A RESTful API generated by [generator-rest](https://github.com/diegohaz/generator-rest).

See the API's [documentation](https://github.com/diegohaz/rest/blob/master/generators/app/templates/DOCS.md).

**Commands**

After you generate your project, these commands are available in package.json.

npm test # test using Jest

npm run coverage # test and open the coverage report in the browser

npm run lint # lint using ESLint

npm run dev # run the API in development mode

npm run prod # run the API in production mode

npm run docs # generate API docs

**Playing locally**

First, you will need to install and run [MongoDB](https://www.mongodb.com/) in another terminal instance.

$ mongod

Then, run the server in development mode.

$ npm run dev

Express server listening on http://0.0.0.0:9000, in development mode

If you choose to generate the authentication API, you can start to play with it.

Note that creating and authenticating users needs a master key (which is defined in the .env file)

Create a user (sign up):

curl -X POST http://0.0.0.0:9000/users -i -d "email=test@example.com&password=123456&access\_token=MASTER\_KEY\_HERE"

It will return something like:

HTTP/1.1 201 Created

...

{

"id": "57d8160eabfa186c7887a8d3",

"name": "test",

"picture":"https://gravatar.com/avatar/55502f40dc8b7c769880b10874abc9d0?d=identicon",

"email": "test@example.com",

"createdAt": "2016-09-13T15:06:54.633Z"

}

Authenticate the user (sign in):

curl -X POST http://0.0.0.0:9000/auth -i -u test@example.com:123456 -d "access\_token=MASTER\_KEY\_HERE"

It will return something like:

HTTP/1.1 201 Created

...

{

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9",

"user": {

"id": "57d8160eabfa186c7887a8d3",

"name": "test",

"picture": "https://gravatar.com/avatar/55502f40dc8b7c769880b10874abc9d0?d=identicon",

"email": "test@example.com",

"createdAt":"2016-09-13T15:06:54.633Z"

}

}

Now you can use the eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9 token (it's usually greater than this) to call user protected APIs. For example, you can create a new article API using yo rest:api and make the POST /articles endpoint only accessible to authenticated users. Then, to create a new article you must pass the access\_token parameter.

curl -X POST http://0.0.0.0:9000/articles -i -d "title=Awesome Article&content=Yeah Baby&access\_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9"

It will return something like:

HTTP/1.1 201 Created

...

{

"id": "57d819bfabfa186c7887a8d6",

"title": "Awesome Article",

"content": "Yeah Baby",

"createdAt": "2016-09-13T15:22:39.846Z",

"updatedAt":"2016-09-13T15:22:39.846Z"

}

Some endpoints are only accessible by admin users. To create an admin user, just pass the role=admin along to other data when calling POST /users.

**Deploy**

Here is an example on how to deploy to [Heroku](https://heroku.com/) using [Heroku CLI](https://devcenter.heroku.com/articles/heroku-command-line):

# start a new local git repository

git init

# create a new heroku app

heroku apps:create my-new-app

# add heroku remote reference to the local repository

heroku git:remote --app my-new-app

# add the MongoLab addon to the heroku app

heroku addons:create mongolab

# set the environment variables to the heroku app (see the .env file in root directory)

heroku config:set MASTER\_KEY=masterKey JWT\_SECRET=jwtSecret

# commit and push the files

git add -A

git commit -m "Initial commit"

git push heroku master

# open the deployed app in the browser

heroku open

The second time you deploy, you just need to:

git add -A

git commit -m "Update code"

git push heroku master

**Directory structure**

**Overview**

You can customize the src and api directories.

src/

├─ api/

│ ├─ user/

│ │ ├─ controller.js

│ │ ├─ index.js

│ │ ├─ index.test.js

│ │ ├─ model.js

│ │ └─ model.test.js

│ └─ index.js

├─ services/

│ ├─ express/

│ ├─ facebook/

│ ├─ mongoose/

│ ├─ passport/

│ ├─ sendgrid/

│ └─ your-service/

├─ app.js

├─ config.js

└─ index.js

**src/api/**

Here is where the API endpoints are defined. Each API has its own folder.

**src/api/some-endpoint/model.js**

It defines the Mongoose schema and model for the API endpoint. Any changes to the data model should be done here.

**src/api/some-endpoint/controller.js**

This is the API controller file. It defines the main router middlewares which use the API model.

**src/api/some-endpoint/index.js**

This is the entry file of the API. It defines the routes using, along other middlewares (like session, validation etc.), the middlewares defined in the some-endpoint.controller.js file.

**services/**

Here you can put helpers, libraries and other types of modules which you want to use in your APIs.