RISK Bot Vs. Bot Website

Software Detailed Design

Individual Contributions

Dale Morris

Dale Morris wrote the Architecture Design section.

Xavier Beatrice

Formatting

Xavier Beatrice formatted and designed the document so it looked appealing.

Data Design

Xavier Beatrice created the data design section of the document and elaborated on how the internal data will be organized.

Procedural Design

Xavier Beatrice created the procedural design section of the document and illustrated how each section will be constructed and what it will do.

Louis Wiley

Interface Design

Louis Wiley created the interface design section of the document to detail how users will interact with the front end of our project.

Data Design

Structures

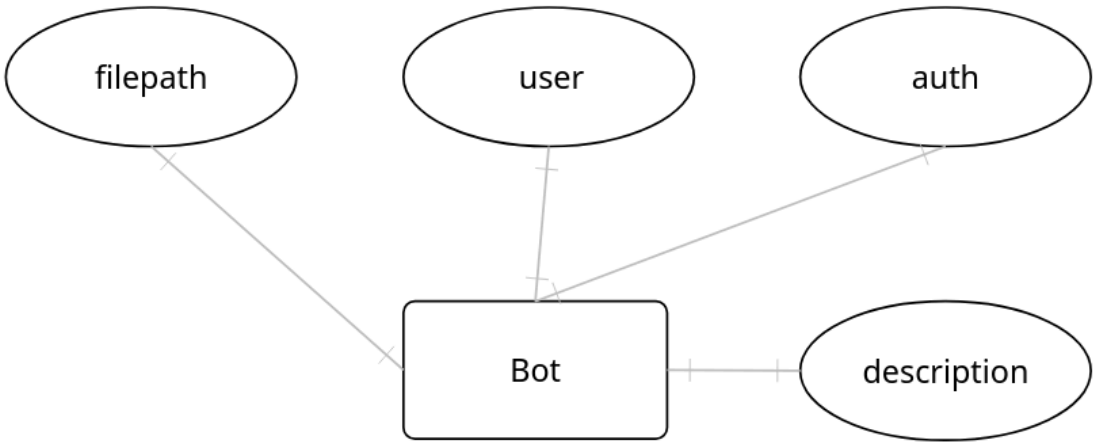
Database

The database will have one main use: acquiring or accessing bots the players created and uploaded to the service. It will have a couple of fields, any necessary Auth fields, a Username, and a path to the bot file. Thus, the main object within the database will be the bot representing the user-created AIs. The description field will be a user-given field about the bot for other users to use to give information about the bot. It'll be a longer text field. Auth will be used for Google authentication. The user will have a short-text username field. Filepath will be an internally generated string containing the username and bot upload date.

Relationships

The relationships within the Bot database will be fairly simple and are just one-to-one relationships for each part of the object. The bot will contain one of each of the attributes.

Diagram



Architecture Design

Input and Output

Input and output information is sent and received via the interfaces described in the Interface Design section of this document. In short, there are interfaces between the user and the client, the client and the server, and the server and the database. On the overall scale of the program, information flows from the user and the database into the server and flows out from the server back to the user and the database.

Processing

Most of the processing for the program is handled by the server. When a user logs into the program, their info is transformed into a query that verifies that a user with the given email and password exists and provides their bot data to the client. When a user creates a new account, their info is transformed into a query that checks if a user with the same username or email address already exists. If a duplicate user doesn't already exist, the info is then transformed into an insertion statement to a database to create a new user.

Similarly, users can provide info to the server about bots which will then be transformed into a selection or an insertion depending on the intended action from the user. Uploaded bots can then be combined with the game logic stored on the server to create a viewable game that can be run through by the client.

Interface Design

Landing Page

The landing page is where users will first be directed when visiting our page. From here they can have the following functions including.

- Sign in to Account
- Create Account
- View Account
- View a Battle
- Submit Your Bot

If the user signs in they are presented a different page where they are not asked to sign up or log in instead the user can only view an account, view a battle, or submit their bot

Account Creation Page

The account creation page will use OAuth so the account creation page will ask if they want to sign up with Google or other services. This page will gather the following information from the user.

- First Name
- Last Name
- Email
- Password in an encrypted form if the user doesn't have a valid OAuth service.

Login Page

This page should ask the user to sign in with either an OAuth or with the email and password used to create the page. The page should have a forgot password function for users who have a created page.

Battle View Page

This page will be where users select the bots they want to play against Bots or watch Bots compete with each other.

Bot Submission Page

This page will have a file submission function and allow the user to name the bot that they are submitting. The page will display information about creating and submitting bots as well as the information about the size limit of files.

Account View Page

This page will be used to view account information that is gathered and any Bots the user has submitted with information about the wins and loses.

Procedural Design

Components

Website

This component's purpose is to act as the user's main interface. Its inputs will include text input or selector fields for logging in, uploading files, selecting an opponent for a match, and defining the name of a bot. The outputs will be text responses for user feedback and the game viewer output, which displays the game to the user. It will perform this transformation from input to output by asking the server for the necessary game data and displaying it. The only real main constraints to the design are that it must use Google auth, it needs to be very lightweight, and it needs to be containerized.

Database

This component stores any basic user information we keep. Its inputs are the SQL queries supplied by the server, and similarly, it outputs back to the server with the requested information queries. The database software obscures the actual process of converting input to output. There are no real design constraints for the database besides using minimal resources and processing power, as the functional differences won't make much difference. It should be containerized.

Server

This component will handle most of the backend work, run the game, and handle database requests. It takes input via the appropriate Rest API and from the database. It outputs queries to the database and the appropriate game or response to the website. It converts the input to output using the database or manually simulating the game as necessary. The server should use only one core and less than a Gigabyte of RAM with minimal disk usage. The server should be containerized.

Key Personnel

- Xavier Beatrice
  - Team-lead
  - Fill-in Developer
- Louis Wiley
  - Front-end Developer
- Dale Morris
  - Back-end Developer