CS 161 Product Test Plan

Section 1: Feedback from Others

Feedback: N/A as of now.

Section 2: Project Overview and Build Instructions For Testing

Detailed Description of the End Product:

In WordSearch, players search for hidden words in a grid of letters. The game can include different categories and difficulty levels, making it suitable for players of all ages. The primary objective of this project is to develop a web-based straightforward Word Search game that provides users a fun and interactive way to learn vocabulary by exposing players to new words within themed puzzles. They aid in spelling practice as players must correctly identify and spell the words they find. These puzzles also improve pattern recognition skills as players search for words in a grid of letters. Wordsearches require focus and attention, contributing to cognitive development.

Testing Instructions for Expert Users to Carry out the Tests:

Building the Frontend:

The frontend is developed using Node.js, currently undergoing active testing with version 18.16. Setting it up manually is straightforward and involves the following steps.

1. Download and Install Node.js
   - Download link: [Node.js](https://nodejs.org/en) (Choose the "Recommended For Most Users" version).
   - After installation, validate by opening a terminal and typing `node -v`. You should see the version number if Node.js is installed correctly.

2. Install Required Dependencies
   - Commands are the same for Windows, MacOS, and Linux.
   - Navigate to the project folder in the terminal (e.g., `<Your Local Path>/CSS566-Final-Project/`).
   - Change directory to the source -> Node.js client folder: `cd src/client`.
   - Install dependencies: `npm install`.

3. Start the Node.js Client
   - Once all dependencies are installed without errors, start the frontend Node.js client with either of the commands:

- `npm run start:dev` for the development environment.
- Wait for the client to start. Once all services are up, a new page will open in the default browser.

Building the Backend:

The backend is constructed using pure Python and the popular microframework Flask. It's currently actively tested with Python 3.11, Python 3.10, and Flask 2.3.2. The manual setup process involves the following steps.

1. Download and Install Python
   - Download Link: [Python](https://www.python.org/downloads/) (Choose the newest version of either 3.11 or 3.10).

2. Install Virtual Environment (Recommended But Not Required)
   - Navigate to the project folder in the terminal (e.g., `<Your Local Path>/CSS566-Final-Project/`).
   `python -m venv env`
   `source env/bin/activate`

3. Install Required Packages
   - Make sure you have the `requirements.txt` file in the `src` folder.
   - Ensure your terminal is in the `src` folder and then install all required packages using `pip install -r requirements.txt`.
   Install certifi `python3 -m pip install certifi`

4. Start the Flask Server
   - Once all packages are installed and you are still in the `src` folder, start the Flask application using the provided commands based on your OS.
   - Navigate to `http://127.0.0.1:5000/` in your browser to verify if the Flask application is running correctly.

Requirements for Hardware, Software, and/or Password Key:

- Hardware: To test the project, you need a computer, a mouse or a trackpad, and a keyboard.
- Software: To test the project, the software needed is an IDE and a Command Line
- Interface (CLI). Such as VSCode
  Password Key: N/A

Section 3: Automation (Complexity of Testing Instructions)

Time and Effort to Set Up the Test Plan For Execution:

- Cloning the Project, Installing Dependencies, and Starting Project : 15 min
- Download and Setup for All Testing Tools: 20 min
- Time for Testing Features: 35 min (approx)
- Time for Backend: 30 min (approx)

Testing Tools:

- Frontend testing: Jest, Selenium (for end-to-end testing).
- Backend testing: pytest, Postman (for API testing)

Section 4: Testing Information

Time to Finish the Tests: TBD, since some features are still under development.

Test Cases for Each Feature:

Page	Feature	Description	Testcase	Time Taken
Landing Page	Landing Page	First page user lands on. Should be able to see leaderboard, user history, sign up button, normal game, daily rewards puzzle and create your own puzzle buttons.	Open the Ui and check if the buttons are visible and clickable	<2 minutes
User Authentication	Sign Up	Users should be able to sign up using email and password.	Can navigate to the UI and sign up with email id and password	<1 minute
	Log Out	User should be able to log out of the app	user should be able to click on user icon on top right corner and log out	< 1 minute
	Log In	User should be able to log in using email and password.	Can navigate to the UI and log up with email id and password	< 1 minute
	Forgot Password	Click on the user icon and click on the forgot password.	Clicking forgot password will show you a screen to reset account	<2 minutes
	User history	User will be able to see all the games played and the scores on each	Navigate to landing page and should be able to see on the screen	< 1 minute

Normal Game	selecting difficulty level	User should be able to select the difficulty level from the drop down.	Navigate to normal game page and select from dropdown	< 1 minute
Normal Game/ Daily Reward	Timer	Timer should be running from the time the difficulty level is set.	User can see on screen	< 1 minute
	Selecting words	User can select the words from the list of words displayed from the grid	Matching words should result in the word being stricken from the	< 2 minutes
	storing and retrieving puzzles	User should be able to see the game and be able to select the word.	The right word should result in a popup saying the correct answer. The wrong word shows pop up saying incorrect word selected	<5 minutes
Leaderboard	Share Score	On finishing the game the user should be able to share the result to the leaderboard.	If the user is logged in it should be shared with the user name or anonymously if user is not logged in	< 2 minutes
	View Leaderboard	The user should be able to see game specific leaderboard on the game tab	The user should be able to see game specific leaderboard on the game tab	< 2 minutes
Design Puzzle	User should be able to create a puzzle	User should be able to create a puzzle	Can test using postman	< 2 minutes