UnicomTIC Management System

Parameshwararaj Kavishna

UT010336

### 🔹 **1. Introduction**

The **UNICOMTIC Management System** is a web-based platform developed to digitally centralize and automate student, faculty, subject, and departmental data for academic institutions. It reduces manual administrative tasks and provides a unified system for seamless operations.

### 🔹 **2. Project Objectives**

* Efficient management of student and faculty information.
* Automated handling of course assignments and subject allocations.
* Role-based visibility for critical functions like update/delete.
* Proper exception handling to avoid system breakdowns.
* A responsive design that works flawlessly across devices.

### 🔹 **3. Technologies Used**

Frontend - C#

Backend - PHP

Database – MySQLite

### 🔹 **4. MVC Architecture Implementation**

The application follows the **Model-View-Controller (MVC)** design pattern:

* **Model**: Contains data-related classes such as Student.php, Subject.php.
* **View**: Comprises the user interface—HTML and frontend templates.
* **Controller**: Manages requests and routes them between Model and View components.

### **🔹 5. Key Technical Features**

#### ✅ **Enum (Enumeration)**

Used to manage constant status values such as Active, Inactive, Suspended for students or Core/Elective for subjects, ensuring consistency and readability throughout the code.

#### ✅ **DTO (Data Transfer Object)**

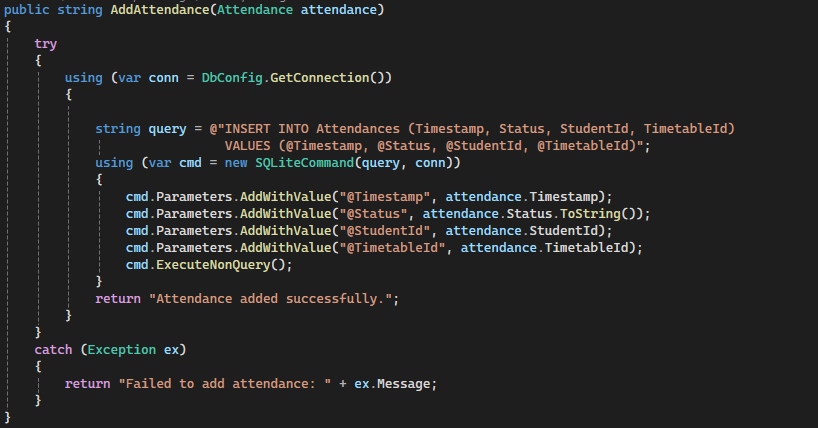
Custom DTOs are implemented to transfer data between layers—especially from backend models to frontend views—ensuring clean and decoupled data flow.

#### ✅ **Mapper Classes**

Mapper classes handle conversion between DTOs and Model objects (e.g., StudentMapper.php). This promotes clarity and separation of concerns in the codebase.

#### ✅ **Try-Catch Exception Handling**

Critical operations such as data insertions or updates are wrapped in try-catch blocks to handle errors gracefully:



|  |  |  |  |
| --- | --- | --- | --- |
| This is the example |  |  |  |
| 🔹 **6. Button Hide Logic (Role-Based UI Controls)** Button visibility is controlled based on user roles. **🔹 7. User Interface Responsiveness** The UI is built using **Bootstrap** and custom CSS, enabling responsive layout adaptation across all screen sizes—desktops, tablets, and mobiles. Key features include:   * Responsive navigation bar * Auto-scaling forms and tables * Dynamic interactivity using JavaScript/jQuery   This enhances user experience regardless of the device being used 🔹 **8. Future Enhancements**  * Implementation of **Role-Based Dashboards** (Admin, Faculty, Student) * Development of **REST APIs** for mobile app integration * Addition of **Student Portal** for marks, announcements, and schedules * Integration of **Attendance Tracking** and faculty performance analytics  **🔹 9. Conclusion** The **UNICOMTIC Management System** stands as a robust and scalable administrative tool. By integrating clean architecture (MVC), well-structured code practices (DTO, Mapper, Enum), and user-centric features (role-based controls, responsive UI, exception handling), it significantly improves academic and administrative workflows. This system enables institutions to focus more on education and less on routine operations.  ***"In addition, I will be adding some screenshots."***    ***Frontend Page*** |  |  |  |
| ***Student Registration Form*** |  |  |  |
|  |  |  |  |
|  |  |  |  |