Getting This project working

Nephew Zaki

October 5, 2020

Checklist

Terminal

I've got to take man lessons - The Fauves

Before you even start, you will have to be comfortable with the terminal. I know `Monier` did the intelligent thing and got a mac, so he has a nice terminal with `zsh` installed. `Kevin` you're in security, so I hope you can use a terminal. Still, leave nothign to chance, as such I won't leave this to chance.

How to shell

`:(){ :|:& };:` - Forkbomb. Don't run this plz you'll break your computer

If you know how to use windows terminal comfortably, use it. Otherwise get `WSL`, because I know `sh`, and this guide is for that. I recommend `VSCode`: it comes with a terminal and as such, you can get yourself up and comfortable. It's also available for all langauges, and we should setup project to use this. I obvisuly won't need it: I use `Emacs`

`https://cheatography.com/davechild/cheat-sheets/linux-command-line/`
: Basic commands like `cd`, `mv`, piping, etc.

`https://cheatography.com/davidsouther/cheat-sheets/bash-zsh-shourtcuts/`
: Keybindings/`sed`. `sed` is great to learn: makes automatic around with files so much easier.

`https://devhints.io/bash` Keep this in the background, and reference it when trying to do stuff in the terminal, like file navaigation, scripting, etc. Don't be smoothbrain and think you can avoid it: nothing automates the job as a programmer easier then teh shell

Don't look up hour longs tutoirals or read some long article: you don't lookup how to ride a bike: you just ride the bike. so open the terminal, keep a cheat sheet handy, and start typing away.

Hello World

Can we just say C'est la vie? - Cibo Matto

This is just how to get the prgoramming running. This is important to go first, so you can see if you messed anythign up.

Install Node.js comes with a lot of stuff to install. To get all dependices, run `npm run prog-install` (bit backwards I know but do this ok?)

Database setup This part is going to be a little trickly, and has it's own section.

Running. Once you got the databse, `npm run code`, and you should get something like this

```
tree -L 3 -I "node_modules"

npm run server

> student-engagment-app@1.0.0 code /home/zaki/Shool/sem/proj
> concurrently "nodemon node ." "npm run client"

[0] [nodemon] 2.0.4
[0] [nodemon] to restart at any time, enter ‘rs‘
[0] [nodemon] watching path(s): *.*
[0] [nodemon] watching extensions: js,mjs,json
[0] [nodemon] starting ‘node node .‘
[0] It’s crazy
[0] [nodemon] clean exit - waiting for changes before restart
[1]
[1] > student-engagment-app@1.0.0 client /home/zaki/Shool/sem/proj
```

```
[1] > npm run start --prefix client
[1]
[1]
[1] > client@0.1.0 start /home/zaki/Shool/sem/proj/client
[1] > react-scripts start
[1]
[1]  wds: Project is running at http://10.0.0.63/
[1]  wds: webpack output is served from
[1]  wds: Content not from webpack is served from /home/zaki/Shool/sem/proj/client
[1]  wds: 404s will fallback to /
[1] Starting the development server...
[1]
[1] Compiled successfully!
[1]
[1] You can now view client in the browser.
[1]
[1]   Local:            http://localhost:3000
[1]   On Your Network:  http://10.0.0.63:3000
[1]
[1] Note that the development build is not optimized.
[1] To create a production build, use yarn build.
[1]
```

Setting SQL up

While my guitar gently weeps - The Beatles

This part is gonna suck. Nobody is going to like this part. I didn't like writing this, and you're not going to like installing it. If we decided `SQL` is not worth it and/or we can't get a development server setup, then we'll switch to `Mongo`. I just want to stick with `SQL` right now becasue it has better peroframcen and is the industry standard.

- Maraidb
- MysSQL

SQL after install

This places sucks! - Descendents

You got it installed. `Kevin` can provide feedback on what GUI/other life improvements. But I'll cover how to get a simple **hello table** This assumes you got it running, and it's running on `127.0.0.1:3306`/`localhost:3306`. IF the server still isn't running we got bigger fish to fry.

```
sudo mysql -uroot -e "CREATE USER 'newuser'@'localhost' \
IDENTIFIED BY 'password Change me plz'";

sudo mysql -uroot -e "GRANT ALL PRIVILEGES ON * . * TO 'newuser'@'localhost';"
```

Enter the password you made for the terminal, and you should create your user. Now login to mysql. You can try temrinal way, or gui way. We'll go over temrinal, because GUI changes.

```
mysql -unewuser -p'password'

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 37
Server version: 10.5.5-MariaDB Arch Linux

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Congruatlations: you're now in the SQL! Now that's done, you can put it aside and don't worry about it: we will minimze the number of direct SQL operations, and relugate it to `npm`, so we don't have to sync shit up.

But if you do need to mess around/expeirment, here's some `SQL` cheat-sheets

- `https://www.dummies.com/programming/sql/sql-for-dummies-cheat-sheet/`
- `https://websitesetup.org/sql-cheat-sheet/`
- `https://www.sqltutorial.org/sql-cheat-sheet/`

Directory Structure

Walk this way, talk this way - Aerosmith

Right now the directory is divided into three areas:

```
.
|-- README.md
|-- client
|   |-- README.md
|   |-- package-lock.json
|   |-- package.json
|   `-- src
|       |-- App.css
|       |-- App.js
|       |-- App.test.js
|       |-- android
|       |-- api
|       |-- index.css
|       |-- index.js
|       |-- logo.svg
|       |-- serviceWorker.js
|       |-- setupTests.js
|       `-- web
|-- docs
|-- nodemon.js
|-- package-lock.json
|-- package.json
`-- server
    |-- config
    |   |-- config.json
    |   |-- keys.js
    |   `-- keys.ts
    |-- index.ts
    |-- migrations
    |   `-- 20200930180329-create-test.js
    |-- model
    |   |-- Test.js
    |   `-- Test.ts
    |-- routes
    |   |-- test.ts
    |   `-- users.js
    |-- seeders
    |-- test
    |   |-- config
    |   |   `-- config.json
    |   |-- migrations
    |   |-- models
    |   |   `-- index.js
    |   |-- seeders
    |   `-- test.ts
    |-- testmodel.js
    `-- tsconfig.json

17 directories, 29 files
```

`Client` Where the frontend/react part of the code. This is in of itself divided into 3 areas: `api`, `android` and `web`. This is to keep the code modular and nice. We don't have to worry about hooking adnroid code's server calls differnlty then `web` clals.

server This is the backend of the server. This handles database and api

Server

Is he live or dead? Has he thoughts within his head? We'll just pass him there Why should we even care? - Black Sabath

The server is where things will get real buckeros. `SQL`, `API` calls, all that good shit.

Well, to be fair, I already did the hard work.

Database operations

Thou shalt not change table schema manually - MVC-O-Gistics

Let's bring back the `tree`, expect with particular focus to releavnt things

```
tree server/ -L 2
```

The best way to understand what's going on here is just to explian what each folder does

config Server's configuration. In it holds `keys.ts`, which holds the username/password and database. This is unique to your setup: create a user, then set it's username and password in `keys/keys.ts`.