1. Create 1 document per team, within the **milestones** folder in the project directory, that describes how, at least, 3 features within your finished product will be tested.
2. The test plans should include specific test cases (user acceptance test cases) that describe the data and the user activity that will be executed in order to verify proper functionality of the feature.
3. The test plans should include a description of the test data that will be used to test the feature.
4. The test plans should include a description of the test environment (localhost / cloud) that will be used to test the feature.
5. The test plans should include a description of the test results that will be used to test the feature.
6. The test plan should include information about the user acceptance testers.

User acceptance testers:

Internal QA Team: Responsible for validating initial functionality, error handling, and session management.

User Acceptance Test Group: Small group of friends who enjoy point and click games selected for testing to confirm user experience and verify data accuracy, with feedback collected on performance for each test.

Feature Test Plan 1 - Scoreboard Functionality:

1. **View Scoreboard (HTTP Status Code: 200 OK)**
    - **Description**: Player views the scoreboard to see their rank and scores of other players.
    - **Test Steps**:
        - Player selects the "Scoreboard" option.
        - Retrieve and display sorted player scores from highest to lowest.
    - **Expected Result**: Scoreboard displays top scores with player's current rank highlighted.
2. **Update Score (HTTP Status Code: 200 OK)**
    - **Description**: Upon completing a game, player’s score is updated in the scoreboard.
    - **Test Steps**:
        - Player completes a game with a new score.
        - Submit score update to the database.
        - Refresh scoreboard and verify updated ranking.
    - **Expected Result**: Player’s score updates accurately on the scoreboard, adjusting their rank if applicable.
3. **Error Handling for Score Update (HTTP Status Code: 500 Internal Server Error)**
    - **Description**: Simulate a server error during score update.
    - **Test Steps**:

        - Trigger a score update with an error simulation (e.g., database down).
        - Check if an error message is displayed, prompting the player to retry or contact support.
    - **Expected Result**: Player is informed of the issue without disrupting other game functions.
4. **Refresh Scoreboard (HTTP Status Code: 200 OK)**
    - **Description**: Player refreshes the scoreboard to see the latest scores.
    - **Test Steps**:
        - Player selects "Refresh" on the scoreboard.
        - Ensure latest scores are retrieved and displayed correctly.
    - **Expected Result**: The scoreboard reflects the most up-to-date scores without any stale data.

Test Data

- **Valid Score Data**: Test usernames with existing scores in the database, including high scores, average scores, and new entries.
- **Invalid Score Data**: Scores that fall below a defined minimum threshold or entries without usernames to validate error handling.

Test Environment

- **Localhost**: Conduct initial tests in a localhost environment to ensure scoreboard feature functions as expected.
- **Cloud**: Once validated locally, perform tests in the cloud environment to confirm consistency in HTTP responses, data retrieval, and score updates.

Test Results

- **Pass**: All HTTP status codes, score updates, and display behaviors align with expected outcomes.
- **Fail**: Document any incorrect score updates, incorrect rankings, or incorrect status codes, detailing issues for debugging.

Feature Test Plan 2 - Inventory Functionality:

Test Environment: The user will test the game on the website, hosting it on their own computer (http://localhost:3000/).

Test Data: An existing username and password with no game progress.

Test Plan & Results:

Step	Desired Result

1.The user will navigate to ‘scene 1’.	Scene 1 appears. (A snowy scene with a snowman.)
2.Click on a carrot in the scene.	The carrot should appear in a slot in the inventory and disappear from the scene.
3.Click on the carrot in the inventory.	The carrot should no longer be highlighted.
4.Click away from the carrot.	The carrot should no longer be highlighted.
5.Click on the snowman.	Nothing should happen.
6.Click on the carrot again, and then click randomly on the screen, anywhere but the snowman.	Nothing should happen.
7. Click on the snowman.	The carrot should appear on the snowman and disappear from the inventory.

Feature Test Plan 3 - Login Functionality (Leo)

1. The test plans should include specific test cases (user acceptance test cases) that describe the data and the user activity that will be executed in order to verify proper functionality of the feature.

   Valid Login (HTTP Status Code: 200 OK)

   - Player enters a username and password
   - If the username/password combination exists login user
   - Change page to the game

   Invalid Login (HTTP Status Code: 401 Unathorized)

   - Player enters an username and password
   - IIf the username/password combination does not exist or is incorrect, display error message
   - Clear fields and prompt user to try again

   Verify User Progress (HTTP Status Code: 200 OK)

- Upon valid login, verify user inventory/progress/score/time is correct

2. The test plans should include a description of the test data that will be used to test the feature.

   Valid User Data: Username and password from database records with saved game data.

   Invalid User Data: Nonexistent username and incorrect password combinations to test error handling.

3. The test plans should include a description of the test environment (localhost / cloud) that will be used to test the feature.

   Localhost: Initial testing will be performed in a localhost environment to validate functionality and HTTP responses.

4. The test plans should include a description of the test results that will be used to test the feature.

   Pass: All functionality and HTTP status codes behave as expected; game data loads accurately for valid logins.

   Fail: Errors in functionality (e.g., incorrect status codes, incorrect game data load, or incorrect error messages) are documented for further review and debugging.