**SOFTWARE DEVELOPMENT PROPOSAL**

**Admirable-ty**

**PREPARED FOR**

***Back Alley Front Company, Inc.***

**PREPARED BY**

**Miguel Palima**

**Olivia Sirak**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **PROJECT NAME** | Admirable-ty | | |
| **EST. START DATE** | 3 August 2020 | **EST. FINISH DATE** | 6 August 2020 |
| **SUBMITTED TO** | *Back Alley Front Company, Inc.* | **COMPANY** | *Back Alley Front Company, Inc.* |
| **CONTACT NAME** | Robin Banks | **ADDRESS** |  |
| **PHONE** | 999-867-5309 |
| **EMAIL** | robin.banks@backalleyfrontco.com |

|  |  |  |  |
| --- | --- | --- | --- |
| **SUBMITTED BY** | Miguel Palima, Olivia Sirak | **COMPANY** | P.S. Development |
| **CONTACT NAME** | Miguel Palima, Olivia Sirak | **ADDRESS** |  |
| **PHONE** | 888-867-5309 |
| **EMAIL** | devteam@psdevelopment.com |

|  |
| --- |
| **PROJECT OVERVIEW** |
| To create a social-networking app that allows users to rate other users on a 0-5 scale and test their likeability or *admirable-ty*. |
| **PURPOSE / GOALS** |
| To create a system that will enable people to rate and differentiate others based on their likeability rating/designation and to assign weighted average ratings to each user whereby ratings from higher-rated users are given greater weight. |

|  |
| --- |
| **OBSTACLES** |
| -Utilizing previously-unused technologies such as Angular.  -Ensure compatability between technologies.  -Tight deadlin. |
| **INDUSTRY / MARKET RISK FACTORS** |
| -Non-adoption of the app by potential users.  -Banning of the app by the app stores of Google, Apple, etc.  -Cancellation of service from infrastructure providers due to updates in their Terms and Conditions of use.  -Acquisition or duplication by nefarious totalitarian states that seek to impose social order. |
| **BUDGETARY RISK FACTORS** |
| Costs should be scalable with app usage. |
| **HARDWARE COMPATIBILITY** |
| The app should be compatible with any web-enabled device. |
| **SOFTWARE EMPLOYED** |
| Angular (front-end), Spring Tools Suite(back-end), AWS Database for MySQL, and Heroku (deployment) |

|  |  |  |  |
| --- | --- | --- | --- |
| **TIMELINE / MILESTONES** | | | |
| **OVERVIEW** | After submission of the project proposal, complete minimum viable product (MVP) for back end, followed by completion of front end for MVP, then adding additional features one-by-one as time permits before deployment. | | |
| **MILESTONE** | | **REPORTING** | **DEADLINE** |
| Project Proposal | |  | EOD, 3 August |
| Complete backend for Minimum Viable Product | |  | 2pm, 4 August |
| Complete front end for MVP | |  | Noon, 5 August |
| Add additional features like hidden ratings for users unrated by viewer | |  | 3pm, 6 August |
| Deploy | |  | EOD, 6 August |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |
|  | |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DEPLOYMENT / DISTRIBUTION** | | | | |
| Online availability via web browser. Eventual expansion into Google and Apple app stores for mobile devices. | | | | |
| **TESTING** | | | | |
| Manual testing using dummy accounts, followed by beta testing with a select group of real users. | | | | |
| **DOCUMENTATION** | | | | |
| Comments written into the code; UML concept models, user view mockups. | | | | |
| **SUPPORT** | | | | |
| For support during pre-deployment and initial launch, contact the developers. For real-world use, contact the dedicated support team. | | | | |
| **TRAINING** | | | | |
| A 1-page user How-To guide. The app itself should be simple and intuitive to use. | | | | |
|  |  |  |  | |
| **COST STRUCTURE** | | | | |
| **OVERVIEW** | Costs should be variable, scaling with usage. | | | |
| **NEEDS / INVESTMENT** | | | | **COST** |
| Heroku app hosting | | | | Free to $500 per month |
| AWS Aurora Database | | | | $0.10 per GB-month plus $0.20 per 1 million requests |
| AWS Aurora Database accessibility - db.t3.small | | | | $243.00 per year if paid upfront |
|  | | | |  |
|  | | | |  |
|  | | | |  |
|  | | | |  |
| **ESTIMATE TOTAL** | | | |  |
| **PAYMENT TERMS** | | | | |
| Costs incurred should be monthly (or annually if discounts on subscriptions are offered). | | | | |

|  |  |
| --- | --- |
| **TERMS & CONDITIONS** | |
|  | |
| **PROPOSAL MAY BE WITHDRAWN IF NOT ACCEPTED BY DATE OF** |  |
|  | |

|  |  |  |  |
| --- | --- | --- | --- |
| **ACCEPTANCE OF PROPOSAL** | | | |
| **AUTHORIZED CLIENT SIGNATURE** |  | **DATE OF ACCEPTANCE** |  |

App Architecture

Features and Priority

Minimal Viable Product:

- User sign-up – form entry of user’s first name, last name, city, state, country, email, and date of birth

- Home page for non- logged-in users

- Sign-up complete page

- User log-in

- User log-out

- Navbar

- All users page

- User pages with name and city

- Weighted average rating display for each user

- Ability to rate other users

- Ability for a user to delete ratings he/she gives to other users.

Additional Features – First Priority:

* User profile photos
* User search function
* “Admirable-ty” designation based on weighted average rating
* Weighted average rating weighted heavily towards ratings given by higher-rated users
* Ensure user’s email address is unique before sign-up

Additional Features – Second Priority:

* Anonymous follow (“Track”) and unfollow (“Purge”) function
* Following page of users that one is following
* Ability to re-rate users that overrides previous rating

Additional Features – Third Priority:

* Logo
* Average rating given by user
* Last five ratings given to user

Additional Features – Fourth Priority:

* Show ratings of the users who gave the five most recent ratings to user
* Last five ratings given by user

Additional Features – Fifth Priority:

* Show ratings of the users who received the five most recent ratings given by user

Additional Features – Sixth Priority:

* Show dates of the user’s past five ratings given and received
* Show number of followers/trackers on profile page

Additional Features – Seventh Priority:

* Ability to delete one’s own profile
* Ensure users are over 16

Additional Features – Final Priority:

* Secret self-destruct function that destroys all ratings records for all users (Order 66)
* Users kicked off the app upon getting a 0.00 weighted average rating

Possible Future Features:

* Email verification
* GPS linking to phone to ensure that a user can only rate someone they have met within the past 72 hours

Notes on Ratings

* User designations are as follows:
  + 4.5 - 5.0: “Fleet Admirable”
  + 3.5 - 4.49: “Admirable”
  + 2.5 – 3.49: “Vice Admirable”
  + 1.5 – 2.49: “Rear Admirable – Upper Half”
  + 0.5 – 1.49: “Rear Admirable – Lower Half”
  + 0.01 – 0.49: “Swab”
* Weighted average rating is weighted by rater’s rating on an exponential scale:
  + Weight w = 2 ^ user’s rating
  + 0 star: w =1
  + 0.5 star: w = 1.41
  + 1 star: w =2
  + 2 star: w = 4
  + 3 star: w = 8
  + 3.5 star: w = 11.31
  + 4 star: w =16
  + 5 star: w = 32

Java Class List

com.Admirablety

* AdmirabletyApplication.java

com.Admirablety.configuration

* SecurityConfiguration.java
* ThymeleafConfiguration.java
* WebMvcConfiguration.java

com.Admirablety.controller

* AuthorizationController.java
* RatingController.java
* TrackController.java
* UserController.java

com.Admirablety.model

* User.java
  + Long id, String firstName, String secondName, String city, String state, String country, String email,
* Role.java – who’s tracking who
  + Long id, Long tracker\_id (imported from User doing tracking), Long tracked\_id (imported from User being tracked)
* Rating.java – ratings given by one user to another
  + Long id, Long rater\_id (imported from User doing rating), Long subject\_id (imported from User being rated), Float rater\_rating (rater’s own rating at time he/she is rating taken as a snapshot of ), Integer stars\_given

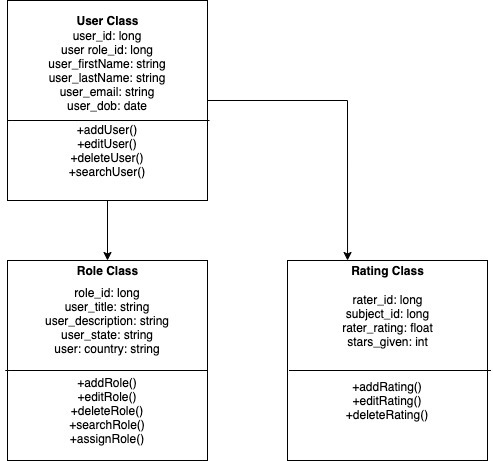
com.Admirablety.repository

* RoleRepository.java
* RatingRepository.java
* UserRepository.java

com.Admirablety.service

* RatingService.java
* UserService.java

UML Diagram



User View Mock-Ups

A screenshot of a cell phone

Description automatically generated

A close up of a map

Description automatically generated

Note: if a rating has already been given to this user, the logged-in user will see the number of stars he/she has given to this user beneath the Track/Purge button. Beneath this will be an input similar to the shown “Give Your Own Rating” that will enable edit of that rating. Also, beneath that will be button to delete that rating altogether.

A screenshot of a cell phone

Description automatically generated

A user will be able to rate someone that he/she has not rated prior on this page (as well as on the similar “Your Tracking” and search results pages). To delete or edit a rating, one must navigate to the that person’s profile.