An Academic Reference Manager

By:

- Daníel Ekaphan Valberg (danielv16@ru.is)
- Grétar Örn Hjartarson (gretarh17@ru.is)
- Jóel Snær Garcia (joelg18@ru.is)

How to Install and Run

Installation

Make sure you have NodeJS installed. This was written with version 12.9.0 but newer versions and older (tested on 10.15.3) works as well. Installing NodeJS includes npm, so you do not have to worry about that.

Whe NodeJS is installed, simply type in terminal/command line in root of the project to install all dependencies: '*$*' *indicates being run in terminal/cmd*

```
$ npm install
```

Next, if you would like to run tests, you must also install Jest into global dependency in npm. You can do that by simply entering

```
$ npm install -g jest
```

> **Note:** There are no setup required for Database, see design documentation for more information

Running

after installing packets you have the following chocies to run: '*$*' *indicates being run in terminal/cmd*

- `$ npm test` to run all tests, to run a specific file simply enter file name after e.g. `$ npm test src/db/statics/UserStatics.test.js`, tests are run on test database collection
- `$ npm run test:coverage` to run all tests and get coverage information, results are outputted into console but a more detailed version is outputted to `/coverage`, to see on browser open `/coverage/lcov-report/index.html/`, tests are run on test database collection
- `$ npm start` to start in development environment, it is connected to a development database collection
- `$ npm run production` to start in production environment, it is connected to a production database collection
- `$ npm run initDevData` to initalize data into development database collection
- `$ npm run initProductionData` to initialize data into production database collection

This will run the server on and listen on port `5000`.

To run HTTP methods either run in the browser for GET methods or use either Postman to execute the requests or cURL.

The routes are accessible at `localhost:5000` followed by any following routes.

Routes

Where '{publication_id}' and '{user_id} appears, replace to desired ID!

To enter authentication, you will have to enter the desired user type into the header in the "Authorization" field, user types are: "*auth*" and "*auth*", emit this for "*anonymous*" user type

Admin users has access to all the following routes

URI	HTTP Method	Permitted User Types	Service	Description
/publications	**GET**	All	Publications	Get information about all publications
/publications	**POST**	Admin	Publications	Add a publication
/publications/{publication_id}	**GET**	All	Publications	Get information about a specific publication (including borrowing history)
/publications/{publication_id}	**DELETE**	Admin	Publications	Delete a publication
/publications/{publication_id}	**PUT**	Admin	Publications	Update a publication
/users	**GET**	Admin	Users	Get information about all users
/users	**POST**	Admin	Users	Add a user
/users/{user_id}	**GET**	Admin	Users	Get information about a given user (e.g. borrowing history)
/users/{user_id}	**DELETE**	Admin	Users	Remove a user
/users/{user_id}	**PUT**	Admin	Users	Update a user
/users/{user_id}/publications	**GET**	Authenticated, Admin	Publications	Get information about the Publications a given user has on loan
/users/{user_id}/publications/{publication_id}	**POST**	Authenticated, Admin	Publications	Register a publication on loan

URI	HTTP Method	Permitted User Types	Service	Description
/users/{user_id}/publications/{publication_id}	**DELETE**	Authenticated, Admin	AutehPublications	Return a publication
/users/{user_id}/publications/{publication_id}	**PUT**	Admin	Publications	Update borrowing information
/users/{user_id}/reviews	**GET**	Authenticated, Admin	Reviews	Get reviews by a given user
/users/{user_id}/reviews/{publication_id}	**GET**	Authenticated, Admin	Reviews	Get user reviews for a given publication
/users/{user_id}/reviews/{publication_id}	**POST**	Authenticated, Admin	Reviews	Add a user review for a publication
/users/{user_id}/reviews/{publication_id}	**DELETE**	Authenticated, Admin	Reviews	Remove review
/users/{user_id}/reviews/{publication_id}	**PUT**	Authenticated, Admin	Reviews	Update publication review
/users/{user_id}/recommendation	**GET**	Authenticated, Admin	Recommendation	Get a recommendation for a given user
/publications/reviews	**GET**	All	Reviews	Get reviews for all Publications
/publications/{publication_id}/reviews	**GET**	All	Reviews	Get all reviews for a given publication
/publications/{publication_id}/reviews/{user_id}	**GET**	All	Reviews	Get a user's review for a publication
/publications/{publication_id}/reviews/{user_id}	**PUT**	Admin	Reviews	Update a user's review
/publications/{publication_id}/reviews/{user_id}	**DELETE**	Admin	Reviews	Remove a user's review

Queries

Queries can be done on the `/users` and `/publications`, the queries are `LoanDate=YYYY-MM-DD` and `LoanDuration=N`, where `YYYY-MM-DD` is Year, Month and Date respectively and `N` is a number.
They work as following:

Route + Query	HTTP Method	Permitted User Types	Result contains
/users?LoanDate=YYYY-MM-DD	**GET**	Admin	Users information + list of publicaitons on loan
/users?LoanDuration=N	**GET**	Admin	Users information + list of publications on loan
/users?LoanDate=YYYY-MM-DD&LoanDuration=N	**GET**	Admin	Users information + list of publicaitons on loan
/publications?LoanDate=YYYY-MM-DD	**GET**	Authenticated + Admin	**Authenticated** users get only publications list, **Admins** get publications + users who loan them
/publications?LoanDuration=N	**GET**	Admin	Publications list + users who loaned them
/publications?LoanDate=YYYY-MM-DD	**GET**	Admin	Publications list + users who loaned them

Hasing used:

hasing is saved into shasum.txt in the root folder.

```
find . -iname "*.js" | shasum -a 256 > shasum.txt
find . -iname "*.json" | shasum -a 256 >> shasum.txt
```