

ECE 421 Project 3 Design

Mackenzie Malainey Lora Ma
mmalaine@ualberta.ca lora@ualberta.ca

Benjamin Kong
bkong@ualberta.ca

April 2022

Contents

1	Backend API	1
1.1	REST Endpoints	1
1.1.1	User Authentication	1
1.1.2	Match Records	3
1.1.3	Notes for Future Development	4

1 Backend API

1.1 REST Endpoints

1.1.1 User Authentication

POST /api/v0/user/login/

Description

Logs in a user.

Request Body Format

Form Data

Request Body Data

user_id = USER_PROVIDED_USERNAME

password = USER_PROVIDED_RAW_PASSWORD

Response Cookies

ADD user_auth_token

Response Status

200 - If successful

401 - If user_id and password do match an existing user

404 - If request body is malformed

Known Issues

Attempting to log in to another account when already logged will automatically log out the other user from the server's perspective. However this might pose a security concern (especially if user specific data gets cached in the future)

Currently accepts and processes requests without any method of form encryption (dangerously insecure)

POST /api/v0/user/logout/

Description

Logs out the currently logged in user.

Request Cookies

CONTAINS user_auth_token

Response Cookies

CLEAR user_auth_token

Response Status

200 - If successful

404 - If header doesn't contain cookie

Known Issues

Need to verify this removes the cookie in browser

POST /api/v0/user/register/

Description

Registers a new user.

Request Body Format

Form Data

Request Body Data

user_id = USER_PROVIDED_USERNAME

password = USER_PROVIDED_RAW_PASSWORD

Response Cookies

ADD user_auth_token

Response Status

200 - If user_id and password match an existing user

401 - If user_id and password do match an existing user

404 - If request body is malformed

Response Cookies

ADD user_auth_token

Response Status

200 - If user_id and password match an existing user

404 - If header doesn't contain cookie

Known Issues

Attempting to log in to another account when already logged will automatically log out the other user from the server's perspective. However this might pose a security concern (especially if user specific data gets cached in the future)

Currently accepts and processes requests without any method of form encryption (dangerously insecure)

Assumes client enforces proper password requirements

1.1.2 Match Records

POST /api/v0/user/records/add

Description

Registers match data for the current user

Request Body Format

JSON

Request Body Data

```
"start_time": START_TIME_IN_SECONDS_FROM_EPOCH_UTC_TIME,  
"game_id": {"Connect4", "OttoToot"},  
"cpu_level": {"Easy", "Medium", "Hard"},  
"duration": DURATION_IN_SECONDS,  
"result": {"Win", "Loss", "Tie"}
```

Request Cookies

CONTAINS user_auth_token

Response Status

200 - If successful

401 - If user_auth_token does match an existing user

404 - If request body is malformed

Known Issues

Does not verify "start_time". Probably best to remove this field and use server time to log time upon recording.

GET /api/v0/user/records

Description

Retrieves all the match data from the current user sorted by most recent matches first

Request Cookies

CONTAINS user_auth_token

Optional Request Query Parameters

limit (default = 10)

Number of records to return at once

offset (default = 0)

Number of records to skip (for pagination)

Response Status

200 - If successful

401 - If user_auth_token does match an existing user

404 - If header doesn't contain cookie

Known Issues

Does not support any form of filtering or sorting other than listed

Does not log out user if user_auth_token does match an existing user

Does not apply a maximum or minimum on limit values

Should include a count of how many records there are in total

GET /api/v0/games/records

Description

Retrieves all the match data from the current user sorted by most recent matches first

Request Cookies

CONTAINS `user_auth_token`

Optional Request Query Parameters

`limit` (default = 10)

Number of records to return at once

`offset` (default = 0)

Number of records to skip (for pagination)

`before`

Returns matches that happened before (UTC timestamp in seconds)

`after`

Returns matches that happened after (UTC timestamp in seconds)

`sort_by` (default = `starttime`)

Value to sort by, can be either `duration` or `starttime`

`asc`

Sort direction, defaults to false unless using `sort_by=duration`

`filter`

Only returns elements that match the filter specification (see examples for more info)

Response Status

200 - If successful

Known Issues

Does not have max and min values for `limit`

Should include a count of how many records there are in total

1.1.3 Notes for Future Development

Admin User APIs should be implemented so that a web-created Admin Console may be implemented. This was not implemented right now due to time constraints.

Check APIs for security vulnerabilities (due to time constraints only minimal security could be implemented)