

Betgogo23

Technical Document



**UNIVERSITY OF
BIRMINGHAM**
JUNIOR SOLUTION

A project presented by Birmingham Innovation Studio

Date: 12/01/2002

1. Introduction

This document explains the functional specification of toto, the None project in which Birmingham Innovation Studio, engages to fulfill all the requests demanded by None 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 None. Following this document, the developers agree to provide the capabilities specified and fulfill all requests and demands of None, 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 manage 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 None.

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 view and update their profile information.
- The web app should allow users to upload a profile picture.
- The web app must display the user's betting history.
- Success message must be displayed after successfully updating the profile.

3. Friend Page:

- The web app must offer a friend page where users can view and manage their relationships with others.
- The web app should provide options for sending friend requests and accepting/declining requests.

- The web app must allow users to block/unblock other users.
- Success message must be displayed after successfully sending a friend request or blocking a user.

4. Bet Page:

- The web app must have a bet page where users can view their current bets and betting history.
- The web app should offer options for creating new bets and modifying existing bets.
- The web app must allow users to set betting limits.
- Success message must be displayed after successfully creating or modifying 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 betting history and active bets.

2. Performance:

- The web app should load user profiles and bet history within 2 seconds.
- The web app should be optimized for low-latency data fetching for real-time bet updates.

- The web app should be able to handle a minimum of 1000 concurrent bet placements.

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 apps.

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 like GDPR.
- The web app should have a secure authentication mechanism, including two-factor authentication.
- The web app should have a data backup and recovery plan.