

# WEB-PROGRAMMING PROJECT

UNIVERSITY TIMETABLE



Università  
degli Studi di  
Messina

Presented to : Armando Ruggeri

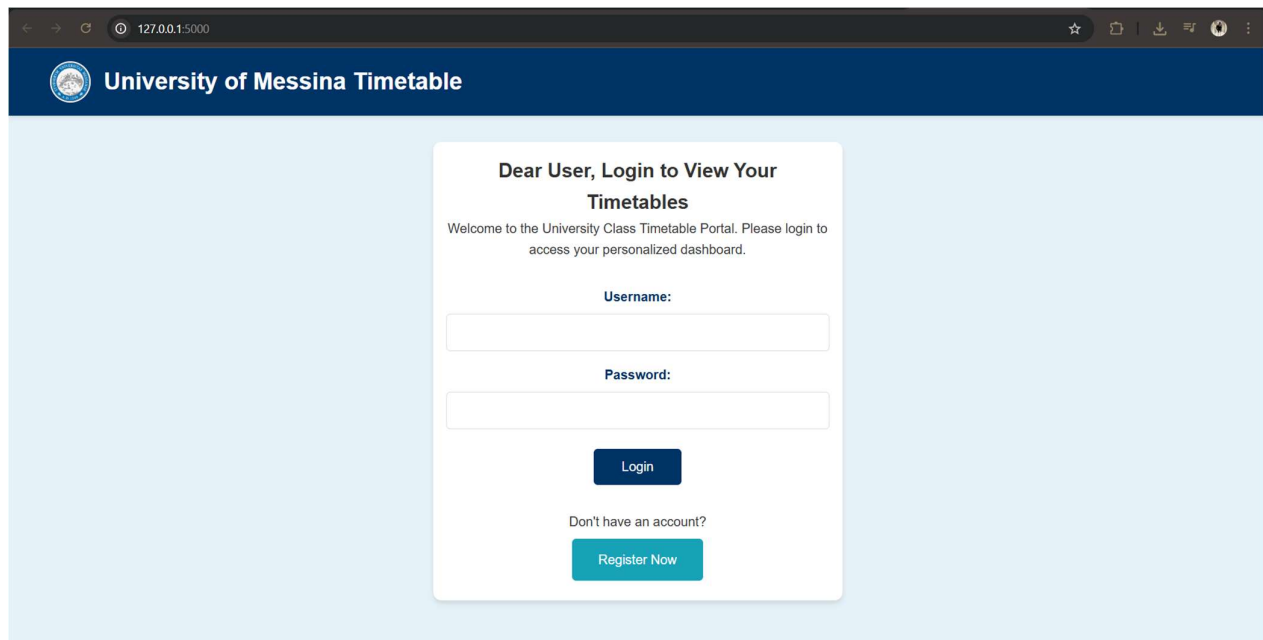
Presented by : Aftab Fakir (533001)

## INTRODUCTION

The University Class Timetable Portal is a web application developed to help students and professors efficiently manage and view class schedules. While students can view updates for their chosen courses in a calendar-based dashboard, professors can add and manage course details and instructions using the application. It has a dedicated section that provides contact details and notifications to improve student-teacher communication.

Python (Flask) is used to develop the backend, which manages database operations, course and schedule administration, user authentication, and notifications. Data is kept in a MySQL database and uses hashing to keep the passwords safe. The frontend of the website uses CSS and HTML to provide a clean and basic interface. Main pages such as the login page, dashboard layout, calendar, notification boxes, and contact cards are carefully structured to enhance usability and accessibility across different devices.

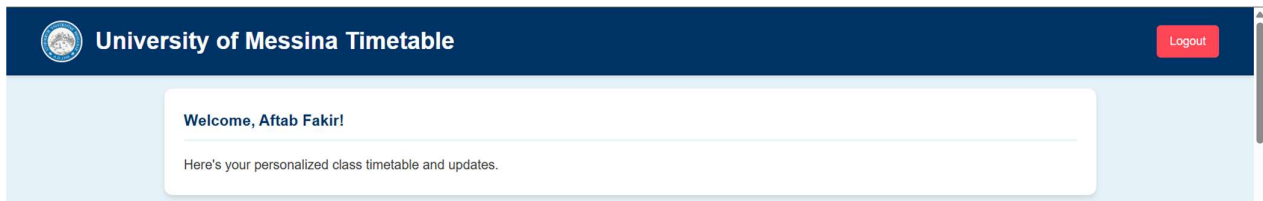
## Home page



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000'. The page has a dark blue header with the University of Messina logo and the text 'University of Messina Timetable'. The main content area is light blue and features a white login box. Inside the box, the text reads: 'Dear User, Login to View Your Timetables', 'Welcome to the University Class Timetable Portal. Please login to access your personalized dashboard.', 'Username:', a text input field, 'Password:', a password input field, a 'Login' button, 'Don't have an account?', and a 'Register Now' button.

The index page of the website shows the user the welcome message asking the user to Login to continue using the website. The user is presented with a login box where the user can use their email and password to login in the portal. Along with the login prompt the user gets the option to register into the website.

## The header section



The screenshot shows the header section of the website. It has a dark blue background with the University of Messina logo and the text 'University of Messina Timetable' on the left, and a red 'Logout' button on the right. Below the header, there is a white box containing the text: 'Welcome, Aftab Fakir!', a horizontal line, and 'Here's your personalized class timetable and updates.'

The header section of the website shows the name of the project along with the logo of the University. Alongside, we have the logout button allowing users to logout from their session. Below the header bar we have the welcome message for the signed-in user.

## The calendar

Monthly Class Schedule						
July 2025						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1	2 WEB101 9:00:00	3	4	5
6	7 WEB101 9:00:00	8	9 WEB101 9:00:00	10	11	12
13	14 WEB101 9:00:00	15	16 WEB101 9:00:00	17	18	19
20	21 WEB101 9:00:00	22	23 WEB101 9:00:00	24	25	26
27	28 WEB101 9:00:00	29	30 WEB101 9:00:00	31		

The user gets a monthly calendar which shows all the lecture updates regarding the courses chosen by the student.

## Notifications and Professor contact details

### Recent Notifications

**Welcome to Web Programming**

Welcome to the Web Programming course! Please check the syllabus.

Course: Web Programming | From: Armando Ruggeri | Date: 2025-07-05 17:44

### Professor Contact Details

**Armando Ruggeri**

Course: Web Programming

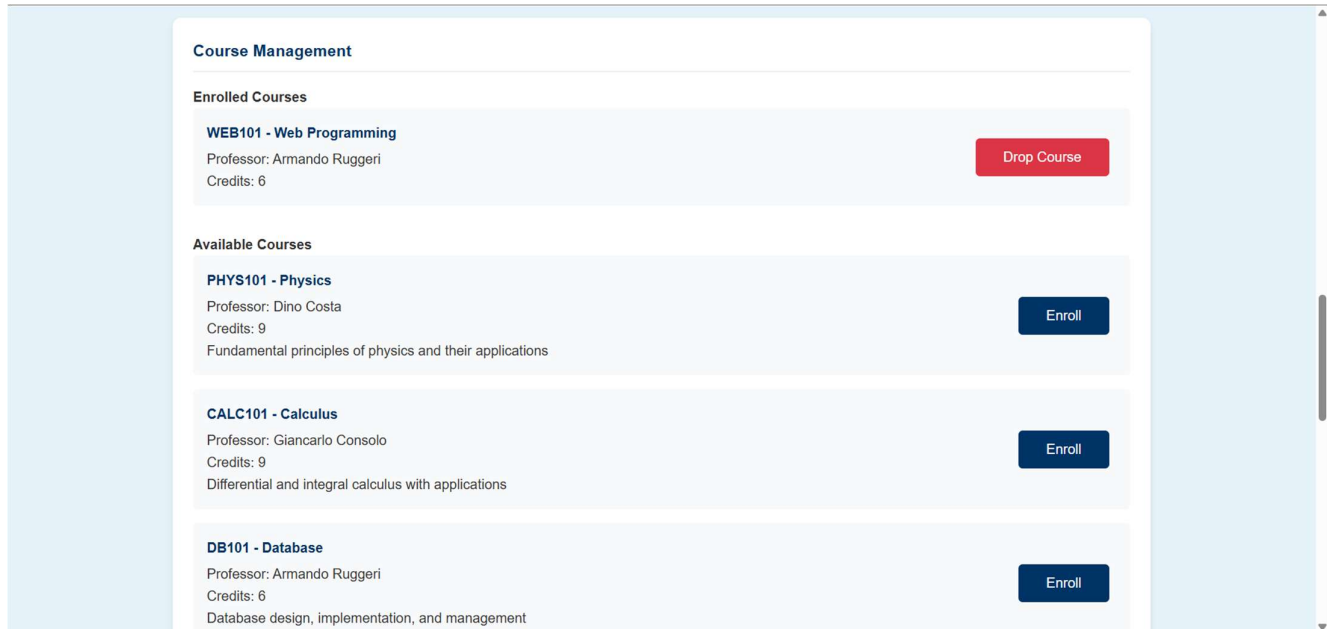
Email: armando.ruggeri@unime.it

Office: Department of Engineering, Room 201

Office Hours: Monday 15:00-17:00, Wednesday 15:00-17:00

A section regarding new notifications and professor contact details is shown below the calendar. Here the student can find contact details and new notifications published by the professor.

## Course Management

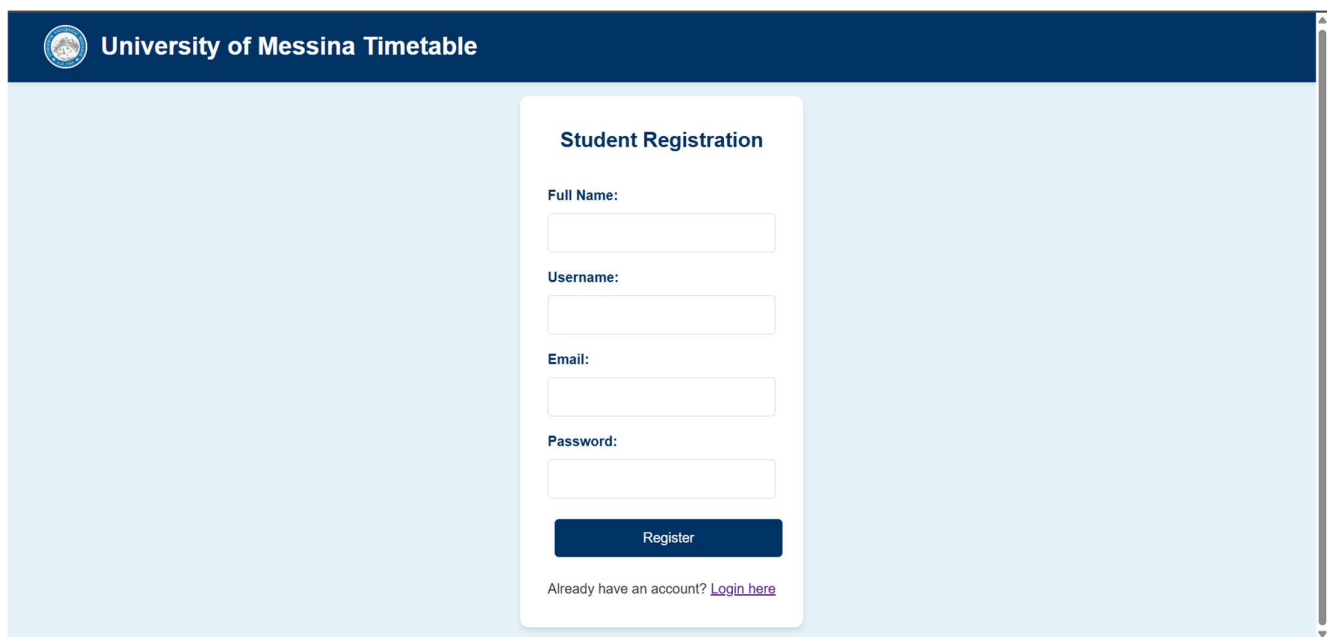


The Course Management interface is divided into two main sections: 'Enrolled Courses' and 'Available Courses'. The 'Enrolled Courses' section displays a single course, 'WEB101 - Web Programming', with details: Professor: Armando Ruggeri, Credits: 6, and a red 'Drop Course' button. The 'Available Courses' section lists three courses: 'PHYS101 - Physics' (Professor: Dino Costa, Credits: 9, description: Fundamental principles of physics and their applications, with an 'Enroll' button), 'CALC101 - Calculus' (Professor: Giancarlo Consolo, Credits: 9, description: Differential and integral calculus with applications, with an 'Enroll' button), and 'DB101 - Database' (Professor: Armando Ruggeri, Credits: 6, description: Database design, implementation, and management, with an 'Enroll' button).

Course Management	
<b>Enrolled Courses</b>	
<b>WEB101 - Web Programming</b> Professor: Armando Ruggeri Credits: 6	<a href="#">Drop Course</a>
<b>Available Courses</b>	
<b>PHYS101 - Physics</b> Professor: Dino Costa Credits: 9 Fundamental principles of physics and their applications	<a href="#">Enroll</a>
<b>CALC101 - Calculus</b> Professor: Giancarlo Consolo Credits: 9 Differential and integral calculus with applications	<a href="#">Enroll</a>
<b>DB101 - Database</b> Professor: Armando Ruggeri Credits: 6 Database design, implementation, and management	<a href="#">Enroll</a>

At the end of the page the user is given the option to manage their courses. The basic information about the course and the professor is shown along the name of the professor. The student has the option to enroll into a new course or leave an existing course they have chosen before.

## Register page



The 'University of Messina Timetable' registration page features a dark blue header with the university's logo and name. Below the header, a white 'Student Registration' form is centered. The form includes input fields for 'Full Name:', 'Username:', 'Email:', and 'Password:', each followed by a text input box. A dark blue 'Register' button is positioned below the password field. At the bottom of the form, a link reads 'Already have an account? [Login here](#)'.

**University of Messina Timetable**

**Student Registration**

Full Name:

Username:

Email:

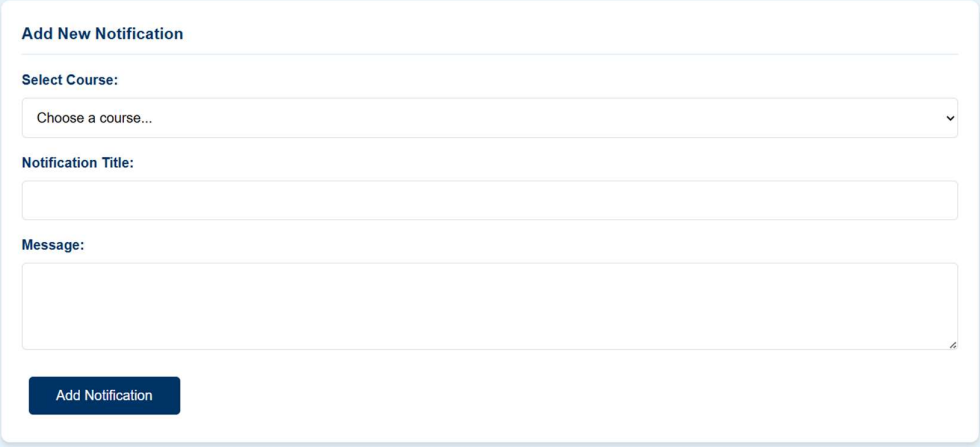
Password:

[Register](#)

Already have an account? [Login here](#)

On the register page the user is asked to enter their Full Name, username, email, and password to finish registering for the website.

## The professors page



**Add New Notification**

Select Course:

Choose a course... ▾

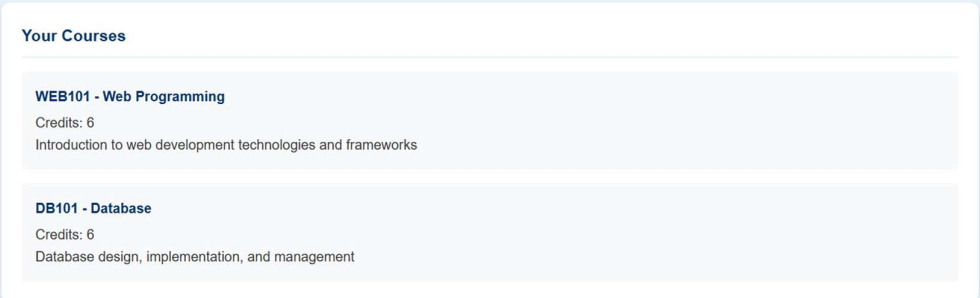
Notification Title:

Message:

Add Notification

This form is used to add new notifications. It includes a dropdown menu for selecting a course, a text input for the notification title, a text area for the message, and a submit button labeled 'Add Notification'.

The professor's page allows the professor to add new notifications to the website. Along with it the user is shown all the courses that the professor is teaching. If a professor is teaching multiple courses, then they have the option to choose which course they want to add the notifications.



**Your Courses**

**WEB101 - Web Programming**  
Credits: 6  
Introduction to web development technologies and frameworks

**DB101 - Database**  
Credits: 6  
Database design, implementation, and management

This section displays a list of courses taught by the professor. Each course entry includes the course name, the number of credits, and a brief description.

## 1. Database Structure

### 1.Users

```
1  CREATE DATABASE IF NOT EXISTS university_portal;
2  USE university_portal;
3
4  CREATE TABLE IF NOT EXISTS users (
5      id INT AUTO_INCREMENT PRIMARY KEY,
6      username VARCHAR(50) UNIQUE NOT NULL,
7      email VARCHAR(100) UNIQUE NOT NULL,
8      password VARCHAR(255) NOT NULL,
9      role ENUM('student', 'professor') NOT NULL,
10     full_name VARCHAR(100) NOT NULL,
11     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
12 );
13
```

1. **Unique Identification:** id serves as the primary key with auto-increment to guarantee unique identification for every user.
2. **Core Credentials:** Stores essential login details - username (50 char limit), unique email (100 char limit), and hashed password (255 char storage).
3. **Metadata Tracking:** created\_at automatically records account creation timestamp using CURRENT\_TIMESTAMP.
4. **Data Integrity:** Enforces mandatory fields (NOT NULL) and prevents duplicate emails (UNIQUE constraint).

### 2.Courses

```
13
14  CREATE TABLE IF NOT EXISTS courses (
15      id INT AUTO_INCREMENT PRIMARY KEY,
16      course_code VARCHAR(20) UNIQUE NOT NULL,
17      course_name VARCHAR(100) NOT NULL,
18      professor_id INT,
19      description TEXT,
20      credits INT DEFAULT 3,
21      FOREIGN KEY (professor_id) REFERENCES users(id)
22  );
23
```

1. **Unique identifier:** id serves as the primary key with auto-increment to guarantee unique identification for every user.

2. **Course code and name:** a unique course code is given to every course to easily know which course the user is using. The name of the course is also put in along with the course code.
3. **Professor id:** at the time of registration every professor is given an unique id and the same id is referred in every course to manage which professor can access the course management.

### 3. Enrollments

```
23
24 ✓ CREATE TABLE IF NOT EXISTS enrollments (
25     id INT AUTO_INCREMENT PRIMARY KEY,
26     student_id INT,
27     course_id INT,
28     enrolled_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
29     FOREIGN KEY (student_id) REFERENCES users(id),
30     FOREIGN KEY (course_id) REFERENCES courses(id),
31     UNIQUE KEY unique_enrollment (student_id, course_id)
32 );
33
```

The enrollments table keeps a record of which student has been enrolled in which course. It uses student\_id and course\_id to match the data and save it in the database. Along with the data it also saves the timestamp of when a student enrolls into a course.

### 4. Notifications

```
CREATE TABLE IF NOT EXISTS notifications (
    id INT AUTO_INCREMENT PRIMARY KEY,
    course_id INT,
    professor_id INT,
    title VARCHAR(200) NOT NULL,
    message TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (course_id) REFERENCES courses(id),
    FOREIGN KEY (professor_id) REFERENCES users(id)
);
```

The notifications table keeps the data is new and existing notifications of the courses. Every new notification is given an unique id that increments automatically. It uses the course\_id and professor\_id as the reference of the course and professor related to the notification.

## 5. Professor contact details

```
CREATE TABLE IF NOT EXISTS professor_contacts (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    professor_id INT,  
    office_location VARCHAR(100),  
    office_hours VARCHAR(200),  
    FOREIGN KEY (professor_id) REFERENCES users(id)  
);
```

This table creates a record of the professor details. It creates an unique id for every entry. It uses the professor\_id to refer to the professor. Office location and office hours are manually put for every professor.

## 2. Core Implementation

Login Page (base.html)

```
1  {% extends "base.html" %}  
2  
3  {% block content %}  
4  <div class="center-container">  
5  <div class="welcome-card">  
6      <h2>Dear User, Login to View Your Timetables</h2>  
7      <p>Welcome to the University Class Timetable Portal. Please login to access your personalized dashboard.</p>  
8  
9  <form method="POST" action="{{ url_for('login') }}" style="margin-top: 2rem;">  
10 <div class="form-group">  
11     <label for="username">Username:</label>  
12     <input type="text" id="username" name="username" required>  
13 </div>  
14  
15 <div class="form-group">  
16     <label for="password">Password:</label>  
17     <input type="password" id="password" name="password" required>  
18 </div>  
19  
20 <div class="form-group">  
21     <button type="submit" class="btn btn-primary" style="width: 23%;">Login</button>  
22 </div>  
23 </form>  
24  
25 <div style="margin-top: 1.5rem;">  
26     <p>Don't have an account?</p>  
27     <a href="{{ url_for('register') }}" class="btn btn-secondary">Register Now</a>  
28 </div>  
29 </div>  
30 </div>  
31 {% endblock %}  
32
```

1. **Session management:** it initializes the user session and connects to the MySQL database using the mysql.connector.
2. **Form Handling:** Handles the form submission through the POST method.
3. **Session & Redirect:** If the credentials are correct, the user's ID and role are saved in the session and they are redirected to their respective dashboard. If authentication fails, an error message is flashed and the user remains on the login page



### 3. Project Structure

```
├── Timetable/
│   ├── app.py
│   ├── scripts/
│   │   ├── sample_data.sql
│   │   ├── database_setup.sql
│   ├── static/
│   │   ├── style.css
│   │   ├── images/
│   │   ├── logo.png
│   ├── templates/
│   │   ├── base.html
│   │   ├── home.html
│   │   ├── login.html
│   │   ├── register.html
│   │   ├── student_dashboard.html
│   │   ├── professor_dashboard.html
```

### 4. Challenges

One of the biggest challenges while creating this project was smooth integration between the frontend and backend aspects, especially with respect to handling user authentication and dynamic presentation of data. Secure handling of user sessions and the use of hashed passwords required to be set up with specificity. Developing a responsive and good-looking interface through HTML and CSS alone, without using frameworks like Bootstrap, also necessitated accurate styling work. Apart from this, maintaining consistency of the data and relational integrity among users, classes, schedules, and reminders in MySQL needed proper planning and debugging.

### 5. Solution

In response to these challenges, a modular and systematic approach was adopted. In backend security, hashed passwords for secure authentication, while Flask sessions provided secure user state management. SQL queries were parameterized carefully to avoid injection attacks and maintain data integrity. At the frontend, regular usage of reusable CSS classes like `.welcome-card` helped achieve standardized structure across pages. Manual testing via different roles of user (student and professor) ensured navigation controls and access controls to work as they should.