Victor Grajski
PUI Assignment 8

Expo Project Page: https://expo.io/@vgrajski/projects/star-wars-starship-databank
GitHub Repository: https://github.com/victor-grajski/assignment_8

Part 1: Description

My project is a React Native app that serves as a front-end for a database of Star Wars starships (ex: Millennium Falcon). Users can view a list of all starships, filter the results by which movie it appeared in, view details about each starship like who piloted it and how fast it is, and users can add their favorites to a dedicated favorites list. My target audience is a fellow Star Wars enthusiast who would like to learn more about the starships that appeared in the movies and save their favorites to compare with friends. I believe the app is interesting and engaging due to its content and the user's ability to personalize their favorites. I also believe the starfield animation on the homepage adds an engaging element to the application.

The app is responsive to portrait and landscape orientation. I designed the landscape orientation so it looks best on a tablet, and the portrait orientation looks best on a smartphone. The app works for both Android and iOS devices by using the React Native and Expo frameworks. Because I built a React Native app rather than a website, it is not hosted directly on the web. However, one can download it from my Expo project page to load it on their Android or iOS device. One can also test it in a simulated device on the web from my Expo project page, however, there may be a queue to test it since their server only has so many resources available.

Part 2: User Manual

Running the app on a native device

1. Download the Expo app using either the iOS App or Android App link at the bottom of the screen
2. Once the Expo app is downloaded, scan the QR code at the top of the page to download and run the app

Running the app via the Expo web simulator

1. Open the Expo project page
2. Click "Open project in the browser" near the QR code at the top right of the page
3. Click the "Open Project" button. This will open a pop-up with a simulated device
4. Click "Tap to Play". You may have to wait in a queue. The mobile web Expo project page should load afterwards
5. Scroll down and click "Open Project using Expo". The simulated app will then open
6. Tap "Got it" or the "x" button to dismiss the modal. Note: the starfield animation and any interactions may be slow due to the limited computing power of the simulator

Once the app is open

1. Using a portrait orientation, scroll up and down the list of starships (Scroll View)
2. Tap the hamburger icon on the top left of the home page (Navigation Bar)
3. Filter to “EPISODE II: ATTACK OF THE CLONES” (List Filtering)
4. Scroll to the starship Slave I at the top of the page (ScrollView)
5. Tap on the starship Slave I (TouchableOpacity)
6. Scroll up and down the page to see its details (ScrollView)
7. Scroll horizontally on the “STATS” carousel to see the full list of stats (Horizontal Carousel)
8. Tap the heart icon on the top right of the detail page (Navigation Bar)
9. Tap the back arrow on the top left of the detail page (Navigation Bar)
10. Once back on the home page, tap the heart icon on the top right to see Slave I saved in the “FAVORITES” list (Navigation Bar)
11. Tap the hamburger icon on the top left of the home page (Navigation Bar)
12. Tap “ALL FILMS” to see the full list of starships, and you’re done! (List Filtering)
13. Repeat this set of instructions using a tablet in landscape orientation

Part 3: External Tools

- React Native
  - I chose to use React Native because I’ve been wanting to teach myself this framework for some time. I was inspired to learn it because of how easy React Native makes it to deploy native apps on both Android and iOS
  - React Native powers my entire app just as React serves as the backbone for a website. I created components for the results page, filters, detail page, etc.
  - It serves as the foundation of the app
- Expo
  - I chose to use Expo because Expo makes it easy to build and test apps on Android and iOS devices
  - I used it as a command line tool where I can scan a QR code to launch the app on my iOS devices. I could also run a simulated Android version using Android Studio
  - Expo made it much easier to develop and test for both Android and iOS because I could write my code once rather than having to develop on Xcode and Android Studio separately
- React Navigation
  - I chose to use React Navigation because it allowed me to easily set up navigation between pages
  - I used it by wrapping my app in a NavigationContainer component so I could navigate between pages
  - React Navigation made it much easier and faster to implement navigation for each detail page dynamically rather than having to hardcode each page
- Starfield React
  - I chose to use Starfield React to introduce a fun animation that I thought was thematically appropriate to the app
  - I used it by downloading the npm package and adding the component to my home page
  - It adds a fun, aesthetically engaging element to my app that I believe delights the user

- React Native Animated Hamburger
  - I chose to use React Native Animated Hamburger because it simplified the process of adding an animated hamburger icon to my navigation bar
  - I used it by downloading the npm package and including the component in my navigation bar component
  - It adds an aesthetically pleasing way of displaying the status of whether the navigation drawer is open or not

Part 4: Mockup Iteration

I did not iterate on my HW7 mockups, and I did not make any changes to it as I was implementing my website save for the starfield animation and the animated hamburger icon.

Part 5: Challenges

My biggest challenge was testing my app through Android Studio. Because I don't own an Android device, I had to simulate one through Android Studio. However, setting up a simulated device and connecting it to my local Expo server was tedious and the instructions provided were unclear. Another challenge I faced was wrapping my head around React Native because it differs in subtle ways from React. For example, styling is subtly different than CSS, and I often found myself writing style code in a CSS style only to have it not work, forgetting that I had to write it in a different style.