ST10219717

PHAKA PHUTI

README FILE

```
SqlConnection conn = new SqlConnection("Data Source=(localDB)\\ProjectModels;
Initial Catalog= ModuleApps;Integrated Security=True;");
```

The desktop application is now quite useful with the data being stored between runs. But having to always use a desktop computer to run the program is maybe not the most flexible user experience. Instead, create a web application that will allow the user to access their data from any device that has a browser.

The software shall persist the data in a SQL database. The user shall be able to register with a username and password. The software shall store only the hash of the password in the database. The user shall be able to log into the software with their username and password. The user shall only be able to see their own data and never that of other users.

The user must be able to add multiple modules for the semester. The following data must be stored for each module: a. Code, Name, Number of credits, Class hours per week, The user must be able to enter the number of weeks in the semester. The user must be able to enter a start date for the first week of the semester. The software shall display a list of the modules with the number of hours of self-study that are required for each module per week. self-study hours per week= number of credits × 10 number of weeks − class hours per week. The user must be able to record the number of hours they spend working on a specific module on a certain date. The software shall display how many hours of self-study remain for each module for the current week. This should be calculated based on the number of hours already recorded on days during the current week

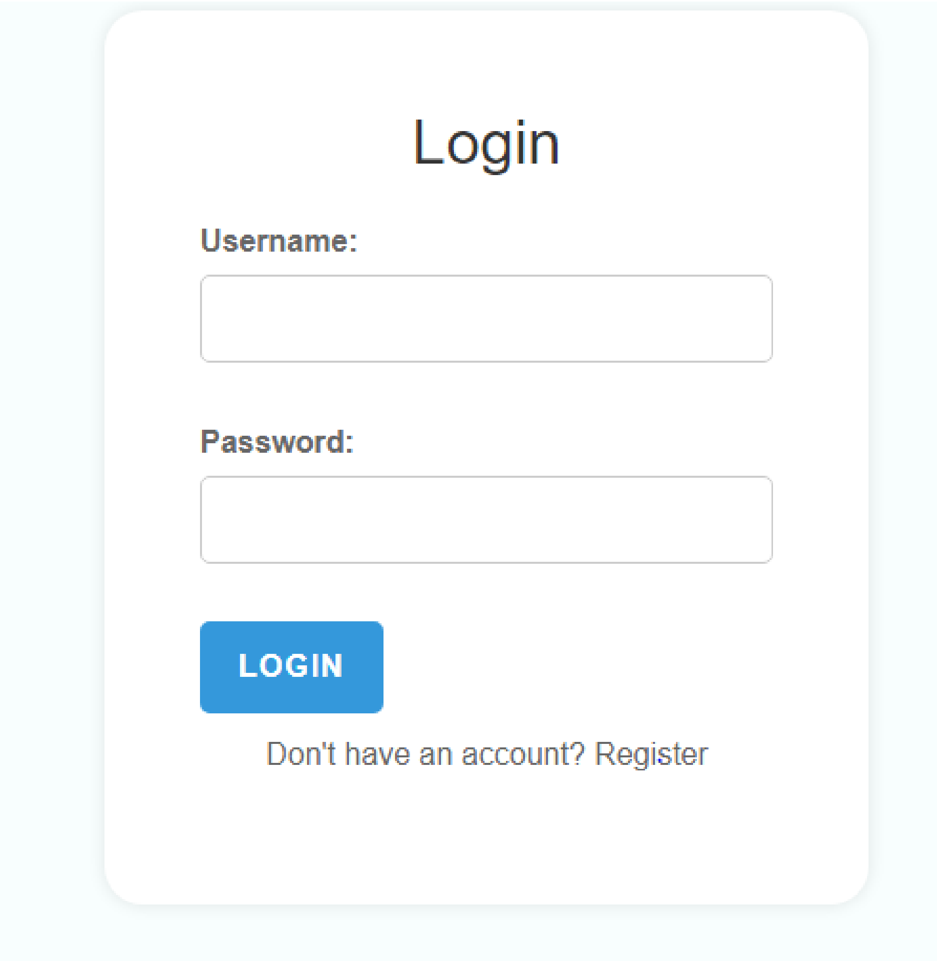
How the app function

Register and login

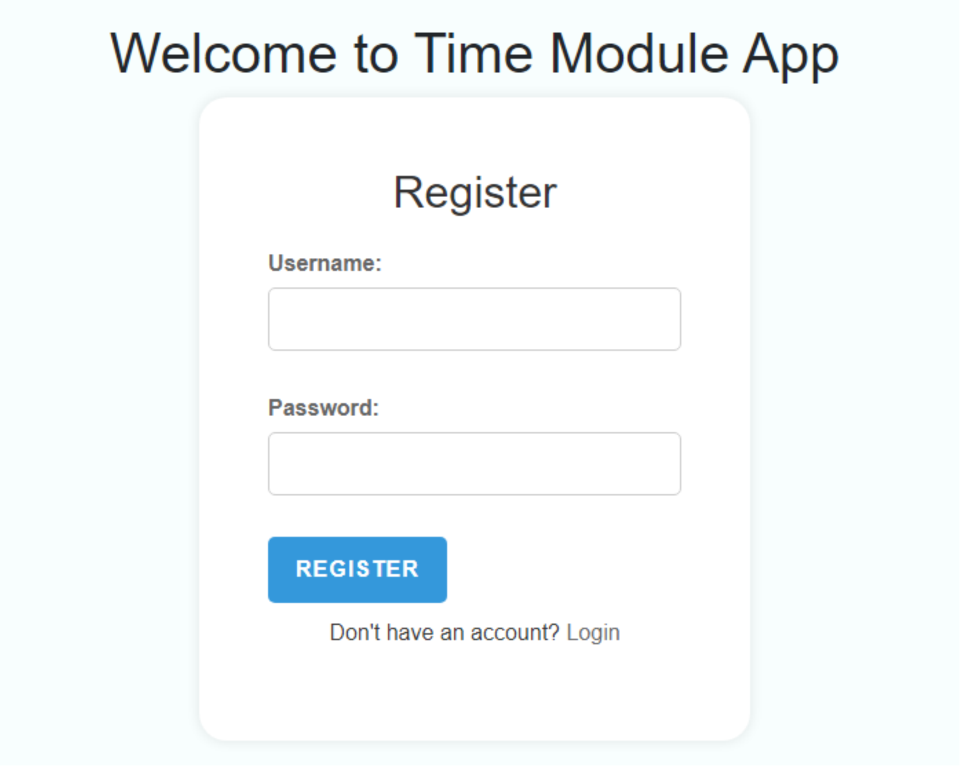
Example : username: phaka and password: 12345 for login

As a user, when you arrive on the page, you are greeted with a welcoming message: "Welcome to Time Module App". There is a registration form presented to you. The form has fields for entering your registration details. You see a field labeled "Username" where you need to input a unique username. This field is mandatory, indicated by the asterisk (*) next to it. There is another field labeled "Password" where you need to set a secure password. The password is typically hidden for security reasons. Below the fields, there is a "Register" button. Clicking this button will submit the information you entered. If you already have an account and accidentally landed on the registration page, there is a link provided: "Don't have an account? Login." Clicking this link will take you to the login page and vise Verser.

The registration page is part of a larger web application. Typically, users navigate to this page when they want to create a new account. The primary purpose of this page is to collect your registration details so If there are any issues with the information you provide (for example, if the username is already taken), the application should ideally provide clear error messages to guide you. If the username and password is correct the application will allow you to open another page.



Don't have an account? Register



Module dashboard

When you first arrive on the page, you see a title at the top saying " Dashboard." and header that says "Module Dashboard." Beneath the header, there's a welcoming message: "Welcome to Your Module Journey."

Frame 1 - Add Module:

The first frame contains an image and a label saying, "Add Module."

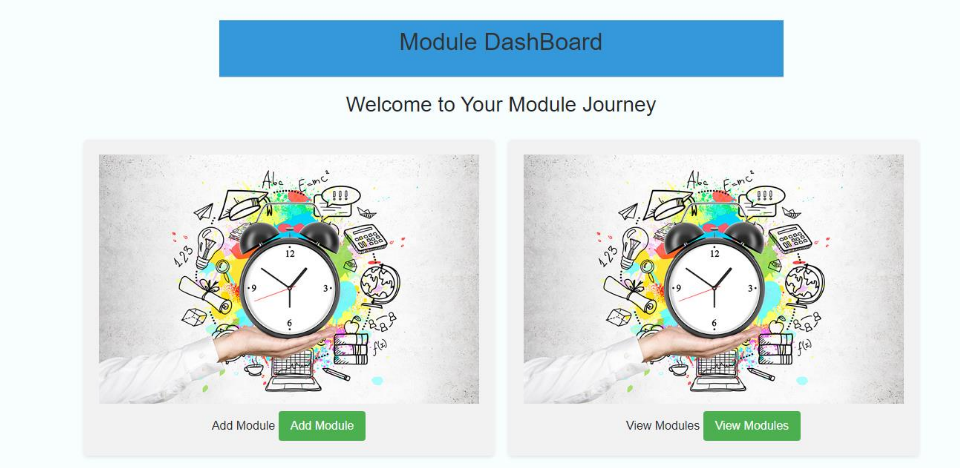
Below the label, there's a button labeled "Add Module."

When you click the "Add Module" button, it directs you to another page where you capture the details. The username is passed as a parameter.

Frame 2 - View Modules:

The second frame contains a similar structure with an image and a label saying, "View Modules." Below the label, there's a button labeled "View Modules." Clicking the "View Modules" button triggers a function (view) that likely redirects you to a page where you can view your existing modules. The username is passed as a parameter.

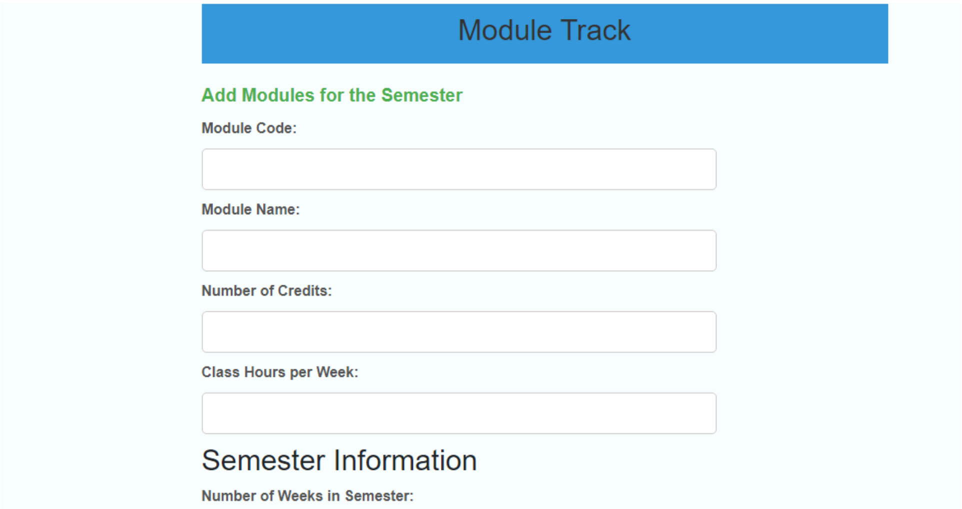
As a user, you interact with the page by clicking the "Add Module" or "View Modules" buttons. These actions take you to different sections of the application.



Module DashBoard

Welcome to Your Module Journey

When you click Add module, you will have the assess to capture the module Code, Name, Credits, Class per week and Semester Information like number of weeks per semester and date of semester. There are two buttons, A button labeled "Add Module." Clicking this button submits the form, adding the module information to the system and A button labeled "Reset." Clicking this button clears all the fields, allowing you to start over if needed. The primary purpose of this page is to allow you to input details about a module you are adding for the semester'.



Module Track

Add Modules for the Semester

Module Code:

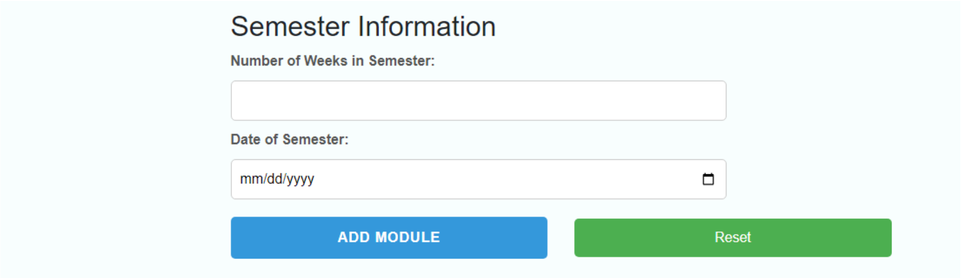
Module Name:

Number of Credits:

Class Hours per Week:

Semester Information

Number of Weeks in Semester:



Semester Information

Number of Weeks in Semester:

Date of Semester:

mm/dd/yyyy

VIEW Module

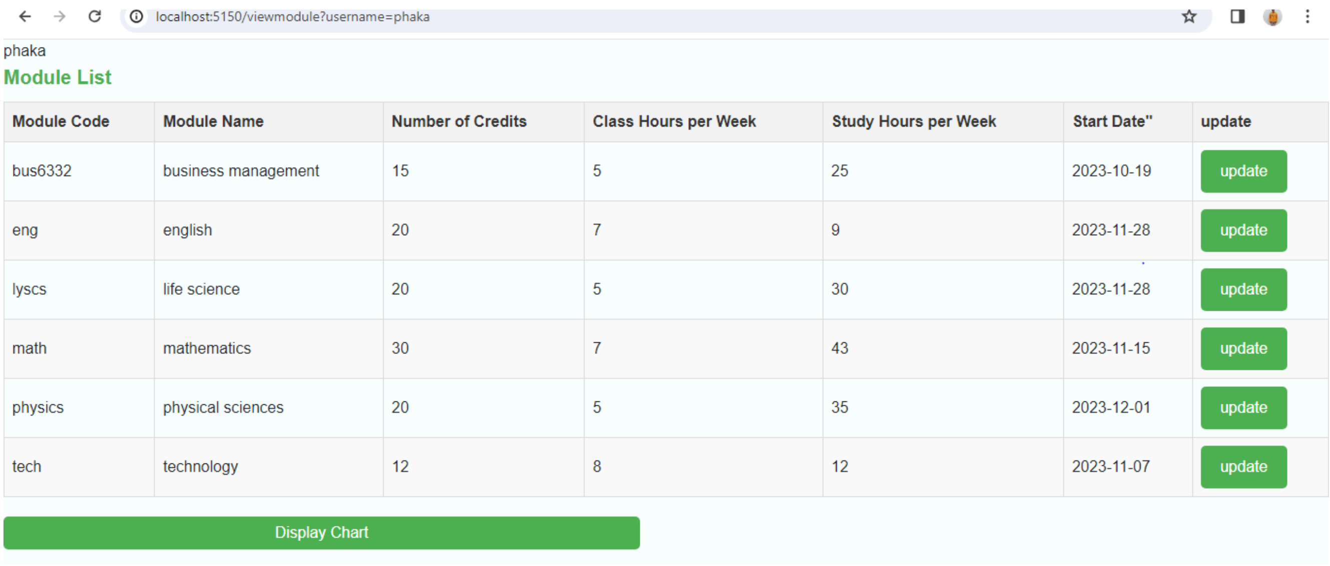
At the top of the page, you see the title "Module Tracker." Below that, it displays the username of the logged-in user. This helps users identify their account. For each module in the table, there is an "Update" button. Clicking this button triggers a function that likely allows the user to update information related to that specific module. The module's unique identifier (id) is passed to this function.

Remaining Study Module Section:

Below the table, there is a form allowing users to input the number of hours they plan to study for a specific module. It shows the Module ID (read-only) and a field to enter the number of hours. After entering the data, users can click the "Submit" button.

Chart Display Button:

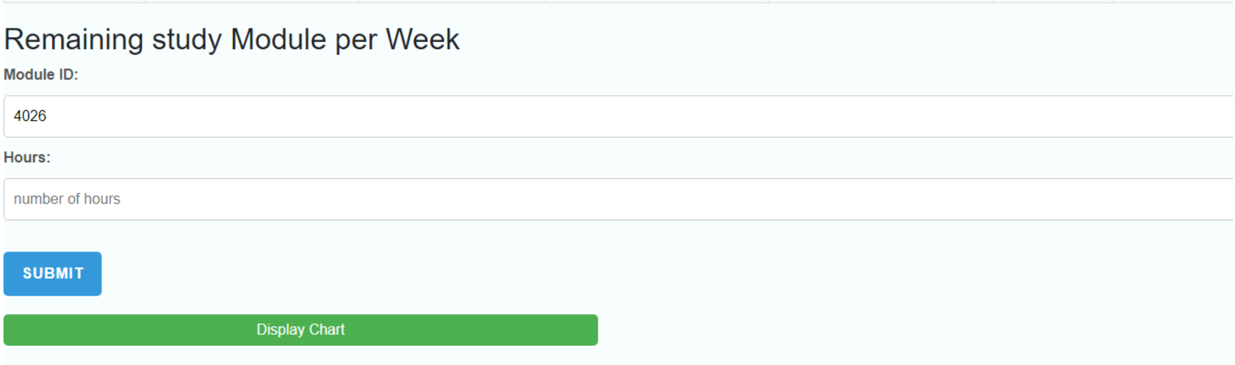
There's a button labeled "Display Chart." Clicking this button triggers a function (display Chart).The chart visualizes the self-study hours per week for each module. Each bar represents a module, and the height of the bar corresponds to the self-study hours. This provides users with a visual overview of their study distribution. The chart allows users to understand how their study hours are distributed among different modules. They can interpret the chart to identify modules that might need more attention.



phaka

Module List

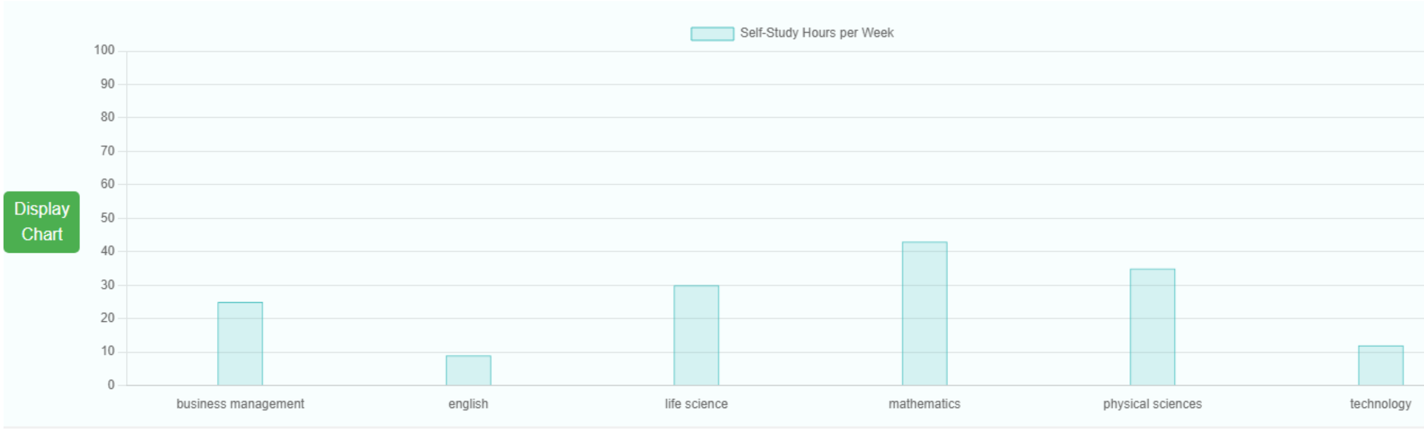
Module Code	Module Name	Number of Credits	Class Hours per Week	Study Hours per Week	Start Date"	update
bus6332	business management	15	5	25	2023-10-19	update
eng	english	20	7	9	2023-11-28	update
lyscs	life science	20	5	30	2023-11-28	update
math	mathematics	30	7	43	2023-11-15	update
physics	physical sciences	20	5	35	2023-12-01	update
tech	technology	12	8	12	2023-11-07	update



Remaining study Module per Week

Module ID: 4026

Hours: number of hours



ClassLibrary

class1

The class establishes a connection to a SQL Server database using SqlConnection. The connection string contains information about the server, database, and authentication. The conn (SqlConnection) is opened and closed appropriately in various methods to ensure efficient database connection management. The user's method is responsible for registering a new user. It takes a username and password, hashes the password using SHA256, and inserts the user into the usertrack table in the database. The login method checks user credentials during login. It compares the hashed password with the stored hashed password in the usertrack table and sets a Boolean check to true if the user is found. The Hash Password method hashes a given password using the SHA256 algorithm. It converts the hashed bytes to a hexadecimal string. The Module method allows the addition of a study module. It inserts the module information into the studymodu table, including the module code, name, credits, class hours, self-study hours, start date, and the associated username. The get_module method retrieves module information for a specific user. It populates lists with information such as module IDs, codes, names, credits, class hours, self-study hours, and start dates. The minusHour method deducts study hours from a module. It checks if the module has enough hours, subtracts the specified hours, and updates the database. Exception handling is implemented to catch and handle errors that might occur during database operations. Error messages are generated and returned for appropriate user feedback.

Class called User for get and setters,

Class called Modules for get setters.