README

This README serves as an *introduction* to the project’s code and should not be treated as an all-inclusive guide. The code itself is extensively commented on the backend and the frontend is intuitive enough, though if anything is unclear, I am available to help explain things :)

Contents

[Backend 2](#_Toc98260295)

[File structure 2](#_Toc98260296)

[Users and Items 2](#_Toc98260297)

[Games 2](#_Toc98260298)

[Deploying 2](#_Toc98260299)

[Frontend 3](#_Toc98260300)

[Running in an development environment 3](#_Toc98260301)

[building 3](#_Toc98260302)

[Todo before deploying 3](#_Toc98260303)

## Backend

The backend is written with Express, MongoDB and socket.io, and is organised as below:

### File structure

#### Middleware

Presently concerns just auth.js, verifying users are authenticated and passing information onto routes.

#### Models

Contains DB schemas for users and items, the exact details of which are clarified below.

Routes  
Contains all the API endpoints, including the games.

### Users and Items

Querying item prices from the Roblox API involves many API requests, and as such is costly on both the frontend and backend. Instead, all a user’s item IDs can be sought with a single API call. Where the corresponding price is not cached in MongoDB, a further request is made *per item* to “resolve” the data needed, which is then stored for future use. This also has the added benefit of reducing delay upon API requests.

It should be noted that as of yet, there is no real reason for user’s details to be stored in MongoDB as well. In the future however, this may be required for managing site currency, or for providing statistics. The .ROBLOXSECURITY of each user is stored – this is sensitive.

### Games

Multiplayer games use web-sockets to communicate with the frontend and rely on accurate timing to decide when a new round should start. The infrastructure is in-place for bets to be made and processed.

### Deploying

All the backend is containerised with Docker, so simply run “docker-compose up –d”. Make sure to build the production version of the frontend to be served by nginx beforehand.

#### Viewing the database

Graphical user interface, text, application, email

Description automatically generatedIt is best to use MongoDB to view the database.

## Frontend

The frontend is written in Vue.js 2 and uses Bootstrap. All styling is done with Sass.

### Running in an development environment

Run “npm install” if need be, then “npm run serve”.

### building

Run “npm install” if need be, then “npm run build”.

## Todo before deploying

* Add and process betting
* Make random numbers provably fair
* Remove .ROBLOXKEY from DB if it’s not needed
* Change credentials in docker-compose.yml on production machine