

Title: Christmas Point & Click Adventure

Members:

Sofia Poulsen sopo6073@colorado.edu onionSoap

Emily Smith emsm2434@colorado.edu emsm2434

Leo Zhu lezh5376@colorado.edu zeolhu

Dorjee Zhang dozh9458@colorado.edu dorjeezzz

Project Description: A 200 word summary of the project (196/200 currently)

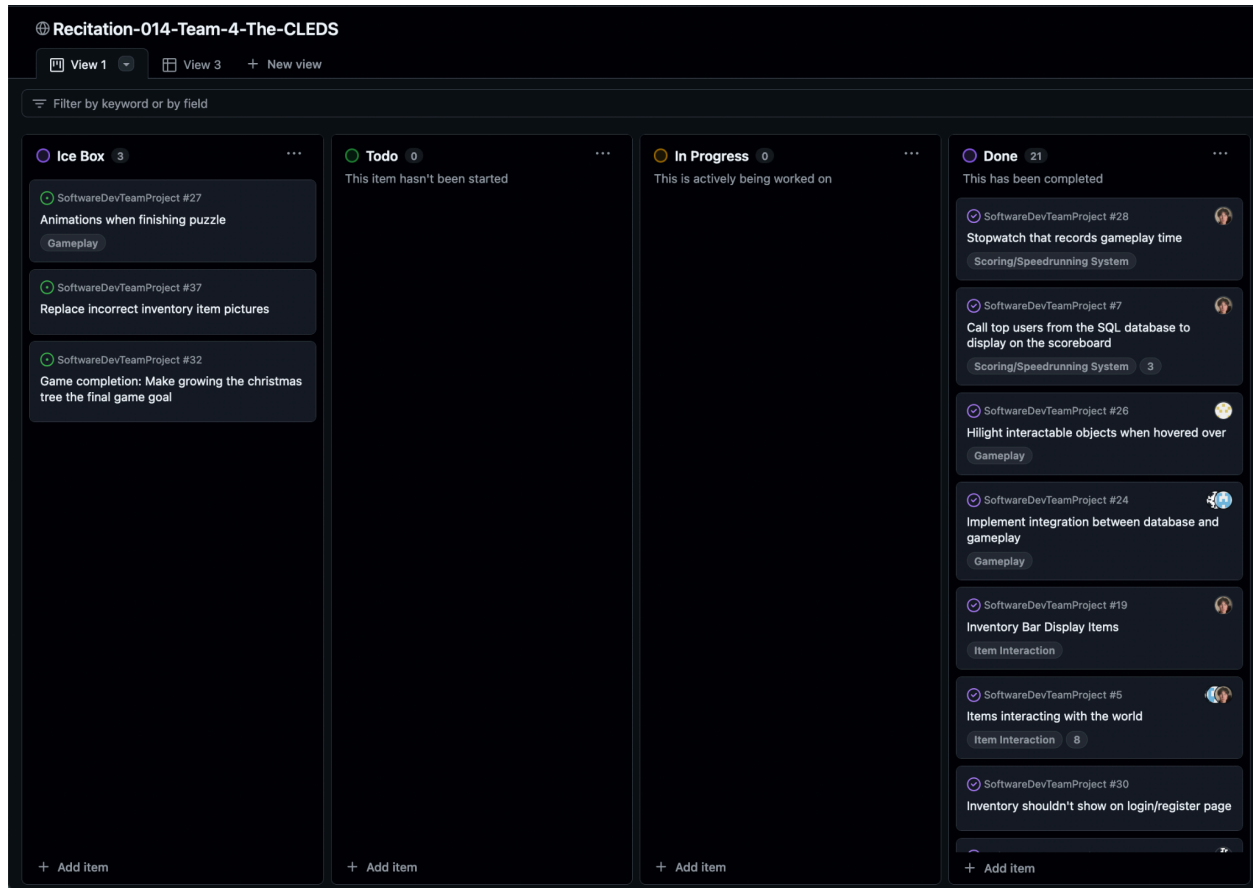
We developed a point-and-click adventure game designed to run within a web browser. This game serves as a demo for a larger project and centers around puzzle-solving mechanics that challenge players to think creatively. Players navigate through static screens, interacting with various objects to transition between rooms, collect items, or use those items to solve intricate puzzles. Each puzzle solved earns the player 1 point, with the game culminating in victory when the player reaches a total of 13 points.

The puzzles vary in complexity, with some requiring specific items from previous pages to complete. These items can either be immediately available or hidden behind other puzzles, creating a layered and interconnected gameplay experience. Certain actions or items on one screen may unlock new possibilities on others, encouraging exploration and strategic thinking.

The ultimate objective is to solve all the puzzles and climb to the top of the leaderboard, which tracks players who complete the game in the shortest time. Progress is saved exclusively for registered users, whose data is securely stored in a database and accessed upon login. This feature ensures a personalized experience and fosters competitive play while preserving the integrity of player achievements.

Project Tracker - GitHub project board:

<https://github.com/users/onionSoap/projects/3/views/1>



Video:

https://github.com/onionSoap/SoftwareDevTeamProject/blob/main/MilestoneSubmissions/video_demo_with_audio_FINAL.mp4

VCS: <https://github.com/onionSoap/SoftwareDevTeamProject>

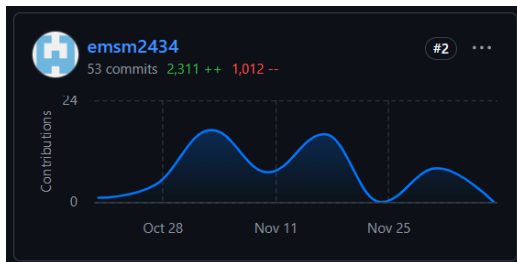
Contributions:

- **Sofia:** I implemented the main gameplay. I created the images for all of the scenes, implemented the 'puzzle-solving' mechanics, swapping scenes and the ability to pick up items. I also worked on integration with the database, figuring out how to send requests to our server with axios, and helped my team members with debugging and merging branches. In the final week I took a high-level view of the project and ensured the entire game was integrated, playable from end-to-end, and that all the bugs were squashed. I worked with handlebars, javascript, and edited the index.js endpoints and the CSS styling.



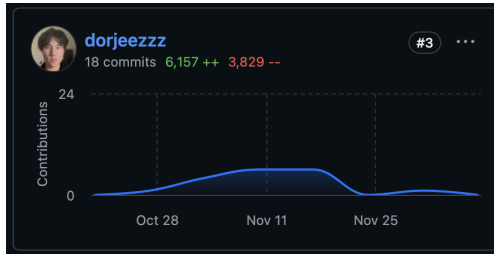
(The additions and deletions are slightly oversized because I kept uploading & deleting SVG images, which each contain quite a few lines.)

- Emily:** I created the database and the default information inserted into it when launching the server. I also created most get/post/and patch requests in index.js, ensuring they communicate with the database efficiently. I helped convert the gameplay from “dummy-data” based to fully integrated with the database, and ensured that each user’s gameplay was unique to them and did not impact other user’s data. I ensured the formatting for partials and pages was adhered to and consolidated most CSS files to the stylesheet. Finally, I created the test cases for testing the basic gameplay mechanics of the game.



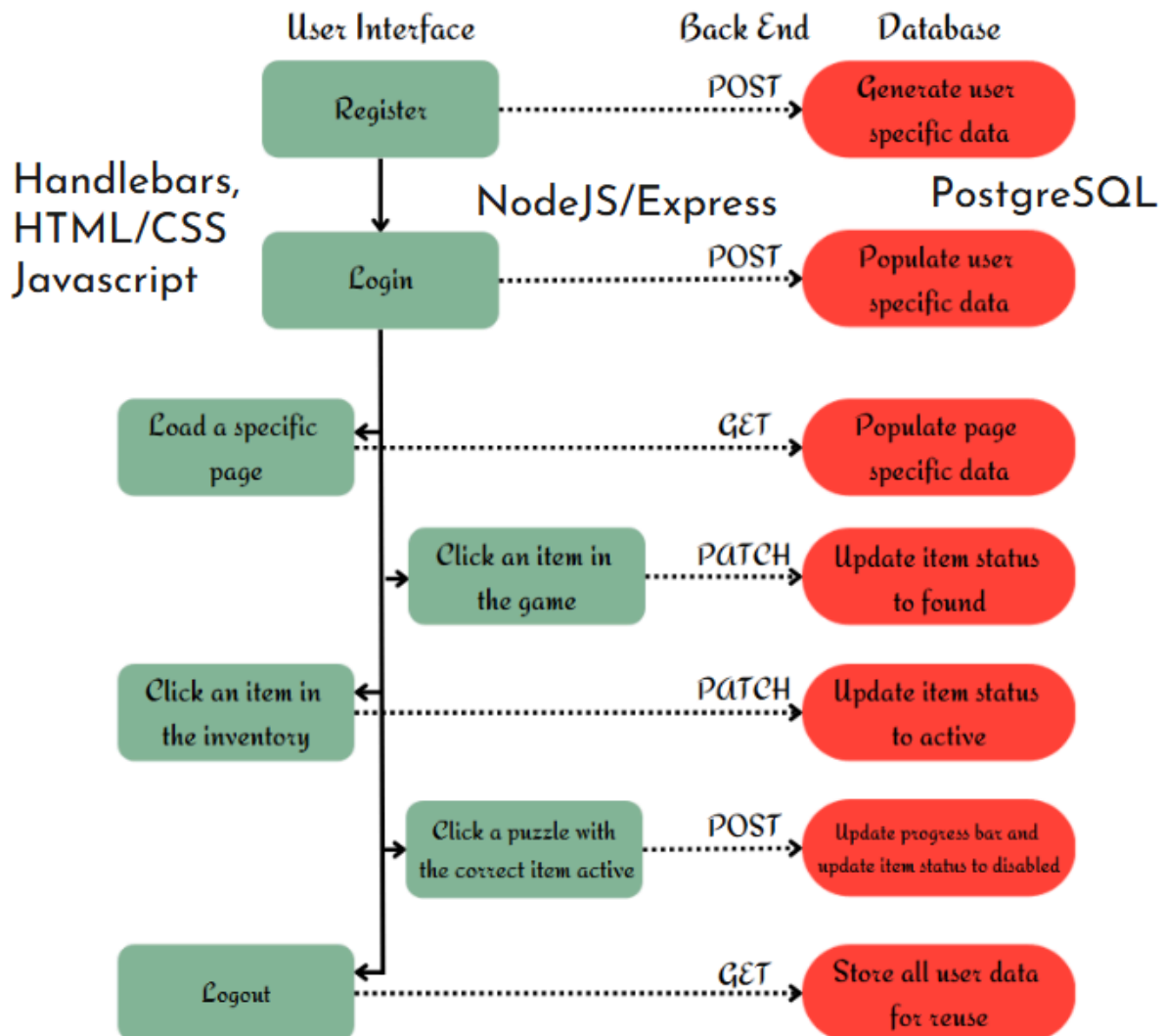
(The dip towards the end was the week of thanksgiving break.)

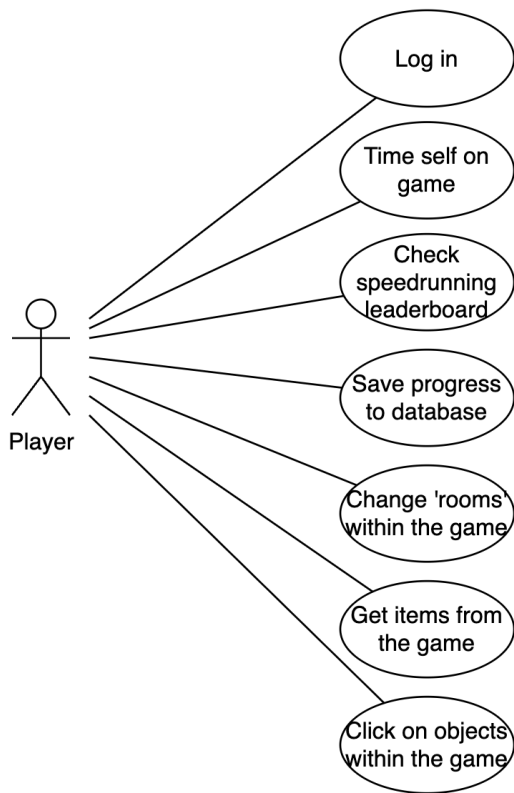
- Dorjee:** I created all of the inventory stylings, connected all items from the database to the items in the front end, revamped the database so that it included a time, created the timer that automatically started upon loading and paused on login and register, created every post request in the index.js to keep the timer continuous with the account as well as making it so that inventory data would be stored to the proper person. I dropped the old database table and created a new one so it would properly store all information. In the final week, I created test cases for the timer and the inventory, and worked on resolving bugs in the inventory such as using unknown or disabled items, preventing the timer from going on infinitely, and sorting the finish times in ascending order.



- **Disclaimer regarding commits:** These commits are only to main, as I had many more commits to both my respective branches, dorjee and dorjee2
- **Leo:** I Implemented the login/register page as well as the overall background scene for our game. Planned out the levels and puzzles for our game and created some test cases for login pages.

Use Case Diagram:





Wireframes:

Main Page (new user)

Login	Progress: 0%	Time 00:00	
page 1	page 2	page 3	page 4

Register to Begin

Username

password

already have an account? [Login!](#)

ITEMS							

LEADERBOARD

Login Page (returning user)

Login	Progress: 0%	Time 00:00	
page 1	page 2	page 3	page 4

Login to Continue Adventure!

Username

password


New? [Register here!](#)




ITEMS							

LEADERBOARD

Game Page | Example (logged in)

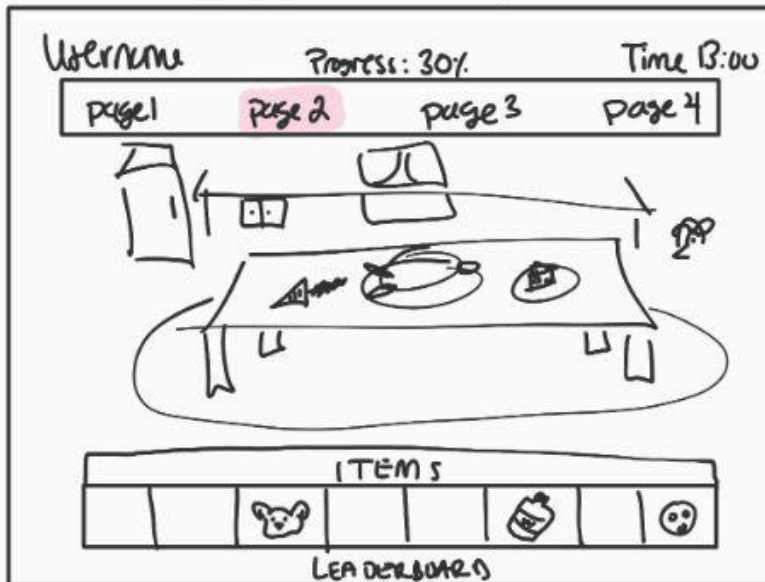
Username	Progress: 30%	Time 13:00	
page 1	page 2	page 3	page 4



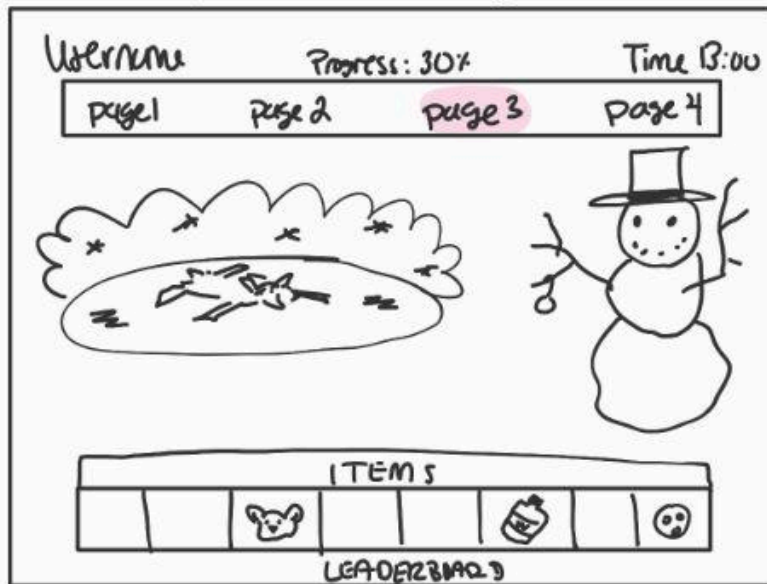
ITEMS							
							

LEADERBOARD

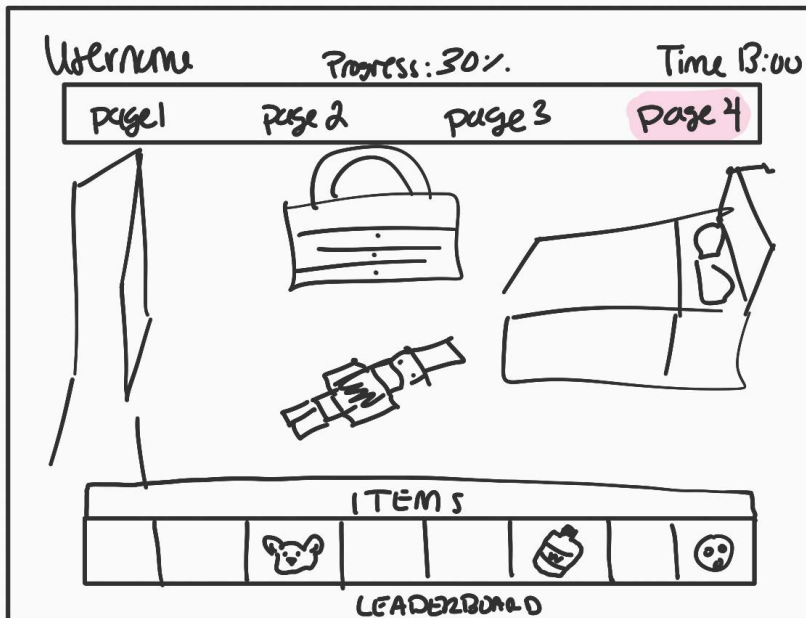
Game Page 2 Example (Logged In)



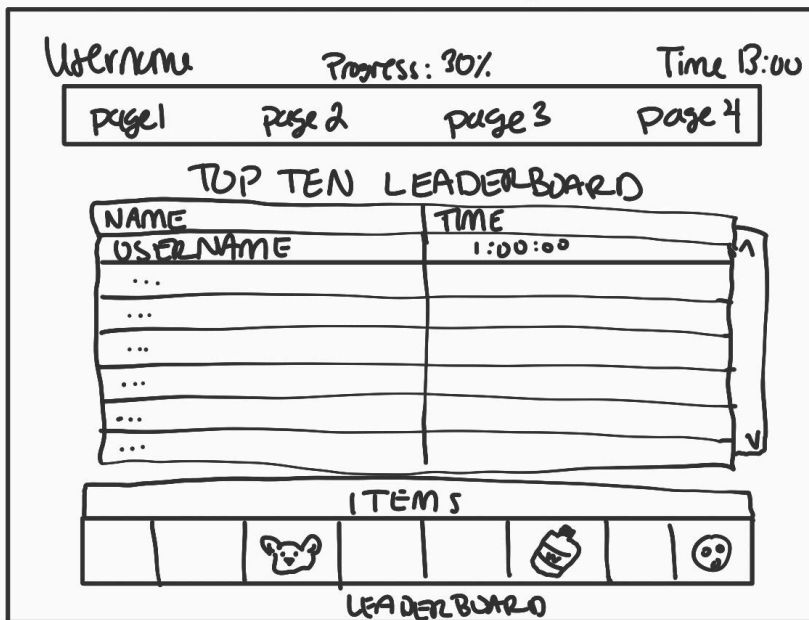
Game Page 3 Example (Logged In)



Game Page 4 Example (Logged In)



Leaderboard (Logged In)



Testing:

Test Cases:

(We edited these tests to reflect the current website, as some of our plans have changed. We performed all of these tests with a fellow CS student who enjoys point and click games.)

Feature Test Plan 1 - Scoreboard Functionality:

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.

Step	Expected Result	Result
<p>View Scoreboard (HTTP Status Code: 200 OK)</p> <p>Description: Player views the scoreboard to see their rank and scores of other players.</p> <p>Test Steps:</p> <p>-Player selects the "Scoreboard" option.</p>	Scoreboard displays top scores with player's current rank.	Pass
<p>Refresh Scoreboard (HTTP Status Code: 200 OK)</p>	The scoreboard reflects the most up-to-date scores without any stale data.	Pass, but the preset data from the test accounts 'admin' and 'beep' appear, so we would remove this in the future.

<p>Description: Player refreshes the scoreboard to see the latest scores.</p> <p>Test Steps:</p> <ul style="list-style-type: none"> -Player selects "Refresh" on the scoreboard. -Ensure latest scores are retrieved and displayed correctly. 		
---	--	--

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. (User: admin
Password: admin & the database was cleared beforehand.)

Test Plan & Results:

Step	Desired Result	Result
1.The user will navigate to 'scene 2'.	Scene 2 appears. (A dough bowl surrounded by a carrot, and two dough ingredients.)	Pass
2.Click on a carrot in the scene.	The carrot should appear in a slot in the inventory and disappear from the scene.	Pass
3. Click on the butter in the scene.	The butter should appear in a slot in the inventory and disappear from the scene.	Pass
3.Click on the carrot in the inventory.	The carrot should be highlighted in the inventory.	Pass

4. Click on the butter in the inventory.	The carrot should no longer be highlighted, and the butter should now be highlighted.	Pass
5. Navigate to Scene 1.	Scene 1 appears. (A snowman and a festive house.)	Pass
6. Click on the snowman.	Nothing should happen.	Pass
7. Click on the carrot again, and then click randomly on the screen, anywhere but the snowman.	Nothing should happen.	Pass
8. Click on the snowman.	The carrot should appear on the snowman and disappear from the inventory.	Pass

A user clicked through the game, following these steps, and all tests were passed.

Feature Test Plan 3 - Login Functionality

1. 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.

2. 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.

3. 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.

Step	Expected Result	Result
Valid Login (HTTP Status Code: 200 OK) - Player enters a username and password ('admin' 'admin')	HTTP Status Code: 200 OK. If the username/password combination exists login user change page to the game.	Pass, redirects to page 1.
Invalid Login (HTTP Status Code: 401 Unauthorized) - Player enters an invalid username and password ('a', 'a')	HTTP Status Code: 401 Unauthorized If the username/password combination does not exist or is incorrect, display error message. Prompt user to try again.	Pass, an error message pops up on the login page.
Verify User Progress (HTTP Status Code: 200 OK) - Upon valid login, verify user inventory/progress/score/time is correct	HTTP Status Code: 200 OK.	Pass, tried with the admin user and timer, progress & inventory stayed consistent.

4. 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.

Feature Test Plan 4 – New User Registration:

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

Test Data: A user will create a new account (username: test) and verify all default information is correct.

Test Plan & Results:

Step	Desired Result	Result
1.The user will navigate to the registration page.	The registration page appears.	Pass
2.The user will enter a known username (admin) and any password.	An error should appear on the screen.	Pass
3. The user will now enter a unique username (test) and any password.	There should be a success message and the page should be redirected to login.	Pass
3.The user will log in using the unique username (test) and their password.	Scene1 page should load. Their unique username, 0/13 progress bar, and a freshly started timer should be at the top of the screen.	Pass
4.The user will click between the different scenes and see all items are in their default positions.	Scene1 items (flour, lights), Scene2 items (carrot, butter, sugar), Scene3 items (none), and Scene4 items (wreath, star) should be loaded onto the screen.	Pass

A user clicked through the game, following these steps, and all tests were passed.

General Observations:

I observed a fellow CS student clicking through the website, giving them no instructions.

- What are the users doing? What is the user's reasoning for their actions?

When logging in and registering the user, unprompted, attempted to enter in impossible data. This is probably because they are a CS student and like to look at the edge cases of applications, and were checking if it was made correctly. They also attempted a SQL injection, but luckily this did not work. They also, near the end, clicked on random areas with random items. This is likely because the game appeared to be a bit confusing, so they were not sure how to proceed.

- Is their behavior consistent with the use case?

Yes, as shown in the 'Test Cases' section above.

- If there is a deviation from the expected actions, what is the reason for that?

There is none.

- Did you use that to make changes to your application? If so, what changes did you make?

The user could not recognize what the 'sugar' was, so in the future we could use a more clear icon in the inventory or include alt text. They thought it was tissue paper. However, at the time of testing we decided not to make any changes because the most important part was the functionality of the game, and we were short on time.

Deployment: <https://softwaredevteamproject.onrender.com/login>