This is a guide for beginners of a MERN full stack app. When I started the full stack development,

I had trouble finding a step by step tutorial. I try to introduce the knowledge of full stack framework,

development and deployment environment in this article. I hope it will help my fellow software engineers, who is new for the full stack development.

* Mongo DB: A document-based open source database, that provides you scalability and flexibility.
* Express JS: A structured base designed to develop web applications and APIs.
* React JS: A Javascript Front-end library for building user interfaces. Maintained by Facebook.
* Node JS: A javascript runtime built on Chrome’s V8 JS engine.

I am using the Windows Subsystem for Linux(WSL). The Ubuntu 18.04.1 LTS is what I used. Please use the URL to install the system(https://docs.microsoft.com/en-us/windows/wsl/install-win10)

Step1. Get package for development

$sudo apt-get update

$sudo apt-get install npm

$sudo apt-get install nodejs

Step2. Create project under main directory

$mkdir book4xchg

$cd book4xchg

$npm init

$npm i -S mongoose express body-parser morgan cors

We also initialize the directory via "npm init", which creates the package.json for project building environment. The terminal will prompt you to enter details, just pressing enter until it finishes. There are also some packages for back end to be installed, such as as mongoose, express, body-parser, morgan and cors.

* Express: It’s the server framework (The E in MERN).
* Body Parser: Responsible to get the body off of network request.
* Nodemon: Restart the server when it sees changes (for a better dev experience).
* Cors: Package for providing a Connect/Express middleware that can be used to enable CORS with various options.
* Mongoose: It's an elegant MongoDB object modeling for node.js

Step3. Backend development

Introduce Server.js and Data.js in main directory for the backend. Replace the dbRoute with your own user and password in MongoDB Atlas. Here is how to setup mongoDB Atlas(https://docs.atlas.mongodb.com/getting-started/)

const dbRoute = 'mongodb://<your-db-username-here>:<your-db-password-here>@clusterone-n3x4p.mongodb.net/test?retryWrites=true&w=majority';

Step4. Start back end server.

$ node server.js

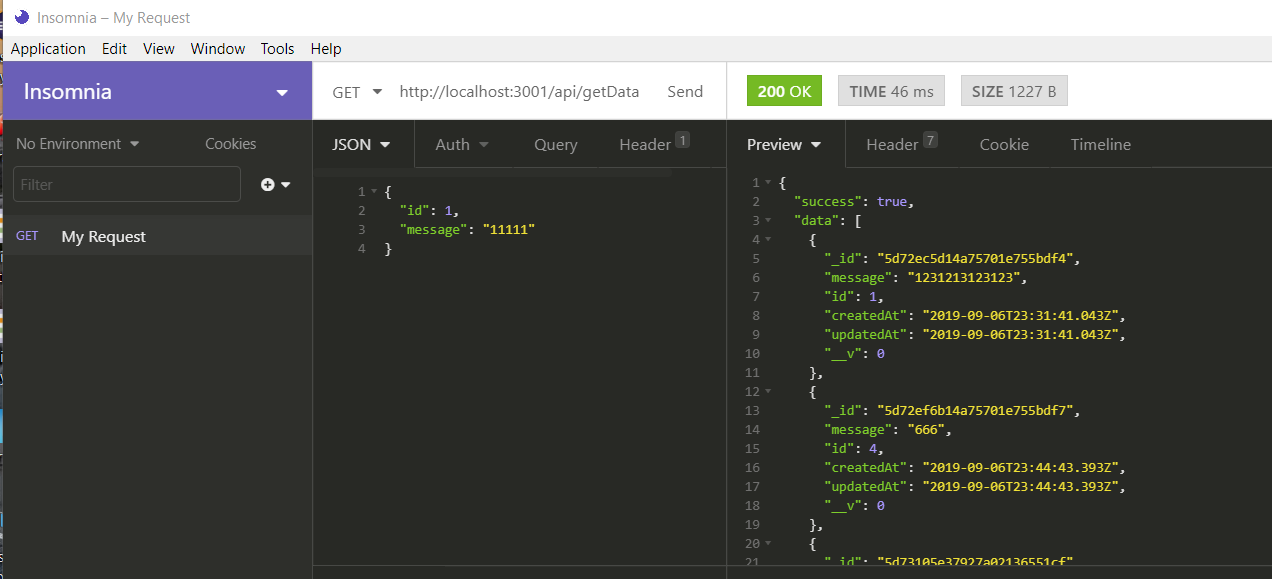
Modify package.json to start the server. So that we may “npm start” instead of “node server.js”

"scripts": {

"start": "node server.js"

},

You may use Insomnia to verify the backend server. The JSON will be similar to what we have in data.js.



Step 5. Front end framework and development

$ create-react-app client && cd client

$ cd client

$ npm i -S axios

we use create-react-app to front end frame work. Here is the command to install the package: $ npm i -g create-react-app. Now we can create app and axios module. Please modify the client/src/app.js and client/src/index.js.

Step6. Front end start

$npm start

Open the browser and type [https://localhost:3000](https://localhost:3000/). Here we go!

Step7. Upload code to github.

Enter the root directory of the project. Please use this guide to create a new repository at github(https://help.github.com/en/articles/create-a-repo). Then run the following command to push the code to github

$ echo modules > .gitignore

$ git init

$ git add A

$ git commit -m "firstcommit"

$ git remote add original <https://github.com/mail2mz/book4xchg.git>

$ git push original master

Github will prompt you to input the user and password for github.

**ShortCut**: The steps to reclone the repository from github.

$mkdir local\_folder

$cd local\_folder

$npm install && npm start

$cd local\_folder/client

$npm instll && npm start

Step8 Deploy to Heroku.

We have to modify some code to make the code working on Heroku. Here are the four items to be modified:

1. Port. const API\_PORT = process.env.port || 3001;

2. MongoDB. Use process.env.MONGODB \_URI for local mongodb. I am using MongoDB.Altas. Thus no change is needed.

3.Integrate REACT server into the backend server.

const path = require('path'); //add path module

...

// append /api for our http requests

app.use('/api', router);

if (process.env.NODE\_ENV === 'production') {

// Serve static files from the React app

app.use(express.static(path.join(\_\_dirname, 'client/build')));

// The "catchall" handler: for any request that doesn't

// match one above, send back React's index.html file.

app.get('\*', (req, res) => {

res.sendFile(path.join(\_\_dirname+'/client/build/index.html'));

});

}

4. Add heroku-postbuild to package.json

"heroku-postbuild": "cd client && npm install && npm run build"

5. Modify proxy

Add " "proxy": "[http://localhost:3001](http://localhost:3001/)" to client\pacckage.json

Remove localhost in client\src\app.js

I assume everyone has an account in keroku and have Heroku client installed already. Here are the steps for a new heroku app

$heroku create

$heroku git:remote -a whispering-castle-49989

$git push heroku master