STEP 1 – BACKEND CONFIGURATION

* I downloaded, installed, and configured my Mobaxterm. I also connected my EC2 instance to it

Text

Description automatically generated

* I need to update my newly configured EC2 server using the command

---------- sudo apt update

Text

Description automatically generated

* I will also upgrade my server running the command.

---------- sudo apt upgrade

Text

Description automatically generated

* Next, I need to get the location of Node.js on my Ubuntu software running the command

---------curl -fsSL https://deb.nodesource.com/setup\_18.x | sudo -E bash –

Text

Description automatically generated

* I will install Node.js with the command below

--------- sudo apt-get install -y nodejs

Text

Description automatically generated

* The node comes with its own package manager called NPM which I will use to install node dependencies later. I will verify node installation as well as the package manager with the command below

---------- node -v

---------- npm -v

Text

Description automatically generated

* I will create a new directory for my To-Do project

--------- mkdir Todo

Text

Description automatically generated

* I will list and change to the newly created directory with the command

---------- cd Todo

Text

Description automatically generated

* I will run the below command to create a new file in my newly created directory

----------- npm init

Text

Description automatically generated

* I will run the below command to confirm that I have the package.json file

----------- ls

Text

Description automatically generated

* I will run the command to install express

---------npm install express

Text

Description automatically generated

* I will create a file with the command below and run "ls" command to be sure it was created

---------------touch index.js

Text

Description automatically generated

* I will install dotenv with the command

------------ npm install dotenv

Text

Description automatically generated

* I will open the index.js file with the command below and type below code into it.

-------- vim index.js

Text

Description automatically generated

* It is time to start my server by typing the command below, but first I will go back to my EC2 instance, under inbound rules I will open port 5000. This way my server will be able to run.

--------- node index.js

Text

Description automatically generated

* I will also copy my instance public address and go to my web browser to check of it will work

Graphical user interface, application, Word

Description automatically generated

* Next, I will create routes that will define various endpoints that the To-do app will depend on and 'cd' into it

-------- mkdir routes && cd routes

Text

Description automatically generated

* I will create a new file called api.js and vim into it to copy code in the file

--------- touch api.js

----------vim api.js

Text

Description automatically generated

* Next I will change directory back to Todo and install the mongoose using rpm

-----------npm install mongoose

Text

Description automatically generated

* I will create a new folder, cd into it and touch a file named todo.js

------- mkdir models && cd models &&touch todo.js

Text

Description automatically generated

* I will open the file todojs and paste the code below into it

---------- vim todo.js

Text

Description automatically generated

* I will change back to routes directory to update the api.js file and copy the below codes into it

------ cd routes

------ vim api.js

Text

Description automatically generated

* I will make use of mlab database to store my data

Graphical user interface, application

Description automatically generated

1. I will change directory back to Todo and create a file named .env

------- touch .env

Text

Description automatically generated

* I will vi into it and add a connection string to access the database.

------vi .env

Text

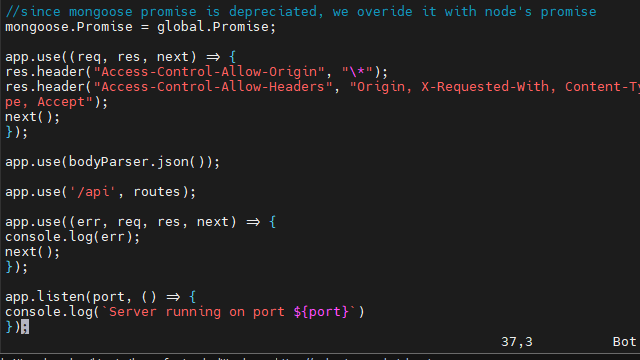
Description automatically generated

Text

Description automatically generated

* I need to update the index.js file to reflect the use of .env by typing the below code into it

------ vim index.js



* I will start my server using the command:

------ node index.js

Text

Description automatically generated

* There is error which is because EADDRINUSE. This means there is a process ID running that I have to kill. To check the process ID I will run the command

----------- lsof -I tcp:5000

Text

Description automatically generated

* Then I will kill the process ID with the command

------- kill -9 15562

Text

Description automatically generated

* There was an error, so I went back to fix it, created a new database in 'mlab' and redid the entry I have this now. Then I will run the command again

-------- node index.js

Text

Description automatically generated

* I will install postman to test my API. Create a POST request to the API and a GET request

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

* The result above shows that my API are working correctly sending and receiving.

* Now I must create my user interface for a web client to interact with the application via API. I will run the command

----------- npx create-react-app client

Text

Description automatically generated

* This will create a new folder called client. Before testing the react app there are some dependencies that need to be installed

---------- npm install concurrently --save-dev

Text

Description automatically generated

----------- npm install nodemon --save-dev

Text

Description automatically generated

* In Todo file I will open a file named package.json and change some parameters

Text

Description automatically generated

* I will change directory into 'client' and open the package.json file to add the key value pair so the application can be accesses directly from the browser.

Text

Description automatically generated

* I will change directory back to 'Todo' and run the command below

-------- npm run dev

Text

Description automatically generated

* I will go to my EC2 instance to edit the inbound rules by adding port 3000. This way I will be able to access the internet via my web browser.

Graphical user interface

Description automatically generated with medium confidence

* I will open another terminal and change directory to "Todo" "client" and "src'". I will create a new directory inside "src" called component and change directory into it.

---------- cd Todo && clients && src

------- mkdircomponents

--------- cd components

Text

Description automatically generated

* I will touch the following inside the component’s directory

----------- touch Inputs.js ListTodo.js Todo.js

Text

Description automatically generated

* I will open the Input.js and paste the following into it

---------- vi input.js

Text

Description automatically generated

* Next, I will change directory into "clients" and install the axios package

--------- npm install axios

Text

Description automatically generated

* I will change directory back to "components and "vi" into my ListTodo.js file and paste the following code into it

------------- cd src/components

------------ vi ListTodo.js

Text

Description automatically generated

Text

Description automatically generated

* I will also write the following code into my Todo.js file

-------- vi Todo.js

Text

Description automatically generated

* I will make an adjustment to my react code. Delete the logo and adjust the App.js

------- cd ..

-------- vi App.js

Text

Description automatically generated

* I will also vi into App.css and paste the following code into it

--------- vi App.css

A screenshot of a computer

Description automatically generated with medium confidence

* I will also "vi" into the index.css file and copy the code below into it

--------------- vim index.css

Text

Description automatically generated

* I will go back to Todo directory and run the command below

---------- npm run dev

Text

Description automatically generated

* I will go back to my web browser and refresh it. It will look thus:

A black screen with white text

Description automatically generated with low confidence