KINGDOM OF CAMBODIA
NATION RELIGION KING

Software Engineer

GROUP: I4-GIC-A

Report Progress Week 4

Name of Students	ID of Students	Score
1. HAN Raby	e20220580	....
2. CHEA Chanminea	e20220335	....
3. HENG Oulong	e20221390	....
4. LO Bunleang	e20220722	....
5. HUN Lyhorng	e20220188	....

Lecturer: Mr. ROEUN Pacharoth

Academic year 2025-2026

Course Enrollment & Classroom Scheduling System

I. HAN Raby

➢ Backend Development (Controllers & Services)

❖ Student Home & Course Display

Enhanced StudentController to:

- Retrieve the currently authenticated student using Spring Security.
- Fetch all available courses.
- Determine enrollment status for each course using EnrollmentRepository.
- Pass enriched course data (with enrollment status) to the student home page.

❖ Course Browsing & Course Detail

Updated StudentCourseController to:

- Display all courses for students.
- Show detailed course information.
- Check enrollment status per course dynamically.
- Display related courses while excluding the currently viewed course.

❖ Enrollment Management

Improved StudentEnrollmentController to:

- Display all enrollments of the logged-in student.
- Handle course enrollment using EnrollmentService.
- Handle dropping courses with validation and authorization.
- Implement proper redirect handling and flash success/error messages.

❖ Student Profile Management

Enhanced StudentProfileController to:

- Display student personal information and enrolled courses.
- Update profile details (name, email, date of birth).
- Upload and update profile images.
- Allow students to drop courses directly from the profile page.
- Integrate StudentProfileService for clean separation of logic.

❖ Student Schedule Feature

Implemented StudentScheduleService to:

- Retrieve the authenticated student.
- Fetch enrolled courses.
- Load schedules for enrolled courses only.

Updated StudentScheduleController to:

- Group schedules by day of the week.
- Provide both timetable and list-based schedule views.

- Frontend Development

  Student Home Page
  - Dynamic course cards.
  - Enrollment status handling.
  - Category filtering using JavaScript.

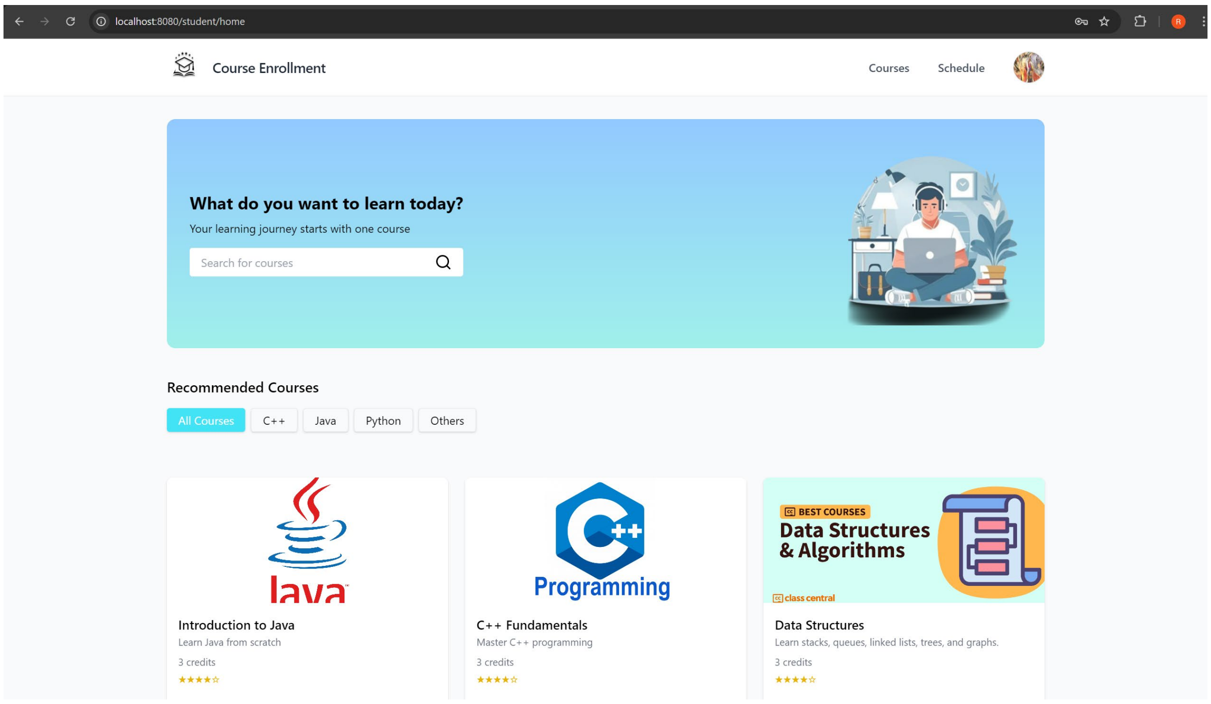
  Courses List Page
  - All courses view with enrollment indicators.

  Course Detail Page
  - Detailed course information.
  - Enrollment actions.
  - Related course recommendations.

  My Enrollments Page
  - List of enrolled courses.
  - Drop course functionality.
  - Enrollment date display.

  Student Profile Page
  - Profile information editing.
  - Profile image upload.
  - Enrolled course management.

  Student Schedule Page
  - Weekly schedule view.
  - Grouped by day.
  - Clean and readable timetable layout.
- Images


localhost:8080/student/home

Course Enrollment

Courses Schedule

What do you want to learn today?

Your learning journey starts with one course

Search for courses

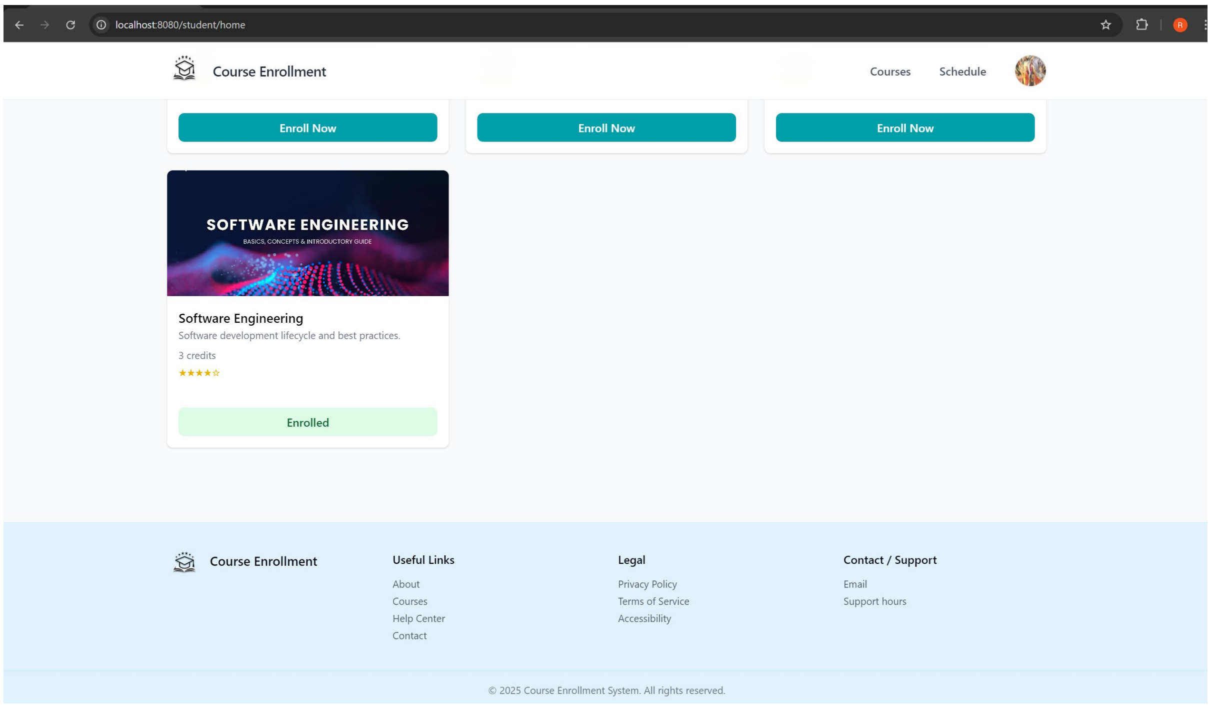
Recommended Courses

All Courses | C++ | Java | Python | Others

Introduction to Java
Learn Java from scratch
3 credits
★★★★☆

C++ Fundamentals
Master C++ programming
3 credits
★★★★☆

Data Structures
Learn stacks, queues, linked lists, trees, and graphs.
3 credits
★★★★☆


localhost:8080/student/home

Course Enrollment

Courses Schedule

Enroll Now | Enroll Now | Enroll Now

Software Engineering
Software development lifecycle and best practices.
3 credits
★★★★☆

Enrolled

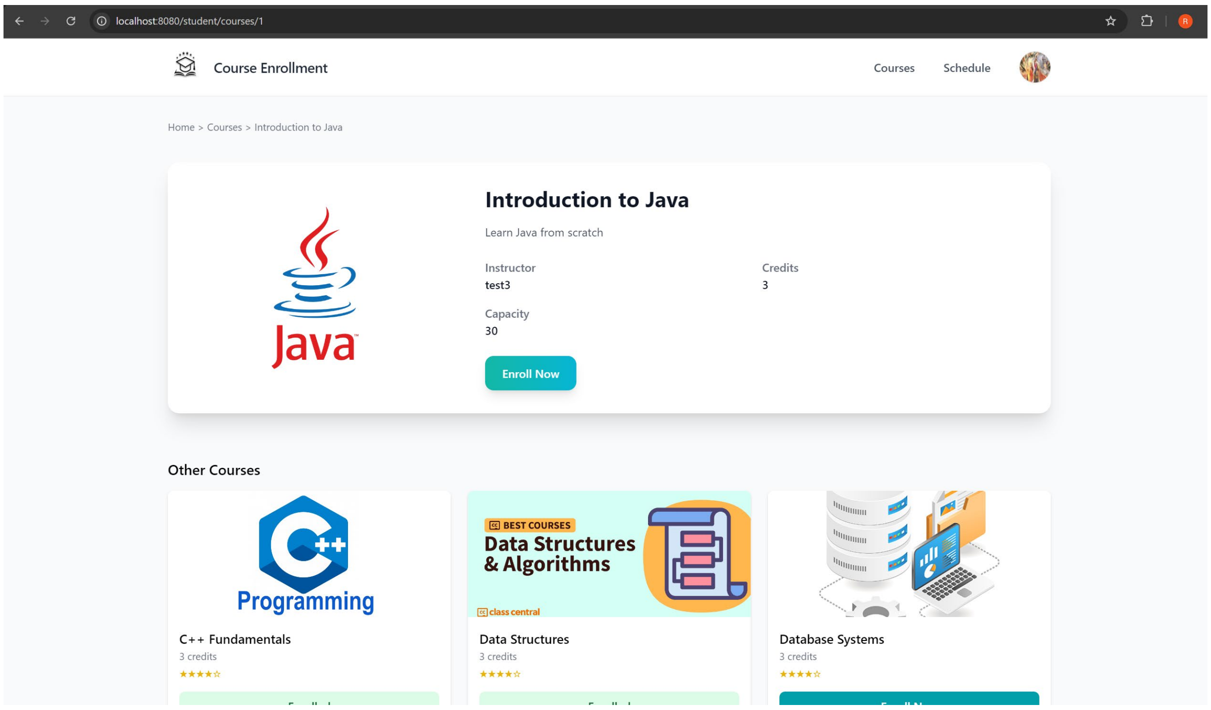
Course Enrollment

Useful Links
About
Courses
Help Center
Contact

Legal
Privacy Policy
Terms of Service
Accessibility

Contact / Support
Email
Support hours

© 2025 Course Enrollment System. All rights reserved.


localhost:8080/student/courses/1

Course Enrollment

Courses Schedule

Home > Courses > Introduction to Java

Introduction to Java

Learn Java from scratch

Instructor
test3

Credits
3

Capacity
30

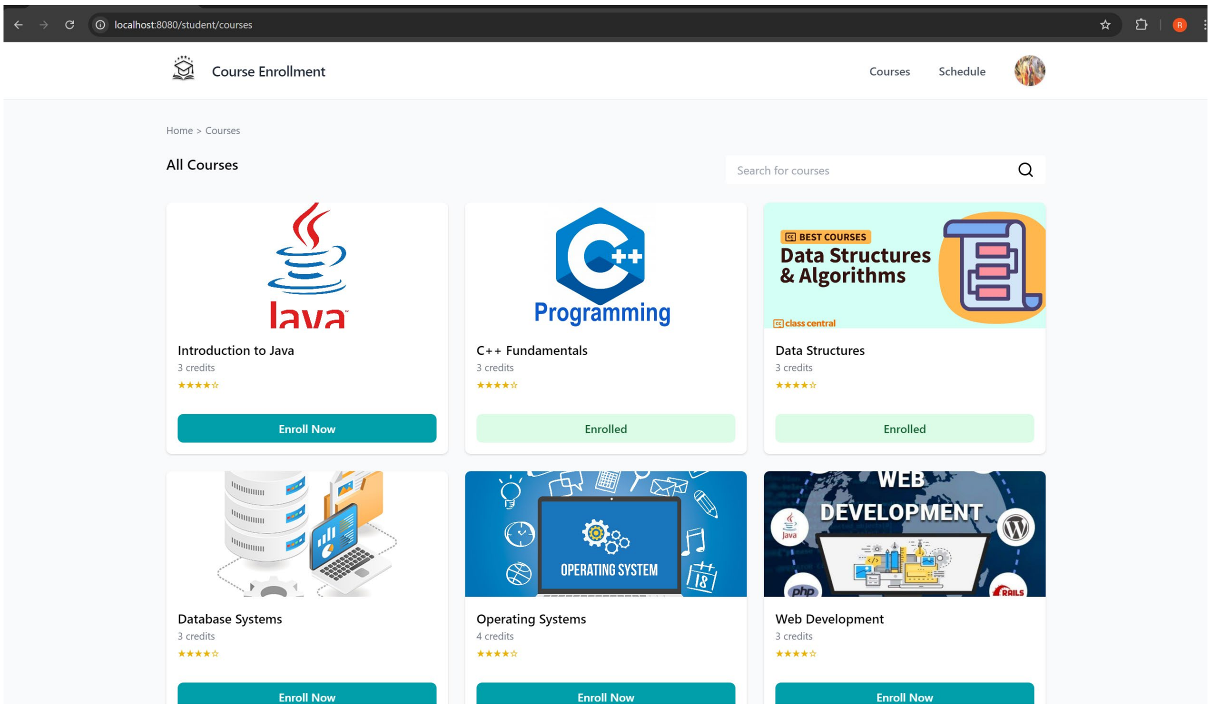
Enroll Now

Other Courses

C++ Fundamentals
3 credits
★★★★☆

Data Structures
3 credits
★★★★☆

Database Systems
3 credits
★★★★☆


localhost:8080/student/courses

Course Enrollment

Courses Schedule

Home > Courses

All Courses

Search for courses

Introduction to Java
3 credits
★★★★☆
Enroll Now

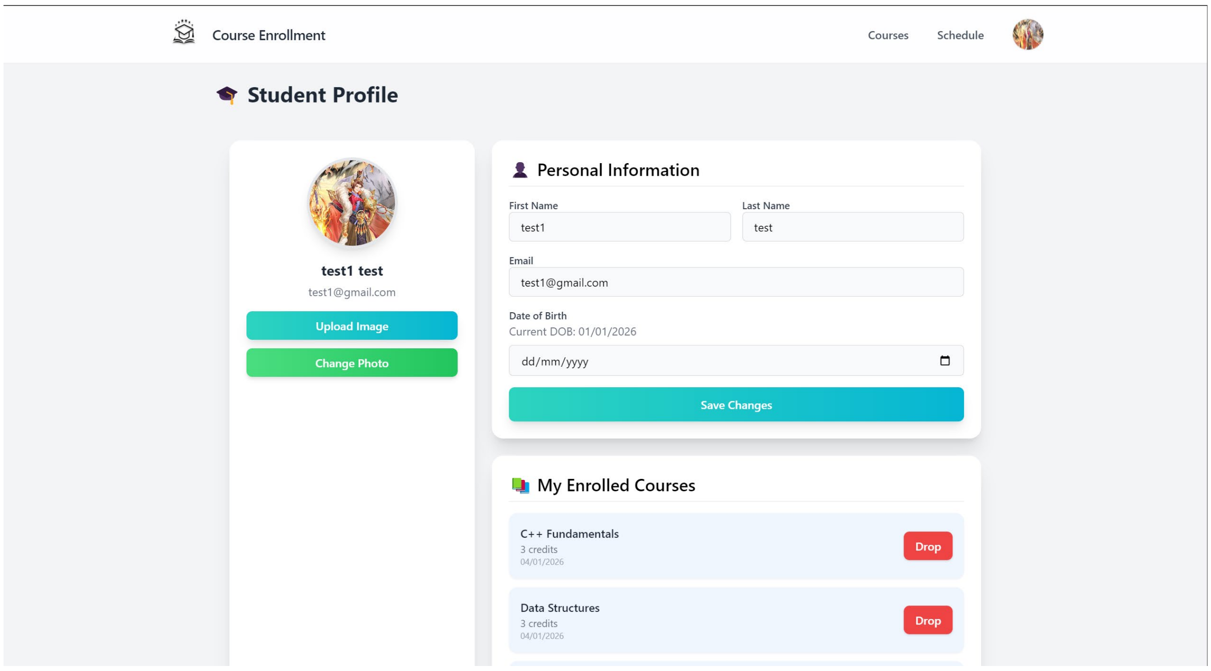
C++ Fundamentals
3 credits
★★★★☆
Enrolled

Data Structures
3 credits
★★★★☆
Enrolled

Database Systems
3 credits
★★★★☆
Enroll Now

Operating Systems
4 credits
★★★★☆
Enroll Now

Web Development
3 credits
★★★★☆
Enroll Now


Course Enrollment

Courses Schedule

Student Profile

test1 test
test1@gmail.com

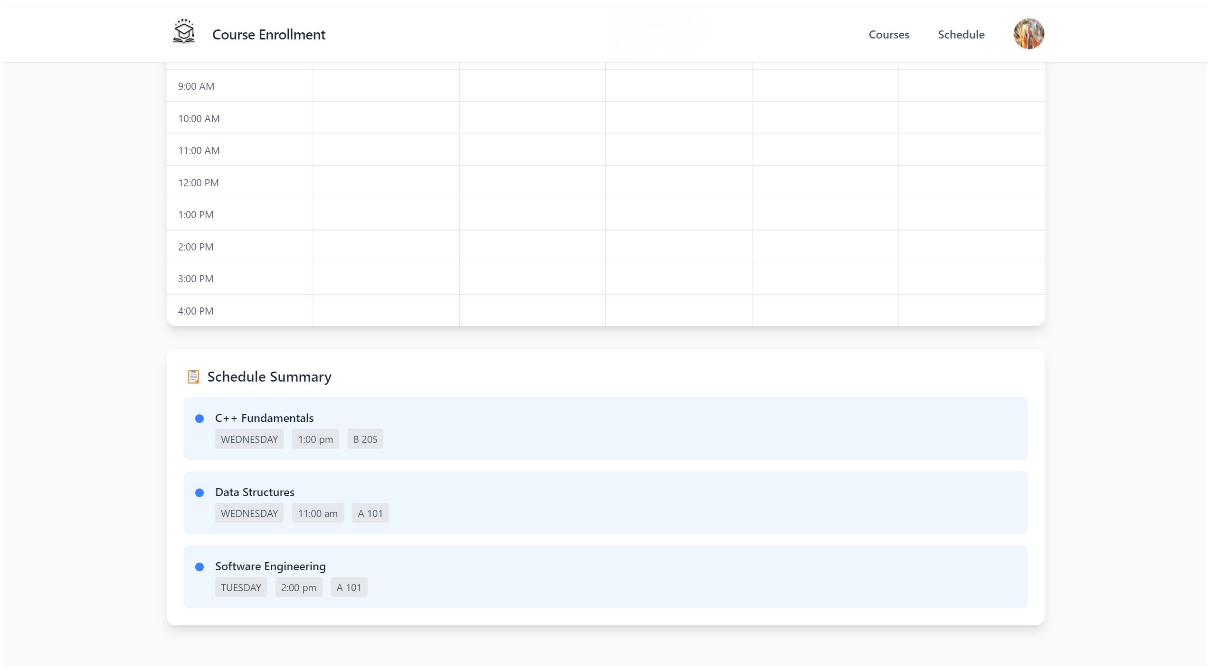
Personal Information

First Name: test1
Last Name: test
Email: test1@gmail.com
Date of Birth
Current DOB: 01/01/2026
dd/mm/yyyy

My Enrolled Courses

C++ Fundamentals
3 credits
04/01/2026

Data Structures
3 credits
04/01/2026


Course Enrollment

Courses Schedule

9:00 AM					
10:00 AM					
11:00 AM					
12:00 PM					
1:00 PM					
2:00 PM					
3:00 PM					
4:00 PM					

Schedule Summary

- C++ Fundamentals
  WEDNESDAY 1:00 pm B 205
- Data Structures
  WEDNESDAY 11:00 am A 101
- Software Engineering
  TUESDAY 2:00 pm A 101

II. LO Bunleang

1. Design in figma

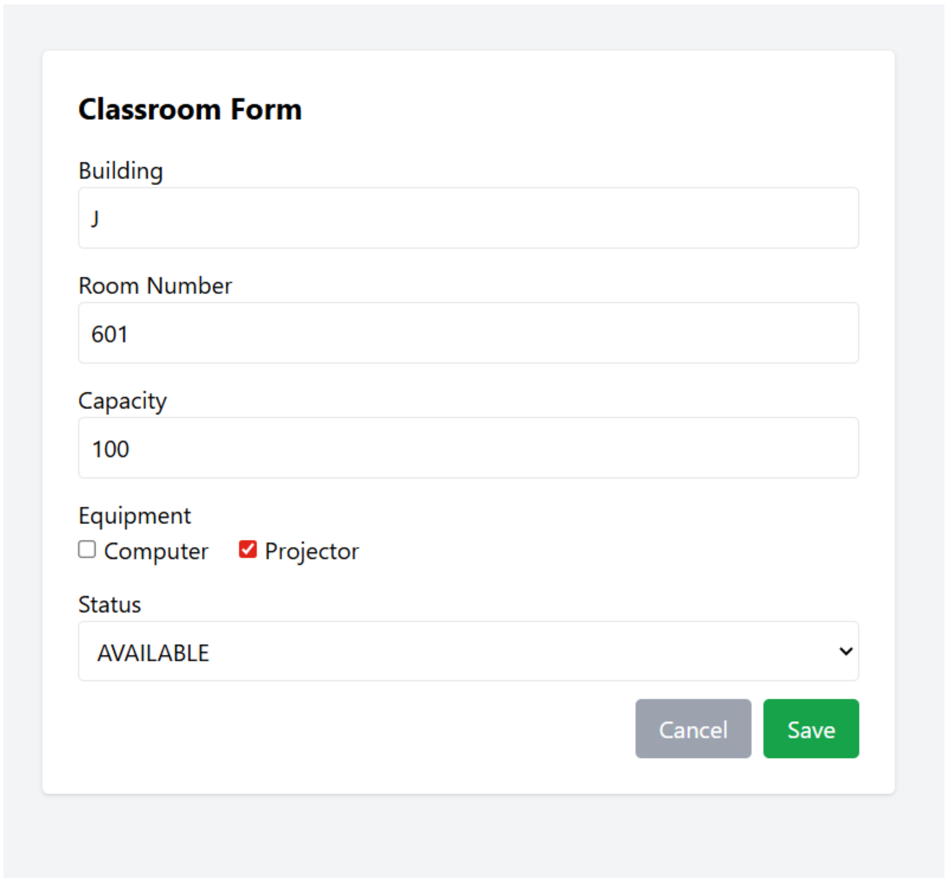
- admin Classroom management
- admin Schedule management
- Student Schedule view
- Link: https://www.figma.com/design/y2mzEZUAfgVZlEZw4WlzoC/Vue-Js?node-id=755-37&p=f&t=bpOdhWVpedvKJyaL-0

2. Implementation and testing

1. Classroom Management (Admin)

- Designed and implemented full CRUD operations for Classroom
- Created admin pages:
    - Classroom list
    - Classroom create form
    - Classroom edit form
- Classroom includes:
    - Building
    - Room number
    - Capacity
    - Equipment (Computer, Projector)
    - Status (AVAILABLE, MAINTENANCE, CLOSED)
- Displayed classroom status clearly using color indicators
- Implemented delete and update actions with proper routing
- Verified data persistence in the database

Create / Edit Classroom (form)



Classroom Form

Building: J

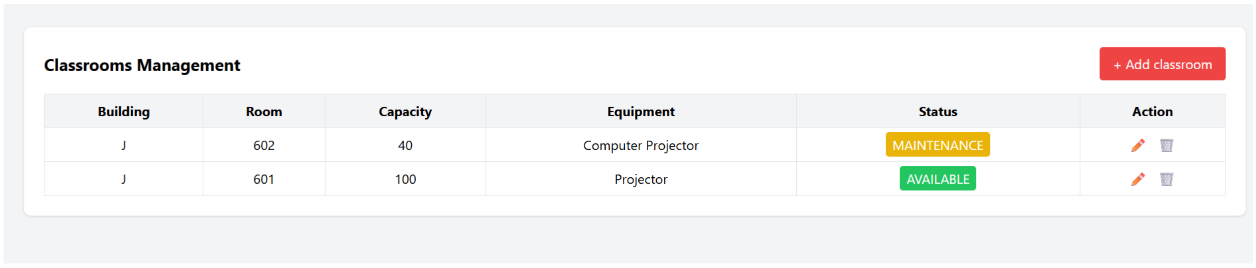
Room Number: 601

Capacity: 100

Equipment: [ ] Computer [✓ Projector]

Status: AVAILABLE

List of Classroom

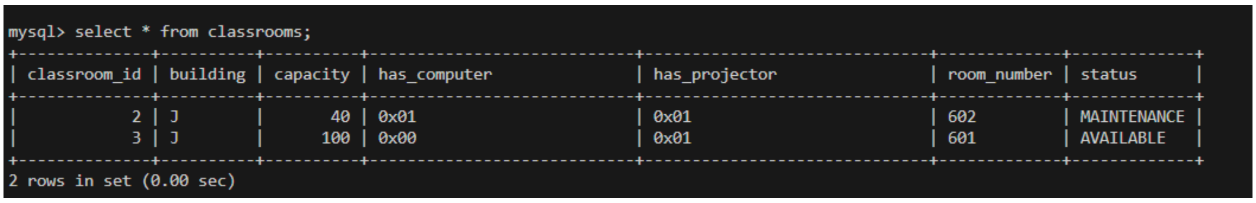


Classrooms Management

+ Add classroom

Building	Room	Capacity	Equipment	Status	Action
J	602	40	Computer Projector	MAINTENANCE	✎ 🗑
J	601	100	Projector	AVAILABLE	✎ 🗑

Classroom table in mysql

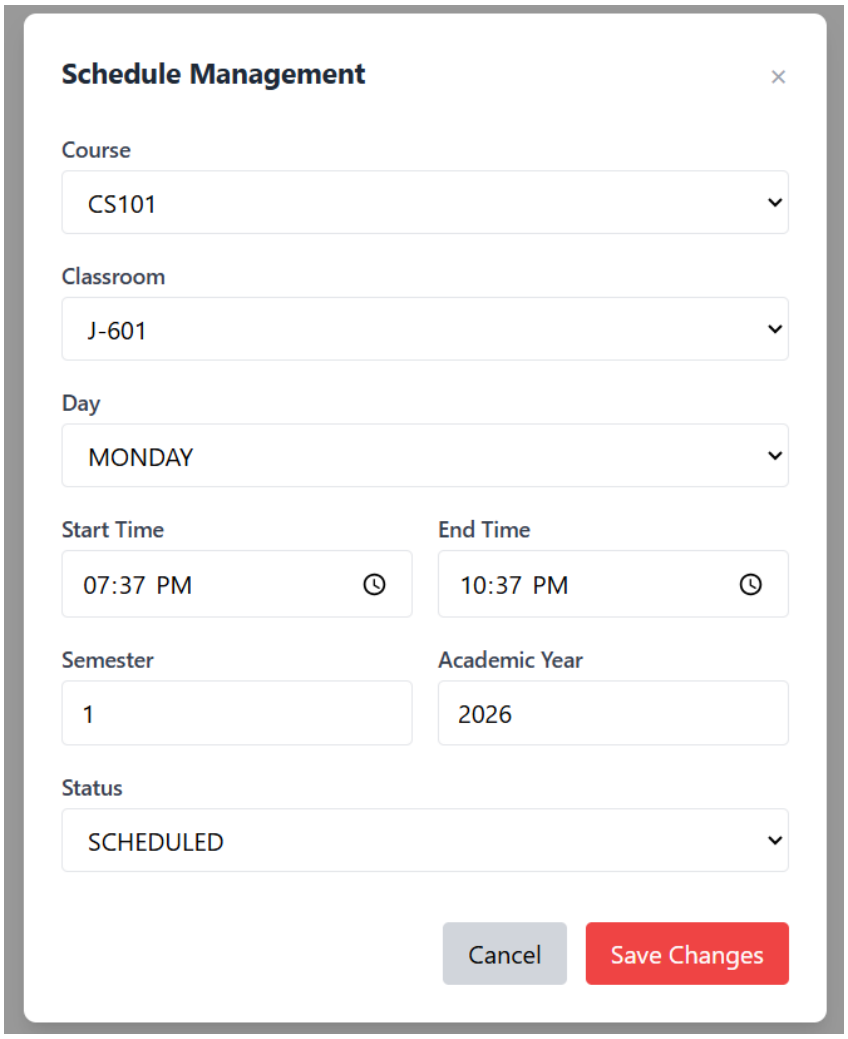


```
mysql> select * from classrooms;
+--------------+----------+----------+------------------------------+------------------------------+-------------+-------------+
| classroom_id | building | capacity | has_computer                 | has_projector                | room_number | status      |
+--------------+----------+----------+------------------------------+------------------------------+-------------+-------------+
|            2 | J        |       40 | 0x01                         | 0x01                         | 602         | MAINTENANCE |
|            3 | J        |      100 | 0x00                         | 0x01                         | 601         | AVAILABLE   |
+--------------+----------+----------+------------------------------+------------------------------+-------------+-------------+
2 rows in set (0.00 sec)
```

2. **Schedule Management (Admin)**

- Implemented **full CRUD operations** for Schedule
- Designed admin pages:
  - Schedule list page
  - Schedule create / edit form
- Schedule includes:
  - Course
  - Classroom
  - Academic year
  - Semester
  - Day of week
  - Start time and end time
  - Status (SCHEDULED, CANCELLED)
- Connected Schedule to:
  - Course table
  - Classroom table
- Populated course and classroom selections dynamically from the database
- Ensured schedules cannot be created without valid course and classroom data

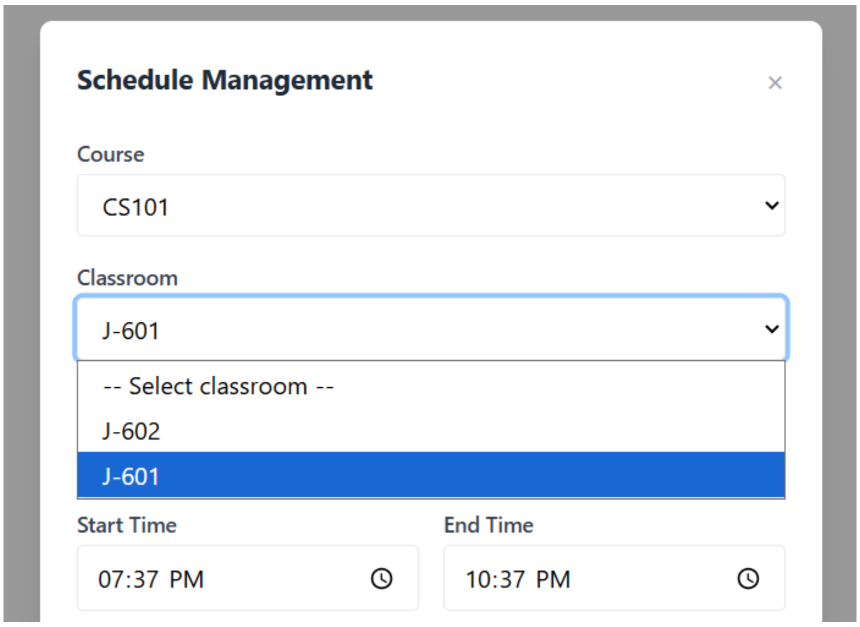
Create/ edit form


Schedule Management

×

Course	CS101
Classroom	J-601
Day	MONDAY
Start Time	07:37 PM
End Time	10:37 PM
Semester	1
Academic Year	2026
Status	SCHEDULED

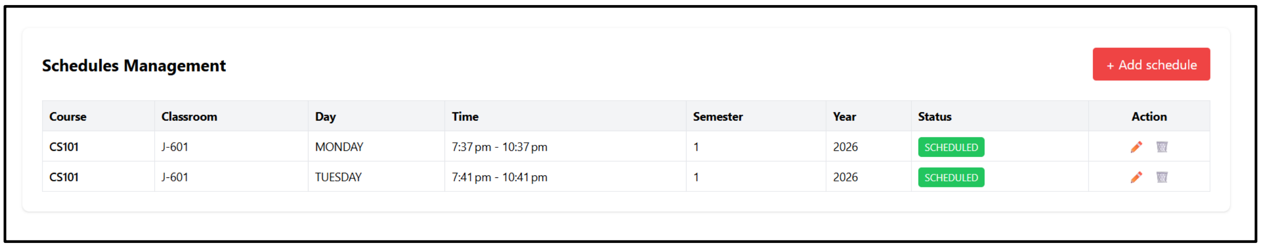
Classroom and course selection depend on real dynamic database


Schedule Management

×

Course	CS101
Classroom	J-601
Start Time	07:37 PM
End Time	10:37 PM

List of schedule

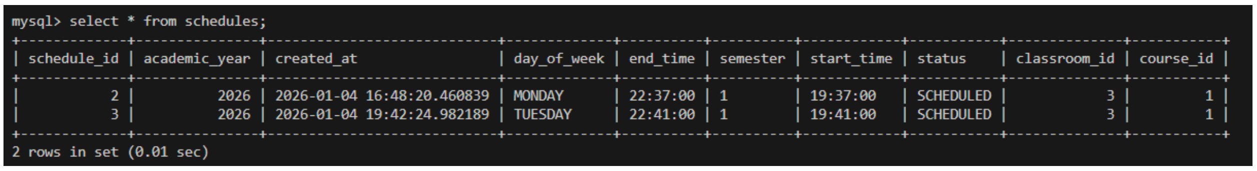


Schedules Management

+ Add schedule

Course	Classroom	Day	Time	Semester	Year	Status	Action
CS101	J-601	MONDAY	7:37 pm - 10:37 pm	1	2026	SCHEDULED	
CS101	J-601	TUESDAY	7:41 pm - 10:41 pm	1	2026	SCHEDULED	

Database

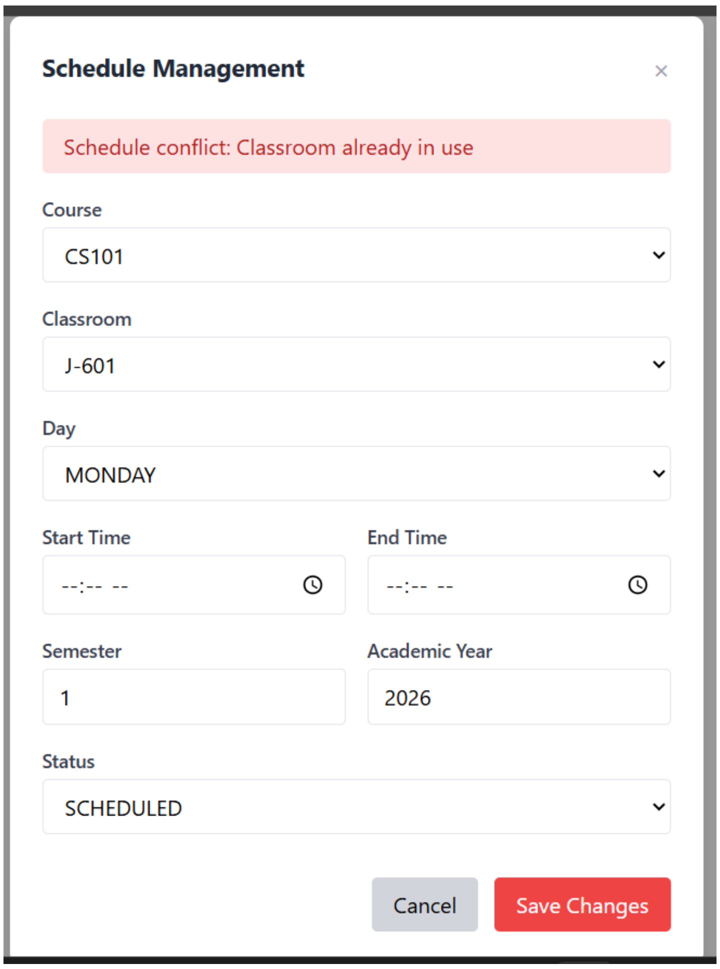


```
mysql> select * from schedules;
+-------------+---------------+----------------------------+-------------+----------+----------+------------+-----------+--------------+-----------+
| schedule_id | academic_year | created_at                 | day_of_week | end_time | semester | start_time | status    | classroom_id | course_id |
+-------------+---------------+----------------------------+-------------+----------+----------+------------+-----------+--------------+-----------+
|           2 |          2026 | 2026-01-04 16:48:20.460839 | MONDAY      | 22:37:00 | 1        | 19:37:00   | SCHEDULED |            3 |         1 |
|           3 |          2026 | 2026-01-04 19:42:24.982189 | TUESDAY     | 22:41:00 | 1        | 19:41:00   | SCHEDULED |            3 |         1 |
+-------------+---------------+----------------------------+-------------+----------+----------+------------+-----------+--------------+-----------+
2 rows in set (0.01 sec)
```

3. Schedule Conflict Validation (Business Rule)

- Implemented **time conflict validation** at the service layer
- Prevented overlapping schedules for:
  - Same classroom
  - Same day
  - Overlapping time ranges
- Used repository-level existence checks for efficiency
- Applied validation for both:
  - Schedule creation
  - Schedule update
- Displayed user-friendly error messages in the admin UI when conflicts occur

Prevent Conflict Schedule


Schedule Management

Schedule conflict: Classroom already in use

Course
CS101

Classroom
J-601

Day
MONDAY

Start Time
--:-- --

End Time
--:-- --

Semester
1

Academic Year
2026

Status
SCHEDULED

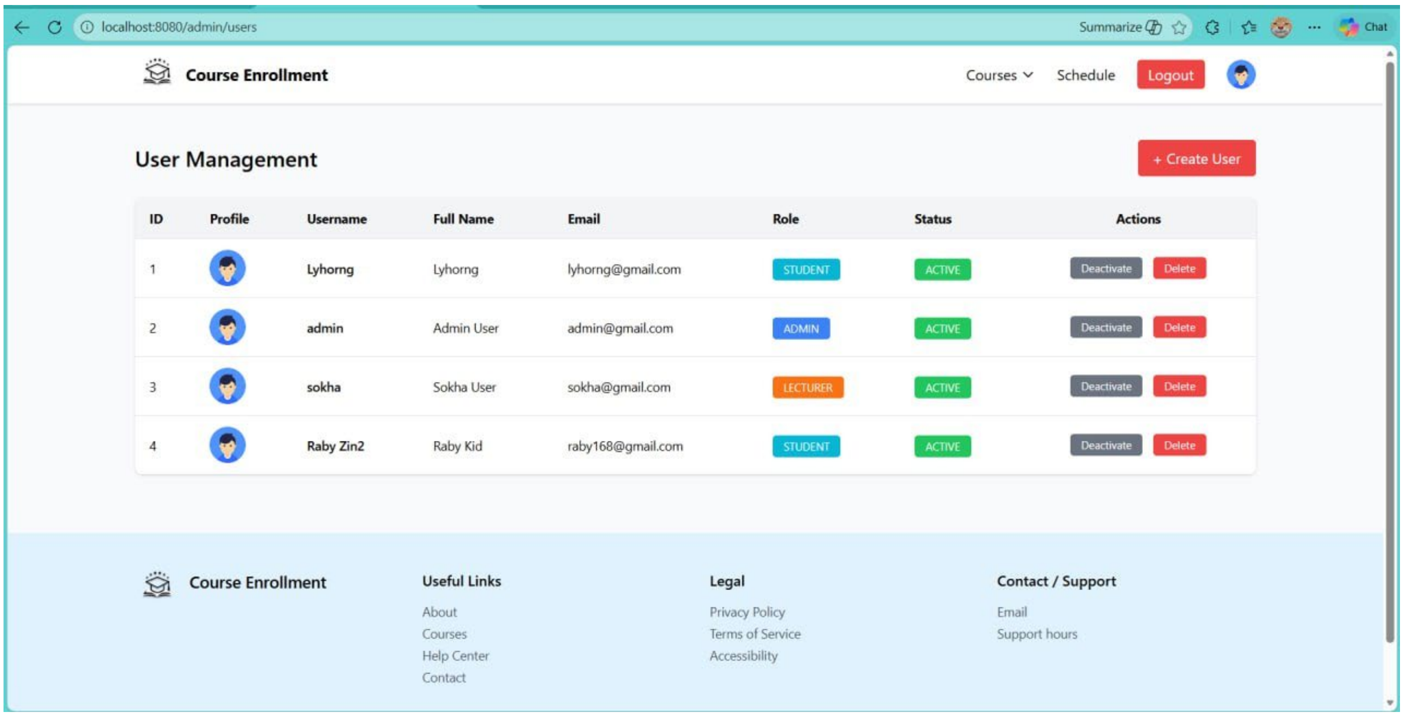
III. HUN Lyhorng

➢ Last week I have finished designing UI/UX in figma for student and finished building homepage with header and footer.

➢ This week I do admin part on creating the user management:
  - Designed and implemented the Admin User Management dashboard using Thymeleaf and Tailwind CSS.
  - Displayed a list of users including profile image, username, full name, email, role, and status.
  - Implemented a Create User button that routes to a dedicated user creation form.
  - Built the Create User form allowing admin to input:
    - Username
    - Email
    - First name
    - Last name
    - Role
    - Status

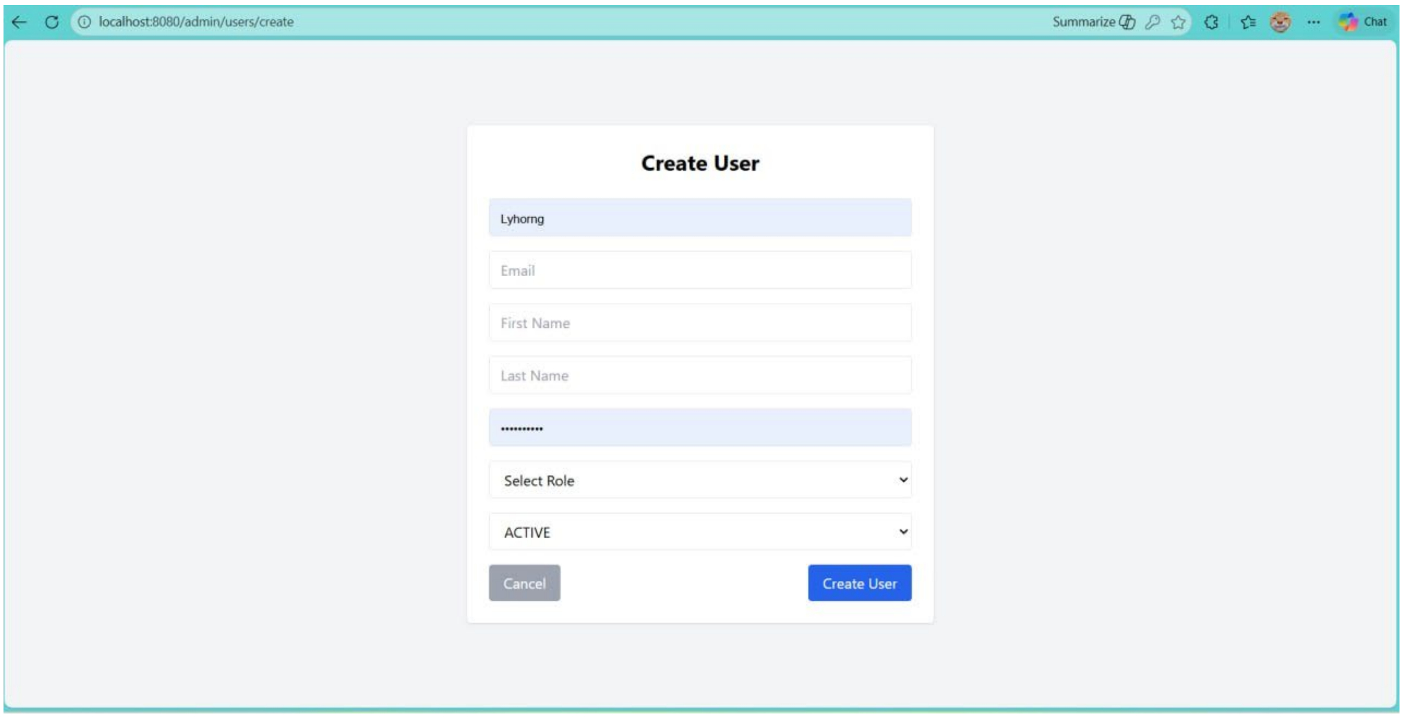
- Password (required)

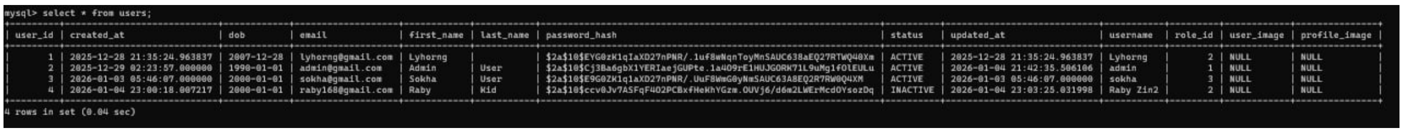
- Ensured the password is not displayed on the users list page for security.
- Created DTOs (UserCreateDto, UserListDto) to separate form data and display data from the entity.
- Implemented UserService to handle business logic:
- Validate unique username and email
- Hash passwords using BCrypt
- Assign roles and statuses
- Save users to the database
- Toggle user status (Active / Inactive)
- Delete users from the database
- Connected the service layer with AdminUserController for clean request handling.
- Applied role-based access control, ensuring only ADMIN users can access admin endpoints.
- Successfully tested the full flow from UI to database insertion.
- Confirmed user data is stored correctly in the users table with hashed passwords and timestamps.



User Management

ID	Profile	Username	Full Name	Email	Role	Status	Actions
1		Lyhorng	Lyhorng	lyhorng@gmail.com	STUDENT	ACTIVE	Deactivate Delete
2		admin	Admin User	admin@gmail.com	ADMIN	ACTIVE	Deactivate Delete
3		sokha	Sokha User	sokha@gmail.com	LECTURER	ACTIVE	Deactivate Delete
4		Raby Zin2	Raby Kid	raby168@gmail.com	STUDENT	ACTIVE	Deactivate Delete



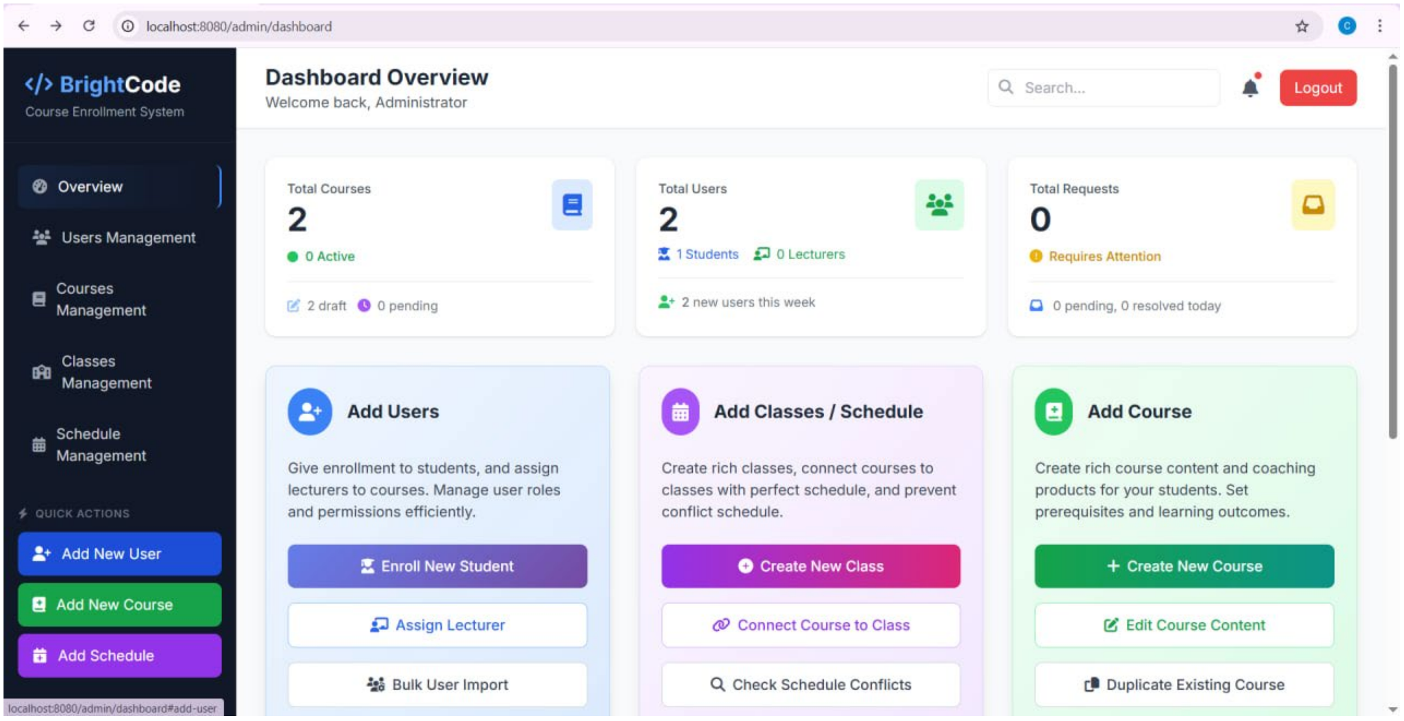


```
mysql> select * from users;
+---------+----------------------------+------------+---------------------+------------+-----------+--------------------------------------------------------------+----------+----------------------------+------------+---------+------------+---------------+
| user_id | created_at                 | dob        | email               | first_name | last_name | password_hash                                                | status   | updated_at                 | username   | role_id | user_image | profile_image |
+---------+----------------------------+------------+---------------------+------------+-----------+--------------------------------------------------------------+----------+----------------------------+------------+---------+------------+---------------+
|       1 | 2025-12-28 21:35:24.963837 | 2007-12-28 | lyhorng@gmail.com   | Lyhorng    |           | $2a$10$EYG0zK1qIaXD27nPNR/.1uf8wNqnToyMnSAUC638aEQ27RTWQ40Xm | ACTIVE   | 2025-12-28 21:35:24.963837 | Lyhorng    |       2 | NULL       | NULL          |
|       2 | 2025-12-29 02:23:57.000000 | 1990-01-01 | admin@gmail.com     | Admin      | User      | $2a$10$Cj3Ba6gbX1YERIaejGUPte.1a409rE1HUJGORK71L9uMg1fOlEULu | ACTIVE   | 2026-01-04 21:42:35.506106 | admin      |       1 | NULL       | NULL          |
|       3 | 2026-01-03 05:46:07.000000 | 2000-01-01 | sokha@gmail.com     | Sokha      | User      | $2a$10$E9G0ZK1q1aXD27nPNR/.UuF8WmG0yNmSAUC63A8EQ2R7RW0Q4XM   | ACTIVE   | 2026-01-03 05:46:07.000000 | sokha      |       3 | NULL       | NULL          |
|       4 | 2026-01-04 23:00:18.007217 | 2000-01-01 | raby168@gmail.com   | Raby       | Kid       | $2a$10$ccv0Jv7ASFqF402PCBxfHeKhYGzm.OUVj6/d6m2LWErMcdOYsozDq | INACTIVE | 2026-01-04 23:03:25.031998 | Raby Zin2  |       2 | NULL       | NULL          |
+---------+----------------------------+------------+---------------------+------------+-----------+--------------------------------------------------------------+----------+----------------------------+------------+---------+------------+---------------+
4 rows in set (0.04 sec)
```

IV. CHEA Chanminea

➢ After building static UI for admin dashboard, I have made it dynamic and more statistical such below details:

- Connect all buttons to services that support our system.
- Add log out button and more button on the left-hand side section.
- For course counting part, we can count total, and others courses by status.
- For user counting part, we can count total users, students, lecturers, and users within a week.
- For counting request part, we can count total alerts if enrollment statuses are pending. Further, if those enrollments change to 'ENROLLED', the resolved today will be counted.
- For enrollment part, we can also track the last enrollment by displaying the enrollment date.



localhost:8080/admin/dashboard

</> BrightCode
Course Enrollment System

- Overview
- Users Management
- Courses Management
- Classes Management
- Schedule Management

QUICK ACTIONS
- Add New User
- Add New Course
- Add Schedule

localhost:8080/admin/dashboard#add-user

Dashboard Overview
Welcome back, Administrator

Search... Logout

Total Courses
2
● 0 Active
2 draft 0 pending

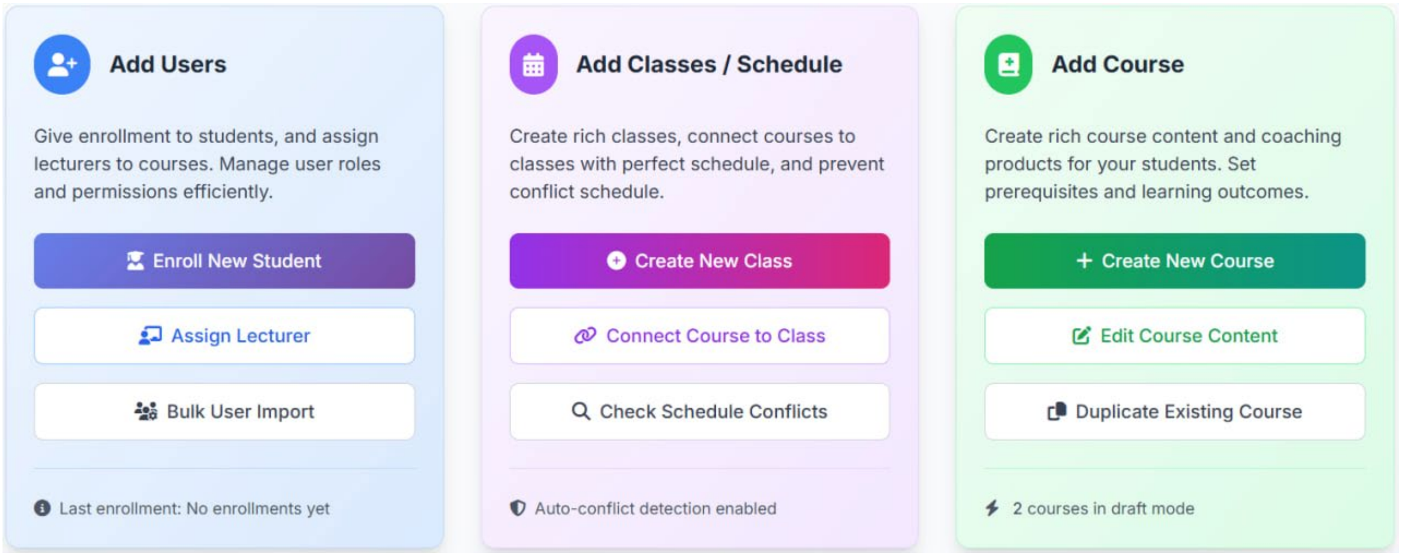
Total Users
2
1 Students 0 Lecturers
2 new users this week

Total Requests
0
Requires Attention
0 pending, 0 resolved today

Add Users
Give enrollment to students, and assign lecturers to courses. Manage user roles and permissions efficiently.
Enroll New Student | Assign Lecturer | Bulk User Import

Add Classes / Schedule
Create rich classes, connect courses to classes with perfect schedule, and prevent conflict schedule.
Create New Class | Connect Course to Class | Check Schedule Conflicts

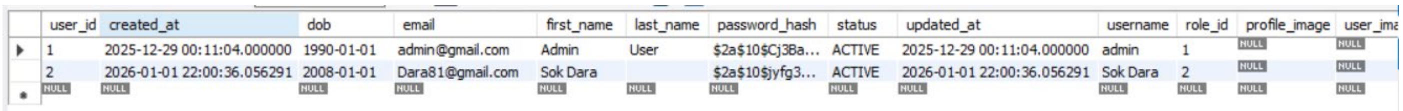
Add Course
Create rich course content and coaching products for your students. Set prerequisites and learning outcomes.
Create New Course | Edit Course Content | Duplicate Existing Course



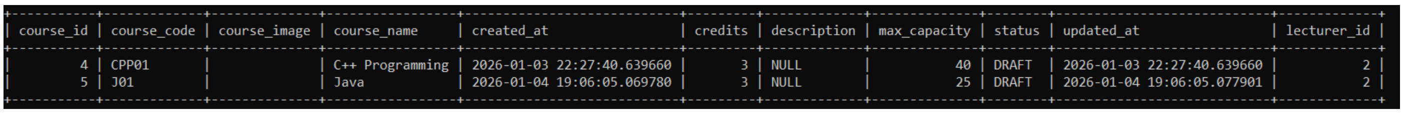
Add Users
Give enrollment to students, and assign lecturers to courses. Manage user roles and permissions efficiently.
Enroll New Student | Assign Lecturer | Bulk User Import
Last enrollment: No enrollments yet

Add Classes / Schedule
Create rich classes, connect courses to classes with perfect schedule, and prevent conflict schedule.
Create New Class | Connect Course to Class | Check Schedule Conflicts
Auto-conflict detection enabled

Add Course
Create rich course content and coaching products for your students. Set prerequisites and learning outcomes.
Create New Course | Edit Course Content | Duplicate Existing Course
2 courses in draft mode



user_id	created_at	dob	email	first_name	last_name	password_hash	status	updated_at	username	role_id	profile_image	user_ima
1	2025-12-29 00:11:04.000000	1990-01-01	admin@gmail.com	Admin	User	$2a$10$Cj3Ba...	ACTIVE	2025-12-29 00:11:04.000000	admin	1	NULL	NULL
2	2026-01-01 22:00:36.056291	2008-01-01	Dara81@gmail.com	Sok Dara		$2a$10$jyfg3...	ACTIVE	2026-01-01 22:00:36.056291	Sok Dara	2	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



```
+-----------+-------------+--------------+-----------------+----------------------------+---------+-------------+--------------+--------+----------------------------+-------------+
| course_id | course_code | course_image | course_name     | created_at                 | credits | description | max_capacity | status | updated_at                 | lecturer_id |
+-----------+-------------+--------------+-----------------+----------------------------+---------+-------------+--------------+--------+----------------------------+-------------+
|         4 | CPP01       |              | C++ Programming | 2026-01-03 22:27:40.639660 |       3 | NULL        |           40 | DRAFT  | 2026-01-03 22:27:40.639660 |           2 |
|         5 | J01         |              | Java            | 2026-01-04 19:06:05.069780 |       3 | NULL        |           25 | DRAFT  | 2026-01-04 19:06:05.077901 |           2 |
+-----------+-------------+--------------+-----------------+----------------------------+---------+-------------+--------------+--------+----------------------------+-------------+
```

V. HENG Oulong

1. Admin Course Management Module

1.1 Course CRUD (Admin)

This week, I completed and stabilized the **Admin Course Management** module.

- Prevented template parsing and binding errors.

3. Course Status Handling (ACTIVE / DRAFT)

3.1 Status Support Added

- Added CourseStatus (ACTIVE, DRAFT) to the system.
- Updated CourseCreateUpdateDto to include CourseStatus status.

3.2 Business Logic Improvements

- Default course status set to DRAFT if not provided.
- Admin can select course status during **create and update**.
- Students can only see courses with status ACTIVE.

4. CourseService Refactoring & Stability

4.1 Service Layer Improvements

- Refactored CourseService to:
    - Cleanly separate **create**, **update**, **read**, and **delete** logic
    - Centralize DTO ↔ Entity mapping
- Added **safe image upload handling** with try-catch.

4.2 Exception Handling

- Added try-catch blocks around:
    - Cloudinary image upload
    - Database operations
- Wrapped checked exceptions into meaningful runtime errors.

5. Enrollment Logic Improvements

5.1 Max Capacity Enforcement

- Implemented course capacity logic:
    - Prevent enrollment if course is full.
    - Calculated current enrollment count dynamically.
- Disabled enroll button when:
    - Course is full

        - Student already enrolled

6. Student Course Module

6.1 Student Course List

- Students can view only **ACTIVE** courses.
- Fixed incorrect mappings and routing issues.
- Cleaned controller paths to avoid duplicate /student/courses/student/courses.

7. Thymeleaf Template Fixes

7.1 Common Issues Fixed

- Fixed:
    - Missing model attributes
    - Incorrect th:field usage
    - Invalid expressions causing template parsing errors

7.2 Admin Templates Updated

- admin/courses.html
- admin/course-create.html
- admin/course-edit.html

Improvements include:

- Clean layout structure
- Placeholder for side panel component (to be integrated later)
- Modern Tailwind-based styling

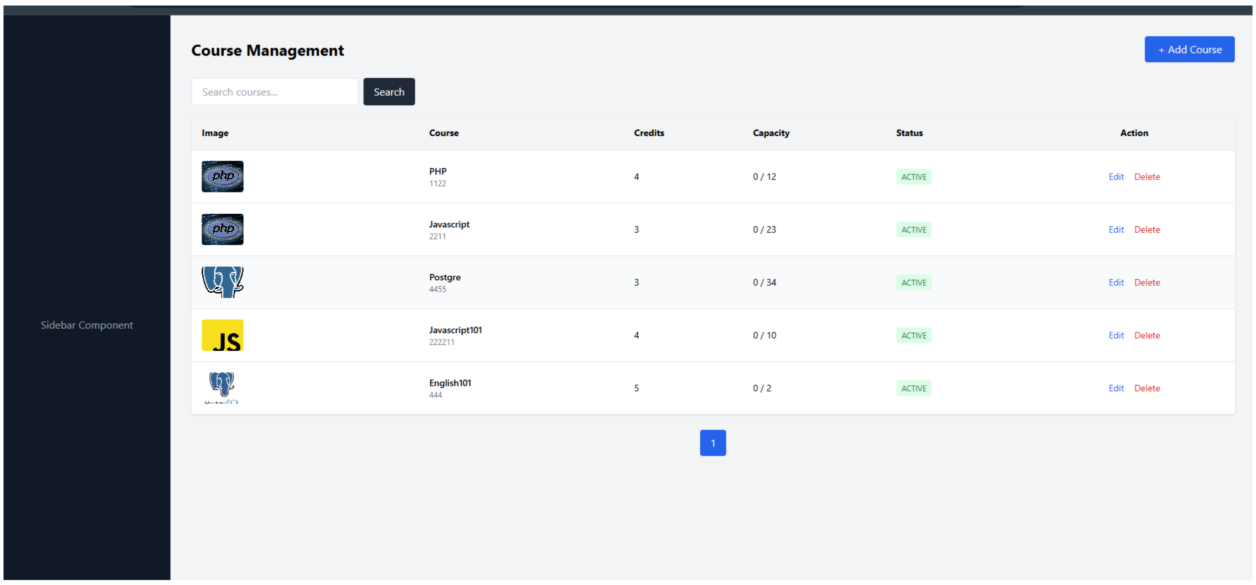
8. Pagination & Search (Backend Ready)

- Added backend pagination logic using:
    - Page
    - PageRequest
- Implemented search by course name (case-insensitive).
- Prepared admin UI for pagination (design-ready, logic connected).

9. Code Quality & Architecture Improvements

- Ensured proper DTO usage:
    - CourseDto → for display
    - CourseCreateUpdateDto → for form submission

Admin CourseEnrollments



Sidebar Component

Course Management

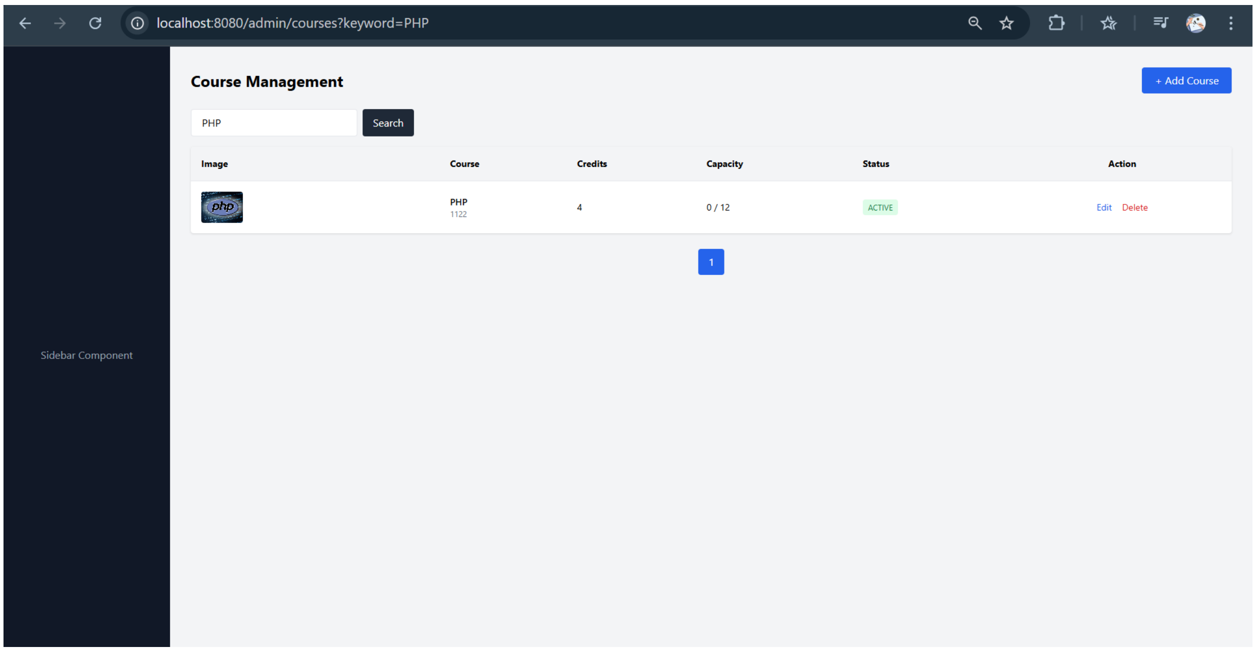
+ Add Course

Search courses... | Search

Image	Course	Credits	Capacity	Status	Action
php	PHP 1122	4	0 / 12	ACTIVE	Edit Delete
php	Javascript 2211	3	0 / 23	ACTIVE	Edit Delete
	Postgre 4455	3	0 / 34	ACTIVE	Edit Delete
JS	Javascript101 222211	4	0 / 10	ACTIVE	Edit Delete
	English101 444	5	0 / 2	ACTIVE	Edit Delete

1

After searching



localhost:8080/admin/courses?keyword=PHP

Sidebar Component

Course Management

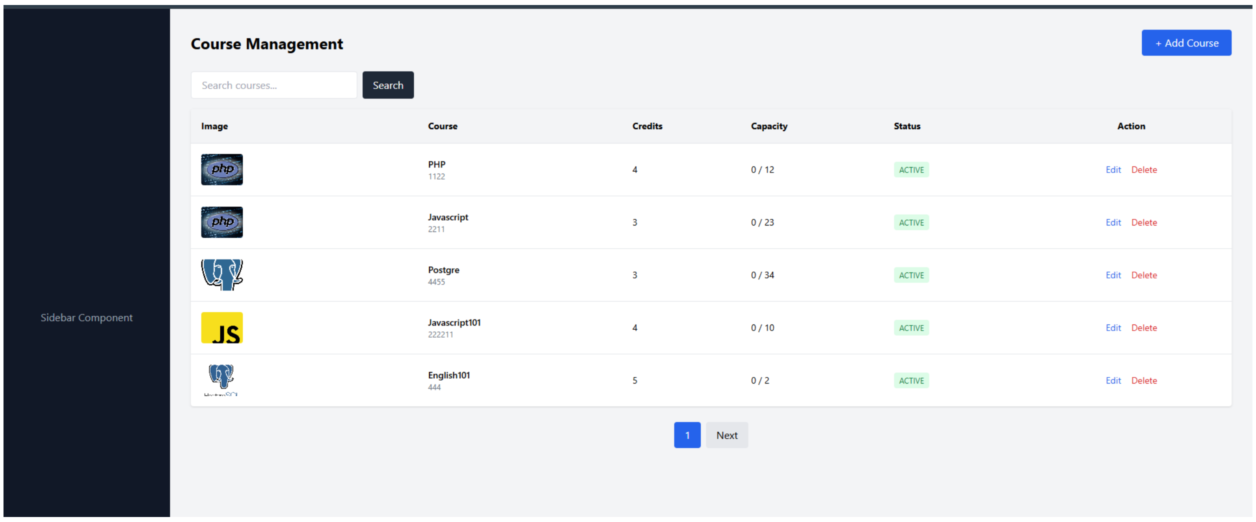
+ Add Course

PHP | Search

Image	Course	Credits	Capacity	Status	Action
php	PHP 1122	4	0 / 12	ACTIVE	Edit Delete

1

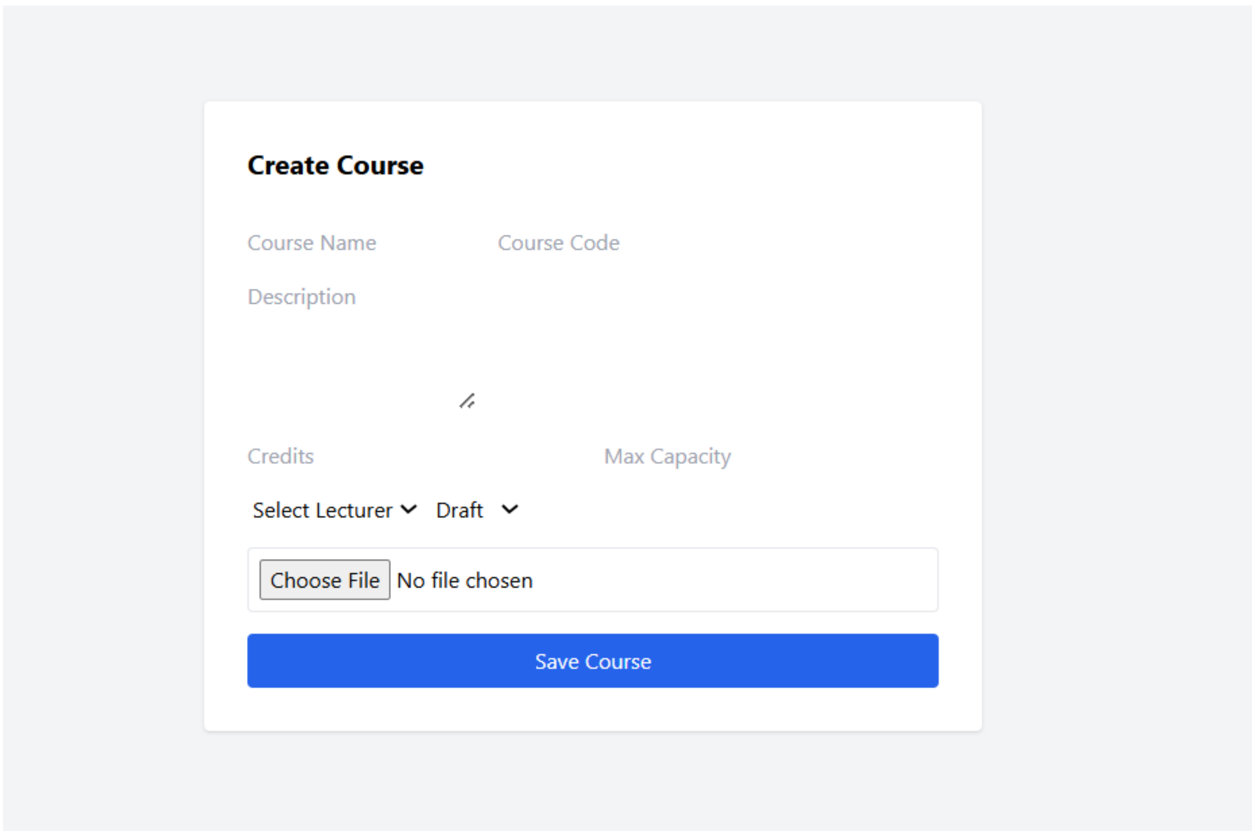
After from page 1 to page 2 (Press Next)



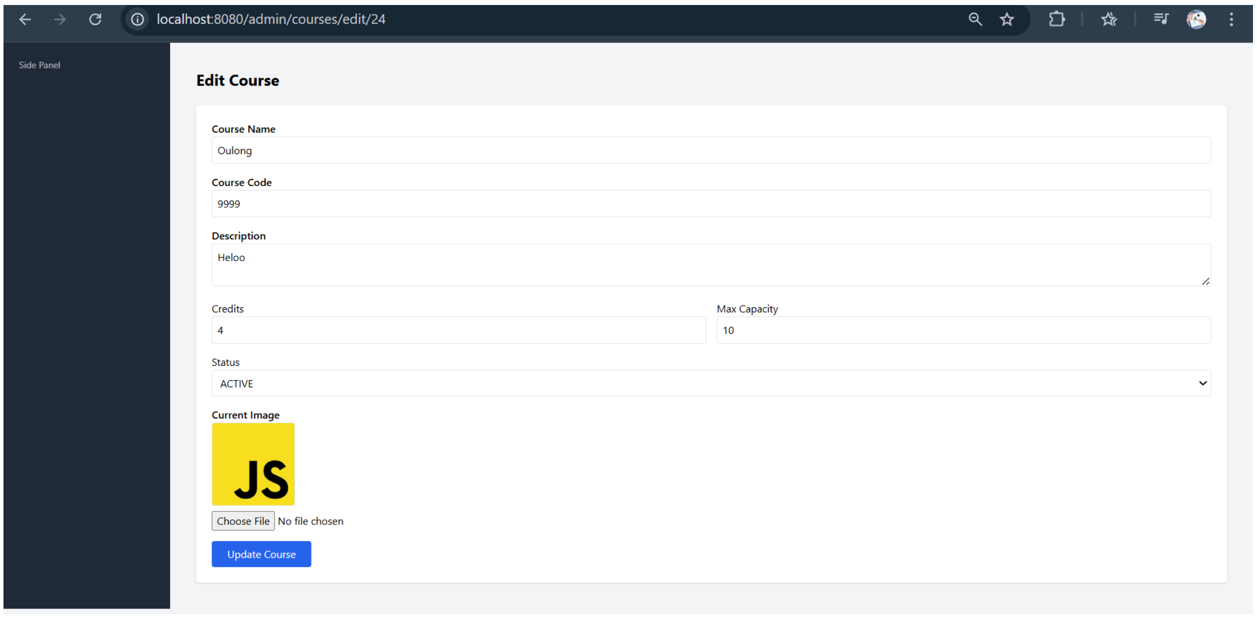
Course Management

Image	Course	Credits	Capacity	Status	Action
	PHP 1122	4	0 / 12	ACTIVE	Edit Delete
	Javascript 2211	3	0 / 23	ACTIVE	Edit Delete
	Postgre 4455	3	0 / 34	ACTIVE	Edit Delete
	Javascript101 222211	4	0 / 10	ACTIVE	Edit Delete
	English101 444	5	0 / 2	ACTIVE	Edit Delete

Create Course



Edit course


localhost:8080/admin/courses/edit/24

Side Panel

Edit Course

Course Name: Oulong

Course Code: 9999

Description: Heloo

Credits: 4

Max Capacity: 10

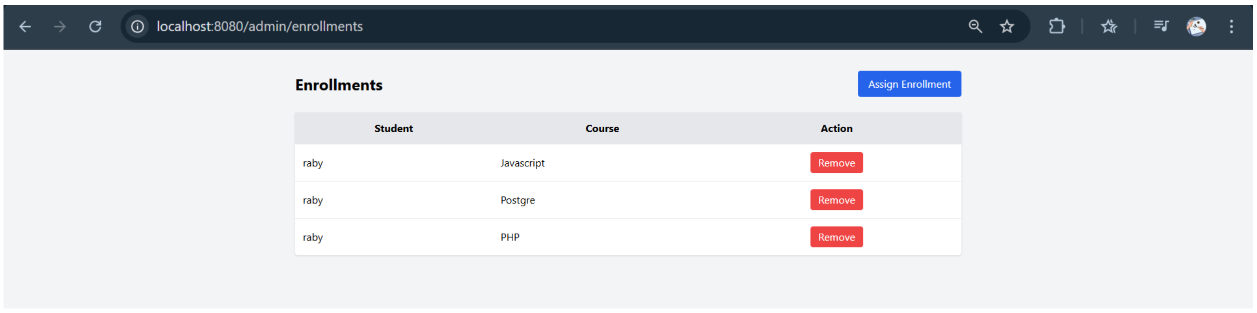
Status: ACTIVE

Current Image

Choose File No file chosen

Update Course

Admin Enrollments

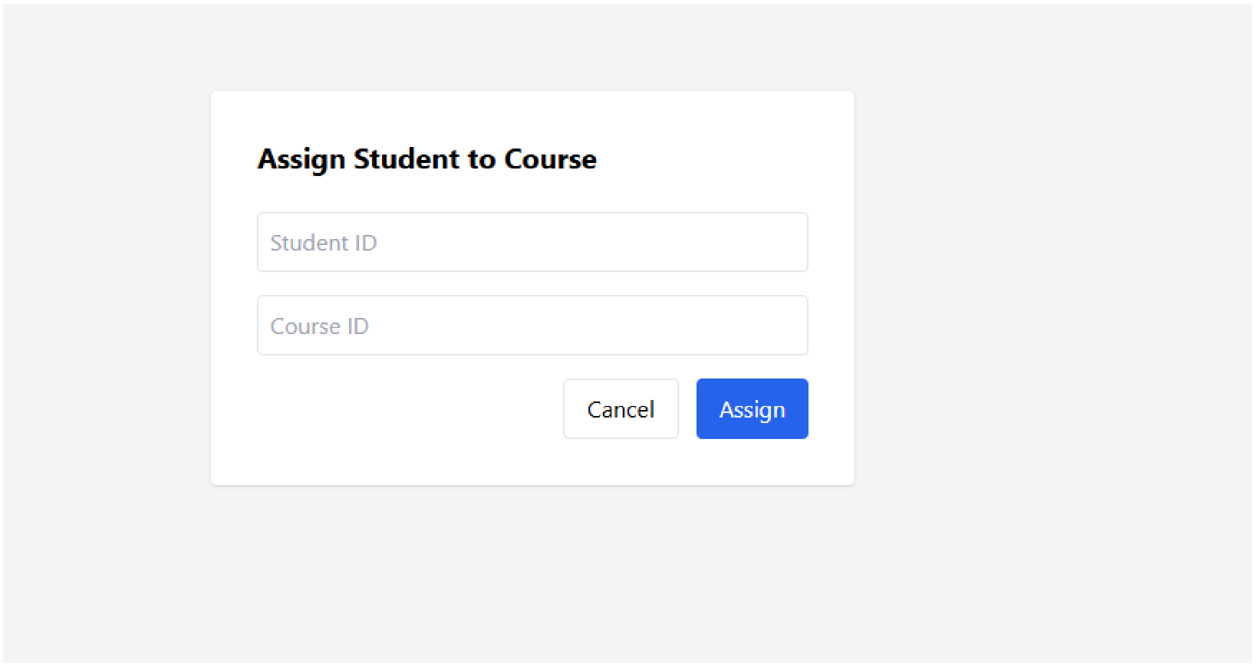

localhost:8080/admin/enrollments

Enrollments

Assign Enrollment

Student	Course	Action
raby	Javascript	Remove
raby	Postgre	Remove
raby	PHP	Remove

Assign enrollments


Assign Student to Course

Student ID

Course ID

Cancel

Assign