**Login UI**  
**Description:** This section describes the login page. The login feature allows players to log in to the system via username and password, and provides “Remember me” and “Reset password” functions.

**User Story (if applicable):**  
*"As a user, I want to be able to log in to the system with my username and password in order to access the home page. At the same time, when I forget my password, I want to be able to reset the password directly by entering my email address in order to quickly restore my account access."*

**1. Acceptance Criteria**

Define what needs to be met for this item to be considered **complete**.

✅ The user can log in using the username and password.

✅ If the password is wrong, the system will give a prompt.

✅ After successful login, you can go to the home page.

✅ If the user select “Remember me”, he/she automatically logs in at the next visit.

✅ The user can click “Reset password” and enter the new password to reset, if the email exists, the user can directly enter the new password to reset, after the user enters the new password (twice confirm), the system successfully updates the password, the user can use the new one to log in again; If the email does not exist, “The email is not registered” is displayed.

**2. Requirements & Specifications**

Provide clear **functional** and **non-functional** details.

* **Functional:**

POST /login/

email = [user@example.com](mailto:user@example.com)

password = 123456

Success: HTTP/1.1 302 Found

Fail: HTTP/1.1 200 OK

Content-Type: text/html/

<html>

<body>

<p>Invalid details: please try again</p>

</body>

</html>

POST /api/password-reset

If “Remember me” is selected, the system returns a long-term JWT Token (valid for 7 days); If it is not selected, the Token is valid only for the current session.

* **Non-functional:** Must handle 1,000 concurrent requests per minute to ensure security.

**Optional:** Include wireframes or process flow diagrams.

**3. Dependencies & Constraints**

* Needs a password for confirmation identify.
* Database: Store user accounts and passwords.
* JWT authentication system: Manages user sessions.(option)
* After the password is reset, the old password becomes invalid immediately.

**4. GWT**

Provide Given-Then-When for main cases and edge cases.

* **Given**: Describes the initial state and preconditions before the test begins
* **When**: Describes the actions taken by a user during the test
* **Then**: Describes the outcome that results from the actions taken in the "when" clause

**Scene1:** Users log in successfully

**Given:** The user is in the login page and has an active account. For example, account: [user@example.com,](mailto:user@example.com,) password: example.

**When:** The user enters the correct email address and password and clicks the “Sign in” button.

**Then:** The user successfully logs in and goes to the home page.

**Scene2:** Users enter wrong password

**Given:** Users are in the login page and they have active account.

**When:** The user enters the correct email address, but enters the wrong password, and clicks the “Sign in” button.

**Then:** The system will prompt the user to re-enter the password.

**Scene3:** Users click the “Remember me” button and log in successfully

**Given:** The user is in the login page, has an active account and check the “Remember me” choice.

**When:** The user exit the page and enter again.

**Then:** The user can log in automatically within 7 days.

**Scene4:** Users do not click “Remember me” button and log in successfully

**Given:** The user is in the login page, has an active account and does not check the “Remember me” choice.

**When:** The user exit the page and enter again.

**Then:** The user needs to log in again.

**Scene5:** User registered but forgot password

**Given:** The user clicks “Forgot password” and enters the registered email.

**When:** The user resets the password and confirms it.

**Then:** The user successfully updates the password.