Build a website with Python Flask. Complete a simple verification system defined below.

Task 1: Home Page

Write your web application using Flask, running as a web server. We want to get a home page described as below:

Home Page:

- **URL:** http://127.0.0.1:3000/
- **Design for Reference:** a simple UI with a signin form, including a text input, password input, checkbox and submit button.

Required Procedure:

1. When a user tries to submit the signin form by clicking the submit button, check if the checkbox is checked by JavaScript in the front-end.
    a. If the checkbox is not checked, show a simple alert dialog with "Please check the checkbox first" message inside.
   b. If the checkbox is checked, submit the signin form.

Task 2: Verification

Users can enter username, password, check the checkbox, and finally submit the signin form to the back-end for verification. Redirect the user to the success or error page based on the verification result.
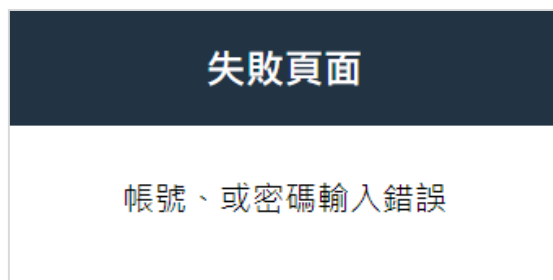
Verification Endpoint:

- **URL:** http://127.0.0.1:3000/signin
- **Method:** POST

Success Page:

- **URL:** http://127.0.0.1:3000/member
- **Design for Reference:** show a simple successful message on this page.

Error Page:
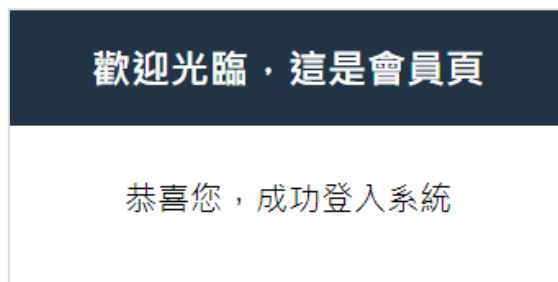
- **URL:** http://127.0.0.1:3000/error?message=自訂的錯誤訊息
- **Design for Reference:** get error message from Query String in URL defined above and show it on this page.

Required Procedure:

1. User enters username and password, checks the checkbox in the **Home Page**.

2. Submit form data to **Verification Endpoint** by **POST** method after user clicking submit button. Execute verification procedure described below in the back-end:

   a. If the username or password is empty, redirect the user to the **Error Page** including "Please enter username and password" message.

   b. **Verify** if the username and password are test:

      i. **If yes**, redirect the user to the **Success Page**.

      ii. **If no**, redirect the user to the **Error Page** including "Username or password is not correct" message.

Task 3: User State Management

Use Flask Session to manage user state in the verification procedure completed in Task 2. Record, trace, and verify user state in the back-end system.

Signout Endpoint:
- **URL:** http://127.0.0.1:3000/signout
- **Method:** GET

Success Page:
- **URL:** http://127.0.0.1:3000/member
- **Design for Reference:** show a simple successful message and a hyperlink for signing out on this page.
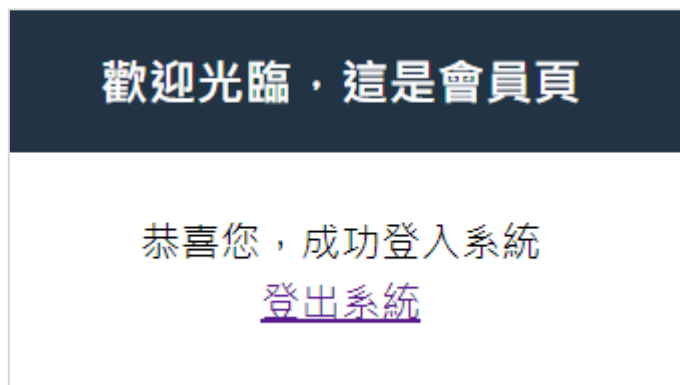
Required Procedure:
1. In the **Verification Endpoint**, if the username and password are verified, set SIGNED-IN state to TRUE and redirect the user to the **Success Page**.
1. In the **Success Page**, if the user clicks the sign out link, redirect to the **Signout Endpoint** where we set SIGNED-IN state to FALSE, and then redirect the user to the **Home Page**.
2. In the **Success Page**, we should always check the recorded SIGNED-IN state in the back-end logic. If it's FALSE, redirect the user to the **HomePage** without showing any content on the page.

Task 4: Dynamic Routing (Optional)

Apply the dynamic routing feature supported by Python Flask for building a simple procedure to calculate squared numbers.

Home Page:

- **URL:** http://127.0.0.1:3000/
- **Feature:** click button to calculate squared number.
- **Design for Reference:** a simple UI with a text input and a button.

Squared Number Page:

- **URL:** http://127.0.0.1:3000/square/某個正整數
- **Design for Reference:** show the squared number based on the number in the path of the URL defined above.

Required Procedure:

1. User enters text in the text input in the **Home Page** and clicks the button.
2. Check if the entered text is a positive number by JavaScript. If not, show a simple alert dialog with "Please enter a positive number" message inside.
3. Redirect the user to the **Squared Number Page** by JavaScript where the entered positive number should be appended to the end of the path.
4. Show the squared number in the **Squared Number Page**.