ADR 1:

Summary:

In order to create a mobile game that fulfills our clients vision, without having to learn both Android Studio and Xcode (native development platforms), we opted to use Unity, which is a cross platform game engine.

Problem:

Deploying apps to the App Store is not the same as deploying apps to the Google Play Store. We needed a solution that allows us to write code once, and then translate that code into deployable versions for both stores.

Constraints:

We have a small team, and no one has experience with either store's native framework. Limited amount of time.

Options:

React Native:

Pros:

- Converts your code into two separate projects. (One for apple and one for android)
- Beginner friendly
- Firebase compatibility

Cons:

- Not the best for building games
- Have to build our own game engine from scratch

Unity:

Pros:

- Converts your code into two separate projects. (One for apple and one for android)
- Exists specifically to make games.
- Firebase compatibility
- Built in libraries we can use specifically for games.
- Various game dev quality of life features

Cons:

- Very clunky
- C# programming

Rationale:

Although React Native is an easier framework to work with, our project is a mobile game at heart. Therefore, we decided to use the framework that is targeted specifically for game dev.

ADR 2:

Summary:

In order to populate the game scene with a scenario, we needed to implement a relational database that can store data we can later use to set up the game scene. We also need this database to be easily modifiable and user friendly. We opted to use Firebase for this solution.

Problem:

Generally working with databases is tedious. Most free solutions are not optimized for mobile devices, and they have a poor/non-existent graphical user-interface. The database will be an essential tool in developing our app. We WILL make many changes and we need a central location that group members can go to see the current state of the database.

Constraints:

This database must be accessible from the internet. Also, this database needs to be scalable (client has big aspirations).

Options:

Firebase:

Pros:

- Free to start
- Google offers a suite of other products we can use for free. Products such as authentication, and analytics.
- Very well documented and supported by the community.
- Specifically supports Unity projects.
- Offers scalability

Cons:

- Gets very expensive as the app scales in popularity.

AWS Amplify:

Pros:

- Prices are much cheaper compared to Firebase at scale.
- Similar to firebase in products
- Open source

Cons:

- Does not support Unity projects “officially”
- Learning curve as it is command line driven.

Rationale:

We opted to use Firebase. It is versatile, free, and all of our group members have experience using Firebase. Firebase is well documented with a large community. It is also one of the only services that “officially” supports Unity projects.