Final Project Writeup: To Do Garden

Jennie Wei

Screen sizes

375x667(iPhone SE) to 1920x1080(24” monitor) fully responsive

Part 1

To Do Garden is a tasklist website where users can create task lists to keep track of tasks they need to complete. Users can customize all aspects of their lists at any point by adding tasks, changing list titles, changing existing tasks, and creating new lists. As users check off tasks, they grow flowers in each task list.

When the user first opens the website, there will be a ‘Welcome list’ that details how to use the website. The user can delete this list once they don't need the instructions anymore. This website is like a planner, so the information it conveys depends on what tasks the user inputs. Through the flower growing feature and tracking flower count, it also displays the user’s productivity and progress on each task list in an aesthetically pleasing manner.

This website is interesting as users can check it to keep track of tasks they need to complete. It is engaging as there is an incentive/rewards system for completing tasks as users get to grow their gardens.

The target audience is anyone who wants a simple way to keep track of tasks on a minimalist and cute interface!

Part 2

- *Create New Task List:*
    - Click on the “+ NEW TASK” button
    - type a title into the title textbox & click “+New Task” to type tasks(optional)
    - click the “Save” button to save the list to the home page or click “X” button to close the list(the new list will not be saved)
- *Open list options menu:*
    - Click on the three dots
    - click “Close” to close it
- *Edit Existing List:*
    - Click the three dots
    - click “Edit List”
    - type into live textboxes to edit titles and existing tasks

    - click “Save” to save the new tasks and other list edits or click “X” to cancel edits
- *Delete List:*
    - Click the three dots on a list
    - click “Delete list” to delete (Note: if the user deletes all lists, when the website refreshes the final list will reappear as I didn't want the website to open with no lists since it looks bare)
- *Add a New Task:*
    - Click the three dots
    - Click “Edit list”, click “+New Task”
    - start typing, click “Save” to save new tasks and other list edits
- *Grow Flower:*
    - Click checkboxes on tasks and the flower to the right will grow automatically
- *See completed tasks:*
    - Click the “>” to display all completed tasks
    - Click the “v” to minimize completed tasks

Part 3

React/Create React App/react-responsive-masonry:

Why: I wanted to learn React and I thought that React components would make this website easier to code

How:

I created task components for many of the things that need to be rendered multiple times or would be easier to manage as their file such as task lists, edit options menu(For each task list), edit mode

What does it add:

React useState allowed my website to easily live update every time the user made a change(changing tasks, flower growing, etc.)

React-responsive-masonry add-on allowed me to easily create a masonry effect with different-sized task lists which looked much better than wrapping flexboxes

Part 4

In my Figma prototype I was going to have all the displayed tasks be live textboxes that users could edit at any time from the dashboard, however, I ended up changing so that

each list had an ‘edit list’ option where users could edit their list on a list popup/overlay after clicking the ‘edit list’ button. This way users would not accidentally edit their list by clicking a task. Additionally, I added a minimize feature to the completed tasks section because I realized it would easily get very long.

Part 5

Initially, I struggled a lot with learning React and using JavaScript for dynamic elements of my project as I couldn’t use anything DOM-related that we learned in class with React components. My project was also very CSS heavy, and I ran into many issues with creating dynamically sized text boxes for the live textareas and accounting for the possibility that the user could type text of any length, so I spent a lot of time formatting those into my layout.

WAVE accessibility test: