Betgogo

Technical Document

UNIVERSITY OF BIRMINGHAM

JUNIOR SOLUTION

A project presented by Birmingham Innovation Studio

Date: 1111-11-12

1. Introduction

This document explains the functional specification of betgogo, the Web app project in which Birmingham Innovation Studio, engages to fulfill all the requests demanded by Safequeen as explained during elementary meetings and calls.

1.1 Purpose of this document

The functional specification document serves the same purpose as a contract. The specifications presented in this document are related to the conversations done by Sami. Following this document, the developers agree to provide the capabilities specified and fulfill all requests and demands of Safequeen, who also agreed to find the product satisfactory if it provides the capabilities as specified.

1.2 Scope

betgogo is a web app that allows user to place bet between each other, this app will have a login and sign up page where users can login or register, reset their password and reset their email. It will have a profile page where users can update their profile and manage their account. there will be a friend page where users can mlanage their relationship with others. there will be a bet page where users can see their current bet, create new bets and modify existing bets.

1.3 References

All meetings, emails and materials exchanges with Safequeen.

2 Requirements

2.1 Functional Requirements

1. Authentication Pages:

• The web app must allow users to sign in and sign up using email and password.

• The web app must include a 'Forgot Password' feature that sends a reset link to the user's email.

• The web app should offer social media login options like Facebook and Google.

• Success message must be displayed after successful registration, login, or password reset.

2. Profile Page:

• The web app must have a profile page where users can update their profile information.

• The web app should allow users to upload a profile picture.

• The web app must offer an option for users to delete their account.

• Success message must be displayed after successfully updating the profile or deleting the account.

3. Friends Page:

• The web app must offer a friends page where users can manage their relationships with others.

• The web app should provide options for accepting or rejecting friend requests.

• The web app must allow users to block or unblock other users.

• Success message must be displayed after successfully accepting/rejecting a friend request, or blocking/unblocking a user.

4. Bet Page:

• The web app must have a bet page where users can see their current bets and the status of each.

• The web app should allow users to create new bets with specific terms and conditions.

• The web app must offer options for modifying or canceling existing bets.

• Success message must be displayed after successfully creating, modifying, or canceling a bet.

2.2 Non-Functional Requirements

1. Reliability:

• The web app must have a minimum uptime of 99.99%.

• The web app should have a failover mechanism to switch to a backup service in case of server failure.

• The web app should offer offline access to saved bet data.

2. Performance:

• The web app should load user profiles and bet data within 2 seconds.

• The web app should be optimized for low-latency data fetching for real-time bet updates.

3. Usability:

- The web app must be user-friendly with an intuitive UI/UX design.
- The web app should offer a tutorial and FAQs for first-time users.
- The web app should be accessible, following WCAG 2.1 guidelines for web applications.

4. Security:

- All user data must be encrypted both in transit and at rest using industry-standard encryption algorithms.
- The web app must comply with data protection regulations.
- The web app should have a secure authentication mechanism, including two-factor authentication.

5. Scalability:

- The web app should be designed to handle an increasing number of users and bets.
- The web app should be cloud-native to allow for easy scaling.
- The web app should be modular to allow for the addition of new features without affecting existing functionality.
- The web app's database should be designed for horizontal scaling to accommodate growing data.