# CENG 382 TERM PROJECT – 2024- 2025 Spring

This is an **individual project**, and all participants are expected to adhere strictly to the principles of **academic integrity**. The work you submit must be entirely **your own**, and any form of **plagiarism**, **unauthorized assistance**, or **code copying** is strictly prohibited.

Any violation of these rules will result in **disciplinary action**, including project failure, grade penalties, or escalation to the university's academic misconduct committee.

You are encouraged to add **optional features** or improvements beyond the core requirements, especially those you personally believe would enhance the project. However, such additions must be **your original work** and properly documented.

This project is not only a technical evaluation but also a test of your **professional ethics and responsibility**.
**Any breach of academic honesty will be taken very seriously.**

## Important Notice:

In order for your project to be **evaluated**, you are required to prepare both a **project report** and a **3–5 minute demo video**. The demo video must be uploaded to **YouTube as an unlisted video**, and the **link to the video** must be included in the **project report**.

## Description of Project

In this project, students will develop a **web-based classroom reservation system** where instructors (users) can submit reservation requests for available classrooms during an academic term defined by the admin. The **admin** is responsible for managing the system by creating terms, approving or rejecting reservation requests, handling scheduling conflicts, and monitoring user feedback.

The platform will be developed using **C# with .NET Razor Pages**, **Entity Framework with MS SQL Server**, and **Bootstrap** for a responsive user interface. Students are expected to apply **S.O.L.I.D principles** and maintain clean, modular, and maintainable code.

The system will have a **role-based structure**, including two panels:

- An **Admin Panel**, where the admin can manage terms, instructor accounts, reservations, and view class-specific feedback.
- An **Instructor Panel**, where users can interact with calendars, submit or edit reservation requests, and provide feedback on classrooms.

Key system features include conflict detection, public holiday checks, an integrated feedback and contact system, a responsive calendar view, security measures (such as password hashing), detailed system logging, and automatic **email notifications**.

# 1. Technologies to Be Used

Participants are required to use the following technologies:

- **C# with .NET (Razor Pages)**
- **Entity Framework with MS SQL Server**
- **Bootstrap** for responsive UI

# 2. Code Design Requirements

- The project **must follow S.O.L.I.D design principles**.
    - For more information about S.O.L.I.D principles, check this link
- Code must be **modular**, **clean**, and **easy to maintain**.

# 3. Role-Based Structure

## Admin Panel

- Admin defines **academic terms (semesters)** and can **edit them later**.
- Admin is responsible for **creating and managing instructor accounts**.
- Admin reviews and **approves or rejects reservation requests** submitted by instructors.
- If two reservation requests **conflict**, both must be flagged and **cannot be approved until the conflict is resolved**.
- Admin sees **color-coded overlaps** and receives **warnings for reservations on official holidays**.
- Admin has access to:
    - A **class list** with average feedback ratings
    - A **detail view** for each class, including:
        - Weekly schedule
        - Instructor feedback with **star ratings**

## Instructor Panel (Users)

- Instructors can:
    - View and interact with a **weekly and monthly calendar** (powered by an external add-on)
    - Submit **classroom reservation requests**
    - Request **modifications or cancellations**
- Instructors can only request reservations **within the active term** defined by the admin.
- When reserving a specific time (e.g., *Tuesdays 09:00–11:00*), the system should apply the request to **all relevant days in that term**.
- Instructors receive warnings if their reservations fall on **official holidays** or **conflict with other reservations**.

## 4. Holiday and Conflict Handling

- System must check selected dates using a **free public holiday API** (e.g., Google Calendar API, government API).
- Reservations on official holidays should **trigger a warning** for both the admin and the instructor.
- **Overlapping reservations must be color-highlighted**, and **cannot be approved** until the issue is resolved.

## 5. Feedback & Contact System

- A **"Contact Us"** section must be included.
    - Instructors select from their registered classes to **send feedback**.
    - Each feedback includes a **5-star rating system** and a comment field.
    - Feedback is **sent to admin via email**.
- Admin views feedback under each class's **detail page**.
    - The average star rating must be displayed in the **main class list**.

## 6. UI & UX Expectations

- The website design should be **visually professional and suitable for an academic setting**.
- All edit operations (e.g., user info, reservations) must be done using **modals or visually appealing inline components**, without full page reloads.
- Navigation should be handled with **nested menus** inside a common **Layout page**.
- The calendar should always load the **current week dynamically** by default.

## 7. Security Requirements

- All passwords must be stored using **hashed and salted encryption**.
- Only the admin can **create and manage user accounts**.

## 8. System Logging

- **All user actions must be logged**, including:
    - Login attempts
    - Reservation submissions, changes, and deletions
    - Admin approvals and rejections
    - Feedback submissions
- Logs should contain the **timestamp**, **user identity**, **operation performed**, and **success or error status**.
- **Errors and unexpected behaviors** must be captured and written into separate error logs.
- Logs should be stored securely and kept **accessible for future auditing**.

## 9. Email Notification System

- The system must send automatic **email notifications** to instructors for:
    - Reservation approval
    - Reservation rejection
    - Reservation falling on an **official holiday**
- The email content must clearly indicate the **reservation details** and the **reason** for any action taken.

## 10. Database Performance

- Database queries must be written with **performance in mind**.
- Use of **eager loading, indexing, or stored procedures** (when necessary) is encouraged to avoid performance bottlenecks.
- The system must remain **responsive even when used by multiple instructors concurrently**.

# Bonus Features (Optional for Extra Credit)

- Dark/Light Mode Toggle: Provide a button to switch between light and dark UI themes.
- Unit Testing: Implement automated tests for core features such as reservation logic, calendar handling, and user validation.

# GRADING

| Criteria | Points |
|---|---|
| Software Architecture & Code Quality: SOLID principles, modular code | 20 |
| Role Management and Authorization: Admin-instructor separation, role control | 10 |
| Weekly Calendar & Reservation Logic: Dynamic weekly view, repeated reservations | 15 |
| Conflict & Holiday Detection: External API for holidays, conflict warnings | 15 |
| Admin & Instructor Functionalities: Term creation/edit, approval workflow | 15 |
| Feedback & Contact System: 5-star feedback, email-based contact form | 10 |
| Logging System: All user/system actions, logged securely | 5 |
| Email Notifications: Automated emails for approvals/rejections | 5 |
| **BONUS FEATURES** | |
| Dark/Light Mode UI Toggle | 5 |
| Unit Testing for Core Logic | 5 |
| **TOTAL** | **110** |