27 August 2021

Project Name: Capstone Project

E-Commerce Online Purchasing Website

Made By Karabo Pheko
8-27-2021

1. **Name:** Karabo Pheko
2. **Date:** 27 August 2021
3. **Project Name:** Capstone-Project (E-Commerce Website)
4. **Description:** A dynamic online shopping of groceries platform/website.
5. **Technologies:**
    - VS Code
    - Notepad++
    - Amazon Web Services, EC2 (Ubuntu) Instance (Host mean stack website for free)
    - Docker (Containerizing the angular application)
    - Node JS
    - Angular (Frontend)
    - MongoDB (Database)
    - Express JS (Backend)
    - Nginx Server (reverse proxy, load balancing)
    - Putty and PuttyGen (Connecting to EC2 Ubuntu Instance via SSH)
    - Git and Github (Version control and project management)
    - Postman and RestEasy (Making API requests)

6. **GitHub Links:**
    - Front-end Angular Application: https://github.com/PhekoK/capstone-project.git
    - Backend Express Server: https://github.com/PhekoK/server.git

7. **Public IPv4 DNS**:

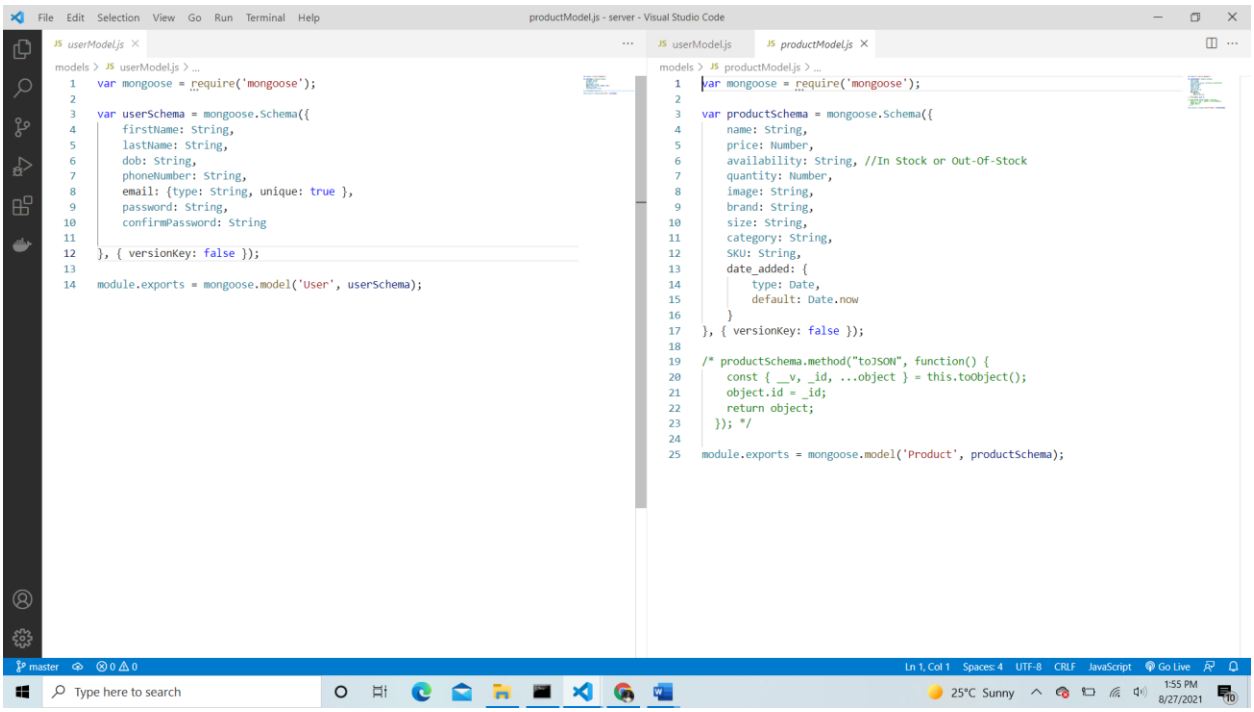
    http://ec2-18-119-139-156.us-east-2.compute.amazonaws.com:80

STEPS CREATING PROJECT – FROM DEVELOPMENT TO PRODUCTION

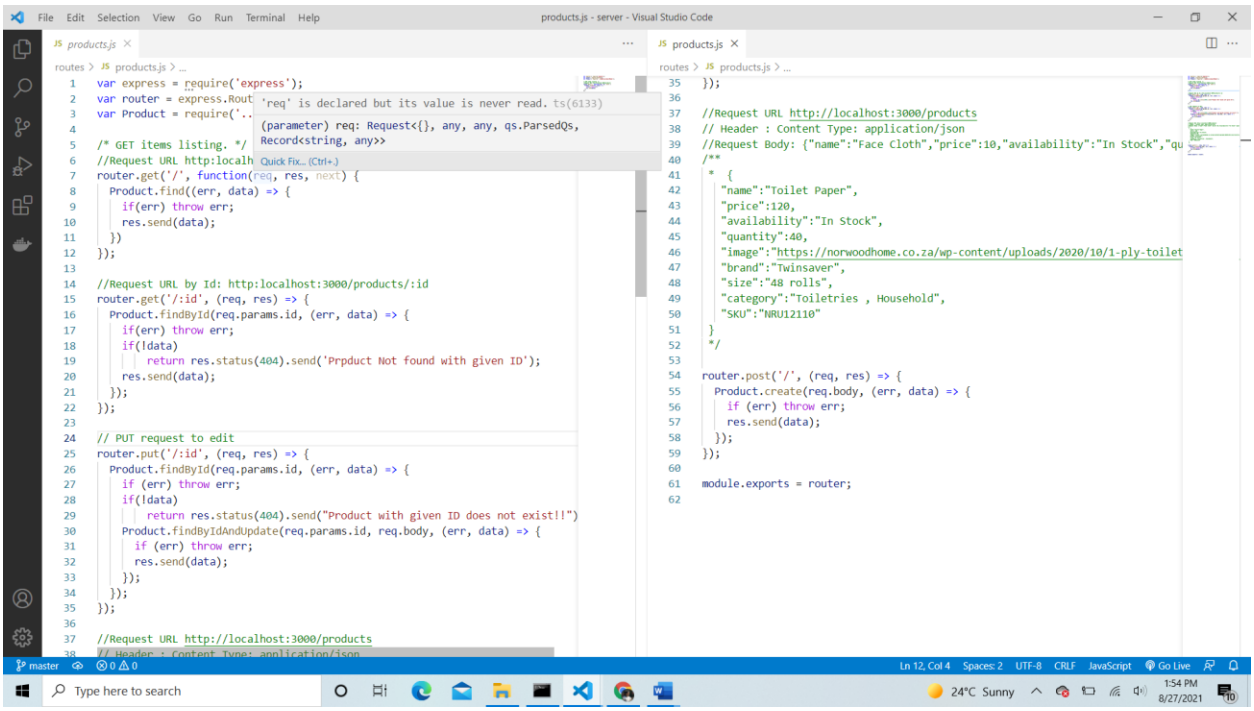
- **Development**

1. Backend - Express server.

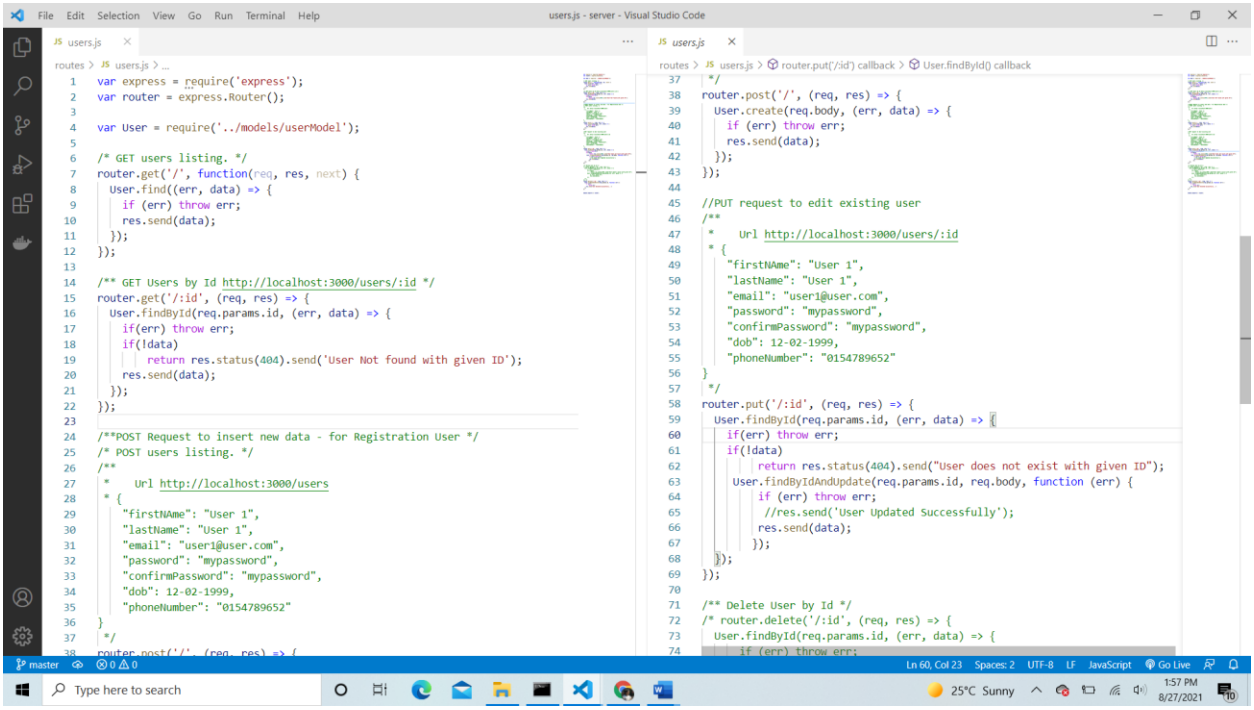
- Created Express Server and Mongoose models.
- User and Product Model



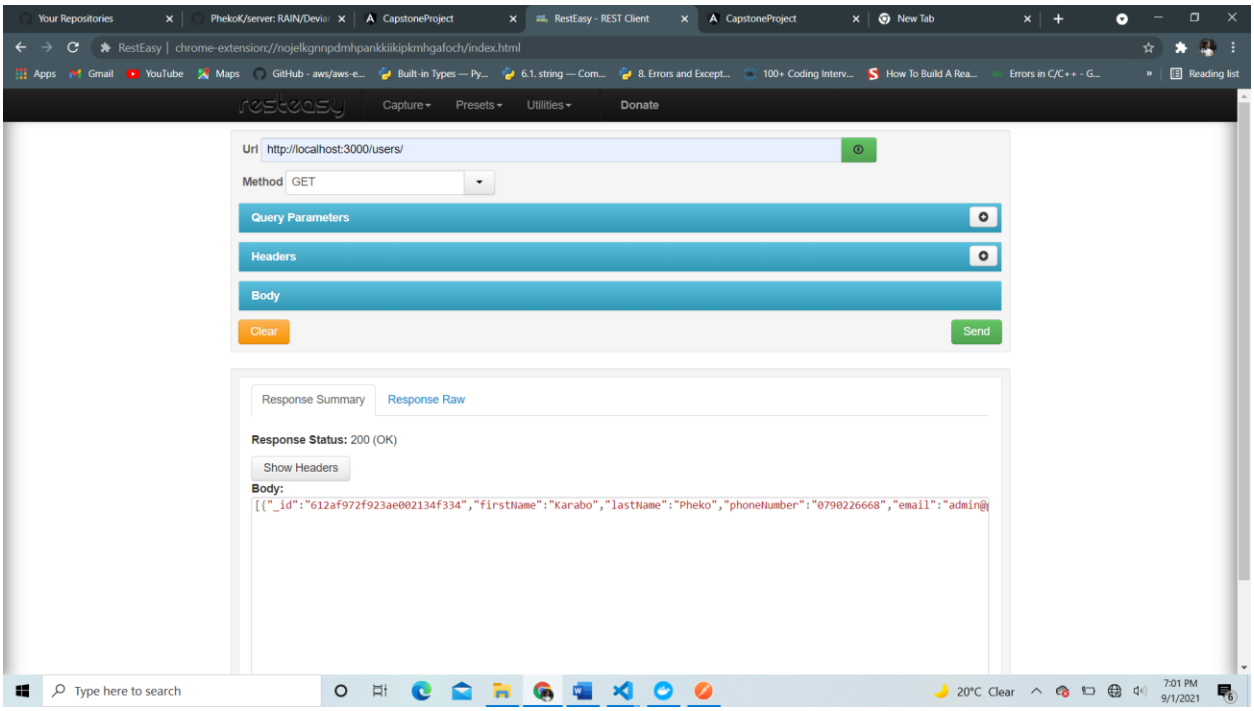
- Added API endpoints(routes) to see if my routing is working.
- Also made use of httpError handler to catch http errors to prevent server from crashing and restarting manually.
- Product route

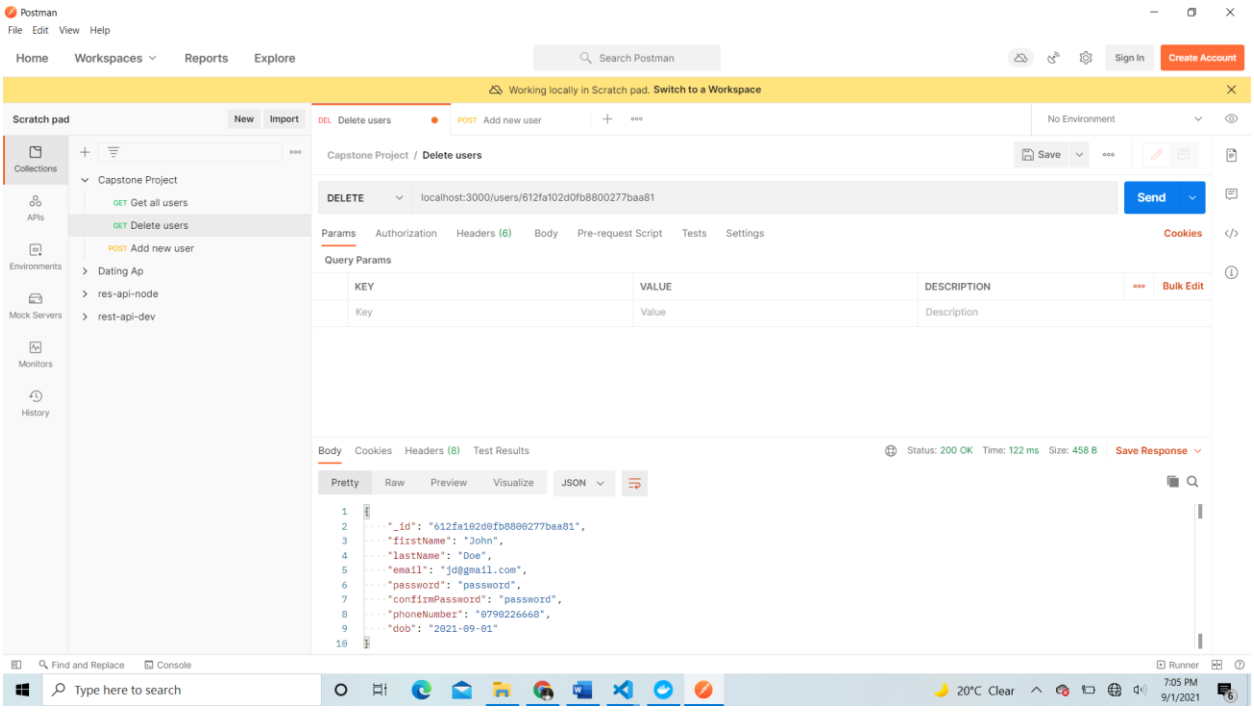


- User route

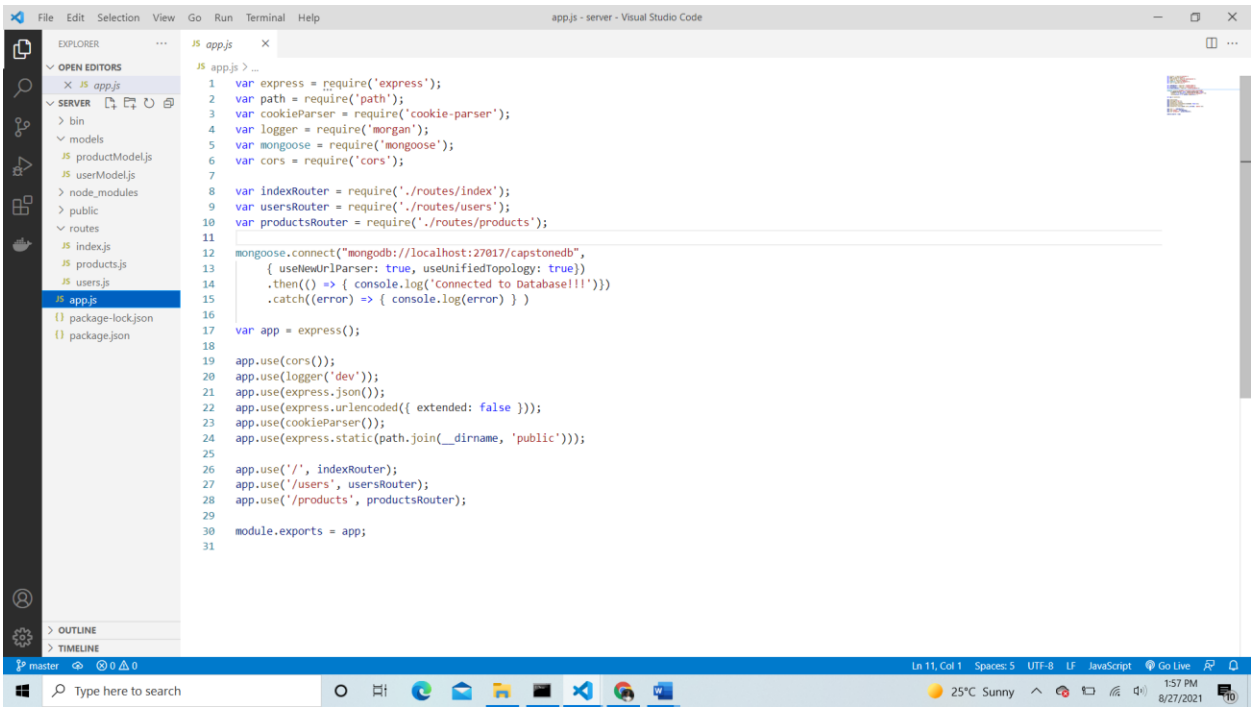


- In Postman/RestEasy. Tested if API requests work before working on the client.





- App.js – contains the middleware and routes of the angular project



2. Frontend – Angular Application

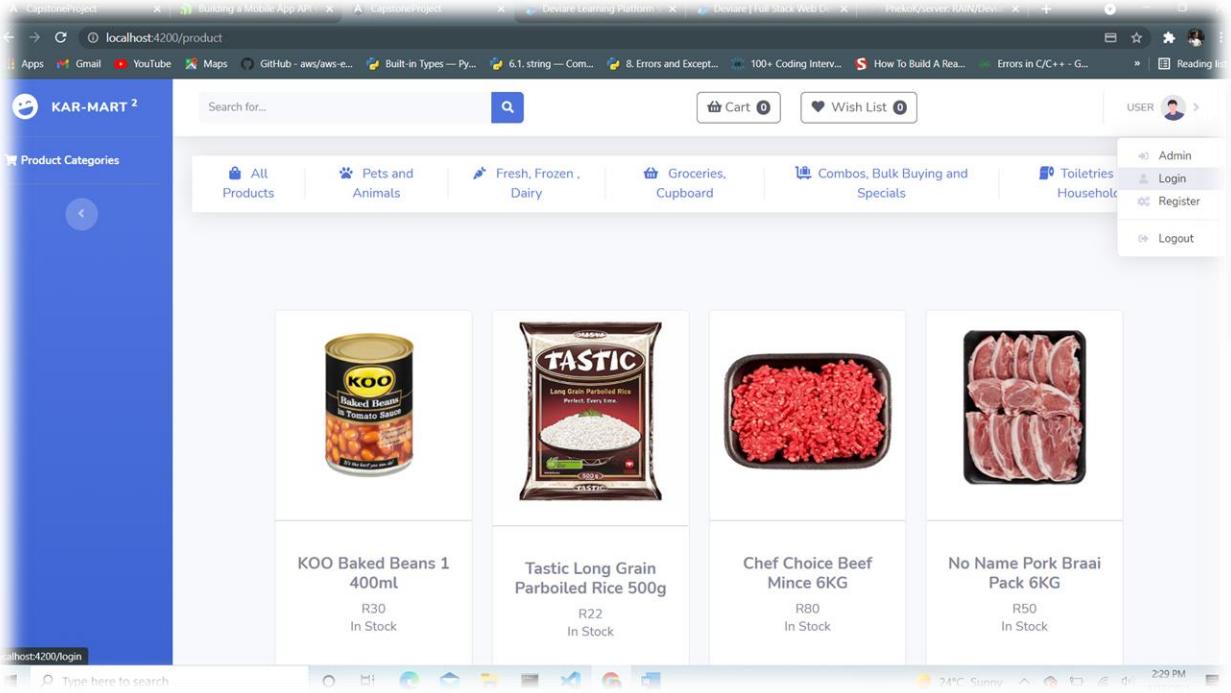
- Used bootstrap to enhance UI for both user and administrator portals.
- And used font-awesome icons.

A. User Portal

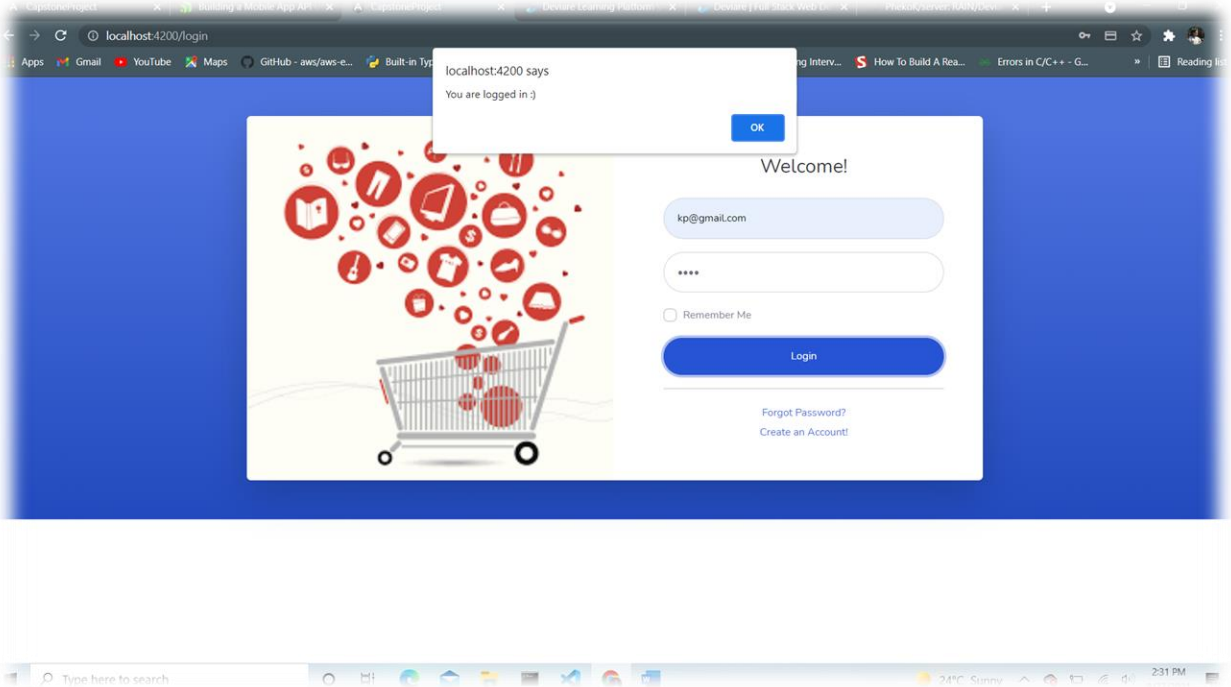
- The user can register and login with their credentials.
- View shopping items in product catalogue page to select which items to purchase.
- Add items to shopping cart, view cart items and delete them.
- If logged in, purchase and proceed to checkout page.
- Review invoice of purchased products.

- User may click the user button to either login/ register with their credentials.

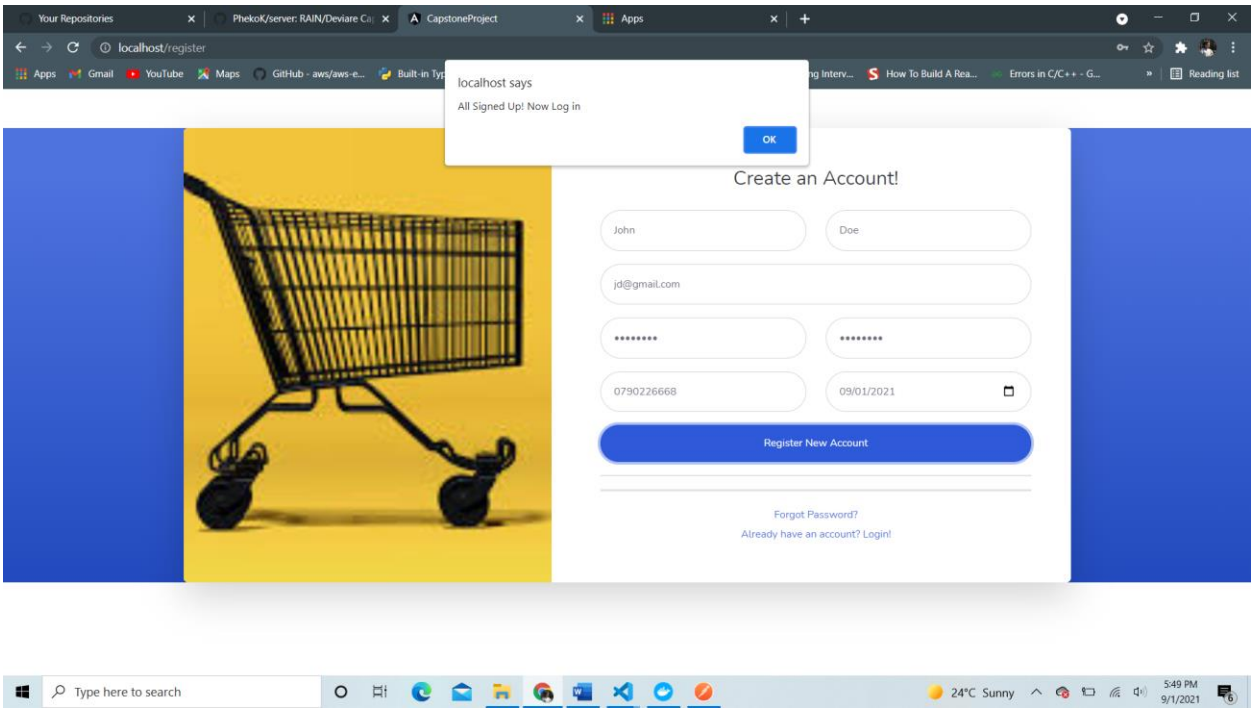
- Product page – with products to choose from



- Login page
- User with existing account may log in. Once logged in, user will navigate to the product catalogue page.

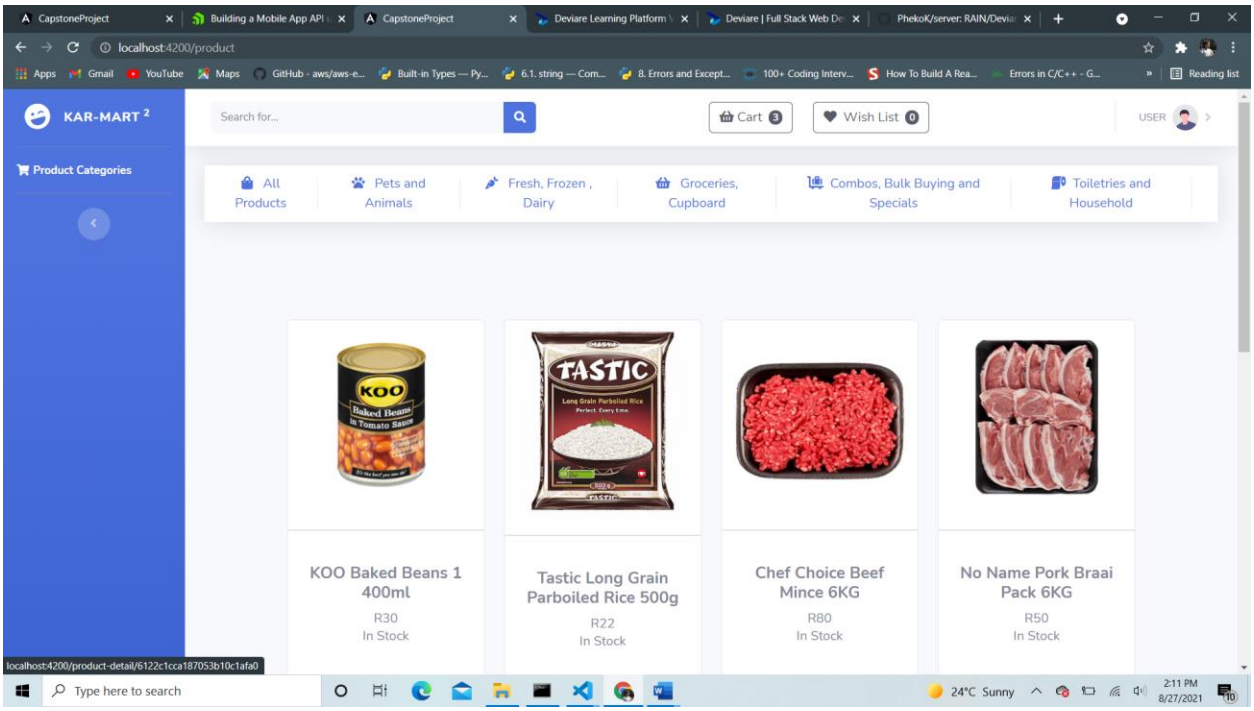


- Register page
- A new user may register their credentials and navigate to login page to sign in.

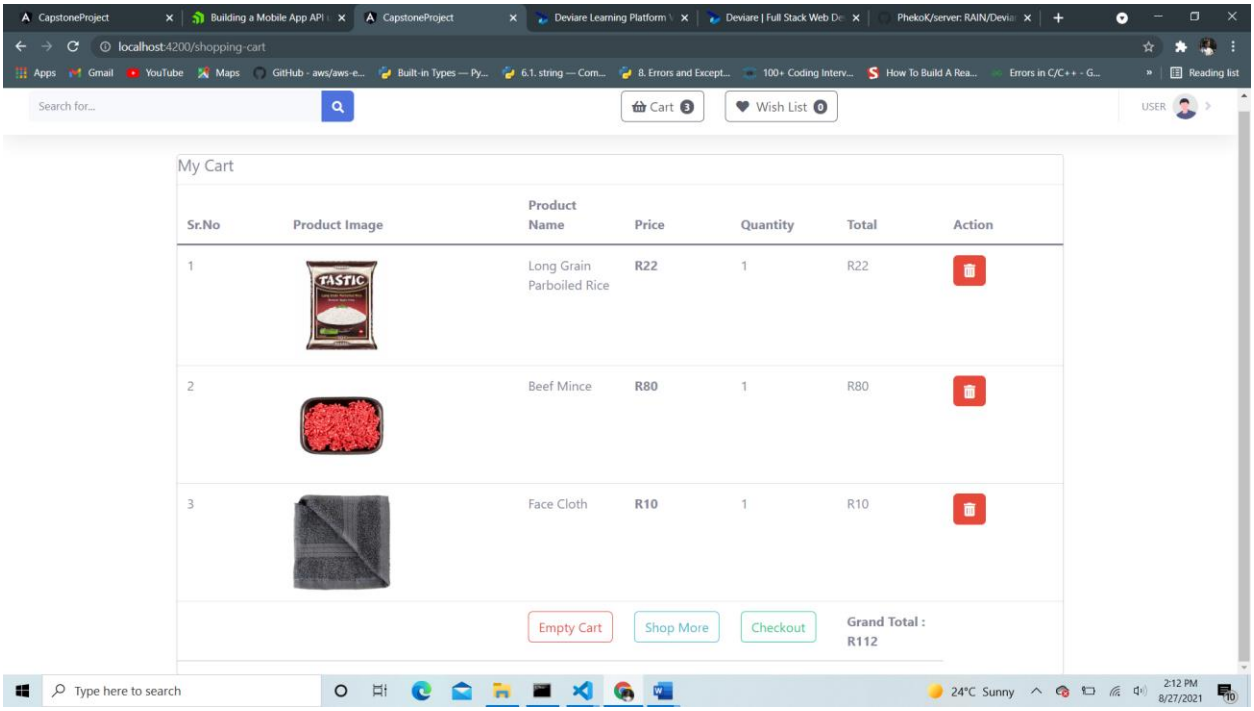


Product catalogue page.

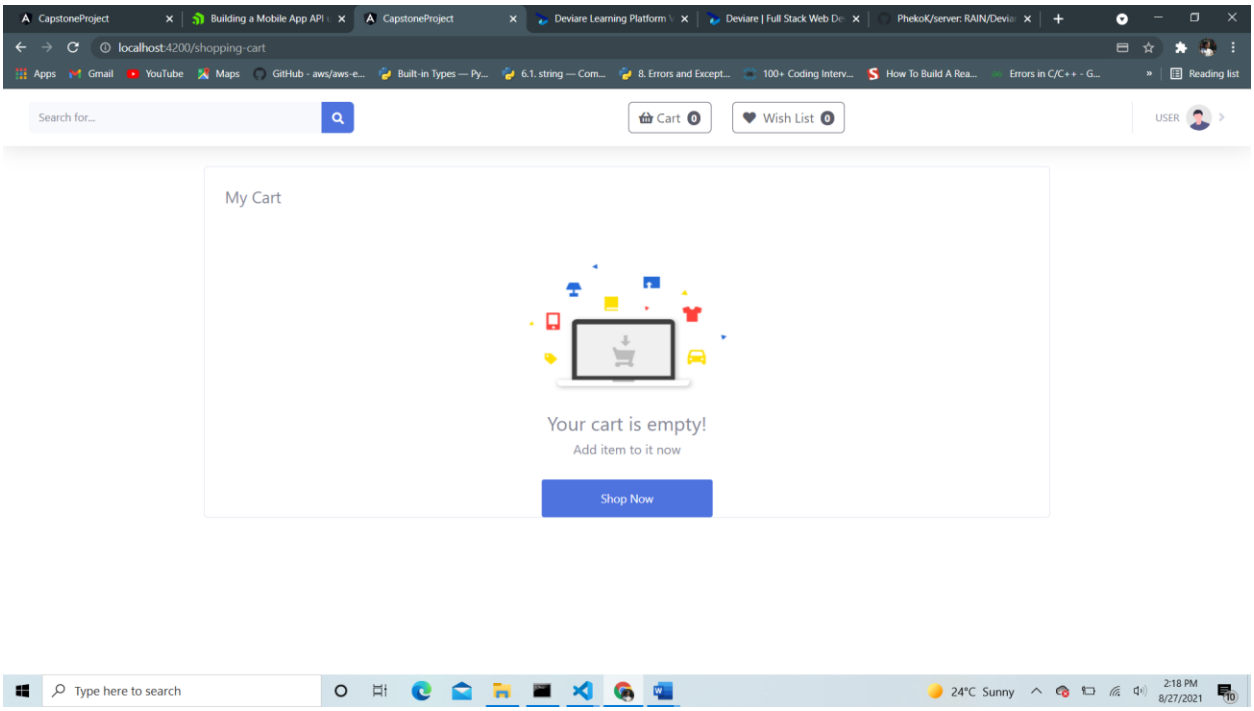
- The page is accessible for viewing items to all users (registered and unregistered) for the user to browse items to shop.
- Once user adds items to shopping cart and they are logged in, they can proceed to checkout page.



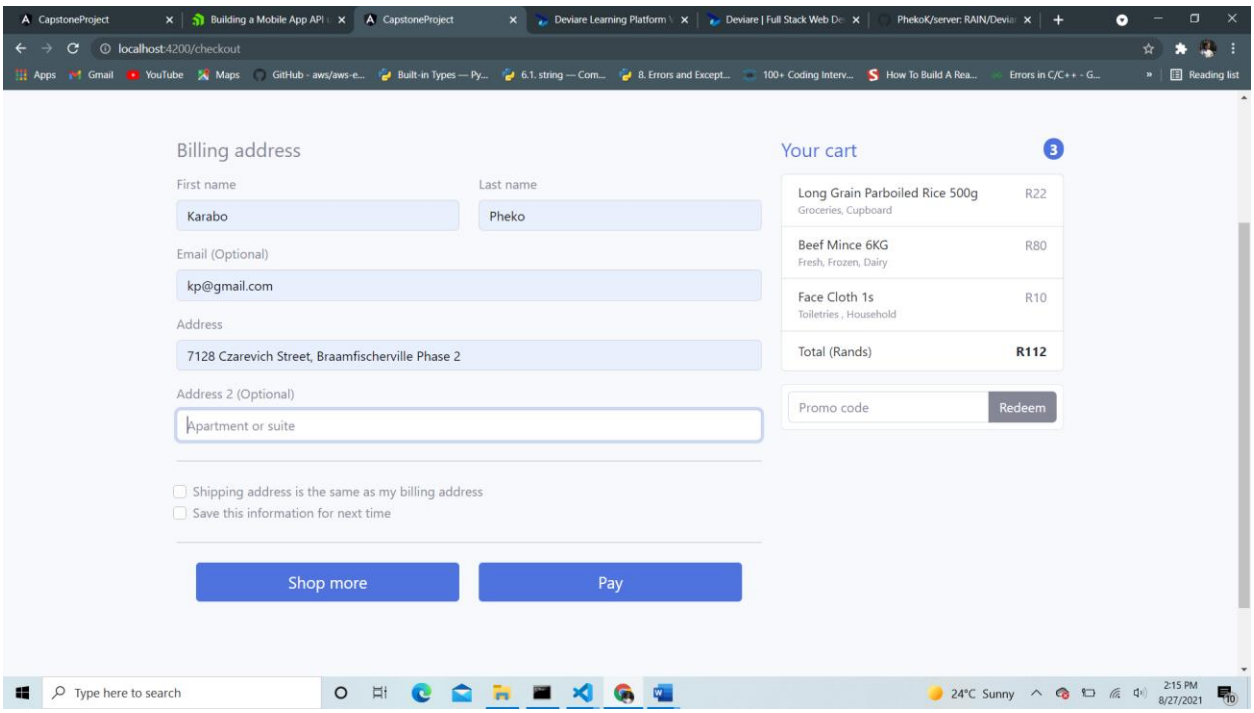
- When user added products in the cart, cart items are show as demonstrated below (if cart is not empty).



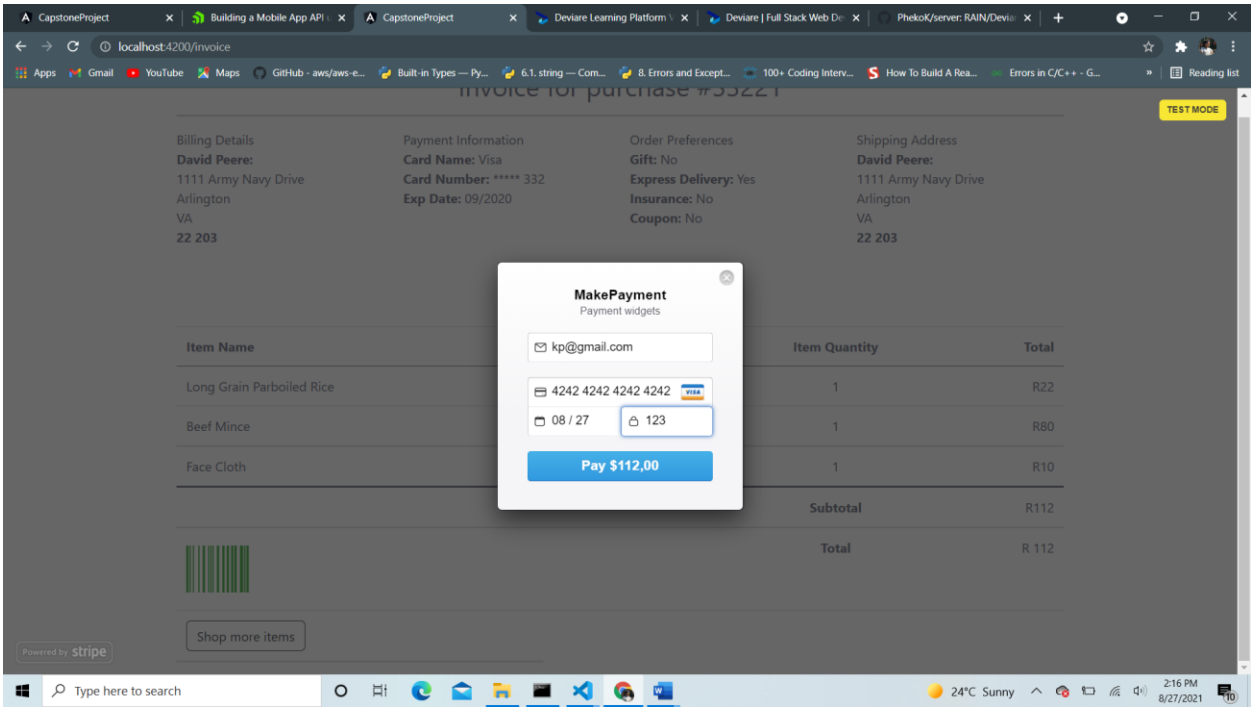
- If cart is empty, user is prompted to product page and browse items, as shown below.



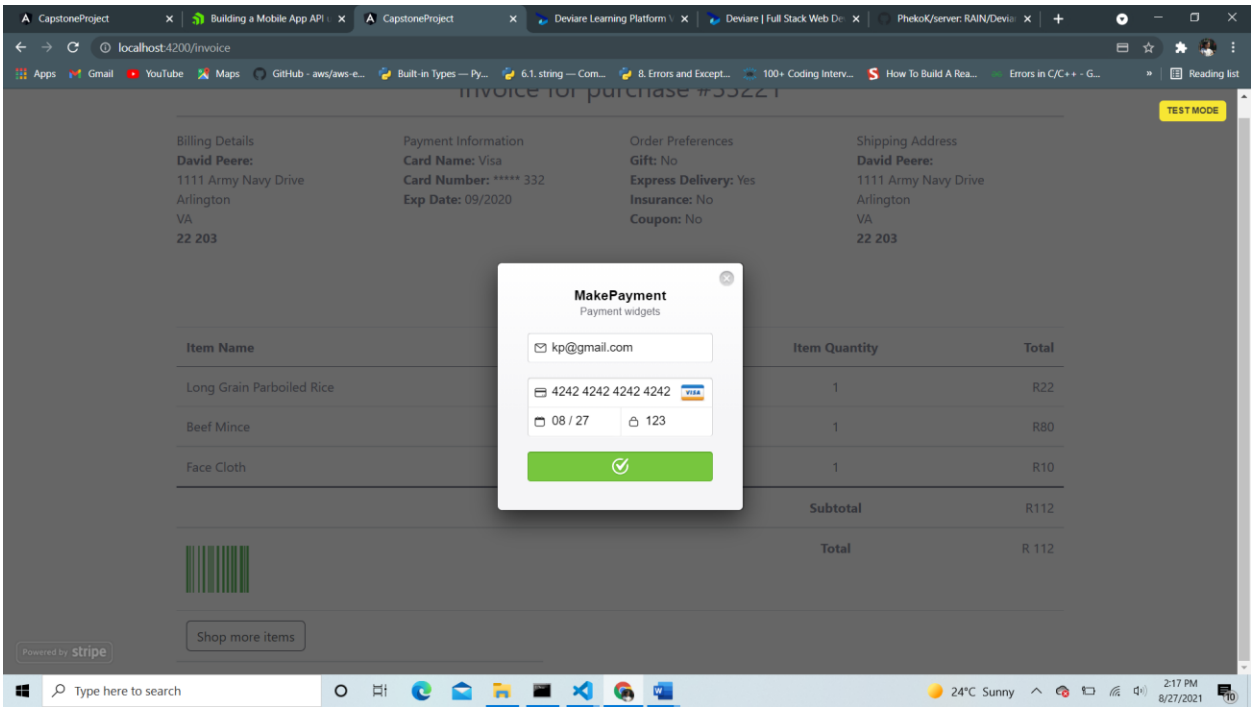
- The checkout page shows billing information that user needs to fill in for shipping/delivery.



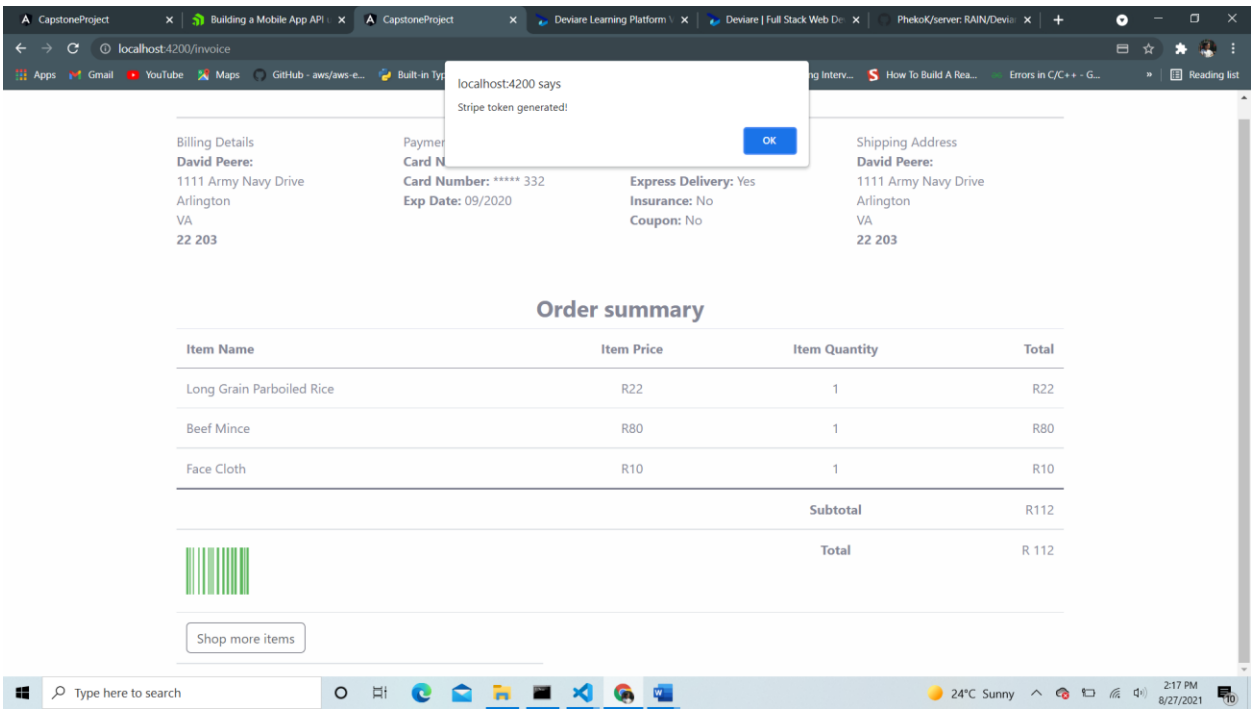
- The ‘pay’ method invokes a method that navigates to the invoice page about the shipping information and an alert generated by strip for ‘card’ payments.



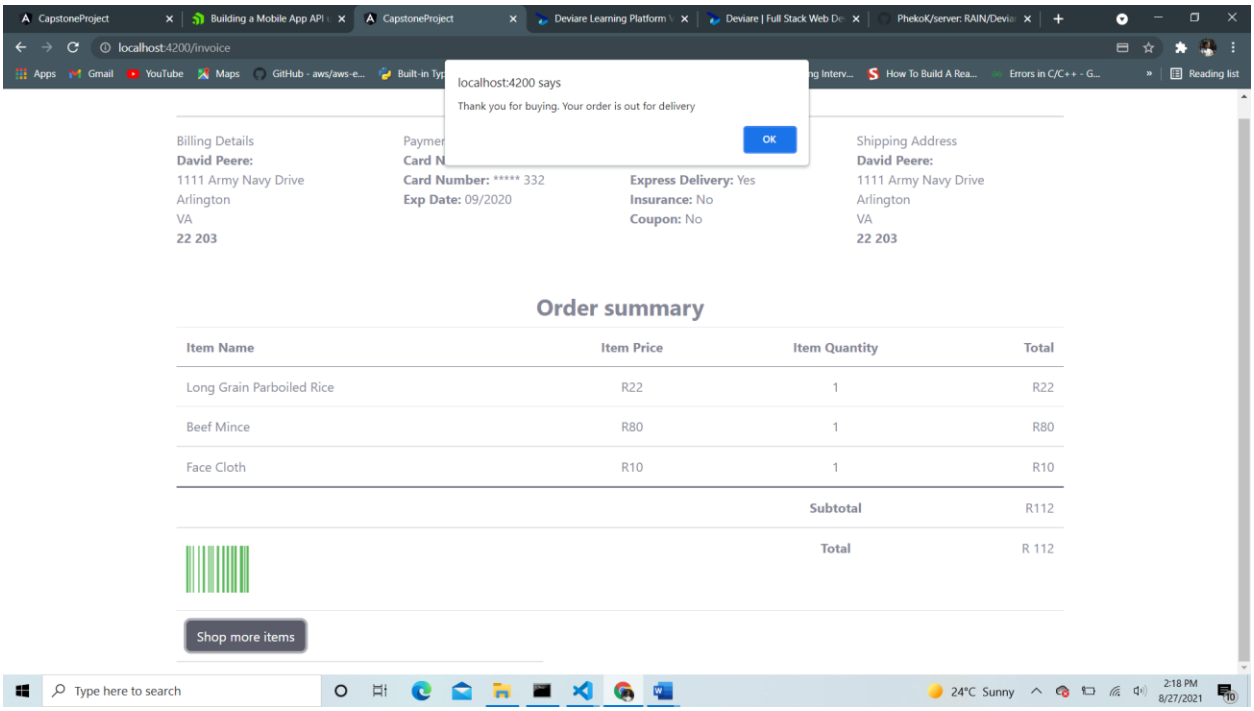
- Once verified, a stripe token will be generated, and user will navigate to invoice page.



- Invoice page is shown, displaying details of users order summary and shipping details



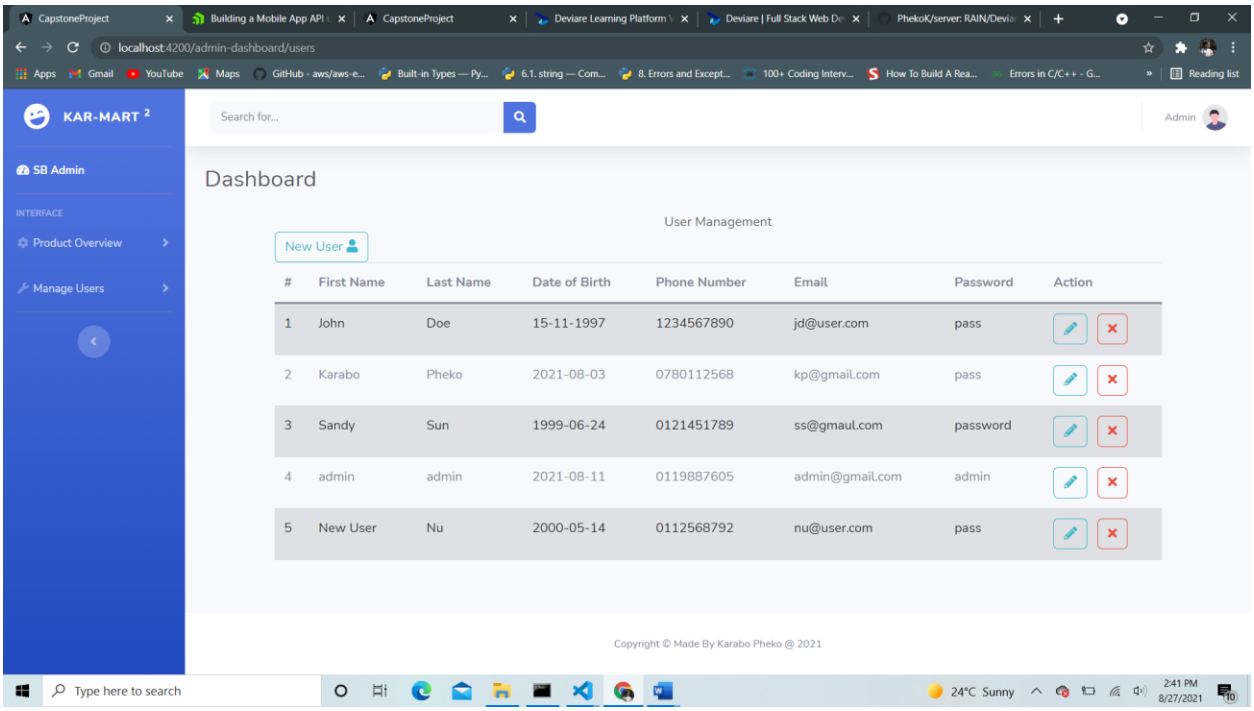
- When done. User may click “shop more” button to navigate to the product portal page. Or sign out.



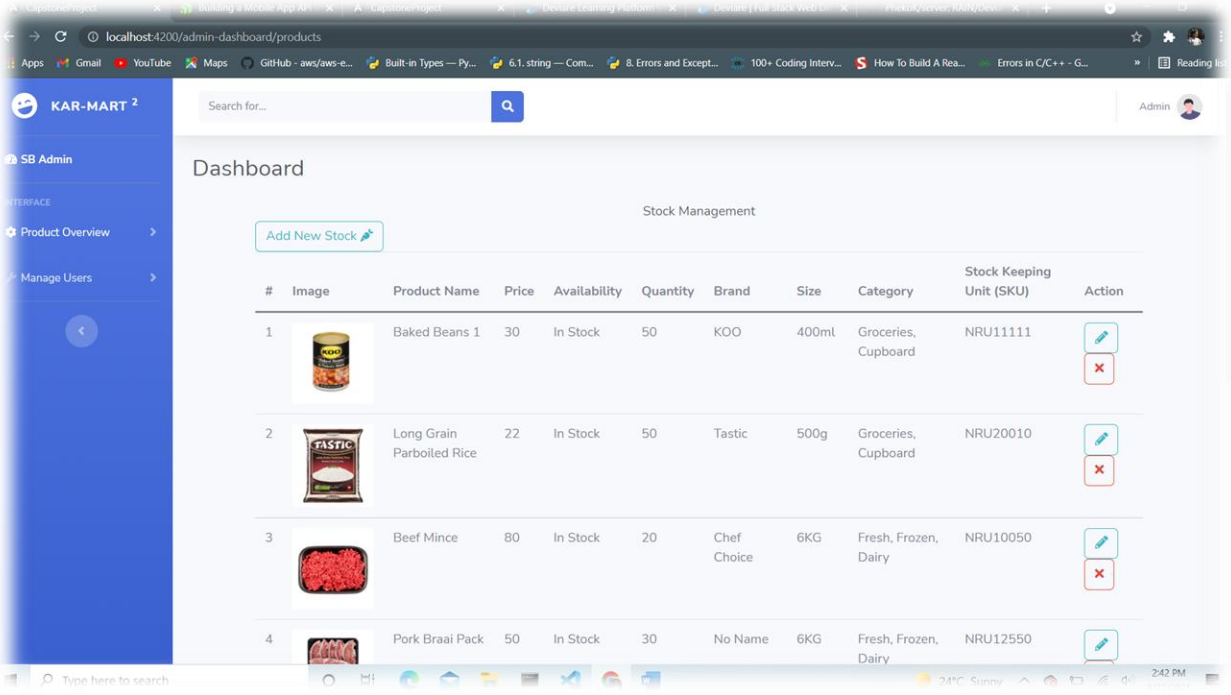
B. Administrator Portal

- An Admin may login and register their credentials.
- Uses email: admin@gmail.com and password : admin to access the admin portal.
- Once logged in. They oversee users and products.
- Add new users, update user credentials, and delete them.
- Add new products, update, and delete them.

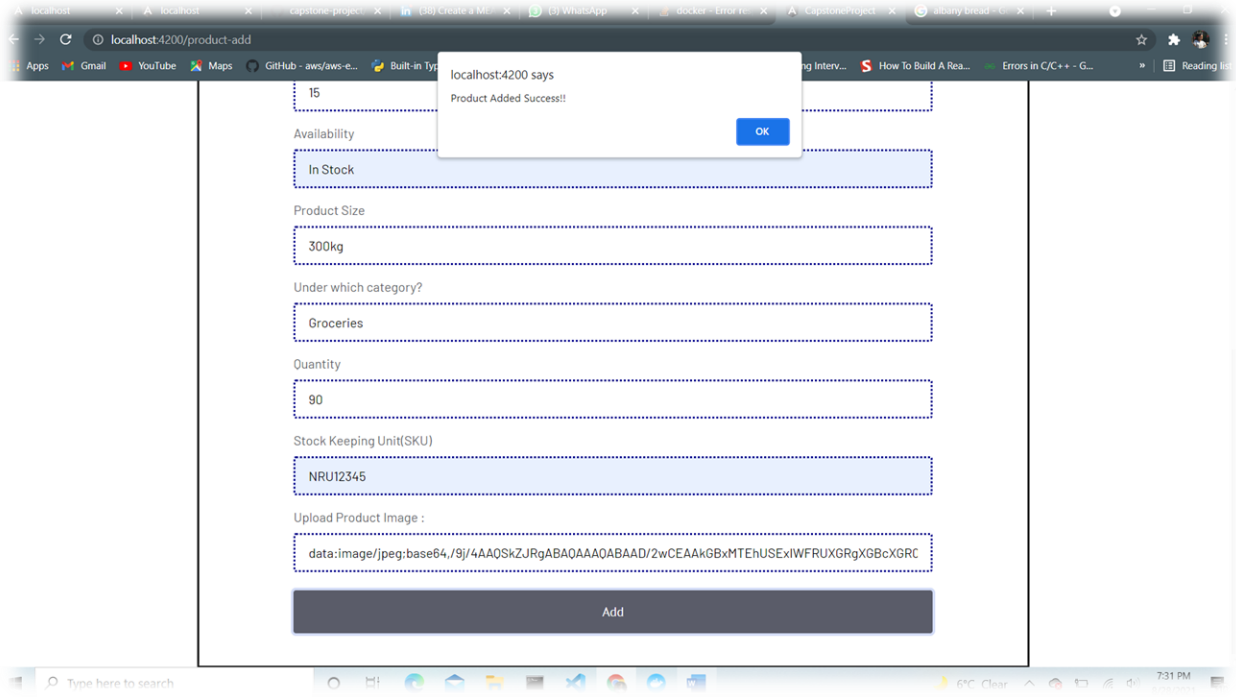
- users – admin can add new user, update current user and delete existing user.



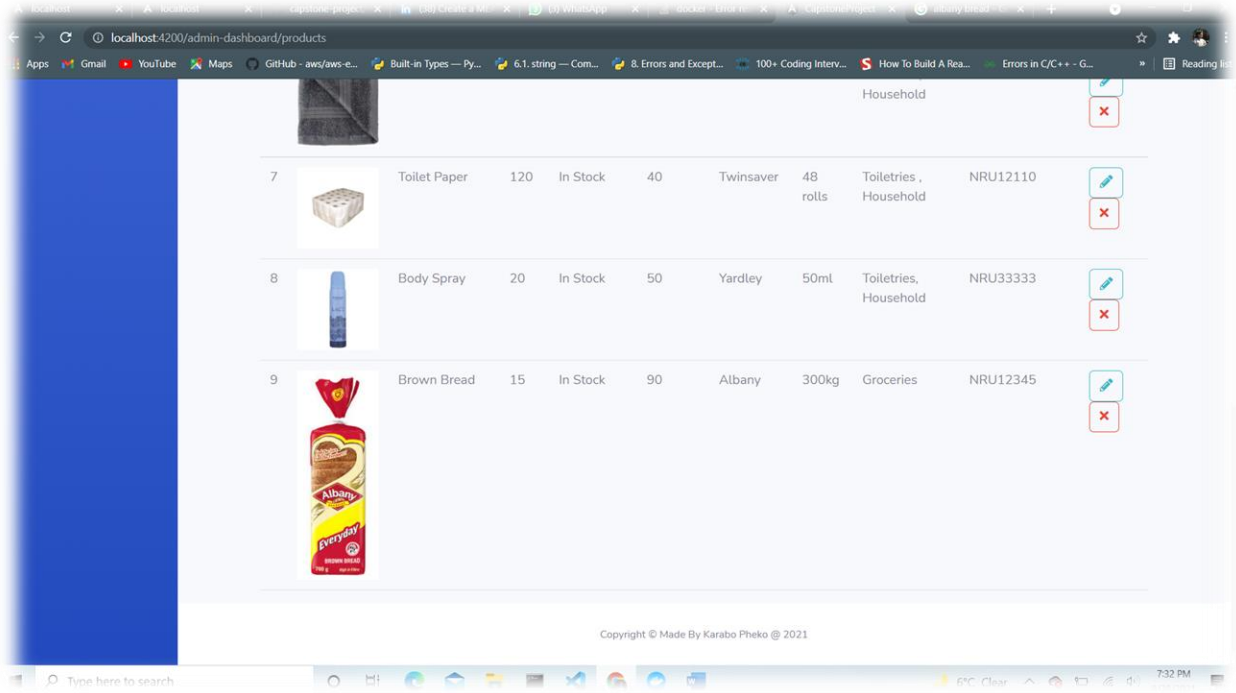
Products: - admin can add, delete, and update products.



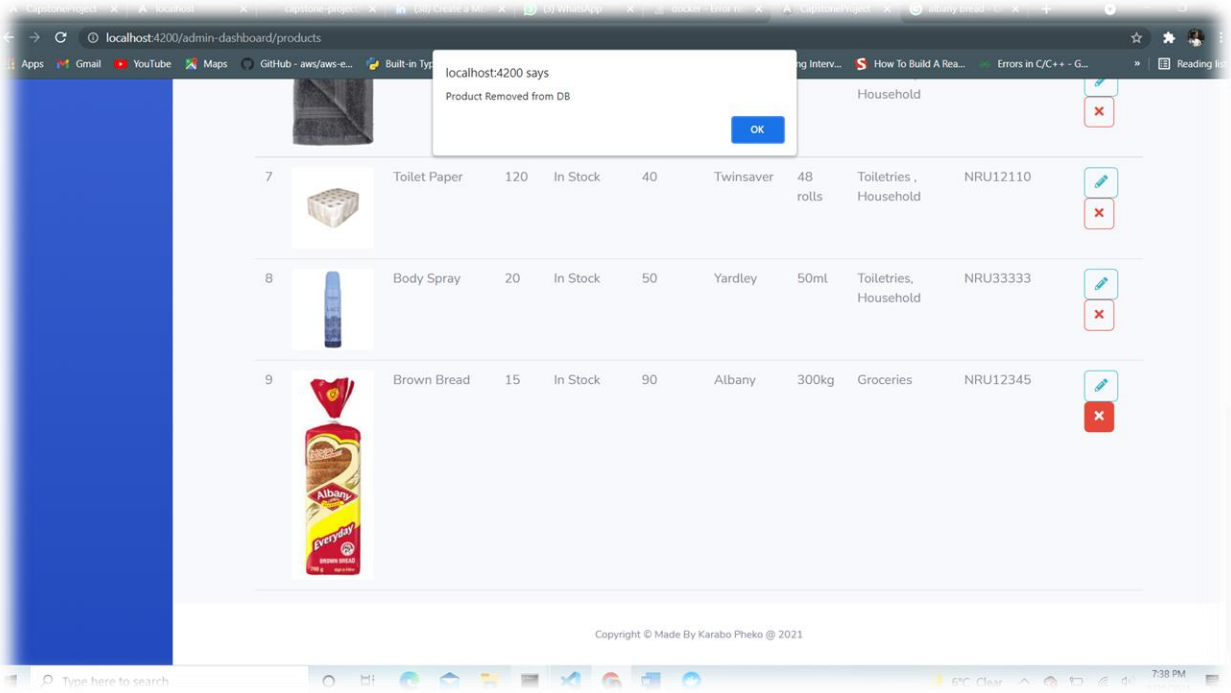
- Product Add – Admin can add a new product item.



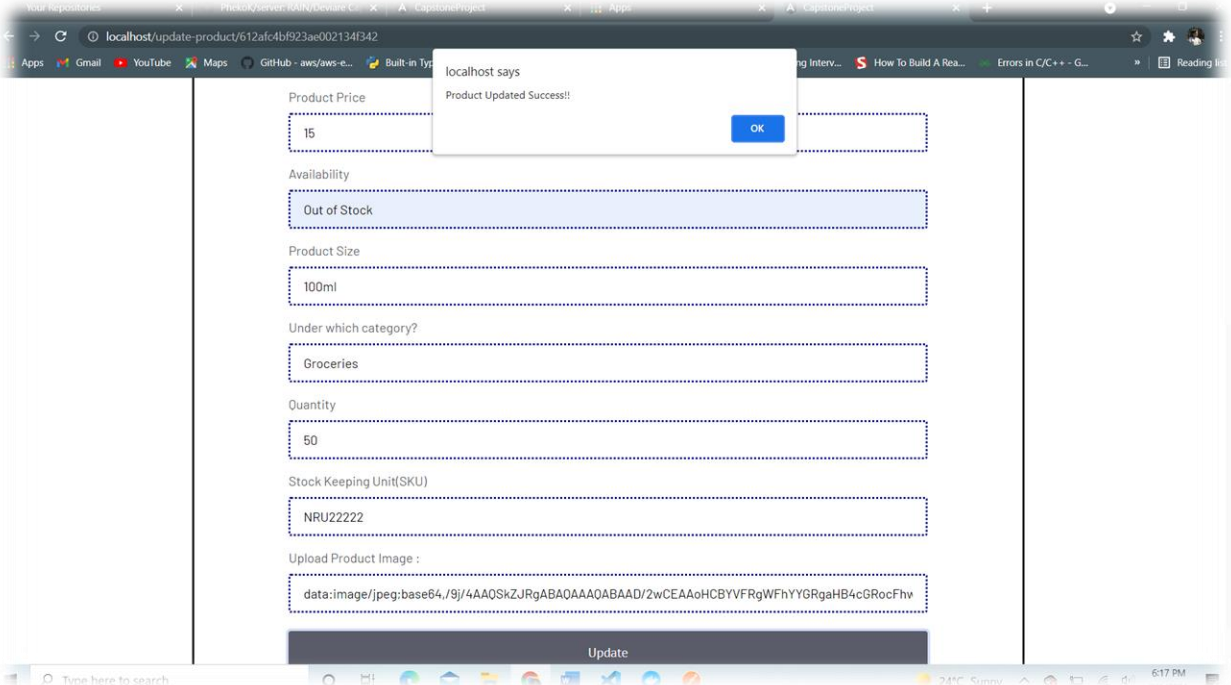
- product added will appear on the product overview page at the bottom of the page for the admin to edit or remove item.



- Removed Item will call an alert prompt informing the admin that a product has been removed from Database.



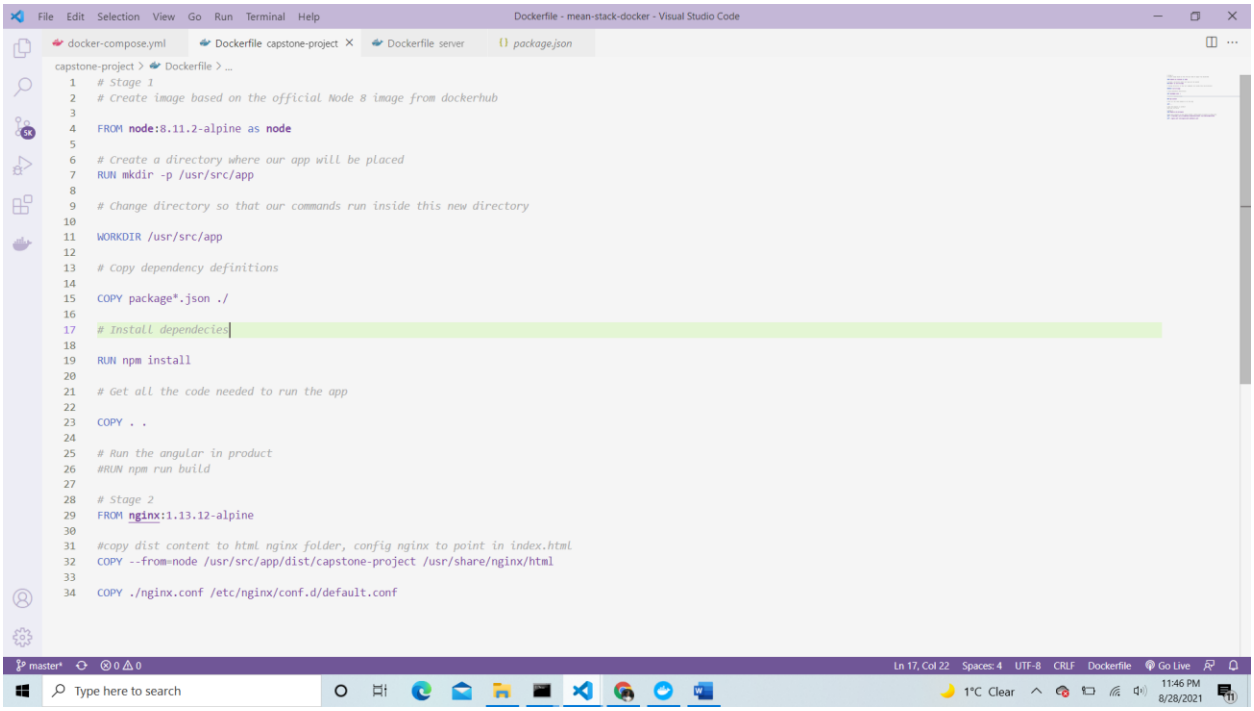
- Admin can edit an existing product.



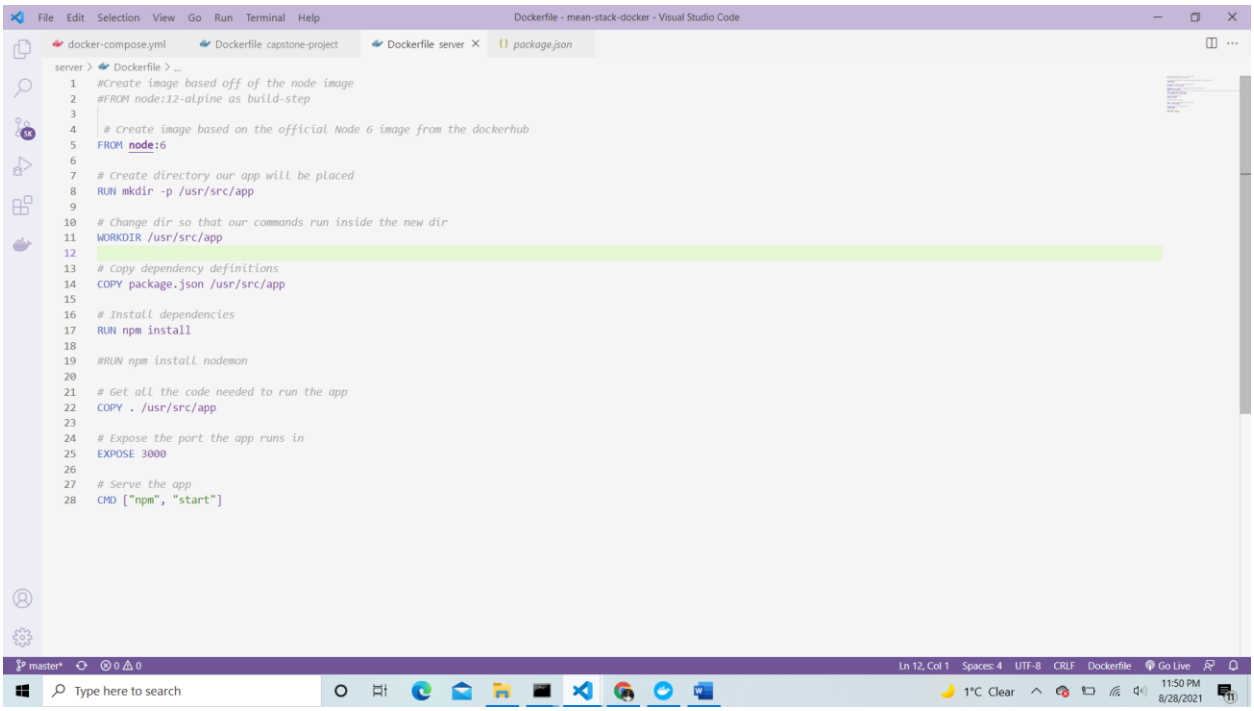
3. **<u>Containerizing Frontend (Angular) & Backend(express) Application using Docker and Dockerhub.</u>**
    - Steps taken to containerize angular application
        1. Dockerized Angular capstone-project app deployed in nginx server.
        2. Dockerized the express server API– named ‘server’
        3. Use MongoDB container
        4. Docker compose
        5. Connecting the 3 Docker containers (angular, server and mongodb database)

    - Created a mean-stack-docker directory and copied capstone client project and express server to the folder.
    - For my angular application to run, I needed the image with NodeJS installed on it and add angular client project into the image and install all the dependencies.
    - Ran the angular project in production mode and copied the folder ‘/dist’ content from the WORKDIR to the nginx html
    - Now the project work in production mode deployed in nginx server (localhost:80)
    - The above instructions are to be written in our Dockerfile from the client side.

    - Dockerfile: mean-stack-docker/capstone/Dockerfile (frontend)



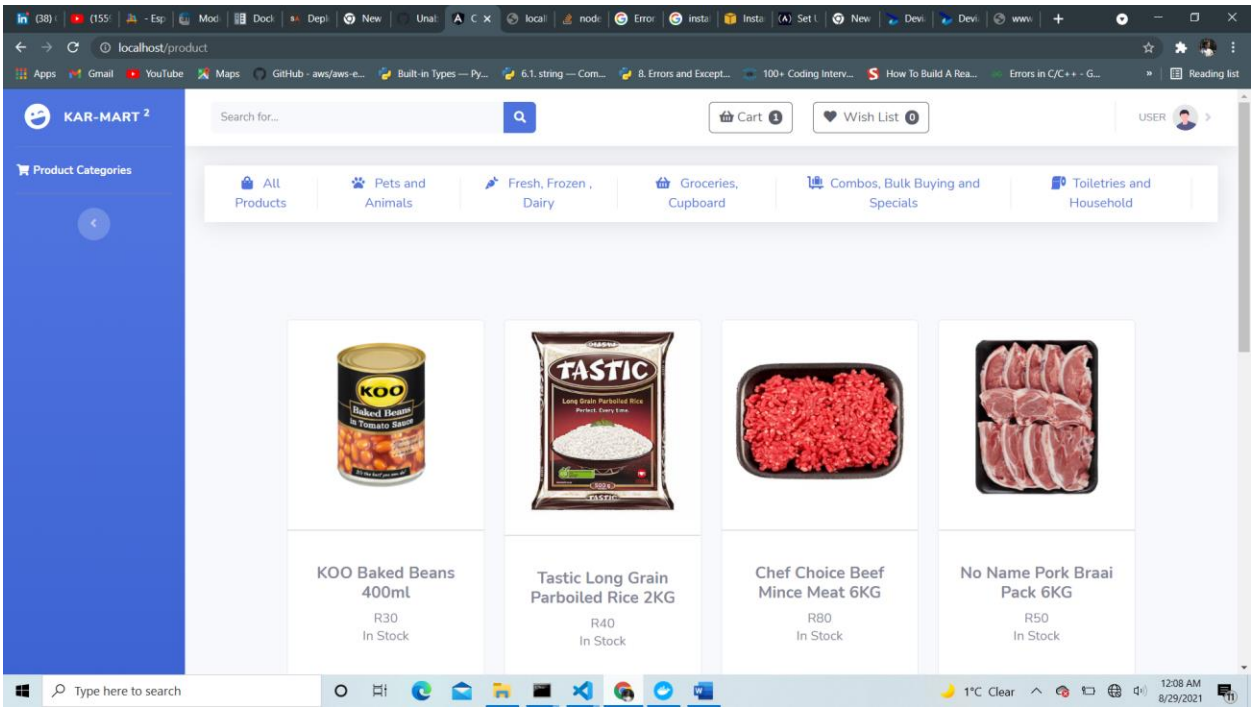
    - Created server directory under the root directory of mean-stack-docker.
    - Inside the mean-stack-docker/server/Dockerfile (backend).



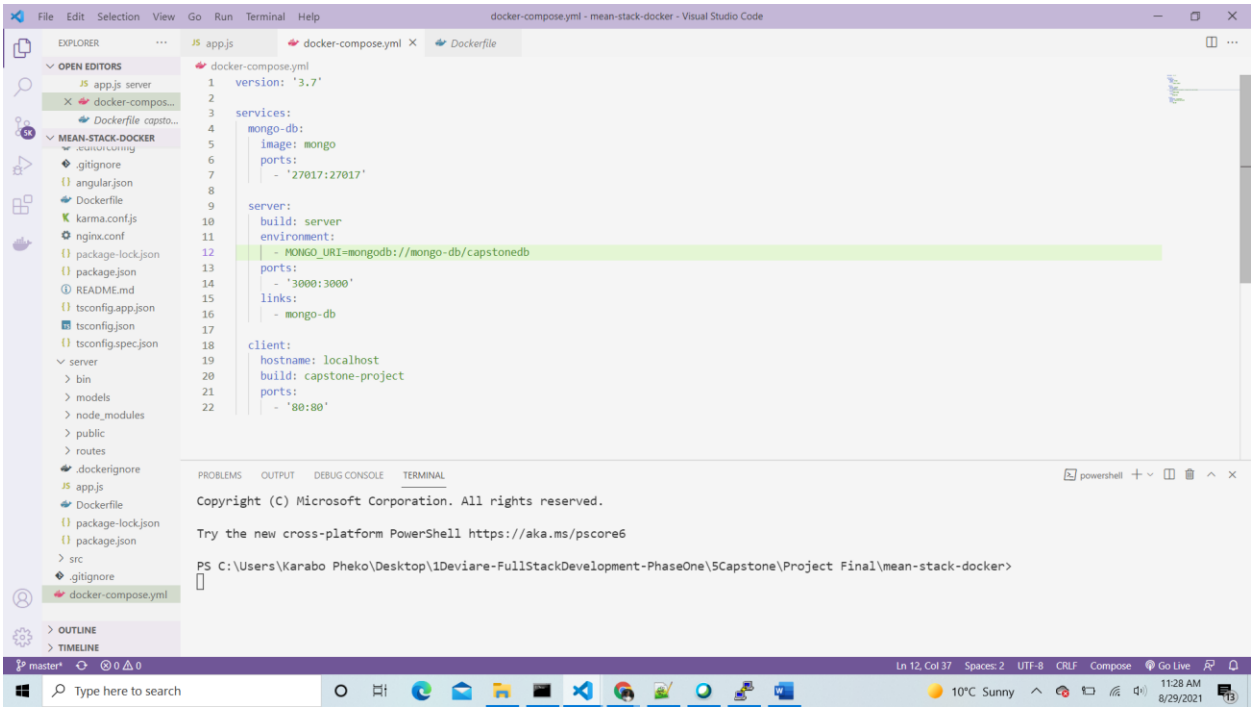
- Ran
- Docker build -t server – to check test if server image is working under port 80.
- Docker run -d --name server -p 3000:3000 server
- Then ran localhost:3000 on browser to test
- MongoDB container
- Assuming I have mongodb image already, I ran docker run -d –name mongodb -p 27017:27017 mongo

Docker Compose

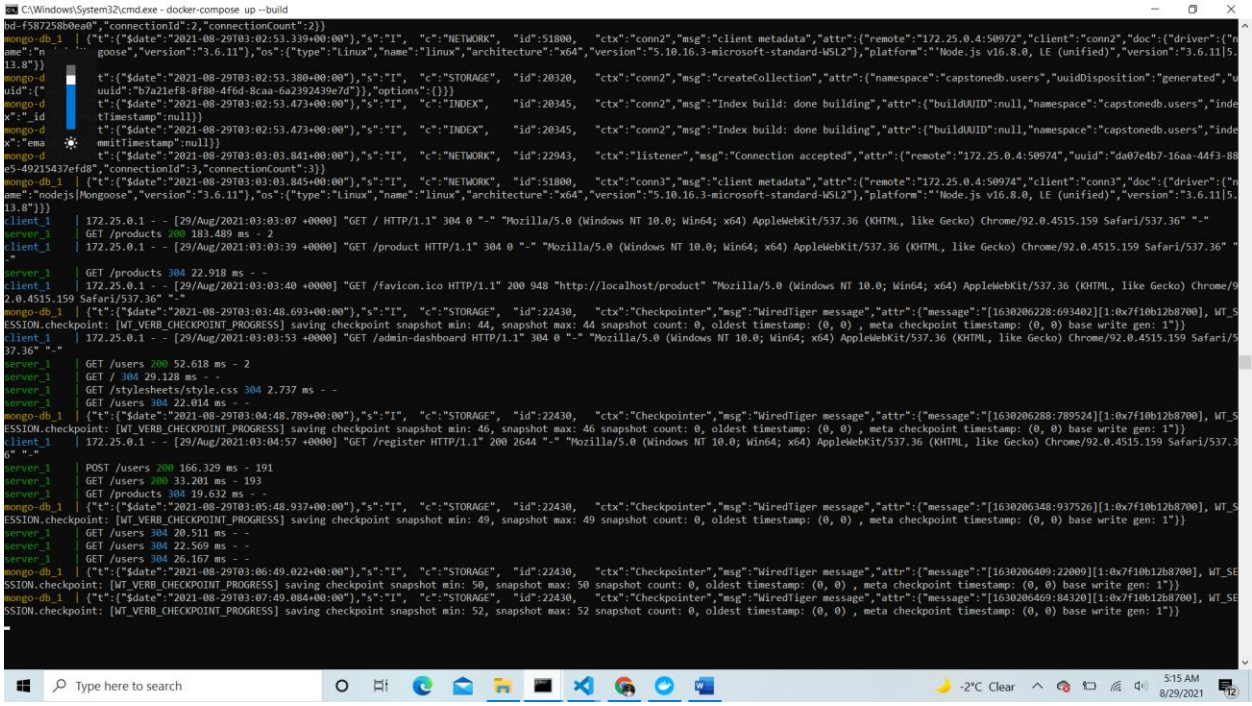
- Created docker-compose.yml file in the mean-stack root directory
- Ran *docker-compose build --cache*
- Ran *docker-compose up*
- Ran localhost:80 in browser and display the following page below.



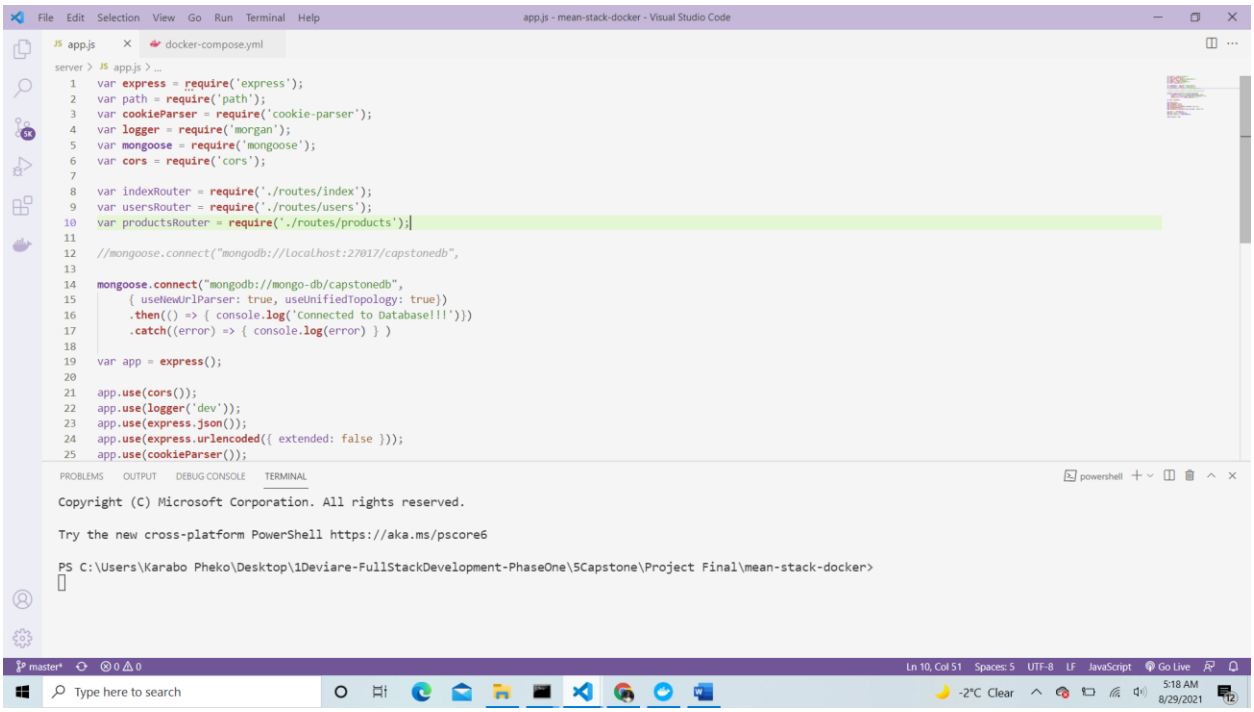
- Docker-compose.yml file



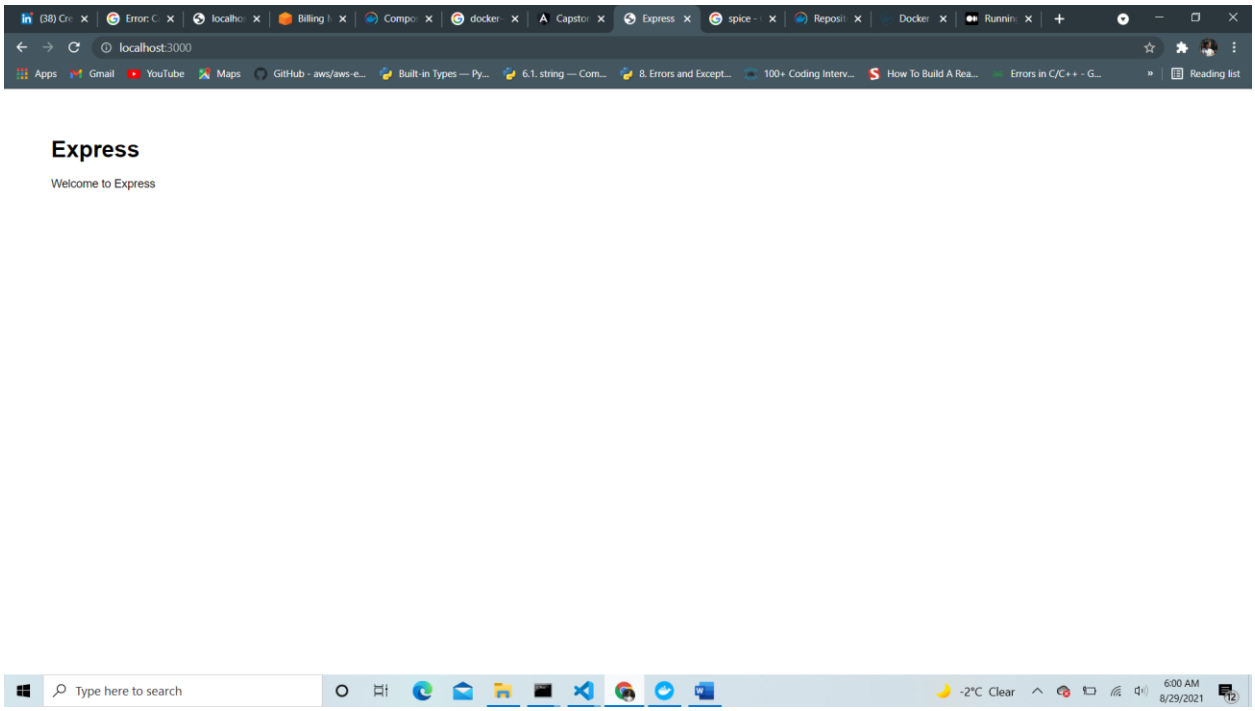
- Ran docker-compose up –build



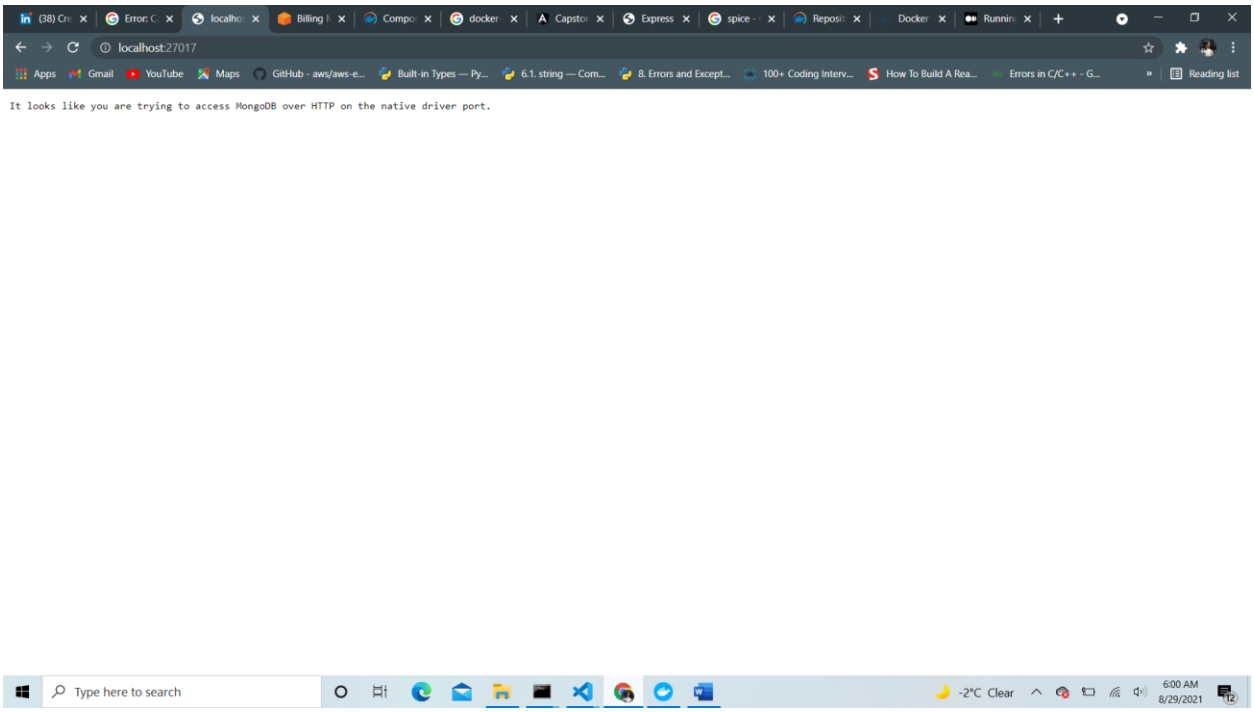
- Minor change in app.js file in server



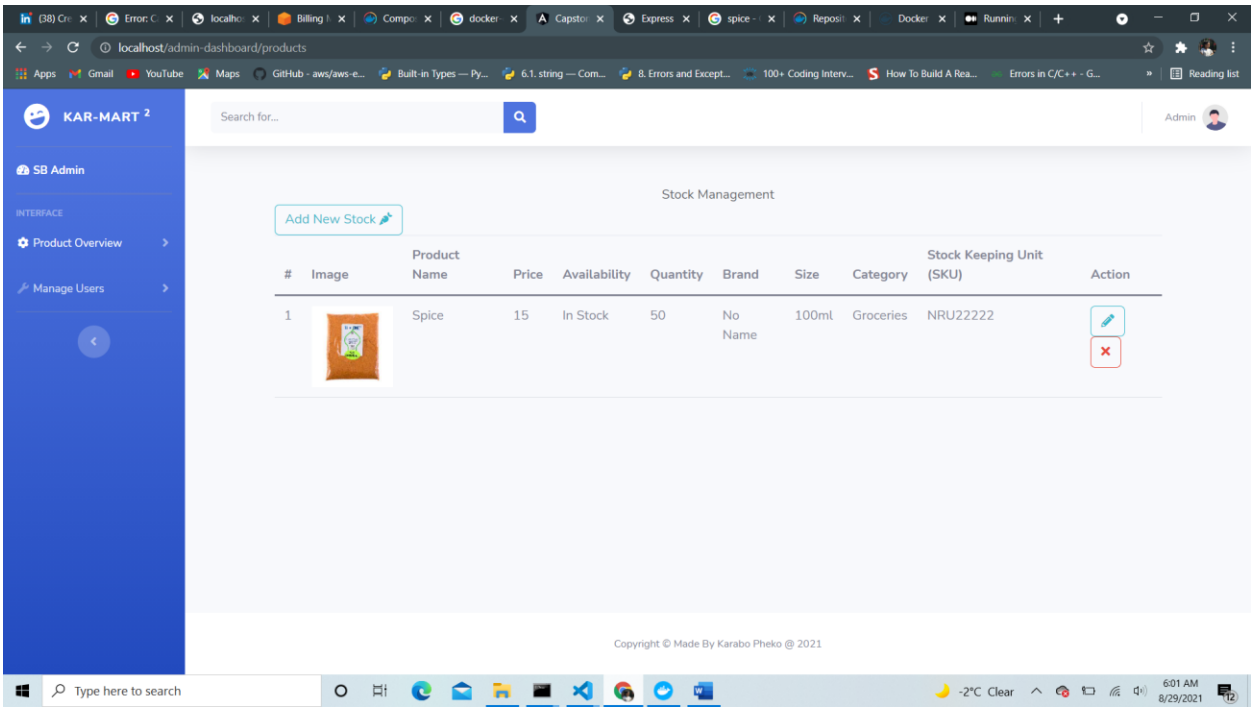
- **<u>Test that server is working</u>**



- **Test mongo image is running properly (localhost:27017)**



- **Finally test your application (localhost:80)**



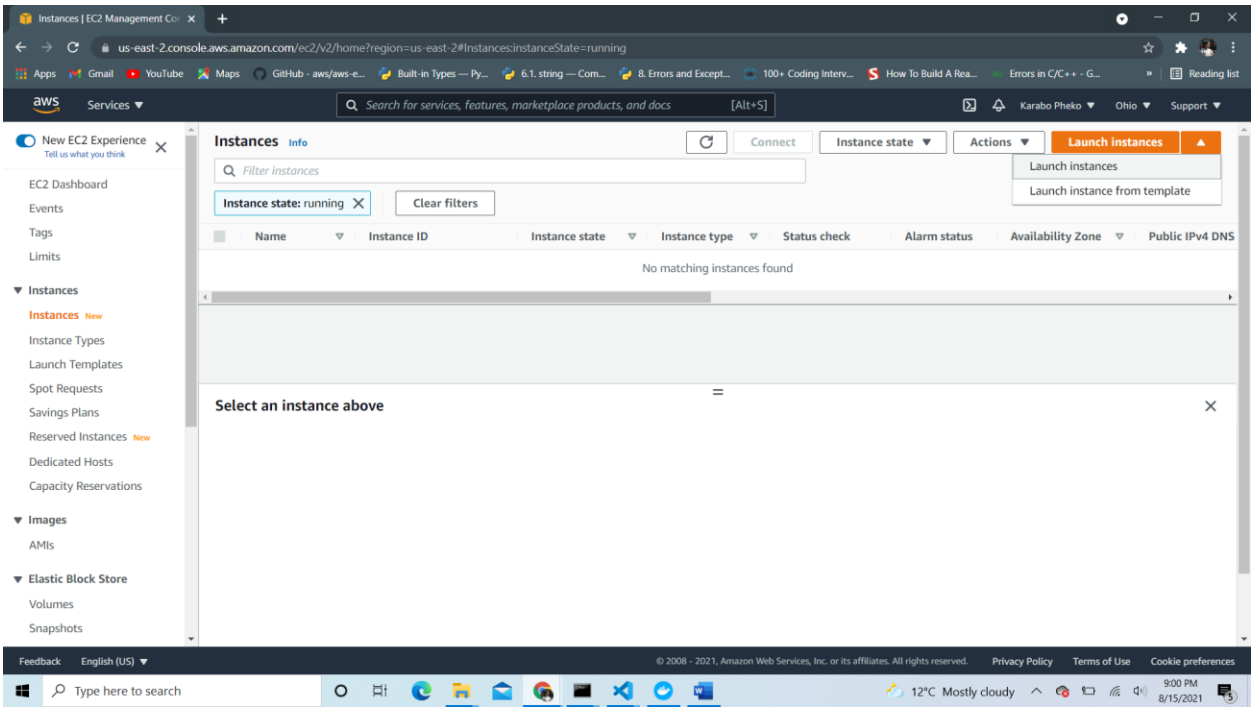
4. AMAZON WEB SERVICES

A. Setting up EC2.

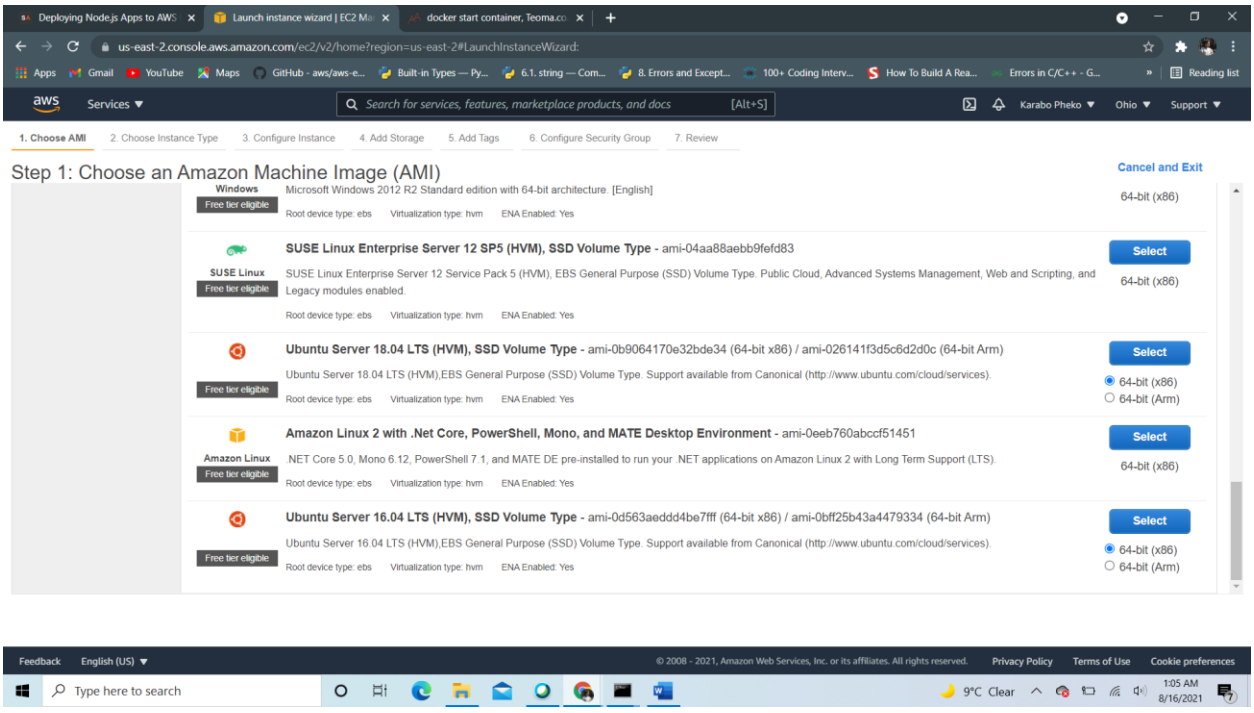
- Since application is containerized, we set an EC2 instance for the container to run on.
- Navigated to AWS and logged in with my credentials.

CREATE A NEW UBUNTU INSTANCE ON AWS EC2

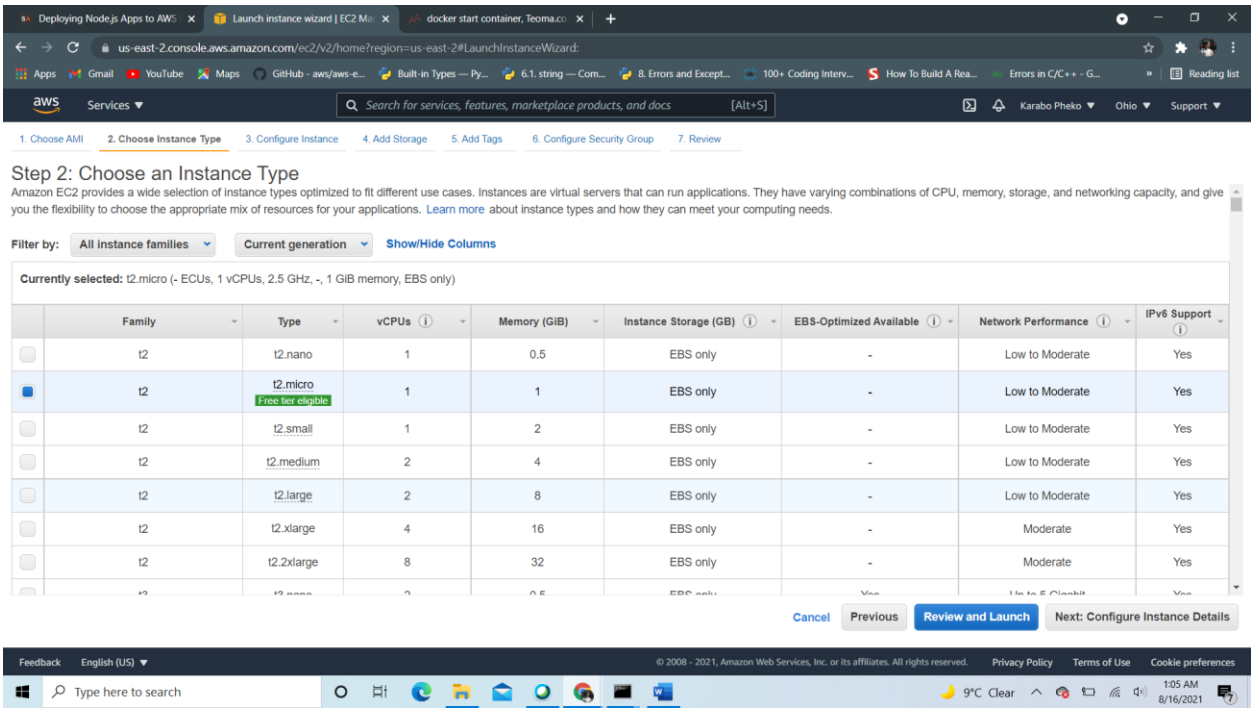
1. Clicked ‘Services’ on the dropdown menu and selected ‘EC2’. Clicked ‘Launch Instance’ button



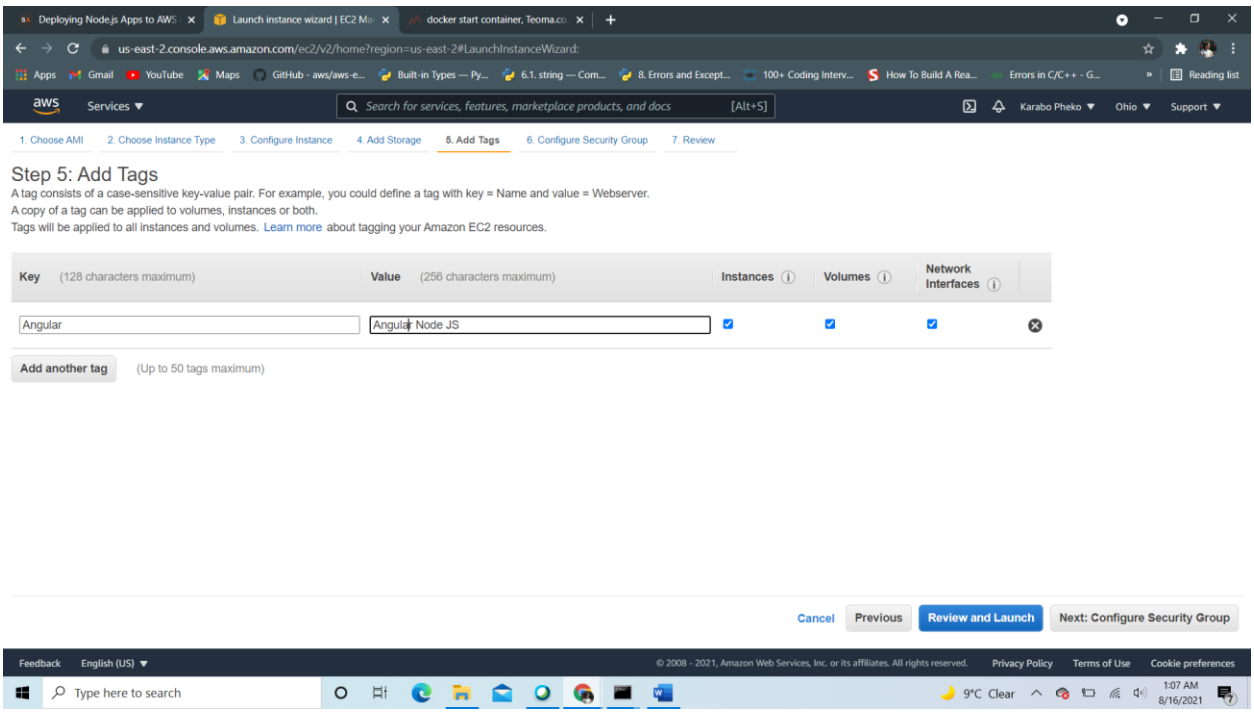
2. Choose AMI you want to work with, in my case, I chose free tier Ubuntu Server 18.04 Lifetime Support.



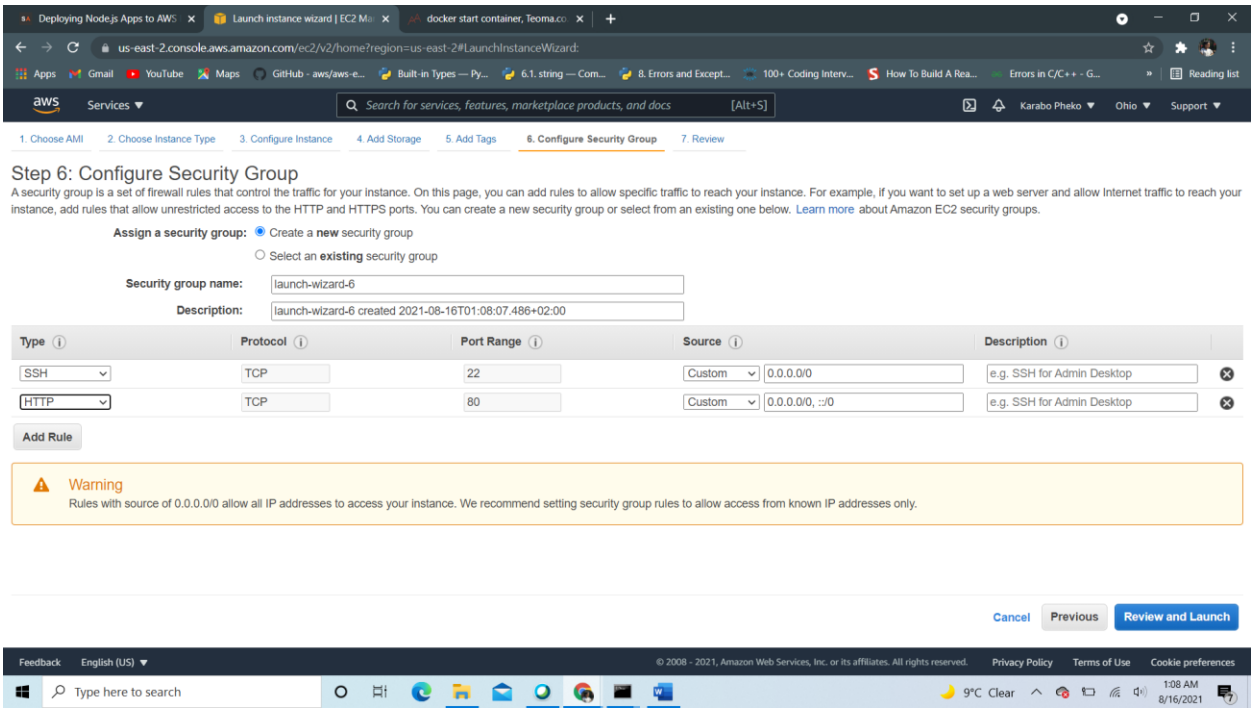
3. Choose Instance type – t2-micro free tier.



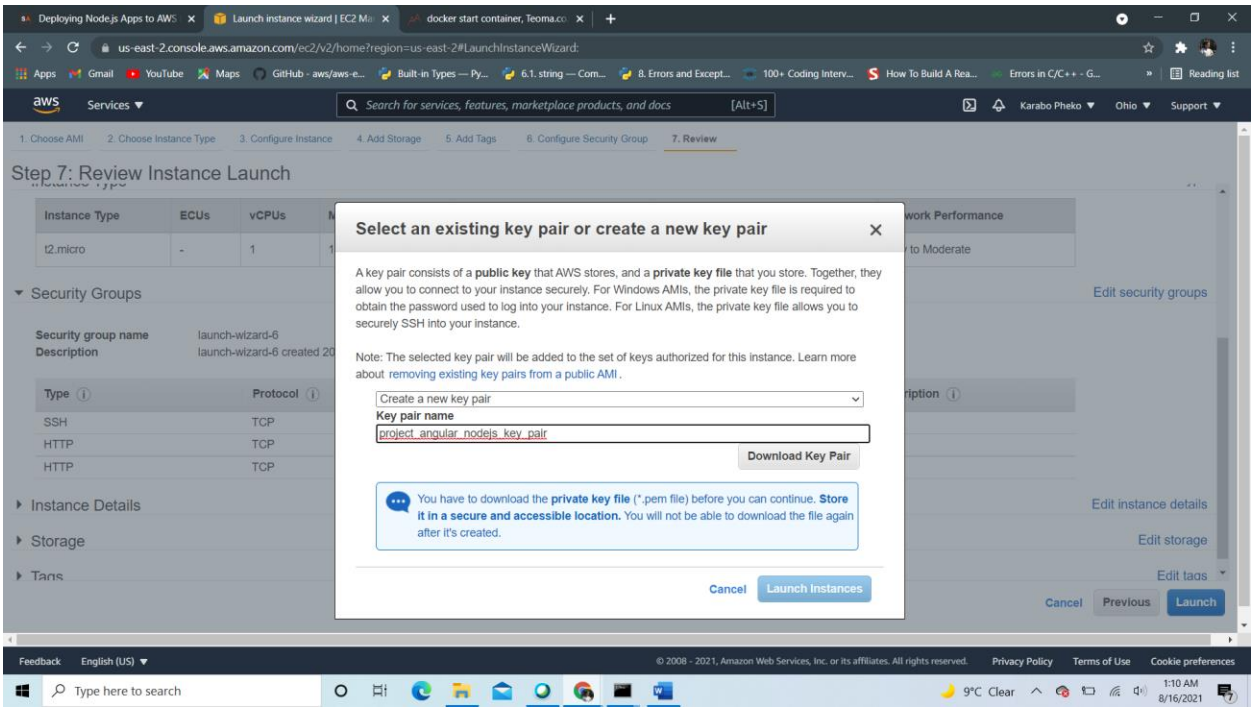
4. Add tags(Optional), in my case I added a Key=Angular and Value=Angular Nodejs



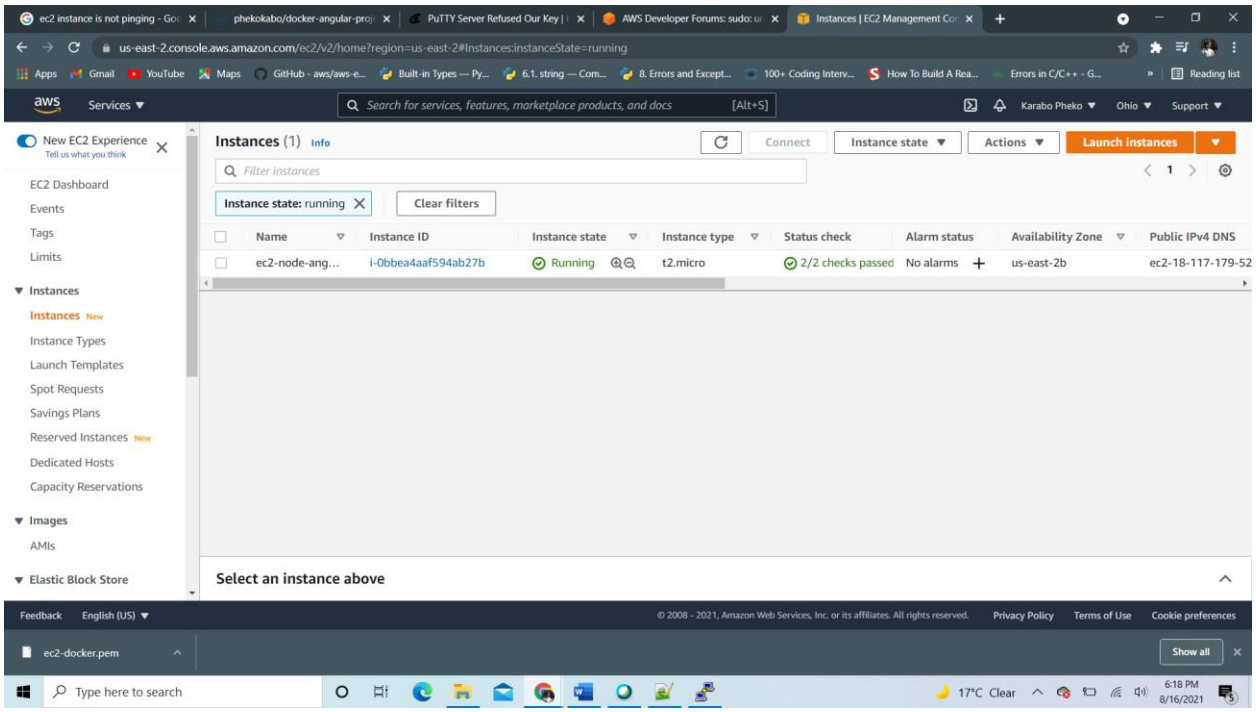
5. Added a new security group, added rule for type HTTP and exposed port number 80 so we can accept connections from the web.



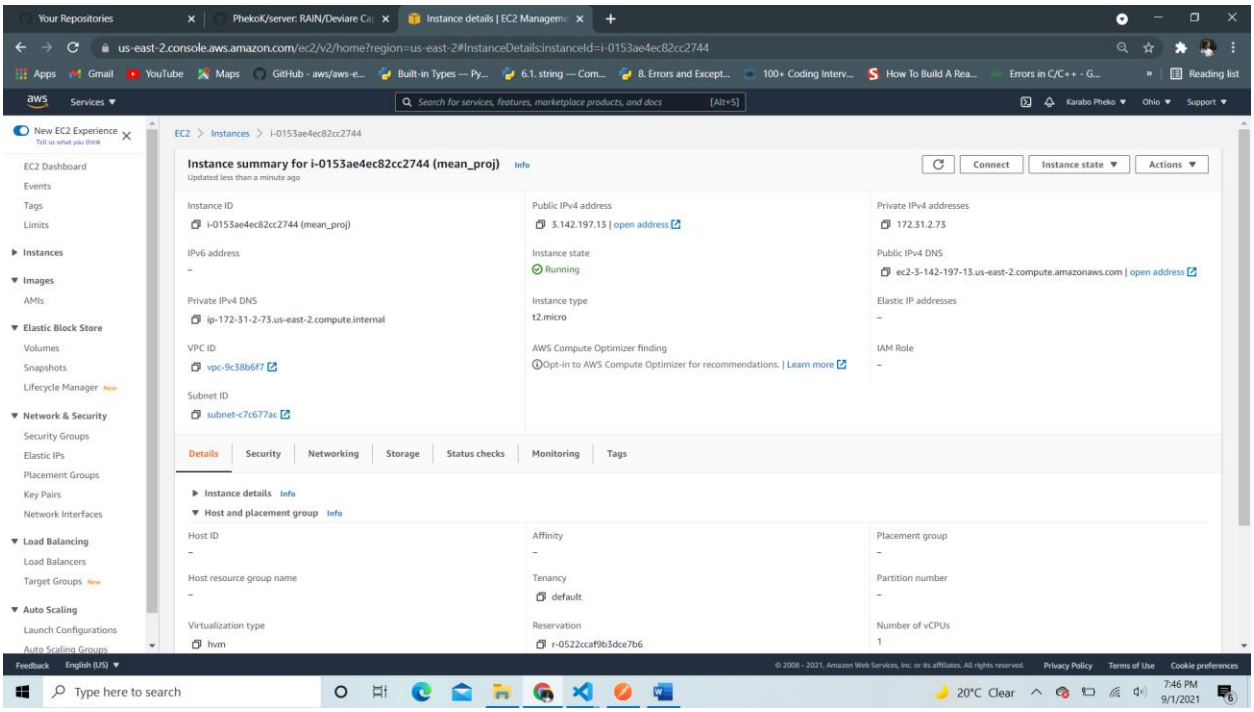
6. After that, clicked Review and Launch button and created a new key_pair to connect to my instance.



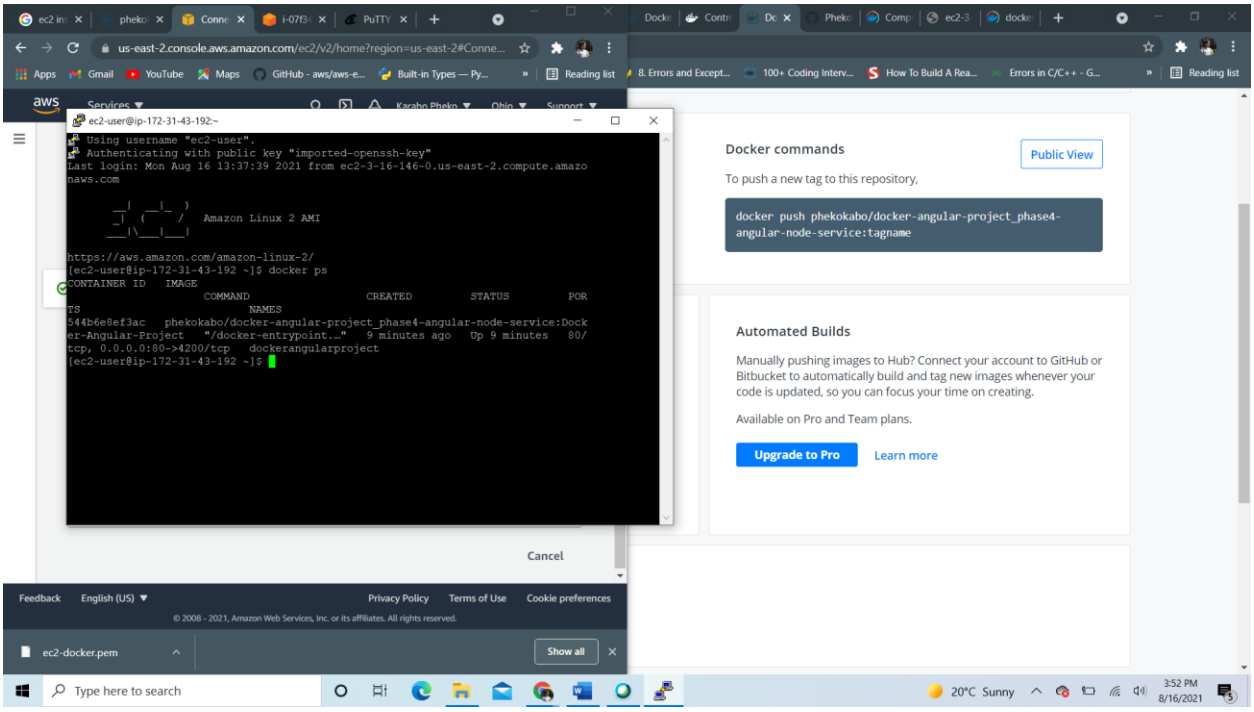
- Checked launched instances after reviewing



7. Launch EC2 instance using putty or virtual machine provided by AWS and SSH into the Amazon virtual machine by using the public IPv4 DNS detailed after I launched the instance.



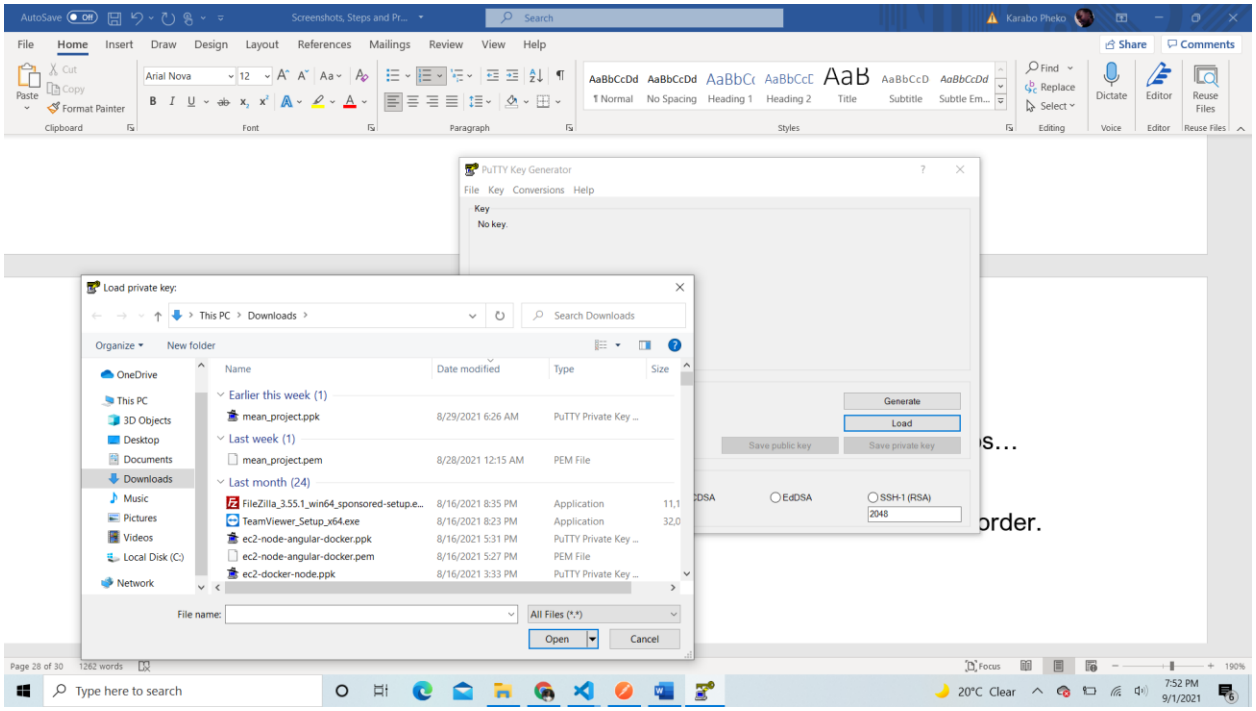
- Then the following instance is launched.

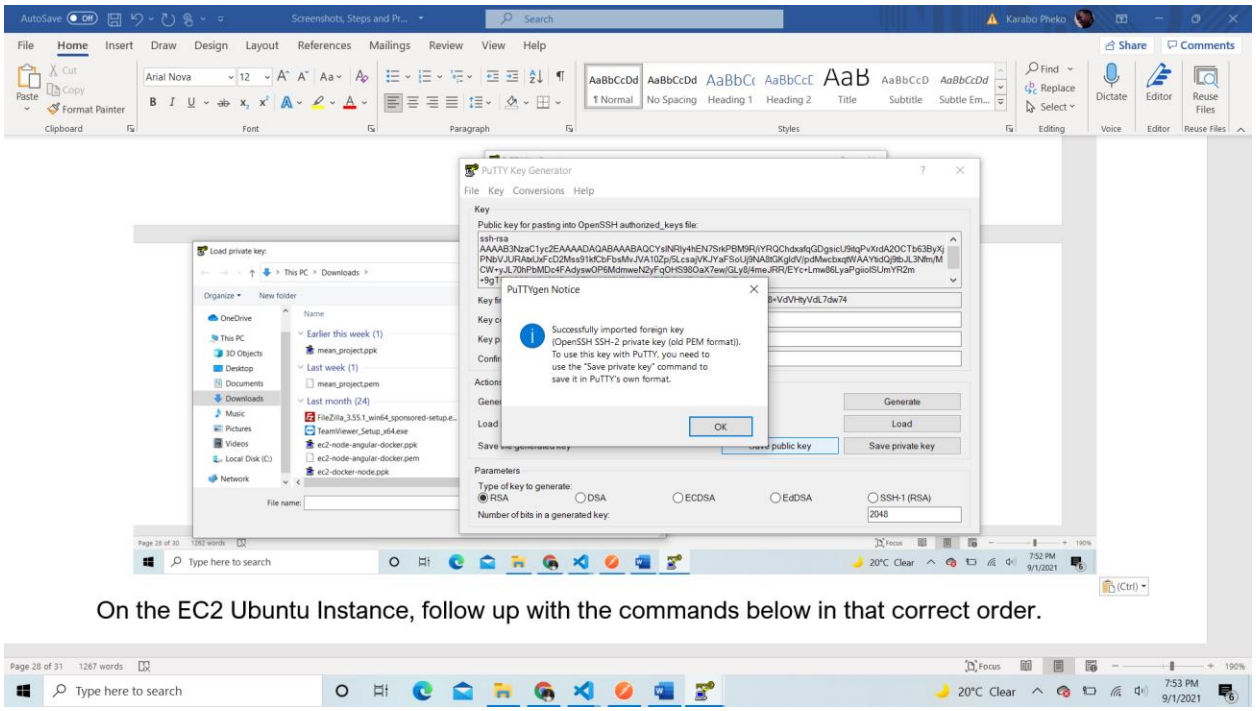


CONNECT TO UBUNTU INSTANCE EC2 VIA SSH

- Alternatively you could use the one provided by putty by the following steps…

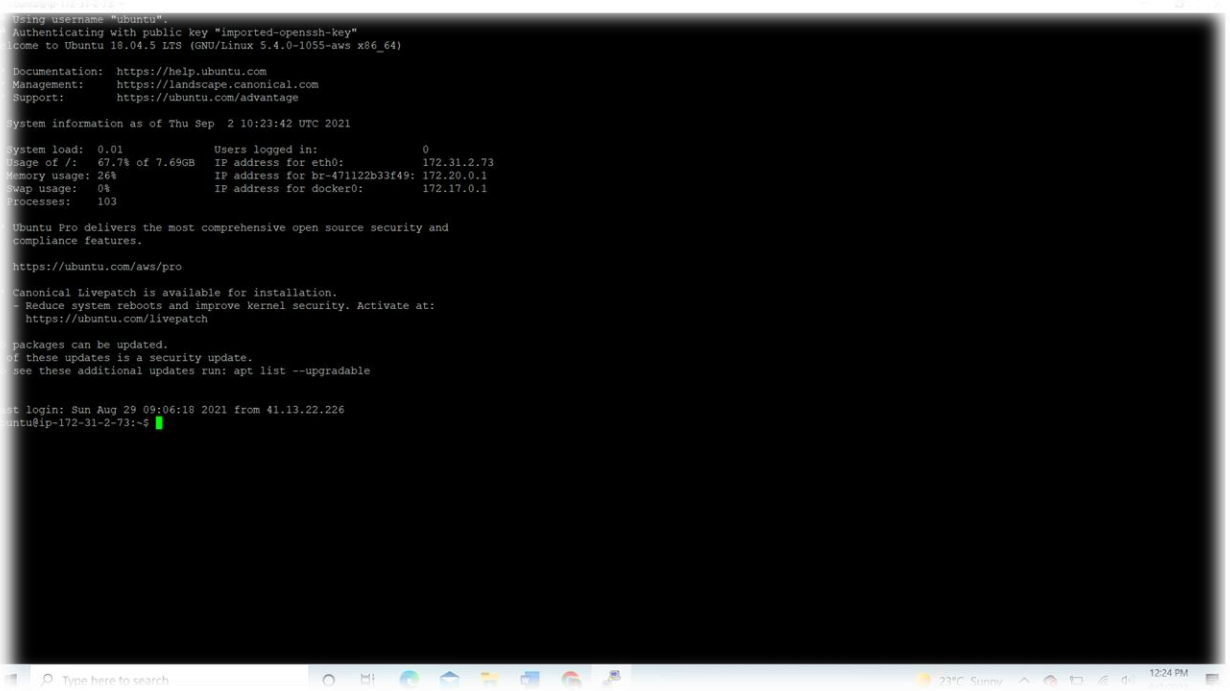
1. Open puTTygen and generate a private key, click load and add key with a .pem extension (Privacy Enhanced Mail) – used to sign me in in my SSH terminal to the virtual machine
2. Once selected, click ‘Save private key’ and your key is generated.
3. Once saved, use the key to connect to putty and open the virtual machine provided putty, by providing the IPv4 address as your address, the login username, i.e ubuntu, and the private key file generated by puTTygen for authentication with the .ppk extension and your instance is launched!





On the EC2 Ubuntu Instance, follow up with the commands below in that correct order.

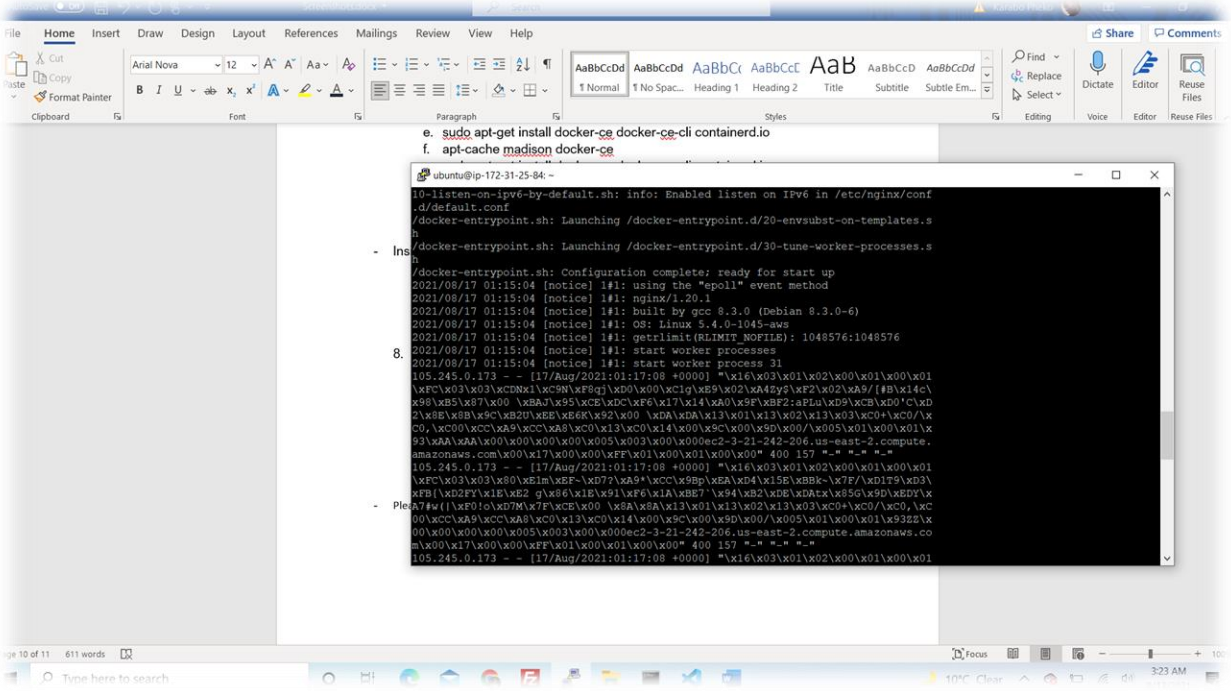
4. Once generated, head to putty to launch your instance and should look like something like this



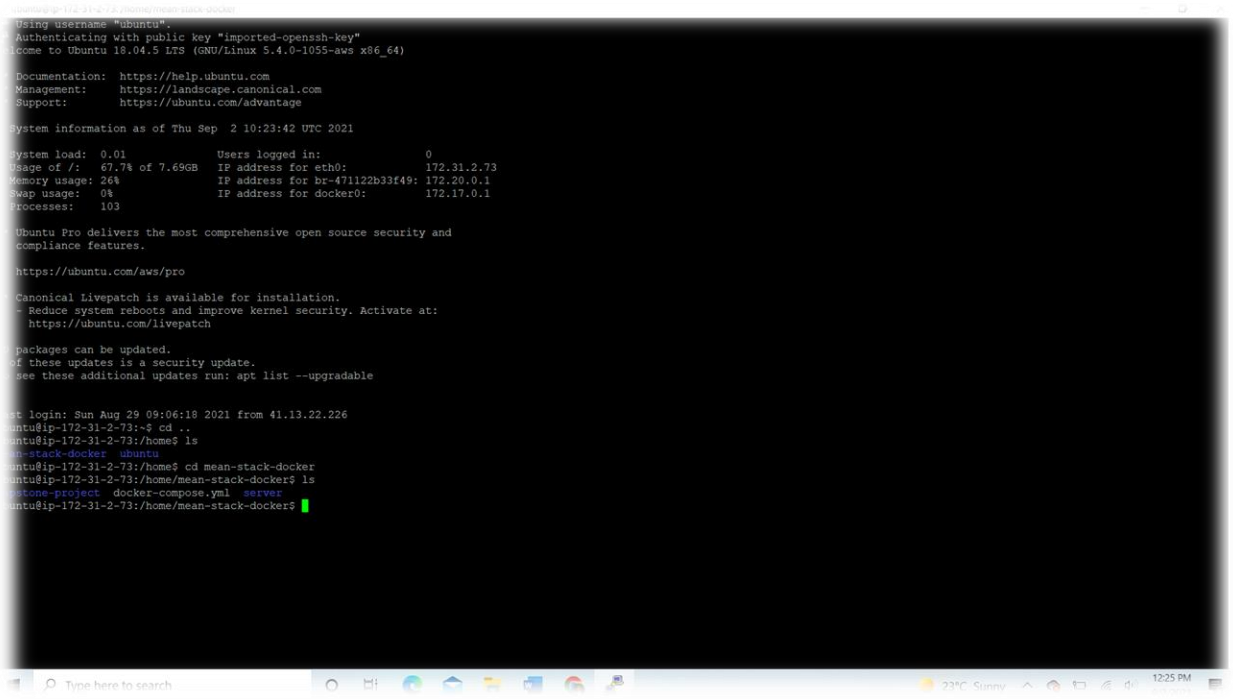
Setup Web Server with Node.js + MongoDB + NGINX

On the EC2 Ubuntu Instance, follow up with the commands below in that correct order.

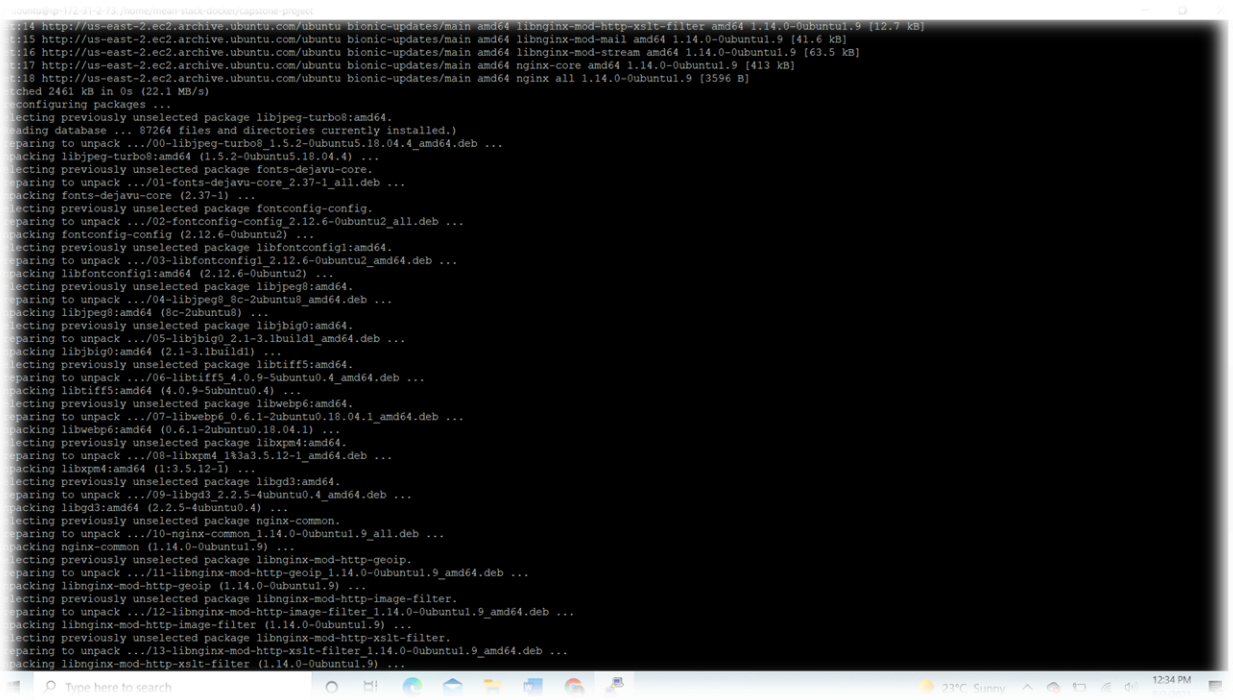
a. sudo apt-get update
b. sudo apt-get upgrade
c. sudo apt-get install \
   apt-transport-https \
   ca-certificates \
   curl \
   gnupg-agent \
   software-properties-common
d. curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
e. sudo apt-get install docker-ce docker-ce-cli containerd.io
f. apt-cache madison docker-ce
g. sudo apt-get install docker-ce docker-ce-cli containerd.io
h. sudo apt install docker.io
i. sudo apt install docker-compose



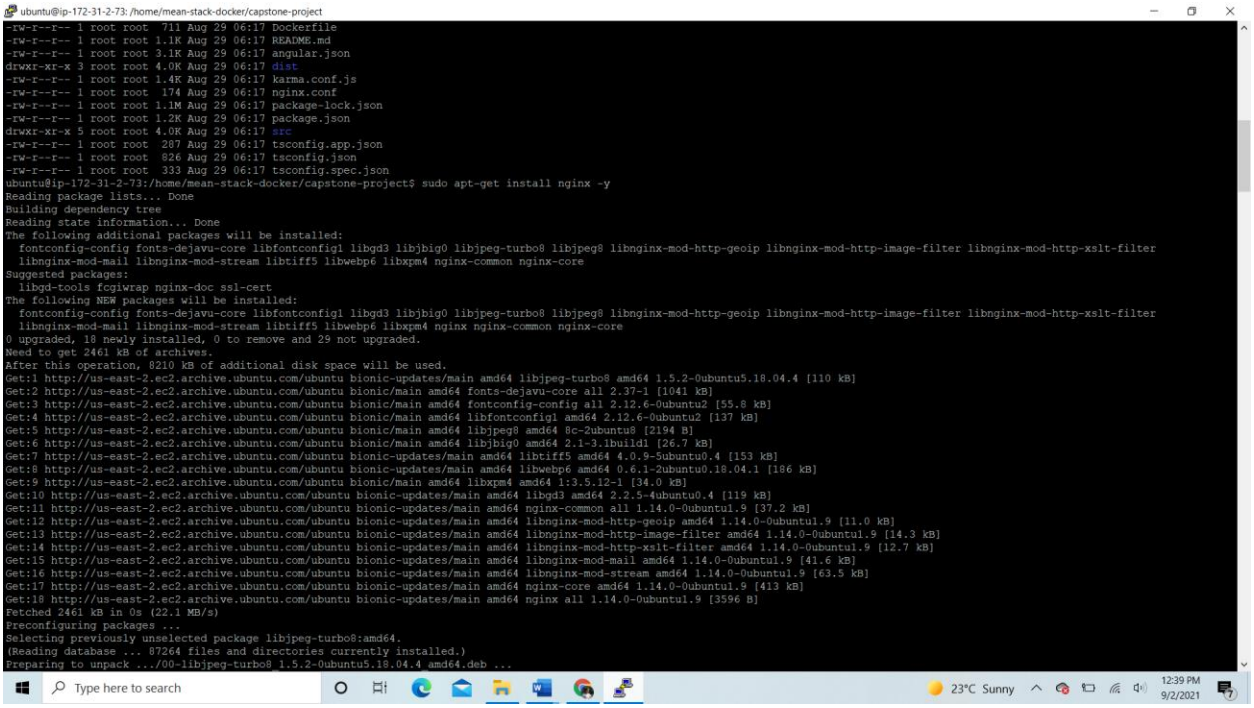
- git clone your backend and fronted application to your instance like so:



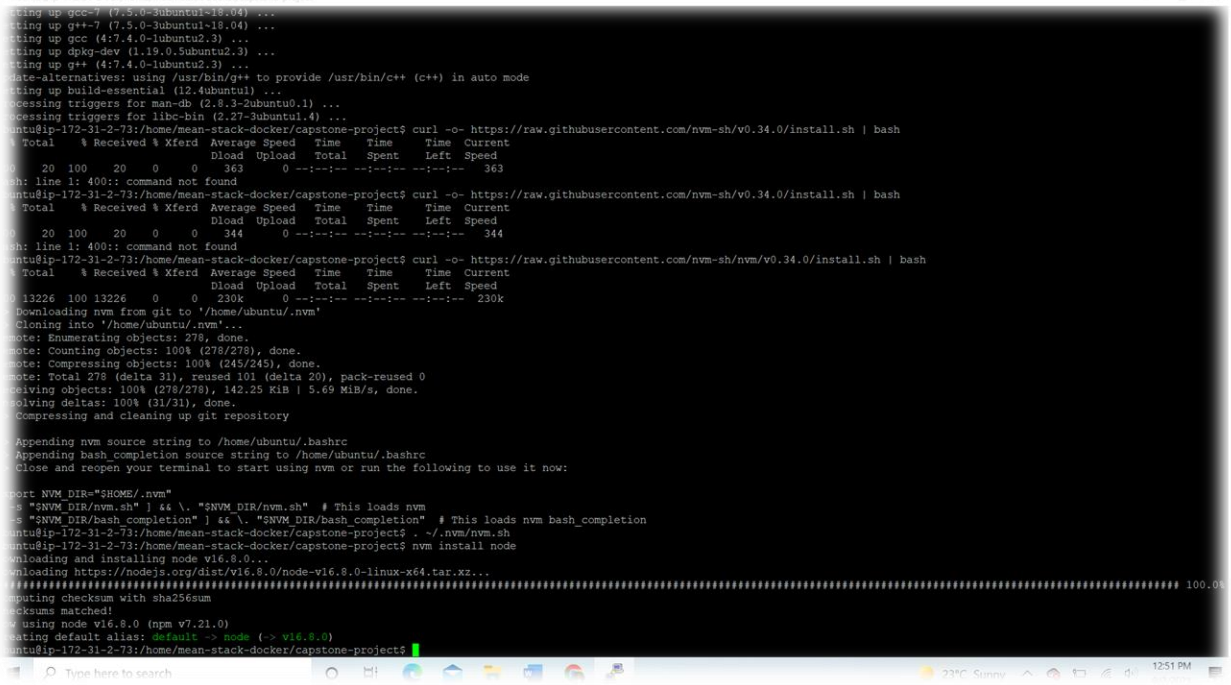
- cd to capstone-project (frontend) and install nginx server



- Run ‘sudo apt-get install build-essential libssl-dev’ to install node



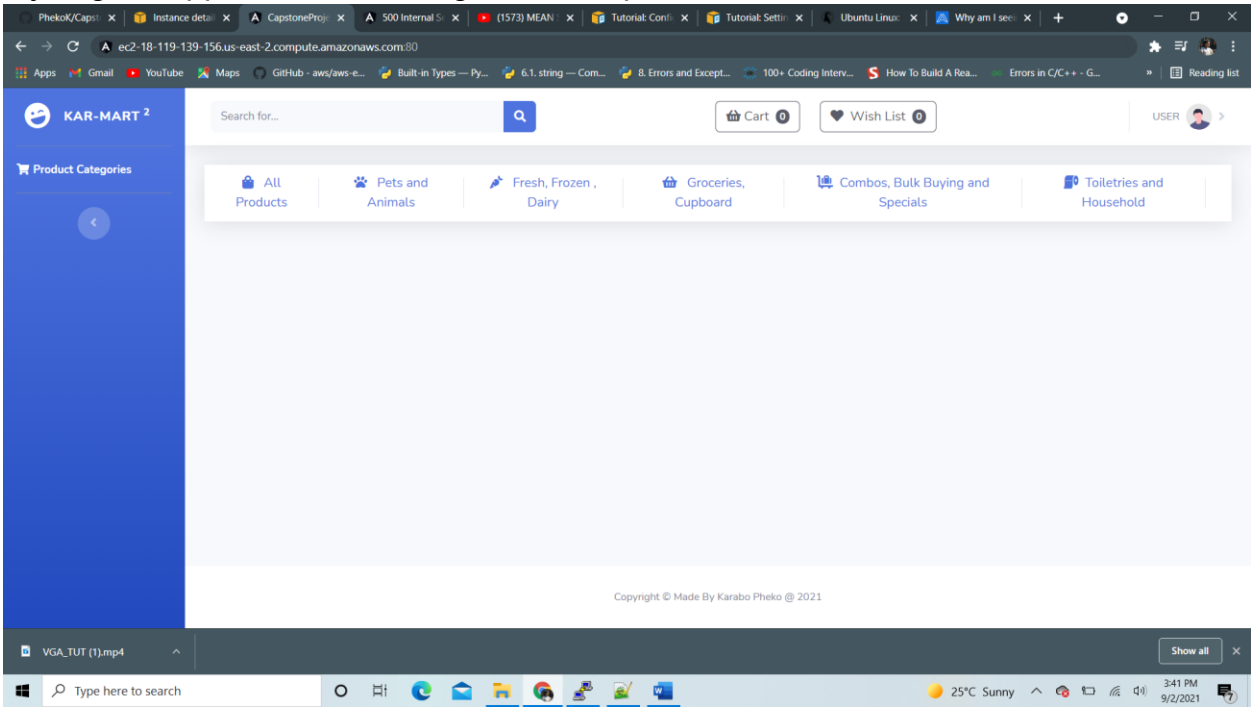
- Run ‘curl -o- https://raw.githubusercontent.com/nvm-sh/v0.34.0/install.sh | bash’ to install node version manager.
- Ran ‘. ~/.nvm/nvm.sh’ to activate nvm
- Ran ‘nvm install node’ to install node where we will use node version 16.8.0



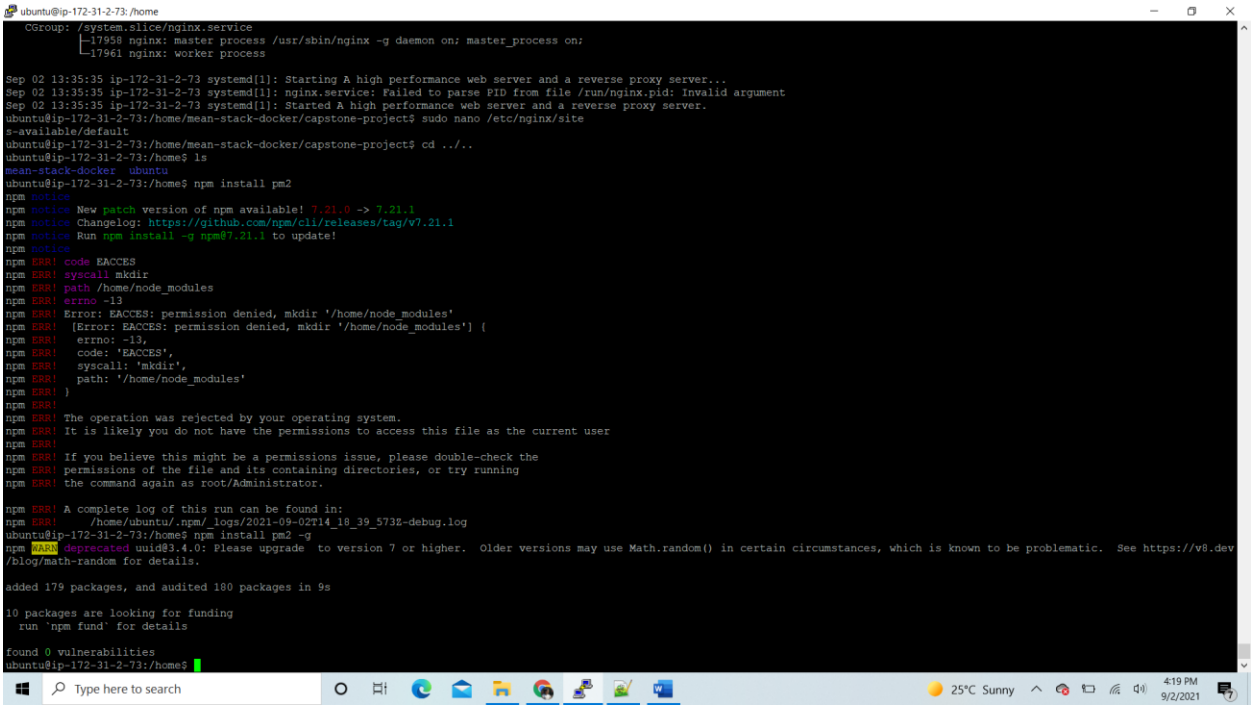
- Created /etc/nginx/sites-available/default file and modified it like so:

```
ADME.md     karma.conf.js  package.json       tsconfig.json
ular.json  nginx.conf     src                tsconfig.spec.json
untu@ip-172-31-2-73:/home/mean-stack-docker/capstone-project$ sudo nano /etc/nginx/sites-ava
able/default
untu@ip-172-31-2-73:/home/mean-stack-docker/capstone-project$ sudo nano /etc/nginx/site
vailable/default
untu@ip-172-31-2-73:/home/mean-stack-docker/capstone-project$ cat etc/nginx/sites-avail
le/default
t etc/nginx/sites-available/default: No such file or directory
untu@ip-172-31-2-73:/home/mean-stack-docker/capstone-project$ cat /etc/nginx/sites-avai
ble/default
rver {
    listen 80;
    server_name ec2-18-119-139-156.us-east-2.compute.amazonaws.com;
    index index.html;
    root /home/mean-stack-docker/capstone-project;
    location / {
            expires -1;
            add_header Cache-Control 'no-store, no-cache, must-revalidate, proxy-revali
te, max-age-0';
            try_files try_files $uri $uri/ /index.html;
    }
    location /product/ {
            proxy_pass http://localhost:3000;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Connection 'upgrade';
            proxy_set_header Host $host;
            proxy_cache_bypass $http_upgrade;
            proxy_redirect off;
    }

untu@ip-172-31-2-73:/home/mean-stack-docker/capstone-project$ sudo systemctl status nginx
nginx.service - A high performance web server and a reverse proxy server
 Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
 Active: active (running) since Thu 2021-09-02 13:35:35 UTC; 31min ago
   Docs: man:nginx(8)
Process: 17483 ExecStop=/sbin/start-stop-daemon --quiet --stop --retry QUIT/5 --pidfile /run/nginx.pid (code=exited, status=0/SUCCESS)
Process: 17954 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
Process: 17941 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
ain PID: 17958 (nginx)
  Tasks: 2 (limit: 1140)
 CGroup: /system.slice/nginx.service
         ├─17958 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
         └─17961 nginx: worker process

p 02 13:35:35 ip-172-31-2-73 systemd[1]: Starting A high performance web server and a reverse proxy server...
p 02 13:35:35 ip-172-31-2-73 systemd[1]: nginx.service: Failed to parse PID from file /run/nginx.pid: Invalid argument
p 02 13:35:35 ip-172-31-2-73 systemd[1]: Started A high performance web server and a reverse proxy server.
untu@ip-172-31-2-73:/home/mean-stack-docker/capstone-project$
```

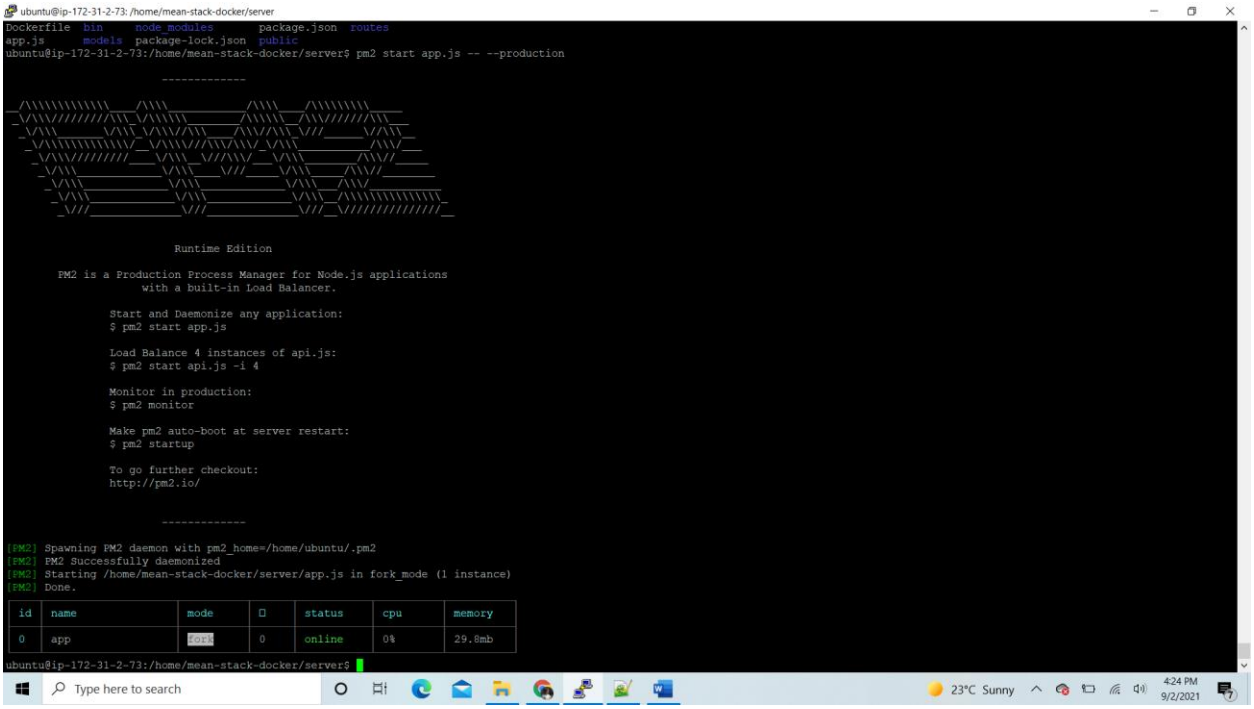
- Checked if nginx is running, and opened browser with the link:
- My angular application is running without express server.
-



- Installed ‘npm install pm2 -g” to make sure our node script is running



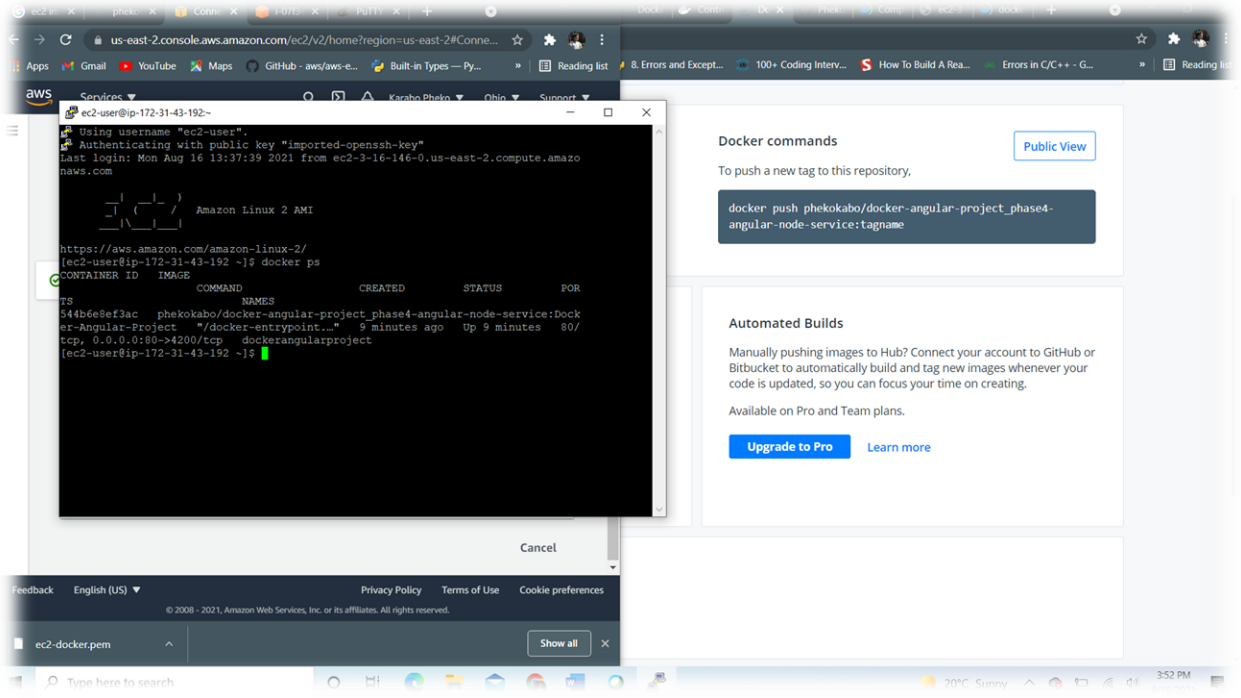
- Run pm2 in production



- Run ‘sudo systemctl status nginx’ to check whether nginx server is running

- Run ‘npm start’ in vs code to start express server and you have the website running like the pictures below.

8. Once connected to ec2 instance and everything up and running, you should be able to access the website using the public ipv4 dns provided below:

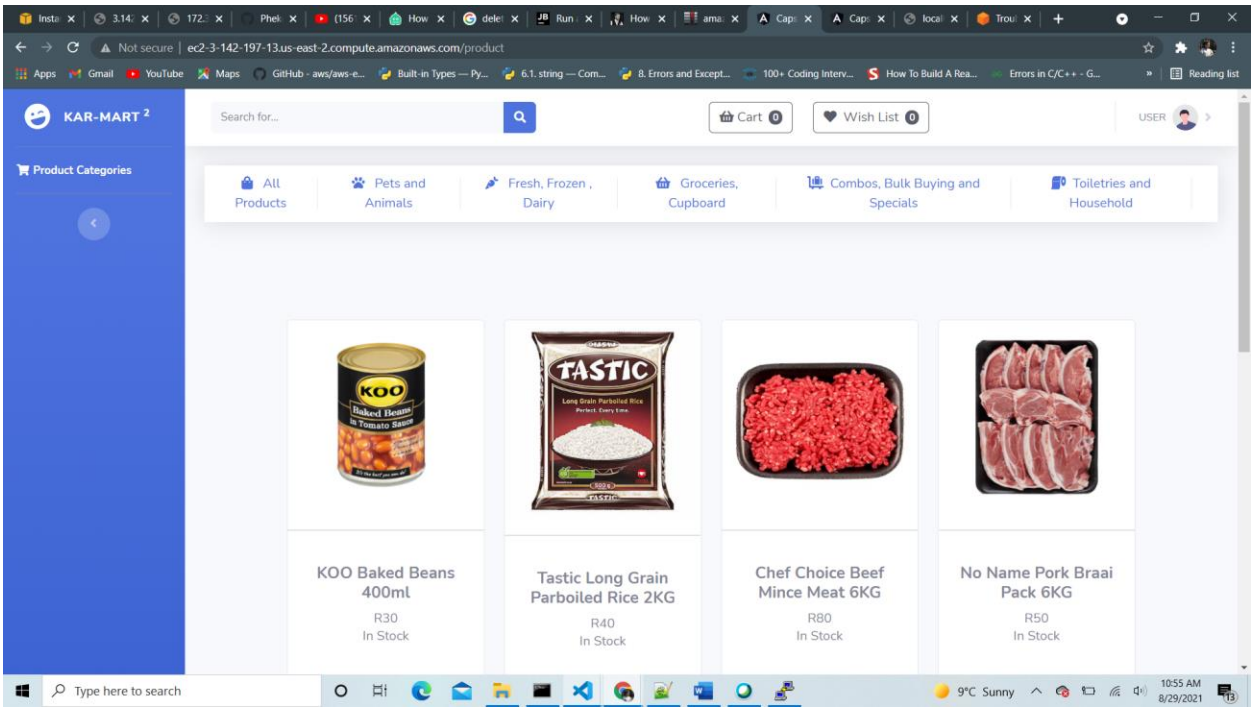


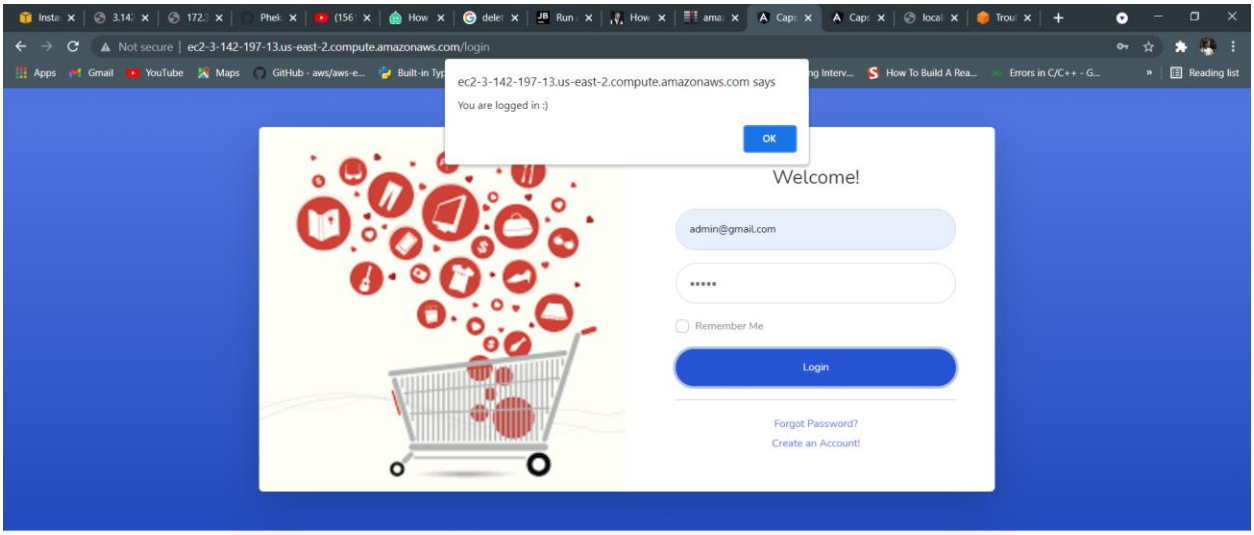
Public IPv4 DNS:

- Use the link below to log into the application

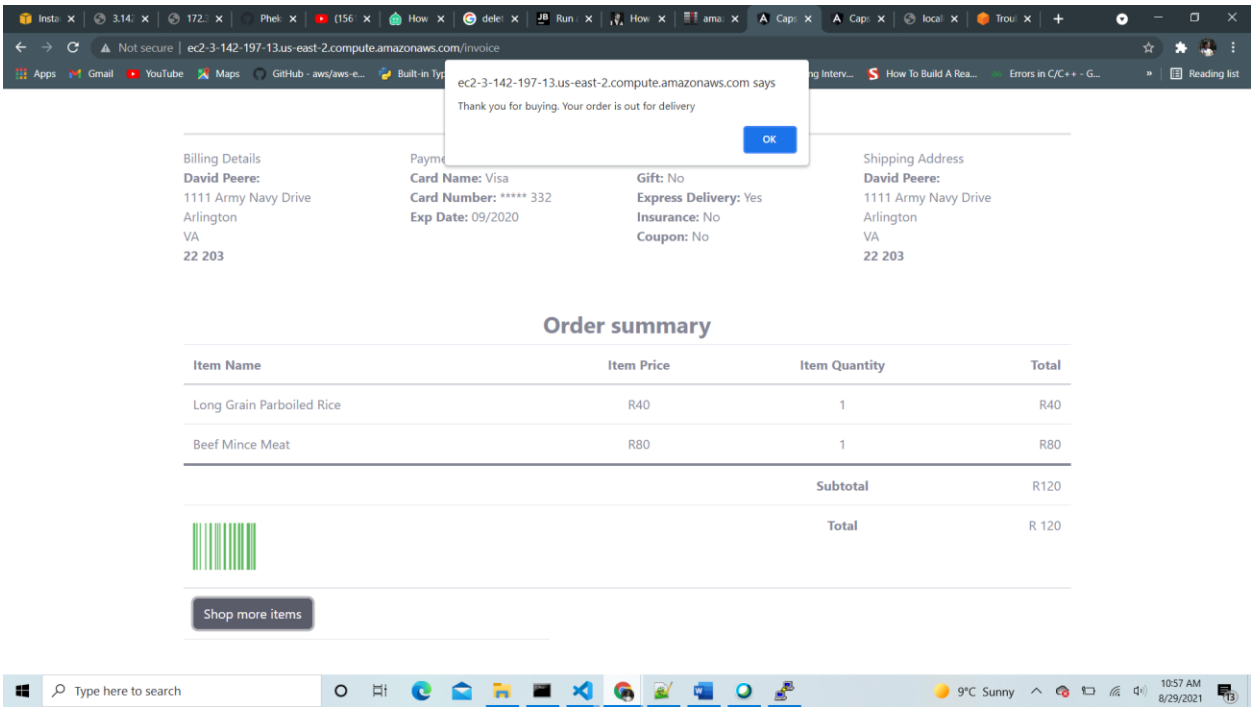
http://ec2-18-119-139-156.us-east-2.compute.amazonaws.com:80 , this lands you to the product page

- The website is now running under EC2 instance


Not secure | ec2-3-142-197-13.us-east-2.compute.amazonaws.com/product
KAR-MART 2
Search for...
Cart 0
Wish List 0
USER
Product Categories
All Products
Pets and Animals
Fresh, Frozen , Dairy
Groceries, Cupboard
Combos, Bulk Buying and Specials
Toiletries and Household
KOO Baked Beans 400ml
R30
In Stock
Tastic Long Grain Parboiled Rice 2KG
R40
In Stock
Chef Choice Beef Mince Meat 6KG
R80
In Stock
No Name Pork Braai Pack 6KG
R50
In Stock
Type here to search
9°C Sunny
10:55 AM
8/29/2021


Not secure | ec2-3-142-197-13.us-east-2.compute.amazonaws.com/login
ec2-3-142-197-13.us-east-2.compute.amazonaws.com says
You are logged in :)
OK
Welcome!
admin@gmail.com
Remember Me
Login
Forgot Password?
Create an Account!


Type here to search
9°C Sunny
10:56 AM
8/29/2021


Not secure | ec2-3-142-197-13.us-east-2.compute.amazonaws.com/invoice
ec2-3-142-197-13.us-east-2.compute.amazonaws.com says
Thank you for buying. Your order is out for delivery
OK
Billing Details
David Peere:
1111 Army Navy Drive
Arlington
VA
22 203
Card Name: Visa
Card Number: ***** 332
Exp Date: 09/2020
Gift: No
Express Delivery: Yes
Insurance: No
Coupon: No
Shipping Address
David Peere:
1111 Army Navy Drive
Arlington
VA
22 203
Order summary
Item Name
Item Price
Item Quantity
Total
Long Grain Parboiled Rice
R40
1
R40
Beef Mince Meat
R80
1
R80
Subtotal
R120
Total
R 120
Shop more items
Type here to search
9°C Sunny
10:57 AM
8/29/2021