**<u>PROJECT NAME</u>: AJEC Re-Cycle**

**<u>Project Description</u>**
The purpose of this web application is to allow users the ability to resale and purchase used bikes. New guests to the site will have the ability to view bike inventory and limited access to bike details. Once guests create a user profile, they will be able to view full bike details, which includes the seller's username and email address. In addition, users have the ability to post bikes for resale with the option to edit and delete the post as needed.

**Timeline**

- Thursday (All Day)
  - AM
    - Get Whimsical Approved
    - Complete Read.Me
    - Complete Kanban
    - Set-up File Framework
  - PM
    - Complete Backend
    - Deploy to Heroku/Atlas
    - Assign Tasks to Team for Frontend Development
- Friday (MVP Due)
  - AM/PM
    - Work on individual tasks
    - Complete MVP for each screen
    - Create Pull Requests for Each Individual Screen
- Saturday (All online from 9AM - 11AM)
  - AM
    - Team Check-in
    - Review Pull- Requests
    - Merge All Pull Request into Development Branch
- Sunday (Flexible, As needed)
  - AM
    - Team Check-in
    - Assign CSS tasks
- Monday (All online from 9AM - 11AM- HOLIDAY)
  - AM
    - Complete individual CSS tasks
    - Create Pull Requests for CSS tasks
- Tuesday (All-Day)

        - AM
            - Review MVP functionality
            - Deploy to Netlify
            - Team Check-in
            - Review Pull- Requests
            - Merge All Pull Request into Development Branch
        - PM
            - Identify PMVP item to integrate into project
- Wednesday(All-Day)
    - AM
        - Team Check-in
        - Work on PMVP as a Group
        - Create Pull Requests
    - PM
        - Review Pull Requests
        - Merge into Development
- Thursday
    - AM
        - Review all Functionality
        - Merge into Main Branch
    - PM
        - Presentation Run-Through
- Friday
    - Presentation

**Team**
Artem Furman, Cindy Mit (Github Czar), Emanuella Altidor, Joel Giroux

**Personal Strengths**
Artem: React components
Cindy: React UI and CSS
Emanuella: React UI, CSS, debugging
Joel: React UI, creating algos for functionality, and debugging

**Personal Challenges (Could Use More Exposure)**
Artem: Debugging personal code
Cindy: Creating functions and conditionals and debugging
Emanuella: Creating functions and conditionals
Joel: CSS, authentication

## GROUP PLANNING

**Team Goals & Values**

- Functional Code
- Interactive, Attractive UI
- Thorough Communication Between Teammates

**Team Communication Preferences**

- Slack at any time; DND is on during sleep hours
- Team operating hours are M-F, 9-5PM; HW: 5:30-9PM. Friday evening through Sunday will be as needed.
- Respond within 1 hours.
- Emanuella is unavailable some days after 10PM.
- Joel is unavailable after 7PM
- Cindy may be periodically unavailable from 5-9:00 PM during dialysis weekdays.

## CODING PRACTICES

**Conventions:**

Branch Naming

- master, development;
- sub-branches use "initial-dev" convention, ie: *mk-feature-forms*
- *hotfix*, *feature*, *refactor*

Practices (Please note, this is an example. You don't need to follow it to a T, but it will prevent further frustrations.)

1. **Never** code on, merge, or push to Master

2. Git Maintainer will create a Development branch and set the upstream so it's available to all group members.
3. Group members should run a *git checkout -b development* and a *git pull origin development* to link their remote branch to the origin branch.

4. Group members can then run *git checkout -b feature-branch-name* from their local Development branch.
5. Make code additions.

6. Once a group member’s code is complete and ready to merge with Development, that group member should run a “safety pull” from Development to check for any updates made while you were coding.
7. If there are no conflicts, you can initiate a pull request on GitHub. Confirm the PR is pulling from your branch and comparing to Development; title the PR “BRANCH NAME to DEVELOPMENT.” This PR and merge will require the git maintair’s approval to finalize the merge.
8. Once that merge is finalized, all group members should add and commit their code to their local branch.
9. Then, they should run a *checkout* to their local Development branch, followed by a *git pull origin development*;
10. Then they should run a checkout to the feature branch they were working on and also run a *git pull origin development*.
11. Continue coding on your feature branch. Return to step 6 as needed.

12. Once your Development branch is ready to merge with Master, the Git Maintainer can make a PR from Development to Master. This PR requires at least 3 of 4 group members’ approval to finalize the merge.
13. Once that merge is finalized, all members should *pull origin master* to all local main and sub branches, if possible.

Express Routes
MongoDB Database & Collection Names
React Component Names & Folders

**Task Tracking**
Kanban Board