1. Test: **Login / Verifying user login info - Unit Testing**

    Data: A test username (string) and password (string) will be used to log in. Some user data will be generated in an object of the type (user).

    Environment: Our Testing environment will be a clone of the Development Environment This environment will include all code that is considered ready to test in preparation for deployment to the Production Environment.

    Results: Dummy user data will be compared to retrieve user data. When logging in with the test username and password some user information will be generated. This user information will be compared to the expected user information.

    Testers: Other group members

2. Test: **Tie between users account and clash royale account - Unit testing**

    Data: To test this we will need a mock user of our site and a real ID to a Clash Royale account.

    Environment: Our Testing environment will be a clone of the Development Environment This environment will include all code that is considered ready to test in preparation for deployment to the Production Environment.

    Results: Test if an actual account was found given a real Clash Royale #ID and if the correct response occurs when given an invalid Clash Royale #ID

    Testers: Other group members

3. Test: **Finding a player's match history - User Acceptance Testing**

    Data: We will need a test user and #ID for an active, inactive, and banned account.

    Environment: Our Testing environment will be a clone of the Development Environment This environment will include all code that is considered ready to test in preparation for deployment to the Production Environment.

    Results: We will have to manually check that the actual match history of the user accounts matches the returned history for the test ID's. General corner cases of active accounts with full match history, inactive accounts with empty match history, and banned accounts with no available history should return as expected.

    Testers: Other group members, beta users with a clash royale account

4. Test: **Generating Random Challenge - Unit Testing [for attendance because missed lab11 due to illness - Derek Norman]**

Description: We want to be able to generate random card decks that the user can complete in order to increment the completed challenges counter.

On the click of a button the api call generates a random set of cards and is displayed under the challenge deck section.

Data: We will need a working card table with all the cards in the game.

Environment: Our testing environment will be a clone of the Development Environment. This environment will include all code that is considered ready to test in preparation for deployment to the Production Environment.

Results: We will have to test if random decks are being generated and the cards are being displayed correctly on the home page.

Testers: other group members, beta users with a clash royale account