POMO PIPELINE

PROJECT STRUCTURE:

CSS (Planned) - One CSS page will be used to apply styling for all pages. This will help keep styling consistent, easier to implement, and easier to upkeep. This page will be overseen by one person who will make sure additions are organized and non-redundant. Anyone can contribute to this page so long as it's approved by the CSS Lead.

HTML (Planned) - All Pomo pages will have their own HTML placing containers and objects according to page functionality. These pages will be overseen by one person, but anyone can contribute so long as it's approved by the page Lead.

JAVASCRIPT (Planned) - In order to reduce round-trip time and maintain simplicity, one JS page will be used to handle the logic for all pages. Each page will have its own section clearly labeled and managed by its respective HTML lead. Anyone can contribute to any section so long as it's approved by that section's HTML lead.

GITHUB (Working) - Will serve as our main repository and will be used to host our website via GitHub pages. Actions like commit, push, and pull requests will also be used to trigger CI/CD workflows via GitHub Actions.

CODING:

ISSUES (Working) - All work done must be attached to an issue. Issues are generated using the following scale: 1Point(< 3 hours of work), 2Point(< 1/2 day of work), 3Point(1 day of work).

DEV BRANCH (Planned) - Development cannot be done on the main branch. All work must be done on a separate DEV branch and must be approved before being pushed onto the MAIN branch.

MAIN BRANCH (Planned) - Pushing a DEV branch into MAIN requires automated and human review by Ari/Jack and the Leads of the pages DEV will affect. This will help aid integration and ensure no DEV push tanks MAIN branch.

CONTINUOUS INTEGRATION:

CODE LINTING (Working) - Code pushed onto the Main branch is checked for programmatic and stylistic errors using different linting tools specific to file type. This will help reduce syntax errors and also maintain code consistency.

1. ESLint, finds syntax errors and reports on patterns detected in JavaScript files. This helps avoid bugs and improves the consistency of JS code. ESLint generates a config file that allows us to run the tool on GitHub.
2. HTMLHint, is a static code analysis tool for HTML, that shows syntax errors and keeps the consistency of HTML in HTML files. HTMLHint also generates a config file that allows us to run the tool on GitHub.
3. StyleLint, an advanced CSS-linting tool that reduces errors and enhances systematics methodology in CSS files. Stylelint generates a configuration file in order to be run.

CODE QUALITY (Working) - CodeFactor is free for 1 private repo and will help automate our code review process. On commit, CodeFactor will review code for errors, auto-fix when able, and provide feedback. We sign up then import it to the project repository. Then CodeFactor is going to automatically inspect any commits and pull requests.

UNIT TESTING (Working) - Code pushed to a DEV branch also undergoes a set of unit tests. These tests will ensure that units and components work correctly before being integrated with the rest of the project. We will be using the Jest framework to build and design these unit tests and Github Actions to implement the tests on push.

INTEGRATION TESTING (Planned) - This set of automated testing will check to see if a DEV branch is compatible with the rest of the project. These tests will be created by the Lead of the DEV branch and the Leads of the pages the merger will affect.

HUMAN REVIEW (Planned) - Will be done when a DEV branch is merged onto the main branch. The review is done by the Leads of the pages the merge will affect. This will ensure pages always have the proper incoming/outgoing data and behavior.

CONTINUOUS DEPLOYMENT:

FINAL TESTING (Planned) - This testing will check the entire app’s behavior and performance are up to standards.

JSDOC GENERATION (Planned) - Done when Main branch code passes final testing. JSDocs improve the readability and clarity of code by generating an HTML page where you can view your documentation. This makes it easier for developers to understand method functionality, variable types, and ultimately maintain code.

PROJECT DEPLOYMENT (Working) - The project is automatically released whenever a merger to the main branch occurs via GitHub Pages.