Role Roulette

Project Proposal

COMP 4350 – A01

January 28, 2020

Group 5

Justin Band

Paul Brown

Chad Hillary

Ali Jasim

Sue Lee

Michael Rivero

Mike Roberts

Marvin Tran

Project GitHub page: https://github.com/chadxhillary/Role-Roulette	Project tracking system – Trello https://trello.com/invite/b/gzInkoqT/e416e5104d072488bf5a5f3d7b9ec238/4350-board-game

1. Vision Statement

Role Roulette is an application that assigns and distributes identities / roles to a group of players participating in a secret-role based game.

2. Project Motivation

2.1 Why are you building this software?

The idea for our application arose from several problems related to playing secret-role based games. The games themselves weren't the problem. The way cards were distributed and the setup involved for the games had issues. For example, if the cards became creased or went missing, players could use this information to their advantage and gain insights into the identities of other players. In addition, cards could be distributed incorrectly causing the game to be played in a way that was unintended. Furthermore, sometimes a player's cards could be accidentally revealed during the middle of a game causing frustration.

2.2 Why should I use your software?

By creating an application that distributes these roles to players directly, we aim to solve the problems that stem from the need for physical cards. Our role distribution app would also speed up game setup times, by removing the need to shuffle and deal the cards. In addition, there is a lot to remember when playing a game for the first time. For new players, remembering their role or objective could require a couple of trips to the rulebook. We see our application removing the need for this by showing each player their objective on their device.

2.3 How is it (going to be) better than similar products?

There are many applications and websites that are specific to one game. We envision that our project will be the central hub for all of these secret-identity games, allowing people to flip between the games at will.

3. Objective and Success Criteria

Our objective is to build both an Android application and a responsive web app that will overcome the problems mentioned in our project motivation by reducing or removing the need for physical cards. We will consider the application a success using the following criteria:

- The application correctly distributes necessary roles and information for at least 3 different games: Avalon, SpyFall, and Ultimate Werewolf.
- The application reduces the average game setup time by 15%.
- Over 50% of the users that have tried the application would prefer to use the application when playing these games.

This success criteria will be measured by introducing it to a group of users. They will play one or more rounds with assistance from the app, and the results and their feedback will be recorded.

4. Technical Details

4.1 Server

We will be utilising AWS EC2 for our server needs for the reasons mentioned in class. Some of the reasons mentioned in class were:

- It is free of charge
- The usage limits are enough for our needs
- Each member has the ability to create their own instances

4.2 Web Application

The web application portion of the project will employ the use of a Javascript stack known as the **MERN** (**M**ongoDB, **E**xpress, **R**eact and **N**ode.js) stack. The individual components of MERN integrate well with each other and provide us a comprehensive end-to-end framework that would allow for faster development and deployment. The stack and its components are widely used with plenty of easily accessible documentation and tutorials. Moreover, experience with the stack and its components would lend aid to our future employability. Having each component of our stack use the same language is also very beneficial for consistency, maintainability, and minimizing overall ramp-up time. Also, since every framework in the stack is open-source, there are no licensing fees whatsoever, which is important in the scope of this course project.

Mongo DB

For our server-side data storage needs, we will be using MongoDB. MongoDB is a NoSQL cross-platform Document-Oriented database. As the number, type, and relation of variables and entities stored will differ from game to game, we believe the schema-less architecture of MongoDB better suits our data needs compared to a fixed schema architecture of a Relational Database Management System (RDBMS). The availability of using document-based or key-value pairs allows easier customization without the need to frequently alter database schemas.

While few of us have experience with this technology, the use of Javascript in MongoDB and its JSON-like notation would make for an easier learning curve. It is also very simple to set up a MongoDB environment. The team also believes that our queries would be simple enough and that the need for a query intensive environment such as SQL is unnecessary. MongoDB is also available as a fully managed service on AWS which is a plus.

Express

We did not necessarily choose Express, it kind of just fell into our lap when looking at the MERN stack. As it turns out, Express is something that suits our project needs and would be a framework that would be useful to learn. It was built to be a minimalist web framework for Node.js and it turns out that it is super fast too. Express has utilities to create an API, which is needed to implement the NoSQL portion of our app and creating an API will also allow us to scale with our project goals.

Since Express was written for Node.js there are a lot of tutorials outlining how to setup Express and Node.js together. Looking into documentation has shown that Express also contains database integration options for MongoDB. This means that we can perform callback and Promise based interactions from Express to MongoDB, giving us more options for front and back end designs.

React JS

Our front end will be built using React. We will be using React because some group members already have experience using React and others would like to learn it since it is a very sought after and marketable skill. React is different from what many members of the group are used to UI wise, so the learning curve could prove to be troublesome during the initial design. However, the copious amounts of documentation and tutorials combined with an active React community outweigh the learning curve.

With React being component based, we would be able to code one UI component and re-use it throughout the UI, letting us spend more time on our projects underlying architecture and features rather than monotonous UI changes.

Node JS

Node.js is going to make up the server-side execution of our project. One of the things we like about Node.js is that it will not be too hard of a transition for many of our group members. Many of us have previously used JavaScript, so getting started with Node.js will likely not be too difficult. Using this technology also just makes sense in regards to the rest of our stack. React and Node.js both provide ways to perform asynchronous operations. There are also nice integration options for MongoDB and Express.

Documentation is plentiful. There are tutorials everywhere about using Node.js server-side, even tutorials relating to using it on an EC2 instance. One of the reasons it has a lot of documentation is that it is so widely used. It is used everywhere from industry to small projects and everything in between because it is simple and based on JavaScript.

4.3 Android Application

As our web app will be using React, we felt that our Android application could leverage some existing code by using React Native. While the two technologies share similarities, the rendering of UI components in React Native uses native platform UI components. By using React Native, the applications codebase would better adhere to the DRY (Don't Repeat Yourself) principle and also support faster development. The learning curve would be less steep due to the usage of JavaScript and React components in both technologies.

5. User Stories

We will be using Trello as our project tracking system. Our board can be found at;
https://trello.com/invite/b/gzInkoqT/e416e5104d072488bf5a5f3d7b9ec238/4350-board-game

5.1 High Priority (MVP)

- As a Host I want to be able to create a room that other players can join
- As a Host I want to be able to end a game/session that the current players have finished
- As a Host I want to be able to select the game type that will be played from a preloaded list
- As a Host I want to distribute identities/roles to all players according to the game type
- As a Player I want to be able to view my identity during a game
- As a Player I want to be able to join a room

5.2 Medium Priority

- As a Player I want to be able to see information about the game I'm playing (game description, basic rules, description of all identities in play)
- As a Player I want to be able to hide my identity
- As a Player I want to be able to view other players roles where applicable
- As a Player I want to see who had what identity at the end of the game
- As a User I want to be able to create my own game type

5.3 Low Priority

- As a Player I want to select my preferred identity before the start of a game
- As a Player I want to be able to send my identity to another player

6. Data Model Diagram