

WeHelp

Assignment - Week 6

This week, we have to build a member system, based on the database tables designed in week 5, with Python FastAPI, MySQL, and any other necessary skills. Refer to [W3Schools Python MySQL](#) tutorial for learning how to connect to MySQL by the official [mysql-connector-python](#) package.

Note: Don't push code including your daily password to a public GitHub Repository.

Task 1: Pages

We have to build 3 pages for this member system.

Home Page:

- **URL:** <http://127.0.0.1:8000/>
- **Method:** GET
- **Design Points:** a signup form with 2 text inputs, 1 password input and a submit button. A signin form with 1 text input and 1 password input and a submit button.

歡迎光臨 · 請註冊登入系統

註冊帳號

姓名

帳號

密碼

登入系統

帳號

密碼

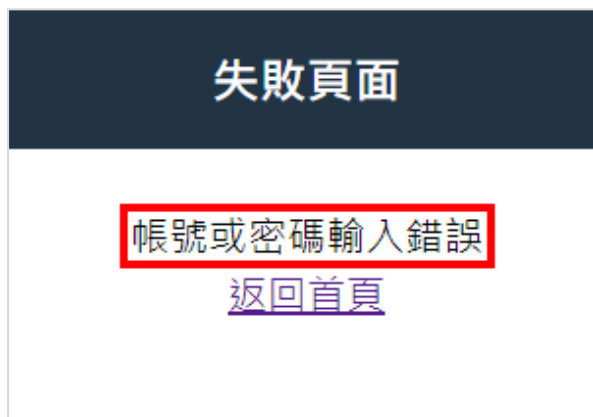
Member Page:

- **URL:** <http://127.0.0.1:8000/member>
- **Method:** GET
- **Design Points:** a page for signed-in members, including member name.



Error Page:

- **URL:** <http://127.0.0.1:8000/error?message=自訂的錯誤訊息>
- **Method:** GET
- **Design Point:** get error message from Query String in URL defined above and show it on this page.



Task 2: Build a procedure for signing up

Users have to sign up for a new member account before signing in.

Signup Endpoint:

- **URL:** <http://127.0.0.1:8000/signup>
- **Method:** POST

Required Procedure:

1. User enters name, username and password in the **Home Page**, and then clicks the submit button. Check if there is any empty input in the front-end, if yes, prevent form submission, if no, submit signup form to the **Signup Endpoint**.
2. In the **Signup Endpoint**, get inputs from front-end and execute procedure described below:
 - a. Check the website database if there is any repeating username in the member table.
 - i. If yes, it means signup failed. Do not insert any data to the member table. Redirect the user to the **Error Page**, show "Repeated username" in the page.
 - ii. If no, it means signup succeeds. Insert input data to the member table. Redirect the user to the **Home Page**.

Task 3: Build a procedure for signing in

Users can sign in to the member page after signing up.

Signin Endpoint:

- **URL:** <http://127.0.0.1:8000/signin>
- **Method:** POST

Required Procedure:

1. User enters username and password in the **Home Page**, and then clicks the submit button. Check if there is any empty input in the front-end, if yes, prevent form submission, if no, submit signin form to the **Signin Endpoint**.
2. In the **Signin Endpoint**, get inputs from front-end and execute procedure described below:
 - a. Check the website database if there exists a username/password pair in the member table, the same as the inputs.
 - i. If yes, it means signin succeeds. Record member id, username and name into the user state. Redirect the user to the **Member Page**, show the current member's name in the page.
 - ii. If no, it means signin failed. Redirect the user to the **Error Page**, show "Username or password is not correct" in the page.

Task 4: Build a procedure for signing out

Users can sign out from the member page after signing in.

Signout Endpoint:

- **URL:** <http://127.0.0.1:8000/signout>
- **Method:** GET

Required Procedure:

1. In the **Member Page**, we should always verify the recorded user state in the back-end logic. If it does not pass the verification, force redirecting the user to the **Home Page** without showing any content on the member page.
2. Add a sign out link/button to the member page. If this sign out link/button is clicked, connect to the **Signout Endpoint** where we have to clear recorded member data in the user state and redirect to the **Home Page**.

Task 5: Build a simple message system

In the member page, add a feature for leaving a message, and show all the past messages in the bottom of the page.

Member Page:

- **URL:** <http://127.0.0.1:8000/member>
- **Method:** GET
- **Design Points:** add leaving message feature, show past messages including author's name and content.

歡迎光臨，這是會員頁

ply，歡迎登入系統

[登出系統](#)

快來留言吧

內容

丁滿：我也來試試看

ply：這是測試留言

ply：好哦好哦

ply：測試測試，測試測試

CreateMessage Endpoint:

- **URL:** <http://127.0.0.1:8000/createMessage>
- **Method:** POST

Required Procedure:

1. When a user enters the **Member Page**, our backend code gets all the past messages from the database and integrates message data to the web page by template engine.

WeHelp
Assignment - Week 6

2. Enter message content and click the submit button to connect to the **CreateMessage Endpoint** for leaving a new message.
3. In the **CreateMessage Endpoint**, get message content from front-end, get member id from user state, and insert a new message record to the message table. After all, redirect back to the **Member Page**.

Task 6: Build a procedure to delete message (Optional)

Users can delete their own message in the member page.

Member Page:

- **URL:** <http://127.0.0.1:8000/member>
- **Method:** GET
- **Design Points:** only show a delete button next to my own message.

歡迎光臨，這是會員頁

丁滿，歡迎登入系統
[登出系統](#)

快來留言吧

內容

送出

丁滿：好了拉
丁滿：哦哦哦哦，慢走不送
精靈：我要回西方了～
精靈：我是精靈
丁滿：測試測試，測試測試

DeleteMessage Endpoint:

- **URL:** <http://127.0.0.1:8000/deleteMessage>
- **Method:** POST

Required Procedure:

1. Show a delete button next to messages belonging to the current signed-in member.
2. If the delete button is clicked, use the built-in confirm function to confirm this deleting action in the front-end by JavaScript. If canceled, do nothing.
3. If confirming deleting, connect and pass member id to the **DeleteMessage Endpoint** where our backend code gets message id from front-end, and delete corresponding message data in the database. After all, redirect back to the **Member Page**.