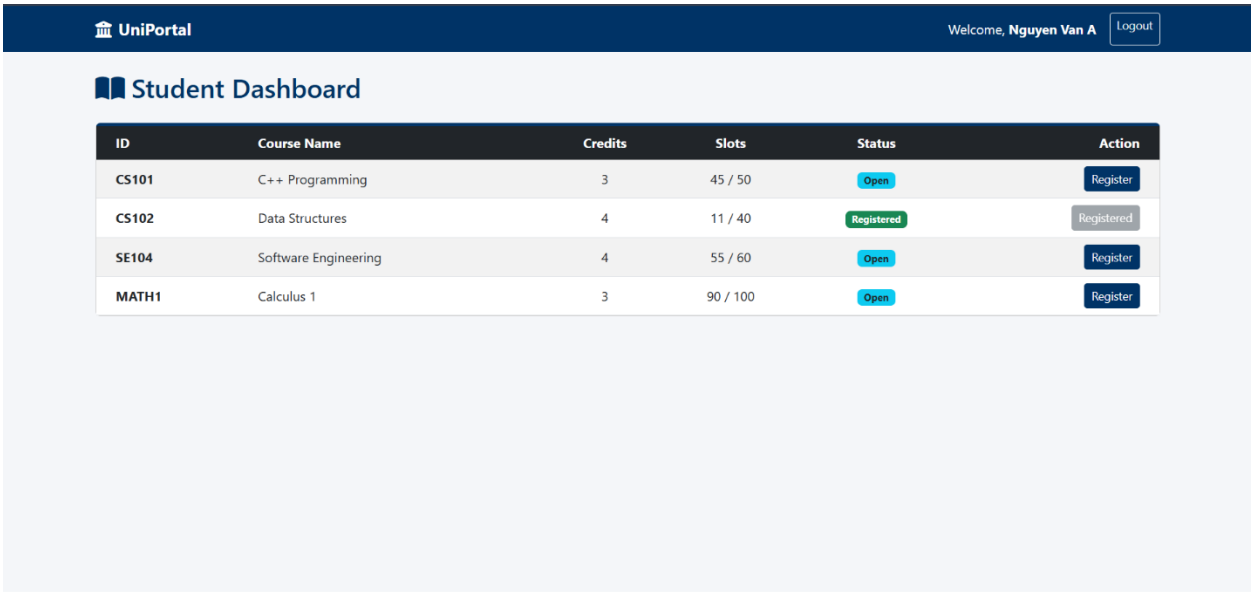
- Upon successful login, students are directed to the Student Dashboard. This interface aggregates course data and registration status into a single view.
- Student Dashboard
- Layout: A tabular view displaying the list of available courses (from the Course and Section classes).

Data Columns:

- ID & Course Name: clearly identifies the subject (e.g., CS101 - C++ rogramming).
-Credits: Indicates academic weight.
- Slots: Real-time capacity tracking (Current Enrolled / Max Capacity).
- Status: Visual badges (e.g., "Open" in Blue, "Registered" in Green) provide immediate feedback on course availability.

Action Buttons:

- Register: Enabled for "Open" courses with available slots.
- Registered (Disabled): Indicates the student is already enrolled, preventing duplicate entries.
- (Note: A "Drop" button would appear for registered courses during the valid period).



9.3. Lecturer Interface (Conceptual Design)

- Class List View: A dashboard similar to the Student view, but listing the classes the lecturer is teaching. Clicking a class expands the list of enrolled students.

- Grade Input Form: A data entry table allowing the lecturer to input or update numerical grades for each student ID. This interface connects directly to the Enrollment class in the Data Model to update the grade attribute.

9.4. Admin Interface (Conceptual Design)

- Management Dashboard: A sidebar navigation menu allowing access to "Course Management," "User Management," and "System Settings."
- Portal Control: A prominent toggle switch or button set to "Open Registration" or "Close Registration." This master control determines whether the "Register" buttons on the Student Dashboard are active or disabled.
- CRUD Forms: Detailed forms for adding new courses (Course ID, Name, Credits) and setting section capacities.