MACH

Documentation for Developers

To install:

- Clone the github repo: https://github.com/423s23/G2-Mach
    - If you do not
    - have git installed, run the following command in a terminal window:
        - “sudo dnf install git-all”
- Run the following commands in a terminal window:
    - “cd /MachProject”
    - “npm install”
        - If “command not found: npm”

          Do “brew install node” then “npm install -g npm” then try again
        - If “npm install” times out

          Do “npm install @expo/ngrok@^4.1.0”

To Run:

- Next, download the expo go app on a device, and login. If you do not have an account, create one.
- Once you have the app downloaded and an account created, return to the terminal and run the following command:
    - “npx expo login”
    - Log in with your account in the terminal
- Open the app on your device, then run the following command in a terminal window:
    - “npx expo start --tunnel”
        - Make sure that you are still in the MachProject directory
- Once it finishes loading and you see a QR code appear in the terminal window, refresh the app on your device. Under Development Servers, you should see a button appear.
    - Clicking on the button will load up the app on your device.
    - If that doesn’t work, scan the qr code on the screen with either your camera app or a qr code scanner.
- After setting up the app for the first time there will be a button on the expo go app, so you won’t need to scan the qr code again. Simply just run “npx expo start --tunnel” in the MachProject directory and click the MachProject button on the expo go app and it will connect.
-

Layout:

- The Mach App launches from the App.js file.

- The components folder contains all the components we use in the app. Currently the navigation bar and the button components.
- The db folder contains the Mach database.
- The documentation folder contains the developer notes, and end-user information.
- The images folder contains images that are used throughout the app.
- The screens folder contains all the different pages that are used throughout the app.
- The tests folder contains all the tests for the app.

Test Suite:

- We have begun implementing a test suite in git.
- Our current test verifies that the App.js compiles correctly. Currently, the test has a bug in it that we are working on fixing

Automated build and test:

- As above, we have an automated test of our build when we push to git.

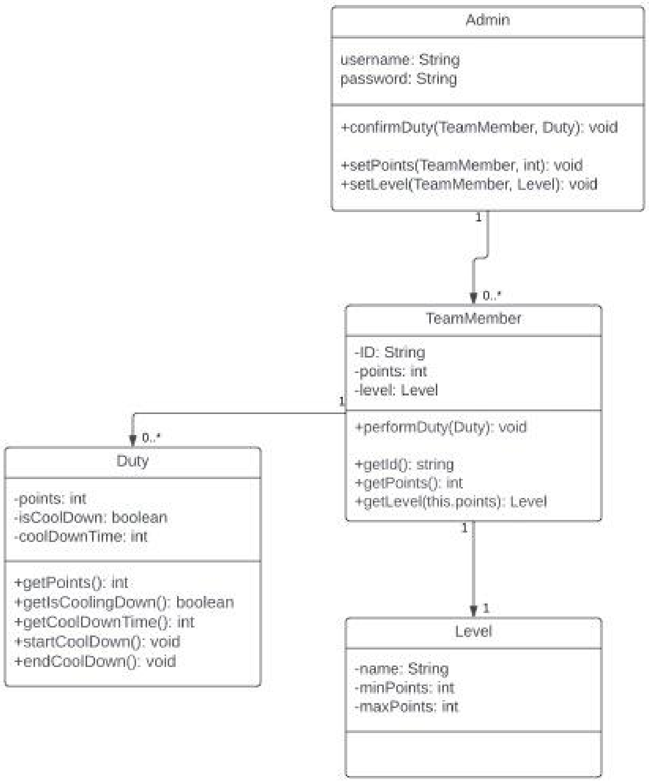
New Release Instructions:

- Not implemented currently, but will be updated for future sprints.

Comments:

- This is how we make a functioning button that can navigate to a different page
```
<Pressable style={styles.press} onPress={() =>
navigationRef.current?.navigate('Settings')}>
   <Text style={styles.buttonText}>Settings</Text>
</Pressable>
```
-
- Class diagram is shown below

Class Diagram


Admin
username: String
password: String
+confirmDuty(TeamMember, Duty): void
+setPoints(TeamMember, int): void
+setLevel(TeamMember, Level): void
1
0..*
TeamMember
-ID: String
-points: int
-level: Level
+performDuty(Duty): void
+getId(): string
+getPoints(): int
+getLevel(this.points): Level
1
0..*
Duty
-points: int
-isCoolDown: boolean
-coolDownTime: int
+getPoints(): int
+getIsCoolingDown(): boolean
+getCoolDownTime(): int
+startCoolDown(): void
+endCoolDown(): void
1
1
Level
-name: String
-minPoints: int
-maxPoints: int