Technologies:

Frontend: React.js, MaterialUI
Middle-tier: Spring Boot, Firebase
Database: PostgreSQL
Other tools: Github (Version control)**,** Figma (Wireframing + Prototyping), Trello (Project management), Diagrams.net (ER Diagrams)

Description of the technologies used:

Our application will be developed using ReactJS and MaterialUI for the frontend, with Spring Boot handling the REST API endpoints. We will use Firebase for user authentication and PostgreSQL for the database. The Spring Boot application will function as a black box, acting as an intermediary between the frontend and backend and enabling efficient communication and data storage. This technology stack offers a robust and scalable solution for building an application with a modern user interface.

API specifications:

Format: HTTP Method | Endpoint | Description

GET api/user - return all users

GET api/user/:id - return a user

GET api/user/search?query=<pattern> - return users with the matching pattern of a string starting with the pattern
For ex: Ame% - where the query string should start with the letters Ame followed by zero, one, or multiple characters.
The query string is case-insensitive. AMe, AME, etc represents the string having the first character as UpperCase and the remaining characters as lower case.
The query string is matched for both first and last names.

GET api/user/signin - login users

POST api/user/signup - create accounts for new users

POST api/creategame - create a game for a specific field

GET api/fields - return list of fields

GET api/fields/:id - return a specific field details

GET api/address/:id - return a specific address

GET api/usermetric/:id - return user metric for a single user

POST api/usermetric/:id - create/update user metric

GET api/chats - returns list of all users for whom the chat thread has been created

GET api/chat/:id - retrieve chats with single/multiple users

GET api/notification?userId=<user-id> - return notifications for a specific user