Quality

This document outlines the necessary documentation indicating the quality standards to be followed in order to maintain high quality across the app and it's documentation.

Table of contents

Coding Standards

Coding standards must be followed to ensure that the code is consistent, error free, easily understandable and can be easily maintained. Below are the standards to be adhered by all the team members during the development of the AIA web app.

JavaScript Coding Standards

Indenting:

- Every block of code must indent using two space characters.
- A block of code must not end with trailing whitespaces.

Semicolons:

- All statements () must be followed by a semi-colon (;)
- Return values must start on the same line as the return keyword.

File-closure:

- All JavaScript code must be declared inside a closure wrapping the whole file.

Camel Casing:

- The first letter of each variable or function should be lowercase, and the first letter of subsequent words should be capitalized.
- There should not be underscores between the words.
- In case a variable contains a jQuery object, the variable must start with a dollar sign ($).

Variables and Arrays:

- All variables must be declared with "var" before they are used and should be declared only once.
- All variables should be declared at the beginning of a function.
- Each variable assignment should be declared on a separate line.

Constants:

- Pre-defined constants should be all-uppercase and words separated by underscores.
- Variables added via PHP should be lower camel case to maintain consistency with other JavaScript variables.

Arrays:

- Arrays should be formatted with one space separating each element and the assignment operator.
- If the line is longer than 80 characters, each element should be broken into its own line.
- Use trailing comma after last element in multi line arrays.

Function Declarations:

- The function keyword must be followed by one space.
- Named functions must not have a space between the function name and the following left parenthesis.
- All functions should attempt to return a meaningful value.

Constructors:

- Constructor functions must be given names with an initial uppercase character.
- It must not be called without a "new" operator.

GitHub Guide

Creating or Duplicating repository

Command	Description
`git init`	Initialize a local Git repository
`git clone ssh://git@github.com/[username]/[repository-name].git`	Create a local copy of a remote repository

`git status`	Check status
`git add [file-name.txt]`	Add a file to the staging area
`git add -A`	Add all new and changed files to the staging area
`git commit -m "[commit message]"`	Commit changes

`git rm -r [file-name.txt]`	Remove a file (or folder)

Branching and merging

- Create a new branch of the specific user story.
- Name of the branch should be same as the 'Tag' name of the user story mentioned in the User Stories document
- After making all the required changes commit the changes to the branch

Command	Description
`git branch`	List branches (the asterisk denotes the current branch)
`git branch -a`	List all branches (local and remote)
`git branch [branch name]`	Create a new branch
`git branch -d [branch name]`	Delete a branch
`git checkout -b [branch name]`	Create a new branch and switch to it
`git checkout [branch name]`	Switch to a branch
`git checkout -`	Switch to the branch last checked out
`git checkout -- [file-name.txt]`	Discard changes to a file
`git merge [branch name]`	Merge a branch into the active branch
`git merge [source branch] [target branch]`	Merge a branch into a target branch

Sharing and Updating project

- After the changes are committed, push them to the branch
- Create a pull request, following which another member must review the code
- If the code is acceptable, the branch must be merged with 'dev' branch

Command	Description
`git push origin [branch name]`	Push a branch to your remote repository
`git push -u origin [branch name]`	Push changes to remote repository (and remember the branch)
`git push`	Push changes to remote repository (remembered branch)
`git push origin --delete [branch name]`	Delete a remote branch
`git pull`	Update local repository to the newest commit
`git pull origin [branch name]`	Pull changes from remote repository
`git remote add origin ssh://git@github.com/[username]/[repository-name].git`	Add a remote repository
`git remote set-url origin ssh://git@github.com/[username]/[repository-name].git`	Set a repository's origin branch to SSH

Inspection and Comparison

Command	Description
`git log`	View changes
`git log --summary`	View changes (detailed)
`git diff [source branch] [target branch]`	Preview changes before merging