
test Documentation

Release 0.1.6

arunnattarayan

Aug 20, 2021

Contents

1 Requirements	3
2 Report	5
3 Contents	7
3.1 Stepup ENV	7
3.2 Usage	7
3.3 Project Structure	8
3.4 API	10

Boilerplate and tooling for authoring data API backends with Node.js, JWT and MongoDB. It is best suited for developing a es6 API endpoint as a standalone (micro)service (demo), backing up web front-ends and/or mobile apps. This Generator will help to us create a express api application skeleton along with these keys features

- components Style coding
- Eslint Standard style guide
- JWT authentication
- Test cases with Mocha
- Docker
- Swagger API doc

CHAPTER 1

Requirements

- Node 10.15.1+
- MongoDB 3.4+
- yeoman-generator 3.2.0+

CHAPTER 2

Report

- Report any issues or feature enhancements in our tracker

CHAPTER 3

Contents

3.1 Stepup ENV

Steps to create a node application skeleton:

```
$ npm install -g yo  
$ npm install -g generator-node-api-boilerplate  
$ yo node-api-boilerplate
```

Now the app was created and start running in your system

3.2 Usage

start the express server:

```
$ npm start
```

Test The unit testcases:

```
$ npm test
```

clean build folder and compile the source code:

```
$ npm run start:dev
```

find Lint issues in source code:

```
$ npm run lint
```

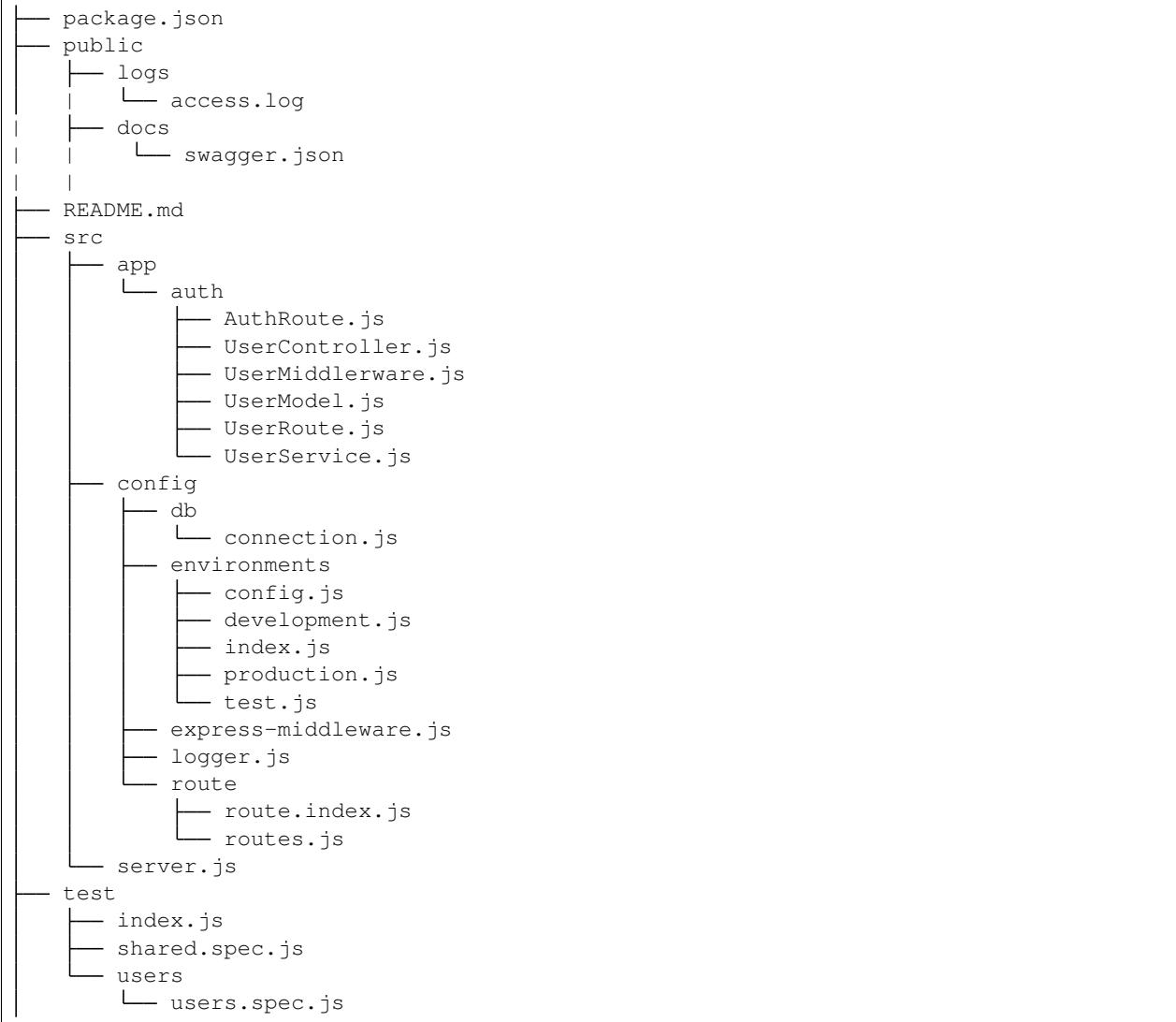
find Lint issues in test cases:

```
$ npm run lint:test
```

Note: npm run lint:test:fix and npm run lint:fix used to fix the lint errors

3.3 Project Structure

Your project will look like this:



3.3.1 src

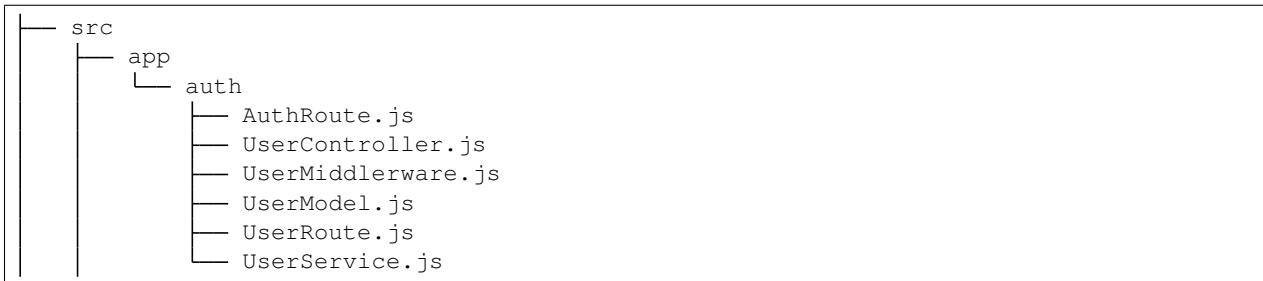
src is the source code directory:



- app contains all needed components
- config has all the configurations like DB, ENV
- server.js is the app bootstrap file

3.3.2 Structure your solution by components

Instead of MVC pattern we recommended components based pattern:



- auth is the component name and that contains route, Controller, middleware, model, and the service files.

3.3.3 config

- Config folder structure:



- DB holds the information of MongoDB connection
- environments use to define ENV variables. We can set different values for different ENV.
- express-middleware defines express middleware need to run this app
- logger.js logger details are defined here
- route creating interface between our components (inside src/app) and application.



(continues on next page)

(continued from previous page)



config.js

It holds common ENV variables across all environments. `development`, `production`, and the “`test`“ are extends this file.

route



Need to link our components in `routes.js` like this

```
import UserRoute from '../../app/auth/UserRoute';

const Routes = [
{
  url: 'users',
  route: UserRoute,
  gaurd: false
};

export default Routes;
```

- url group name of api endpoint
- route component route object
- gaurd (optional) if you want to skip JWT verification set false. By default it sets true

3.4 API

Integrated ‘[swagger API Doc <https://www.npmjs.com/package/swagger-ui-express>](https://www.npmjs.com/package/swagger-ui-express)’. By default it loads in root path.

You can change the API doc path in code (`/src/config/express-middleware.js`).

```
[swagger] () {
  this.exApp.use('/', swaggerUi.serve, swaggerUi.setup(swaggerDocument, {}));
}
```