Software Requirements Specification

1 Requirements Introduction

The San Jose Earthquakes Football Prediction Discord Bot aims to enhance the user experience for fans of the San Jose Earthquakes by providing a platform for predicting scores of football games. This unique bot allows users to submit their score predictions and earn points based on the accuracy of their forecasts compared to the actual game scores. The following document outlines the functional and performance requirements of this Discord bot.

1.1 Scope

The scope of the San Jose Earthquakes Football Prediction Discord Bot includes features such as score prediction submission, scoring calculation, leaderboard display, and user engagement through a user-friendly interface.

1.2 Referenced Documents

- Discord API Documentation
- Discord JS Documentation
- San Jose Earthquakes Football Game Data API

1.3 Requirements

The requirements section specifies various aspects of the bot, including its functionalities, user interactions, scoring system, security measures, and integration with external APIs.

1.4 Precedence and Criticality of Requirements

Requirements will be prioritized based on their criticality and importance to the core functionality of the bot. Critical features such as score prediction submission and scoring calculation will be prioritized over secondary features.

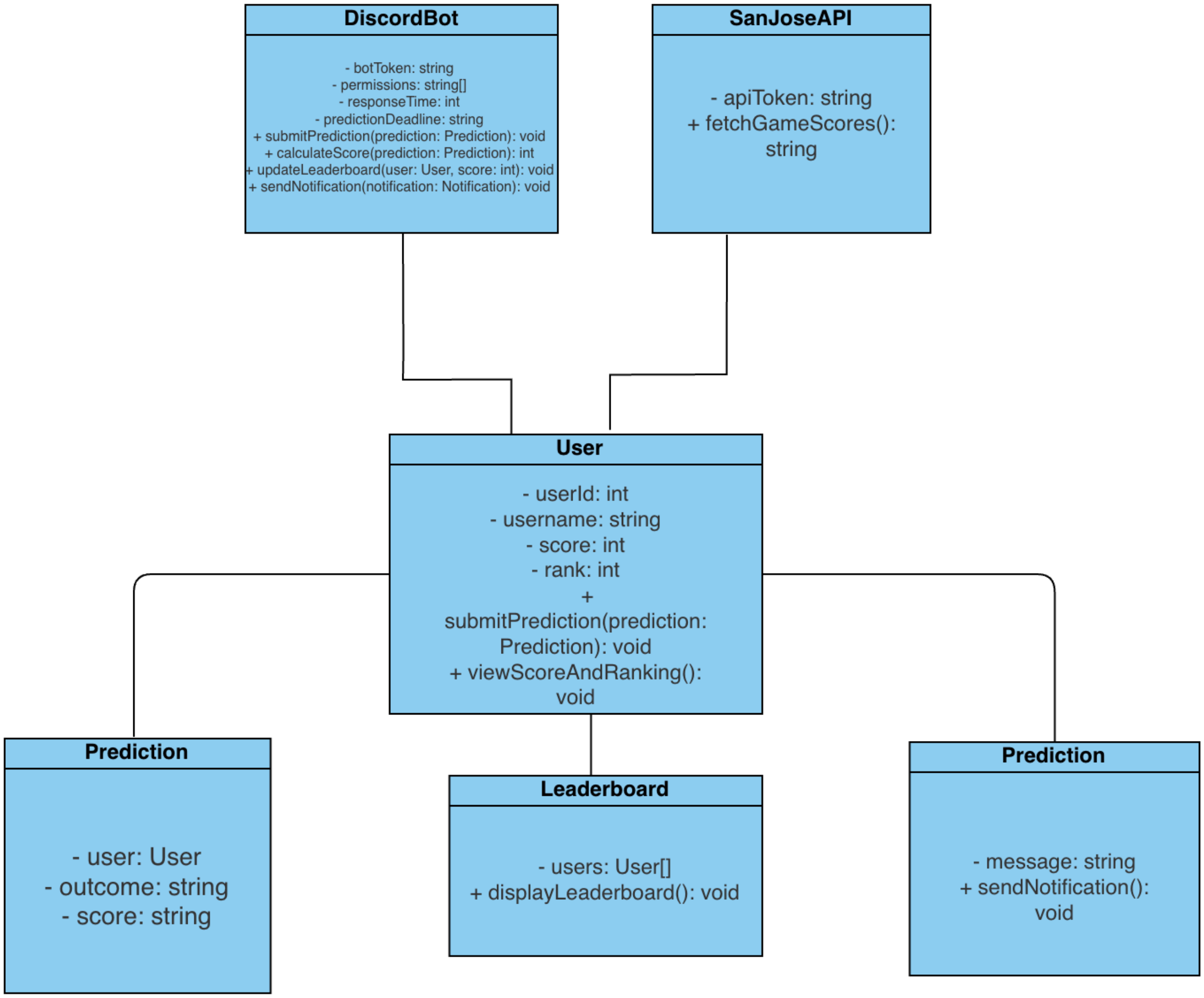
1.5 Evaluation Criteria

The success of the San Jose Earthquakes Football Prediction Discord Bot will be evaluated based on user engagement, accurate scoring calculations, seamless integration with the Discord platform, and ease of use for users of the server.

1.6 Acronyms and Abbreviations

- API: Application Programming Interface
- Discord: Discord Communication Platform

1.7 UML Diagram


DiscordBot
- botToken: string
- permissions: string[]
- responseTime: int
- predictionDeadline: string
+ submitPrediction(prediction: Prediction): void
+ calculateScore(prediction: Prediction): int
+ updateLeaderboard(user: User, score: int): void
+ sendNotification(notification: Notification): void
SanJoseAPI
- apiToken: string
+ fetchGameScores(): string
User
- userId: int
- username: string
- score: int
- rank: int
+ submitPrediction(prediction: Prediction): void
+ viewScoreAndRanking(): void
Prediction
- user: User
- outcome: string
- score: string
Leaderboard
- users: User[]
+ displayLeaderboard(): void
Prediction
- message: string
+ sendNotification(): void

2 Functional Requirements

2.1 Score Prediction Submission

Users should be able to submit their score predictions for upcoming San Jose Earthquakes football games through Discord. The submission process should be user-friendly and intuitive.

2.1.1 Predictions Timeframe

2.1.1.1 Predictions are open to users for a limited amount of time

2.1.1.2 Users cannot submit predictions once the game has started (or close to kickoff).

2.2 Scoring Calculation

The bot shall calculate scores for users based on the accuracy of their predictions compared to the actual game scores and goalscorers.

2.2.1 Points Allocation

2.2.1.1 Users shall earn points for correctly predicting the outcome (win, lose, or draw).
2.2.1.2 Users shall earn additional points for guessing correctly the specific score (1-1 score for example in the match).
2.2.1.3 Users shall earn additional points by predicting the first goalscorer (player on the Earthquakes team to score first in the match).

2.3 Leaderboard Display

The bot shall display a leaderboard that ranks users based on their accumulated points.

2.3.1 Leaderboard Features

2.3.1.1 Users can view the leaderboard at any time using a specific command.
2.3.1.2 Leaderboard shall show the top 5-10 users with the most points

2.4 User Engagement Features

To enhance user engagement, the bot shall include additional features.

2.4.1 Game Notifications

2.4.1.1 Bot shall send notifications about upcoming games
2.4.1.2 Bot shall notify the timeframe of when users can predict
2.4.1.3 Notifications include game start times and prediction submission deadlines.

2.4.2 Users Score Viewing

2.4.2.1 A command allows users to check their current score and ranking.
2.4.2.2 Shall allow viewers to check their individual ranking and score

2.4.3 Squad Details

2.4.3.1 Users can see the starting lineup for the game
2.4.3.2 Users can also see which players are currently injured or healthy

2.4.4 Upcoming Matches

2.4.4.1 Users can see the next matchday/opponent and the type of competition the match is being played in (League Game or Cup Game)

3 Performance Requirements

3.1 Subsystem Divisions

The San Jose Earthquakes Football Prediction Discord Bot is divided into two main subsystems, or sections. The first section, referred to as the Discord Interface Subsystem, is responsible for managing user interactions within the Discord platform. The second section, known as the Prediction Calculation Subsystem, handles the logic and calculations related to score predictions and scoring.

3.2 Discord Interface Subsystem

The following requirements are levied on the Discord Interface Subsystem of the San Jose Earthquakes Football Prediction Discord Bot.

3.2.1 The subsystem shall operate within the Discord platform environment.
3.2.2 The subsystem shall utilize the Discord API for interaction with users.
3.2.3 The subsystem shall consist of multiple commands for user interaction.
3.2.4 One command shall allow users to submit score predictions for upcoming San Jose Earthquakes football games.
    3.2.4.1 This command will be known as the Prediction Command.
3.2.5 The Prediction Command shall provide feedback to users confirming the successful submission of their predictions.
3.2.6 Another command shall allow users to view the leaderboard displaying the top users with the most points.
    3.2.6.1 This command will be known as the Leaderboard Command.
3.2.7 The Leaderboard Command shall display the leaderboard within the Discord interface.

3.3 Prediction Calculation Subsystem

3.3.1 The subsystem shall consist of components responsible for calculating scores based on user predictions and actual game outcomes.

3.3.2 Score calculation shall take into account various factors, including the correctness of outcome predictions and the accuracy of score predictions.
3.3.3 Points shall be allocated to users based on the accuracy of their predictions compared to the actual game results.
3.3.4 Additional points shall be awarded for correctly predicting specific score outcomes and the first goalscorer.
3.3.5 The subsystem shall update user scores and rankings based on the latest game results.

3.4 Response Time

The bot shall respond promptly to user commands and notifications.

3.4.1 Response Time Constraint

3.4.1.1 Bot shall respond within 2 seconds of receiving a command

3.5 Scalability

The bot shall handle a large number of users during peak prediction submission times.

3.5.1 Concurrent Predictions

3.5.1.1 Bot shall handle concurrent predictions for at least 200 users

3.6 Scalability

3.6.1 Runtime

3.6.1.1 The bot shall be running at all times

4 Environment Requirements

4.1 Development Environment Requirements

The development environment must meet specific criteria for successful bot development.

4.1.1 Discord Bot Token

4.1.1.1 A valid Discord bot token is required for integration with the Discord platform.
4.1.1.2 The Discord bot token should be stored in a safe place and not be available for the public

4.1.2 API Integration

4.1.2.1 Integration with the Football API is required for fetching real-time game scores.

4.1.3 Database

4.1.3.1 Need to store the predictions and information in a database to compare results to tally up and distribute points to the users.
4.1.3.2 Sequelize shall be used to specify the format/define the models of the data represented in javascript/typescript and connect it to a relationship database such as PostgreSQL

4.2 Execution Environment Requirements

The bot must run in a stable and secure environment for public use.

4.2.1 Hosting Platform

4.2.1.1 The bot shall be hosted on a reliable server with continuous uptime.

4.2.2 Discord Server Permissions

4.2.2.1 The bot requires specific permissions on the Discord server, including message sending, user information retrieval, and reaction capabilities.

4.2.3 Discord User Permissions

4.2.3.1 The bot shall outline clearly what information it shall take in and store which is the username and predictions for the games throughout the season
4.2.3.2 No personal information shall be taken
4.2.3.3 Permission shall be obtained for the bot to be allowed in the server

5 Future Enhancements

5.1 A.I Component

The potential addition of an A.I component is being discussed and something that, assuming we get the main features of the bot working, could add further features to the project.

5.1.1 Predictive Analysis

5.1.1.1 The bot shall try and predict the upcoming score of the match using historical records such as head-to-head data

5.1.2 A.I Command Feature

5.1.2.1 Users can type in a command to have the same predication as the bot, essentially agreeing with the bot's predication

5.1.3 A.I Participation

5.1.3.1 Since the bot shall be able to predict matchups, it shall also participate in the leaderboard and compete against other users