FULL STACK – II

PROJECT REPORT (2020-2021)

ON

“E-Shopping Website”

Department of Computer Engineering

& Application

Institute of Engineering & Technology

GLA University,

Mathura- 281406.

Submitted To:

Mr. Pankaj Kapoor

Submitted By:

Ashutosh Chauhan

Pranav Tomar

Rishabh Garg

Shailja Tripathi

ACKNOWLEDGEMENT

We take this opportunity to thank all those who have helped us in completing the project successfully.

We would like to express our gratitude to Mr. Pankaj Kapoor, who as our guide/mentor provided us with every possible support and guidance throughout the development of project. This project would never have been completed without his constant encouragement and support.

Our heartiest thanks to Dr. (Prof). **Anand Singh Jalal**, Head of Dept., Department of CEA for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal.

Acknowledgement to any other faculty member Mr. **Pankaj Kapoor,** Assistant Professor. We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

DECLARATION

I/We, hereby declare that the work which is being presented in the Industrial Training "**E- SHOPPING"** in partial fulfilment of the requirements for Industrial Training viva voce, is an authentic record of our own work carried under the supervision of Pankaj Kapoor.

Ashutosh Chauhan(181500148) Signature of Candidate:

Pranav Tomar (181500475) Signature of Candidate:

Rishabh Garg (181500562) Signature of Candidate:

Shailja Tripathi (181400648) Signature of Candidate:

Course: B.Tech. (Computer Science & Engineering) Year: 3rd

Semester: VI

ABSTRACT

This report contains detailed information about each and every single process, during the development of **"E-Shopping website"** by us. In this report, all the types of development procedures are mentioned and the ones being used in the industry are especially distinguished. A brief study on the MEAN stack which is a high-level Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

Contents

Chapter 1. Introduction

1.1 Motivation

We all know that the online shopping field is growing and there are lots of soft wares available to provide these shopping services but not that type of software which can provide the cheap and best authenticated products directly from the manufacturers with no external interference.

1.2 Overview

The main purpose of this project is to create an Online product selling Website that allows users to buy item based on price, quantity and its authenticity. The selected items are displayed in an attractive format and the buyer can simply buy those items white sitting at home. The Administrator will have additional functionalities when compared to the common user.

E-Shopping website is an online web application where the customer can purchase items online, through a web browser.

The user can login using his account details or new customers can create an account very quickly. The items are divided into many categories based on categories based on gender for simplicity. The customer also has the facility to share their feedback. The Administrator is notified by these feedbacks with the help of an e-mail.

1.3 Objective

In "**E-Shopping Website**" project the objective is to provide the cheap and best authenticated products directly from the manufacturers to the buyers.

Another objective of this project is to promote local manufacturers; small businessman can contact us to get their products on display so as to get them on the mainstream online market.

Chapter 2. Introduction of Technology

2.1 Introduction to MERN

MERN is used to build modern web applications. In this blog, we are going to get a brief introduction on MERN and how to install it in your system.

As the acronym of MERN is MongoDB, Node.js, React, and Angular, it includes all the 4 technologies combined into it. It is designed to give a quick and organized way to develop MERN based web apps, websites, web services and APIs.

Using MERN, developers can set up their work with a set of popular tools, which were carefully combined, so the developers need not concentrate on never ending work of system administration, package management, libraries, etc. Instead, they can concentrate on development completely.

Now let's look into individual technologies included in MERN.

2.2 Division of MERN

MEAN is a collection of MongoDB, Express, React, Node.js. It includes all the 4 technologies combined into it. Let's see each in brief.

MongoDB

MongoDB is an open source, NoSQL database designed for cloud applications. It uses object- oriented organization instead of a relational model. In the MERN stack, MongoDB stores the application's data. Because both the application and the database use JavaScript, there's no need to translate the object as it journeys from the application to the database and back. The application can push and pull objects between the back end and the database without missing a beat.

MongoDB is known for its scalability in both storage and performance. You can add fields to the database without reloading the entire table, and MongoDB is well known for its ability to manage large amounts of data without compromising on data access. With just a few clicks, you can expand the resources available to your database, making it perfect for applications with occasional periods of increased activity.

MongoDB's document model is simple for developers to learn and use, while still providing all the capabilities needed to meet the most complex requirements at any scale. We provide drivers for 10+ languages, and the community has built dozens more.

MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time. The document model maps to the objects in your application code, making data easy to work with Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyse your data. MongoDB is a distributed database at its core, so high availability, horizontal scaling, and geographic distribution are built in and easy to use. MongoDB is free to use. Versions released prior to October 16, 2018 are published under the AGPL.

Express

Express is one the most popular and widely used web frameworks in Node.js development zone. Express is a minimal web server built on Node.js that provides all the essential functionality required for delivering web applications to the browser and mobile devices. ExpressJS allows you to handle Routes, Server, and I/O stuff very easily.

```
const express = require('express' 4.17.1 )
const app = express()
const port = 3000

app.get('/', (req, res) => res.send('Hello World!'))

app.listen(port, () => console.log(`Example app listening on port ${port}!`))
```

React

React is a JavaScript library and React applications built on it run in the browser, NOT on the server. Applications of this kind only communicate with the server when necessary, which makes them very fast compared to traditional websites that force the user to wait for the server to re-render entire pages and send them to the browser.

React is used for building user interfaces - what the user sees on their screen and interacts with to use

your web app. This interface is split up into components, instead of having one huge page you break it up into smaller pieces known as components. In more general terms, this approach is called Modularity.

- It’s declarative: React uses a declarative paradigm that makes it easier to reason about your application.
- It’s efficient: React computes the minimal set of changes necessary to keep your DOM up-to-date.
- And it’s flexible: React works with the libraries and frameworks that you already know.

Node.js

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

Used Node packages:

1. **Cors:** CORS is a node.js package for providing a **Connect**/**Express** middleware.
2. **Body-Parser:** Parse incoming request bodies in a middleware before your handlers, available under the req. body property.
3. **Node mailer:** node mailer package is used to sending emails.
4. **Multer:** Multer is used to upload the images to the databases.
5. **Sha1:** sha1 is used to encrypt the password fields before save in database.

Other Used Languages:

1. HTML: HTML stands for Hyper Text Mark-up Language. It is used to design web pages using mark-up language. HTML is the combination of Hypertext and Mark-up language. Hypertext defines the link between the web pages. Mark-up language is used to define the text document within tag which defines the structure of web pages.

2. CSS: Cascading **S**tyle **S**heets, fondly referred to as **CSS**, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

3. Bootstrap 4: Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first web sites. It solves many problems which we had once, one of which is the cross-browser compatibility issue.

4. JavaScript: JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. **JavaScript** is very easy to implement because it is integrated with HTML. It is open and cross- platform. JavaScript is the most popular **programming language** in the world and that makes it a programmer's great choice. Once you learnt JavaScript, it helps you developing great front- end as well as back-end software using different JavaScript based frameworks like jQuery, Node.JS etc.

Chapter 3. Software Requirement Analysis

3.1 Problem Statement

Many-a-times, we have seen that people want to buy products but due to some reasons cannot go physically to the shops to buy them. And since this is a pandemic going on so people really prefer to have the stuff delivered rather than going to the shop. Similarly, the small businesses are having problem in sales due to this pandemic.

So, this E-Shopping website will act as a bridge between the stuck at home public and their urge to shop as well as a helping hand towards the local businesses to expand their reach to the online platforms as well.

Type of users:

Buyer: The person who wants to buy a product from this website needs to have an account first, using which he will be allowed to buy any product. Without an existing account, a new user can only visit the products uploaded by the administrator.

3.2 Modules and its functionalities

Customer:

1. **Default Home Page:** The page that will be opened at first when we open the website whether a person has an existing account on the website or not.

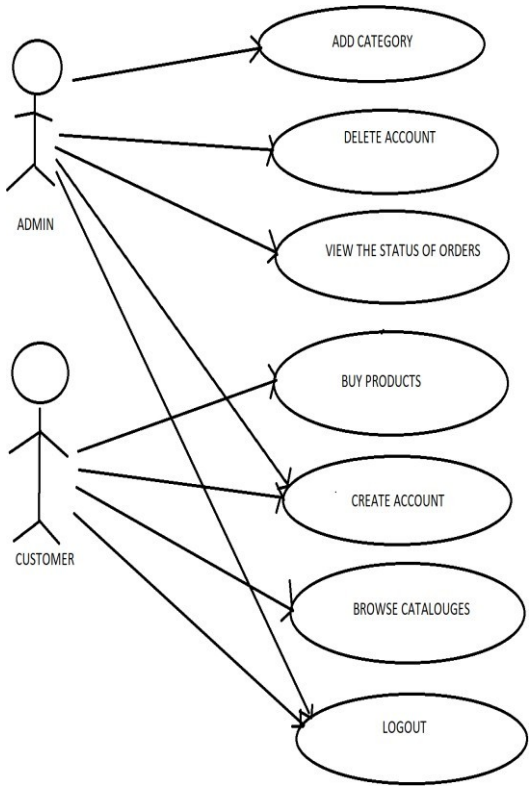
2. **Member Login:** Any member who wants to buy an item, first will login to the website using this module.

3. **Forget and Change Password:** Any user having an existing account can change his password in case he forgets his current password or if he just wants to change for security purpose.

Chapter 4. Software Design

4.1 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different **use** cases in which the user is involved.

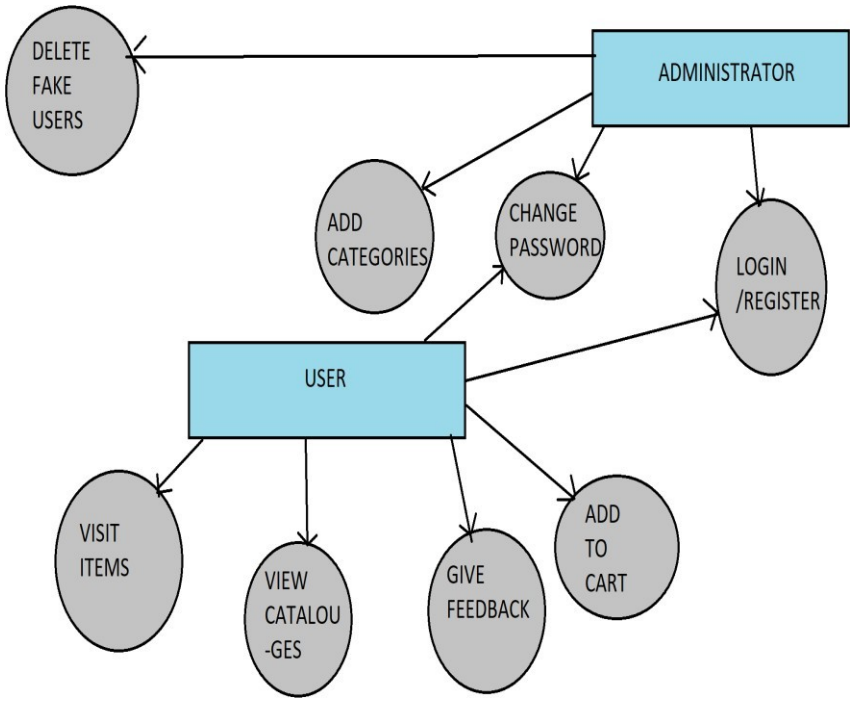


4.2. Dataflow Diagrams

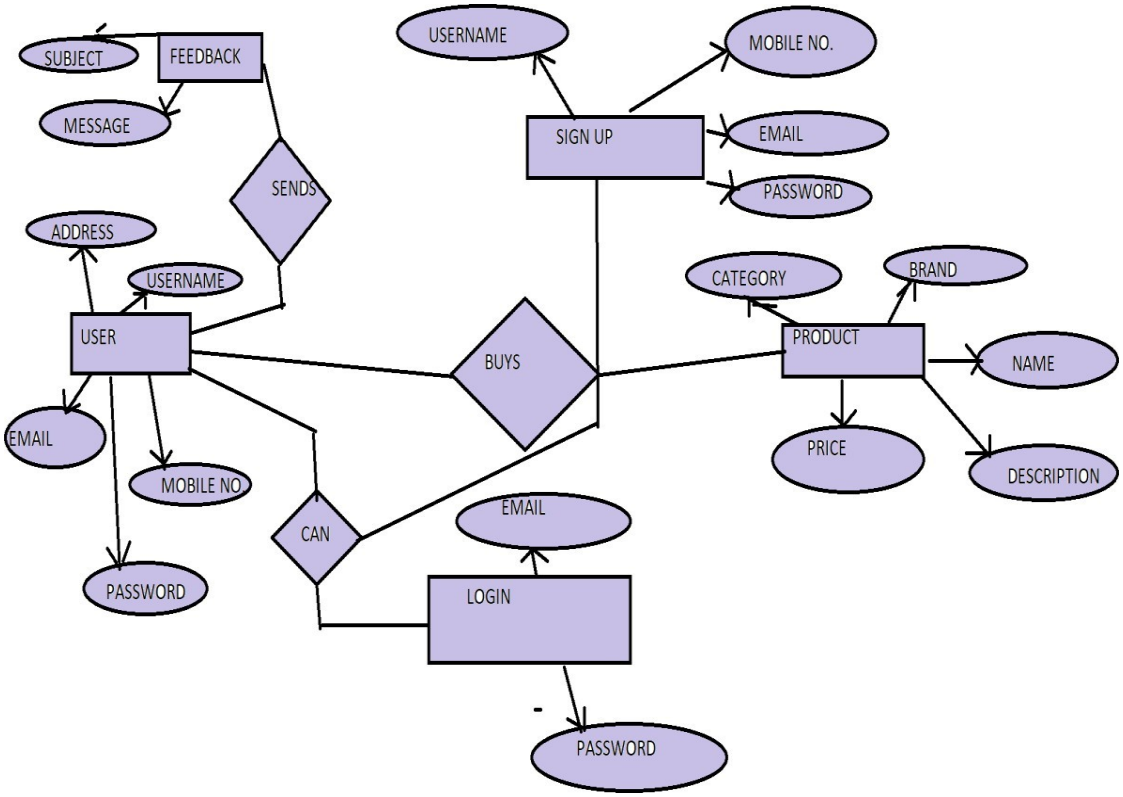
Data Flow diagrams show the flow of data from external entities into the system, and from one process to another within the system.

a) Level -0 Diagram: The level 0 diagram provides a conceptual view of the process and its surrounding input, output and data stores. It is called context level Data flow diagram also.

b) Level -1 Diagram: The level -1 diagram provides a more detailed and comprehensive view of the interaction among the sub-processes within the system.



4.2. Entity Relationship Diagrams


SUBJECT
FEEDBACK
MESSAGE
SENDS
ADDRESS
USERNAME
USER
EMAIL
MOBILE NO.
PASSWORD
USERNAME
MOBILE NO.
SIGN UP
EMAIL
PASSWORD
BUYS
CATEGORY
BRAND
PRODUCT
NAME
PRICE
DESCRIPTION
CAN
EMAIL
LOGIN
PASSWORD

Chapter 5. Creating a MERN Project

5.1 Basic Needs

- Create a separate folder named myProject to store all the project files.
- Go inside this folder open command prompt and run the command.

create-react-app

-The above command will create a new raect project named Frontend containing all the files

In the same root directory as your backend code, which is the todo directory, run:

```
create-react-app client
```

This will create a new folder in your `todo` directory called `client`, where you will add all the react code.

***Important: You should have installed latest version of node and npm on your system.*

Running the React App

Before testing the react app, there are a number od dependencies that need to be installed.

1. Install **concurrently** as a dev dependency:

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

Concurrently is used to run more than one command simultaneously from the same terminal window.

1. Install **nodem**an as a dev dependency:

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

- Run the same command again to generate another React project to design the AdminPanel.

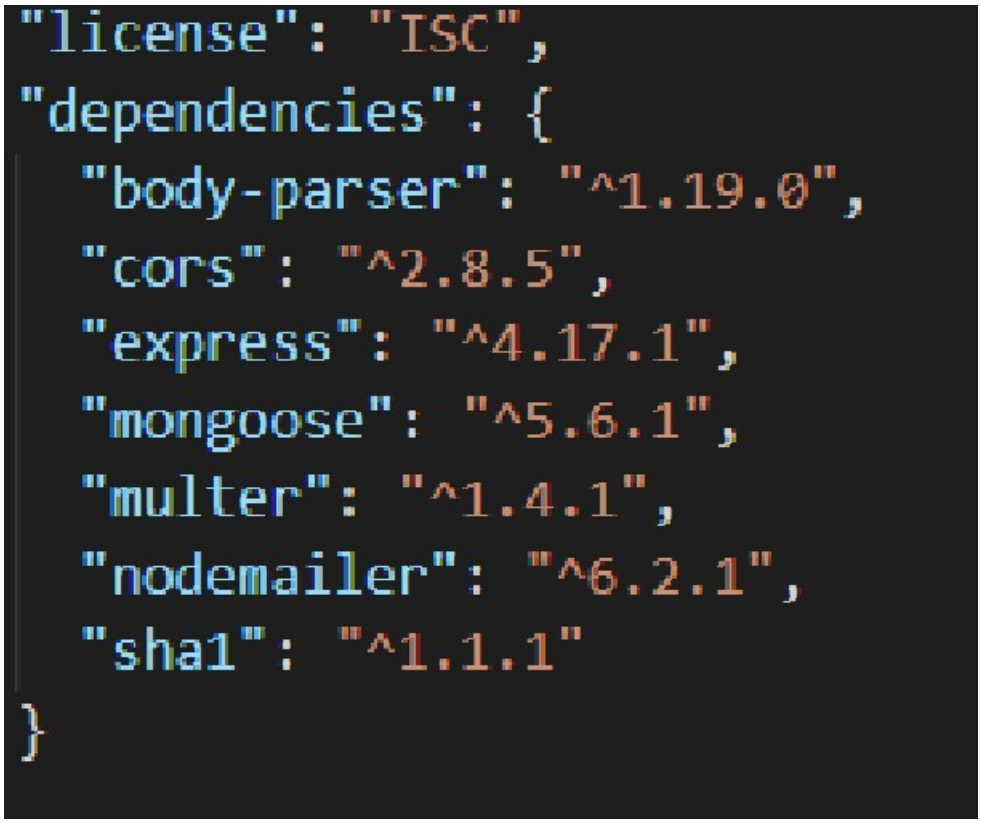
- Now create another folder inside myProject folder named Backend to store all the backend data of the site.

- Go inside the Backend folder and open command prompt and run the command

npm init

- This will generate a package.json file, open the package.json file and set all the

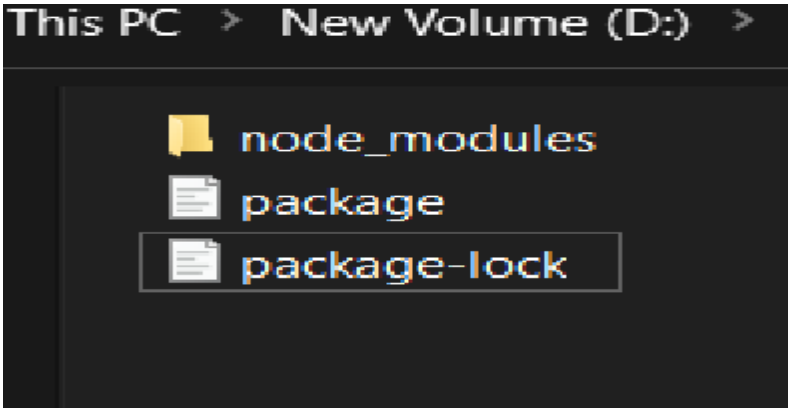
dependencies as given below.



- After this run the command to install all the dependencies

npm install –g node-modules

- Now your backend folder will lode like as shown below.



5.2 Coding the web

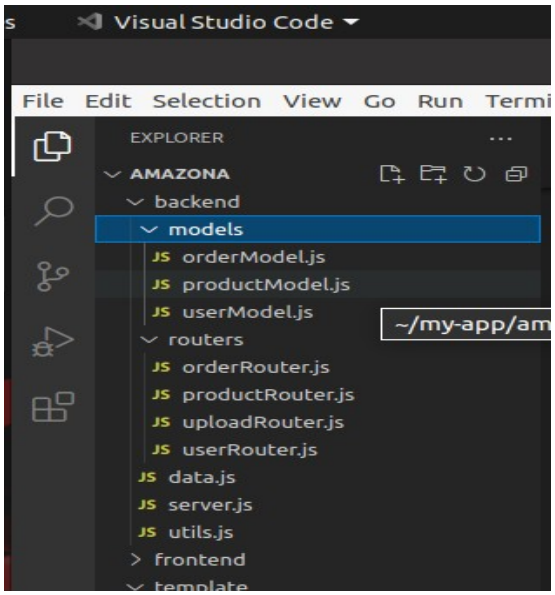
Let's setup Adminpanel

- Go inside the AdminPanel folder and open the folder in command prompt and run the command **ng serve --o** as shown below. This will compile your code.

After this on localhost:4200

- Now create a Login component to allow the Admin to login to admin panel.

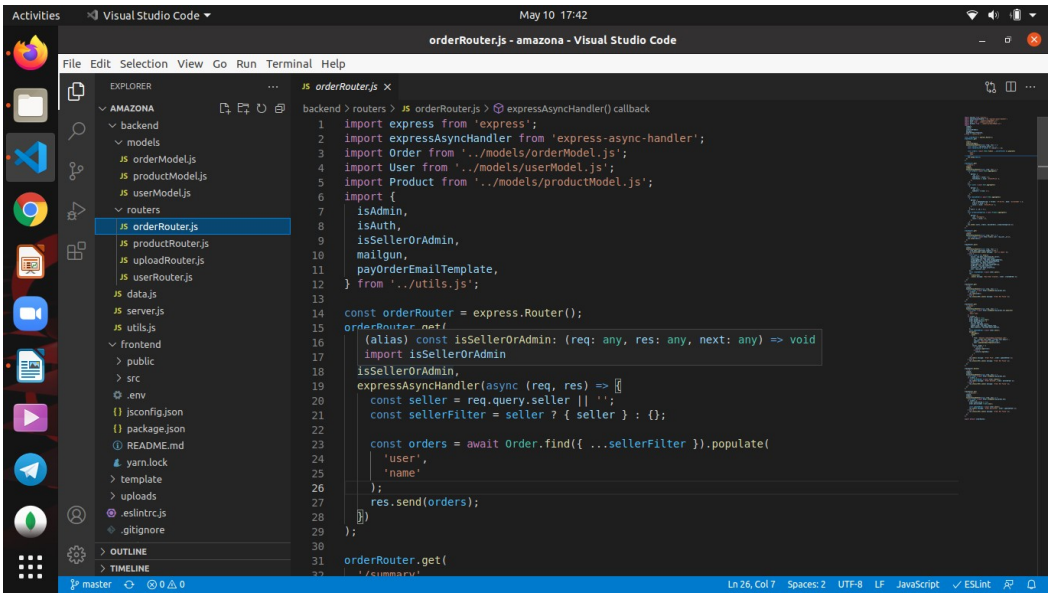
- To create a component open adminpanel folder in command prompt and type command ng g c login this will automatically create the component and add it to the app.module.ts file as show in image below.



]

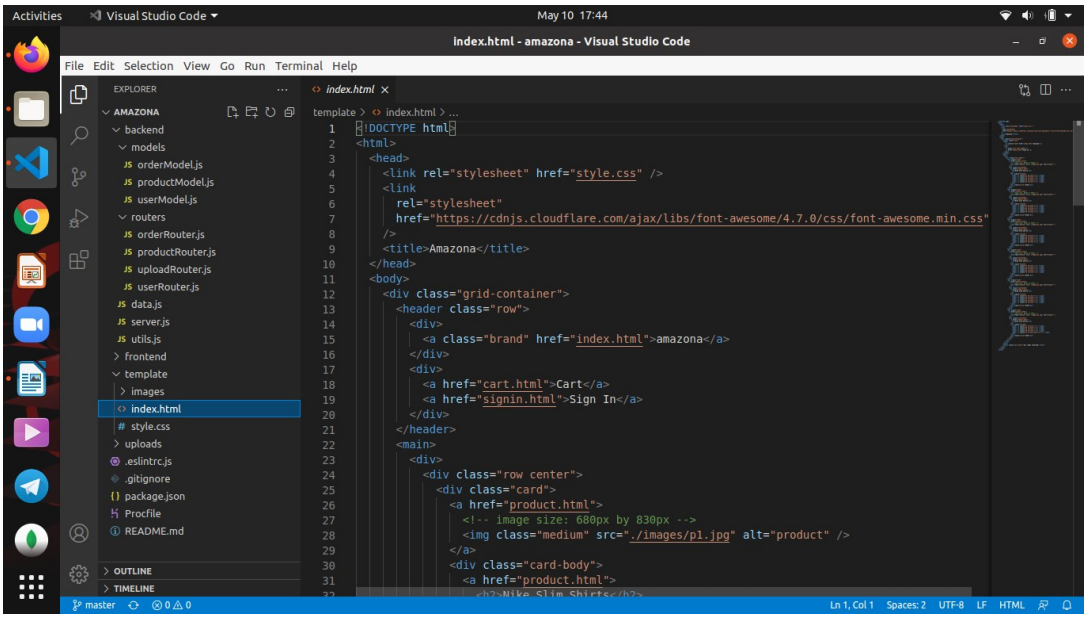
App-routing.module.ts

This file is used to store all the routes. After creating all the components needed you can give routes to them as shown in the image above.



Adding Bootstrap, Fonts and other Styles:

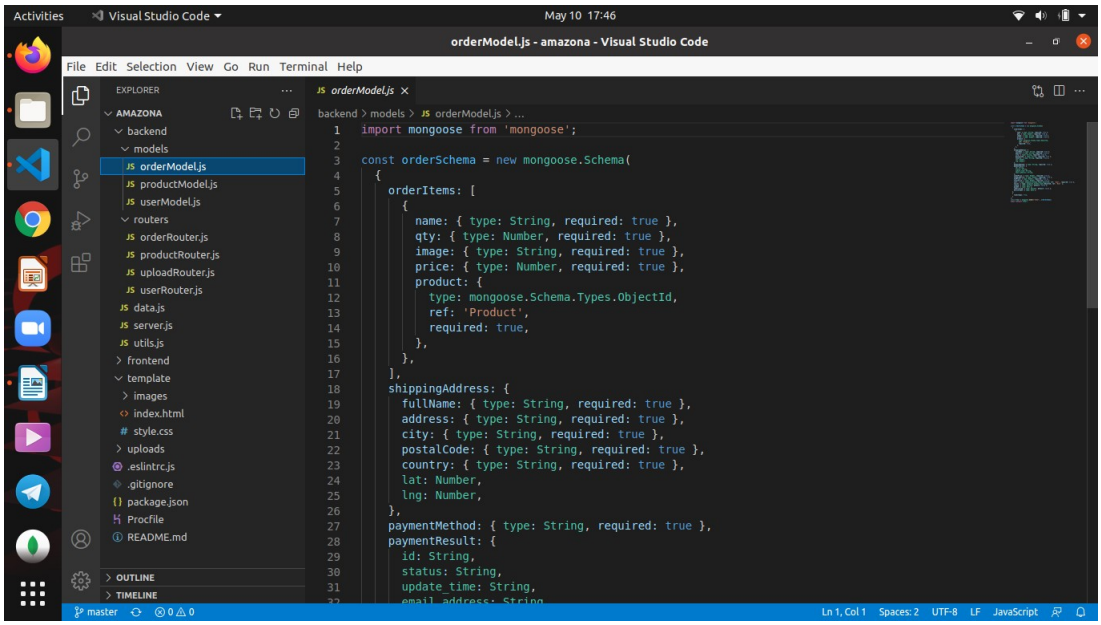
You can add bootstrap, fonts, Styles etc to Adminpanel/src/app/index.html file as shown below:



All set up let's design the login page UI

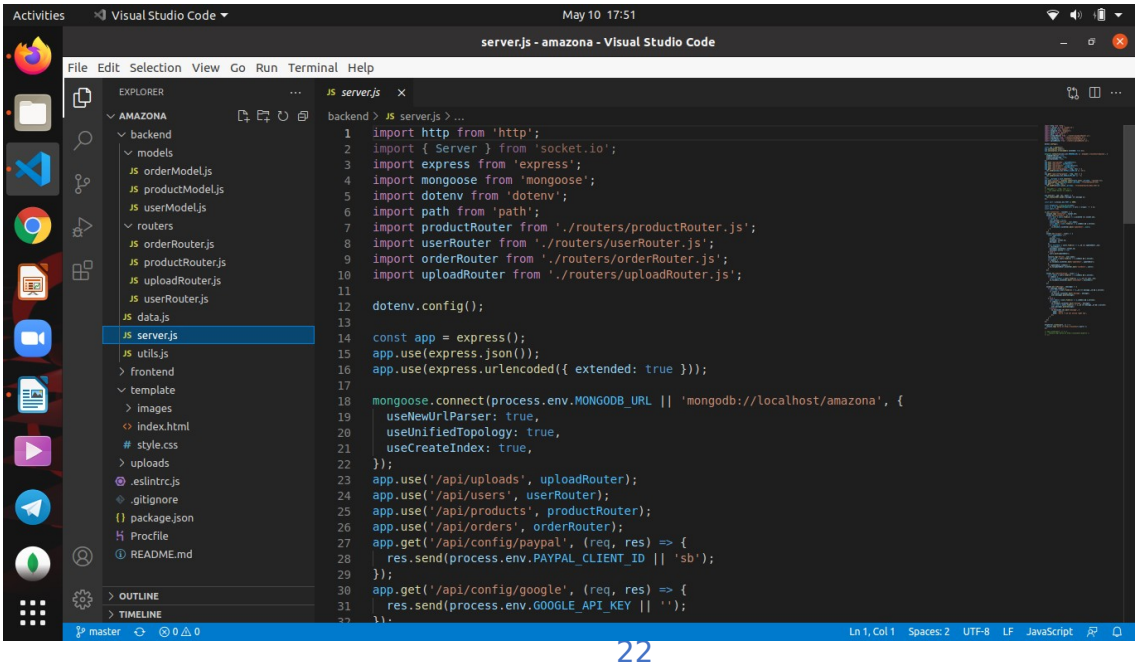
The code below shows the UI design for login page:

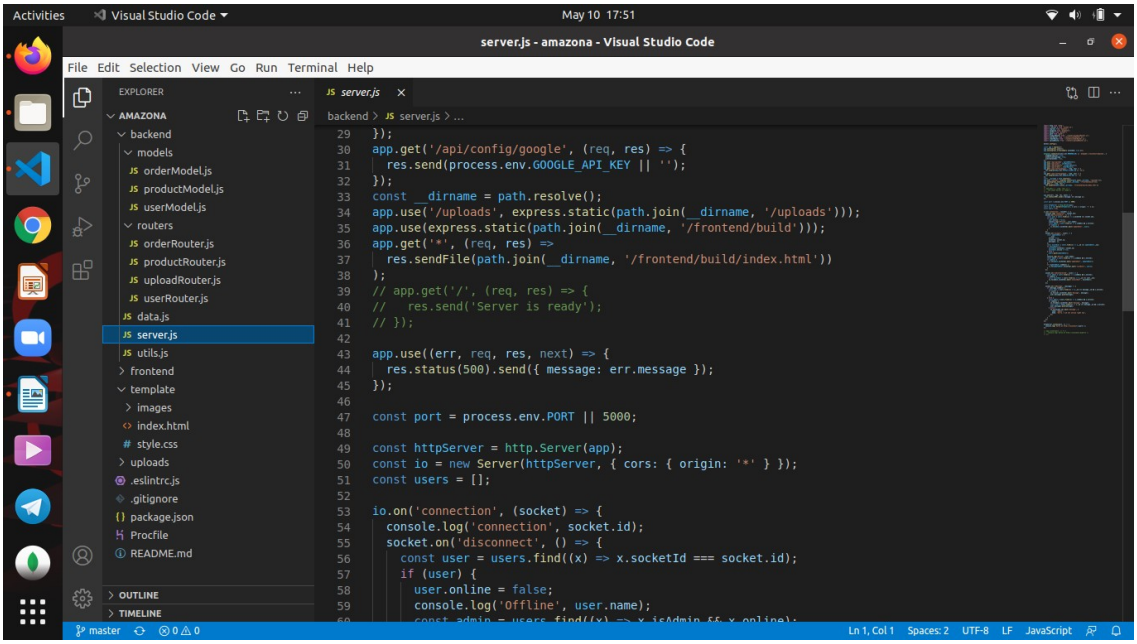
ordermodel.js

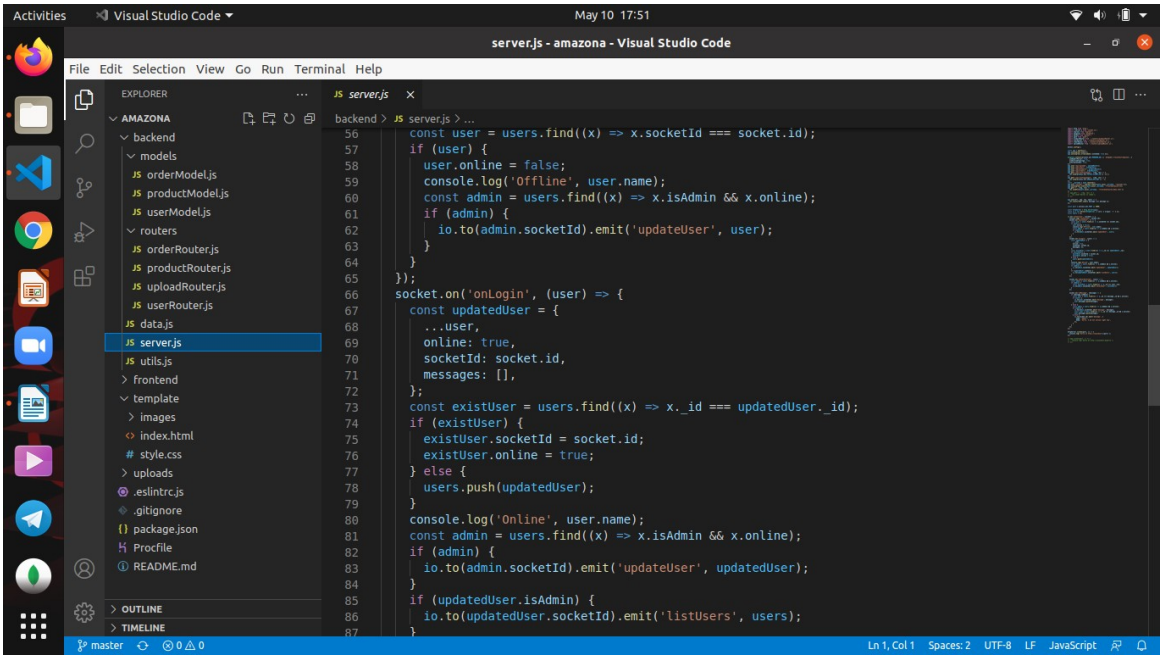


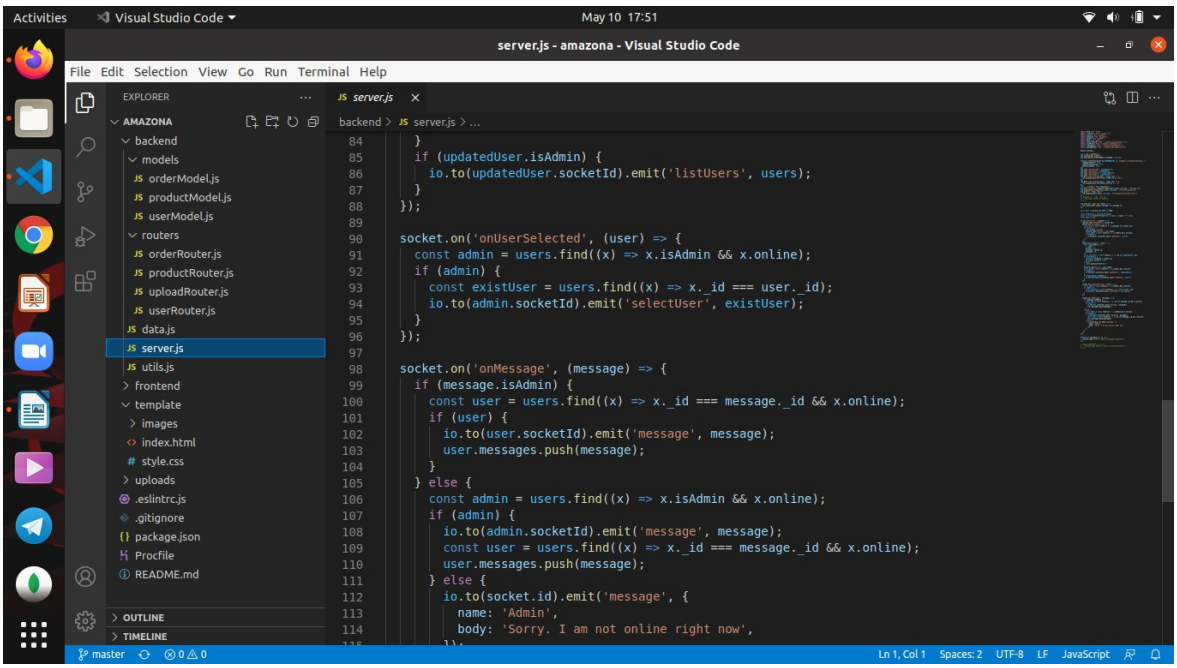
Backend Api file to handle all the database requests

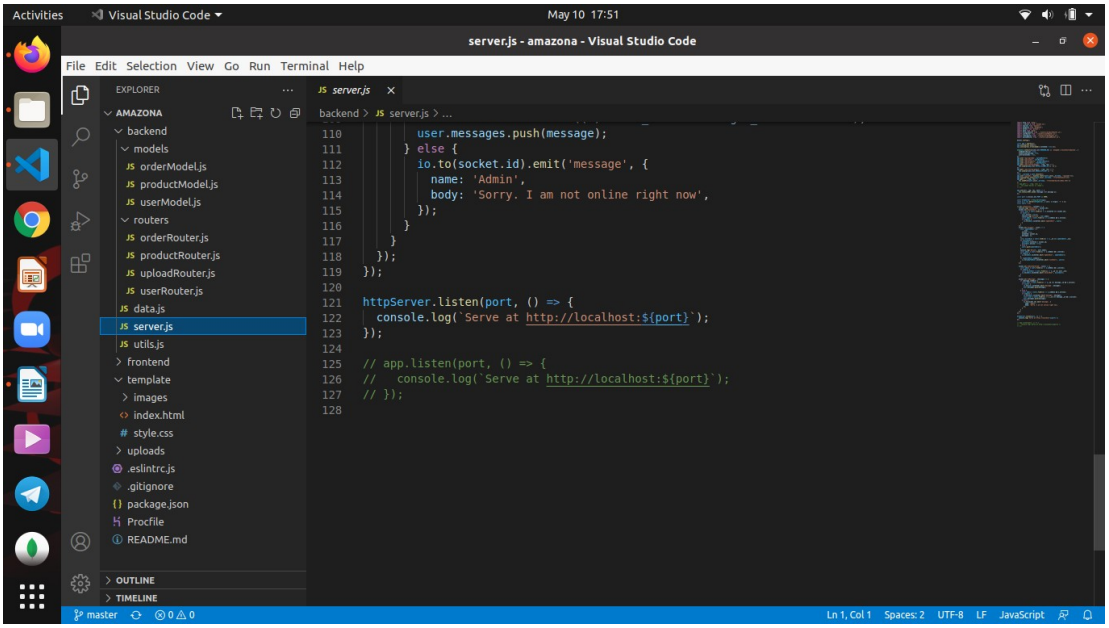
Below is a code snippet of the server.js file to handle all the database related queries:



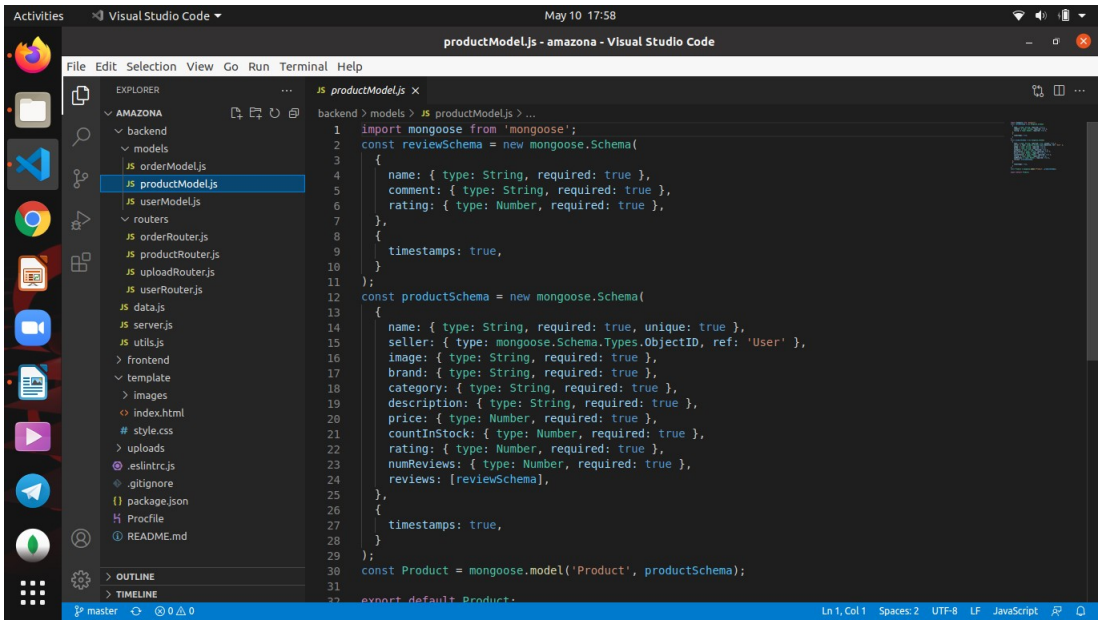

Activities
Visual Studio Code
May 10 17:51
server.js - amazona - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
AMAZONA
backend
models
orderModel.js
productModel.js
userModel.js
routers
orderRouter.js
productRouter.js
uploadRouter.js
userRouter.js
data.js
server.js
utils.js
frontend
template
images
index.html
style.css
uploads
.eslintrc.js
.gitignore
package.json
Procfile
README.md
OUTLINE
TIMELINE
server.js
backend > server.js > ...
});
app.get('/api/config/google', (req, res) => {
res.send(process.env.GOOGLE_API_KEY || '');
});
const __dirname = path.resolve();
app.use('/uploads', express.static(path.join(__dirname, '/uploads')));
app.use(express.static(path.join(__dirname, '/frontend/build')));
app.get('*', (req, res) =>
res.sendFile(path.join(__dirname, '/frontend/build/index.html'))
);
// app.get('/', (req, res) => {
// res.send('Server is ready');
// });
app.use((err, req, res, next) => {
res.status(500).send({ message: err.message });
});
const port = process.env.PORT || 5000;
const httpServer = http.Server(app);
const io = new Server(httpServer, { cors: { origin: '*' } });
const users = [];
io.on('connection', (socket) => {
console.log('connection', socket.id);
socket.on('disconnect', () => {
const user = users.find((x) => x.socketId === socket.id);
if (user) {
user.online = false;
console.log('Offline', user.name);
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
JavaScript


Activities
Visual Studio Code
May 10 17:51
server.js - amazona - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
AMAZONA
backend
models
orderModel.js
productModel.js
userModel.js
routers
orderRouter.js
productRouter.js
uploadRouter.js
userRouter.js
data.js
server.js
utils.js
frontend
template
images
index.html
style.css
uploads
.eslintrc.js
.gitignore
package.json
Procfile
README.md
OUTLINE
TIMELINE
server.js
backend > server.js > ...
const user = users.find((x) => x.socketId === socket.id);
if (user) {
user.online = false;
console.log('Offline', user.name);
const admin = users.find((x) => x.isAdmin && x.online);
if (admin) {
io.to(admin.socketId).emit('updateUser', user);
}
}
});
socket.on('onLogin', (user) => {
const updatedUser = {
...user,
online: true,
socketId: socket.id,
messages: [],
};
const existUser = users.find((x) => x._id === updatedUser._id);
if (existUser) {
existUser.socketId = socket.id;
existUser.online = true;
} else {
users.push(updatedUser);
}
console.log('Online', user.name);
const admin = users.find((x) => x.isAdmin && x.online);
if (admin) {
io.to(admin.socketId).emit('updateUser', updatedUser);
}
if (updatedUser.isAdmin) {
io.to(updatedUser.socketId).emit('listUsers', users);
}
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
JavaScript


Activities
Visual Studio Code
May 10 17:51
server.js - amazona - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
AMAZONA
backend
models
orderModel.js
productModel.js
userModel.js
routers
orderRouter.js
productRouter.js
uploadRouter.js
userRouter.js
data.js
server.js
utils.js
frontend
template
images
index.html
style.css
uploads
.eslintrc.js
.gitignore
package.json
Procfile
README.md
OUTLINE
TIMELINE
server.js
backend > server.js > ...
}
if (updatedUser.isAdmin) {
io.to(updatedUser.socketId).emit('listUsers', users);
}
});
socket.on('onUserSelected', (user) => {
const admin = users.find((x) => x.isAdmin && x.online);
if (admin) {
const existUser = users.find((x) => x._id === user._id);
io.to(admin.socketId).emit('selectUser', existUser);
}
});
socket.on('onMessage', (message) => {
if (message.isAdmin) {
const user = users.find((x) => x._id === message._id && x.online);
if (user) {
io.to(user.socketId).emit('message', message);
user.messages.push(message);
}
} else {
const admin = users.find((x) => x.isAdmin && x.online);
if (admin) {
io.to(admin.socketId).emit('message', message);
const user = users.find((x) => x._id === message._id && x.online);
user.messages.push(message);
} else {
io.to(socket.id).emit('message', {
name: 'Admin',
body: 'Sorry. I am not online right now',
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
JavaScript


Activities
Visual Studio Code
May 10 17:51
server.js - amazona - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
AMAZONA
backend
models
orderModel.js
productModel.js
userModel.js
routers
orderRouter.js
productRouter.js
uploadRouter.js
userRouter.js
data.js
server.js
utils.js
frontend
template
images
index.html
style.css
uploads
.eslintrc.js
.gitignore
package.json
Procfile
README.md
OUTLINE
TIMELINE
server.js
backend > server.js > ...
user.messages.push(message);
} else {
io.to(socket.id).emit('message', {
name: 'Admin',
body: 'Sorry. I am not online right now',
});
}
}
});
});
httpServer.listen(port, () => {
console.log(`Serve at http://localhost:${port}`);
});
// app.listen(port, () => {
//   console.log(`Serve at http://localhost:${port}`);
// });
master
Ln 1, Col 1
Spaces: 2
UTF-8
LF
JavaScript

Below is a code snippet of the productModel.js file to handle all the database related queries:



Chapter 6. Testing

1. Login: while logging in to website user need to enter email and password

   if (email is incorrect || password is incorrect)

   Generate err window (“Email or password is incorrect”);

   else

   Login to website

2. Change Password: user need to enter old password, new password and confirm new password.

   If old password matches, new password and confirm new password field has the same password value, password is changed otherwise it generate an error message that old password did not match or new password and confirm password should be same depending on the condition.

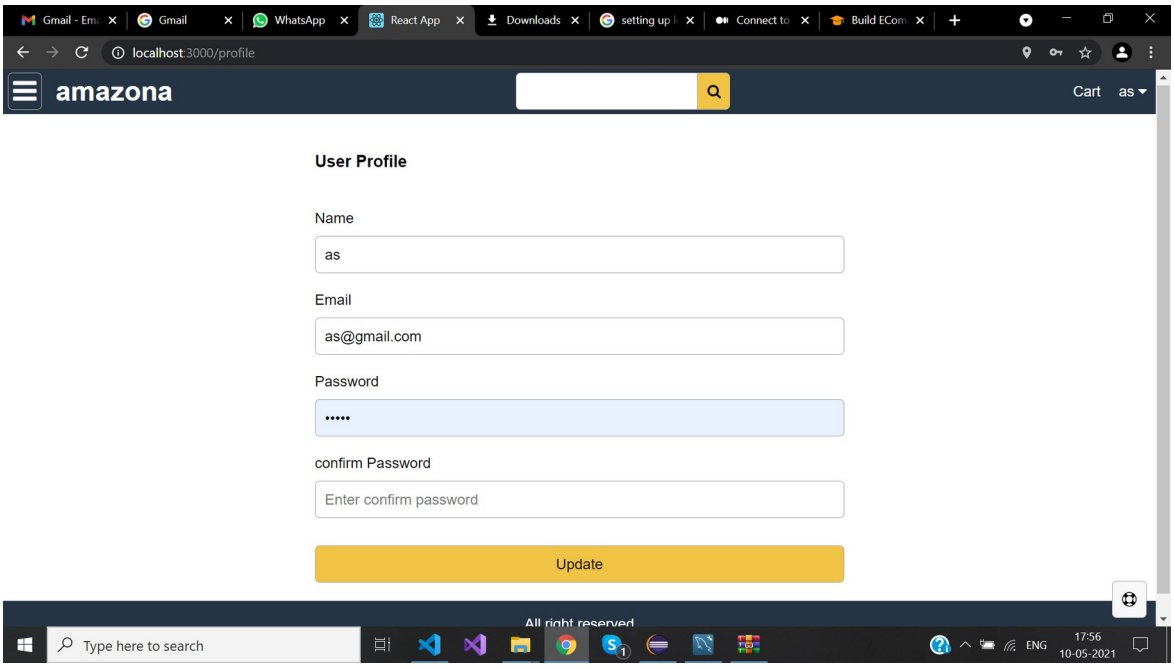
3. Sign up: while registering on the website user needs to enter username, email, password, confirm password and mobile number.

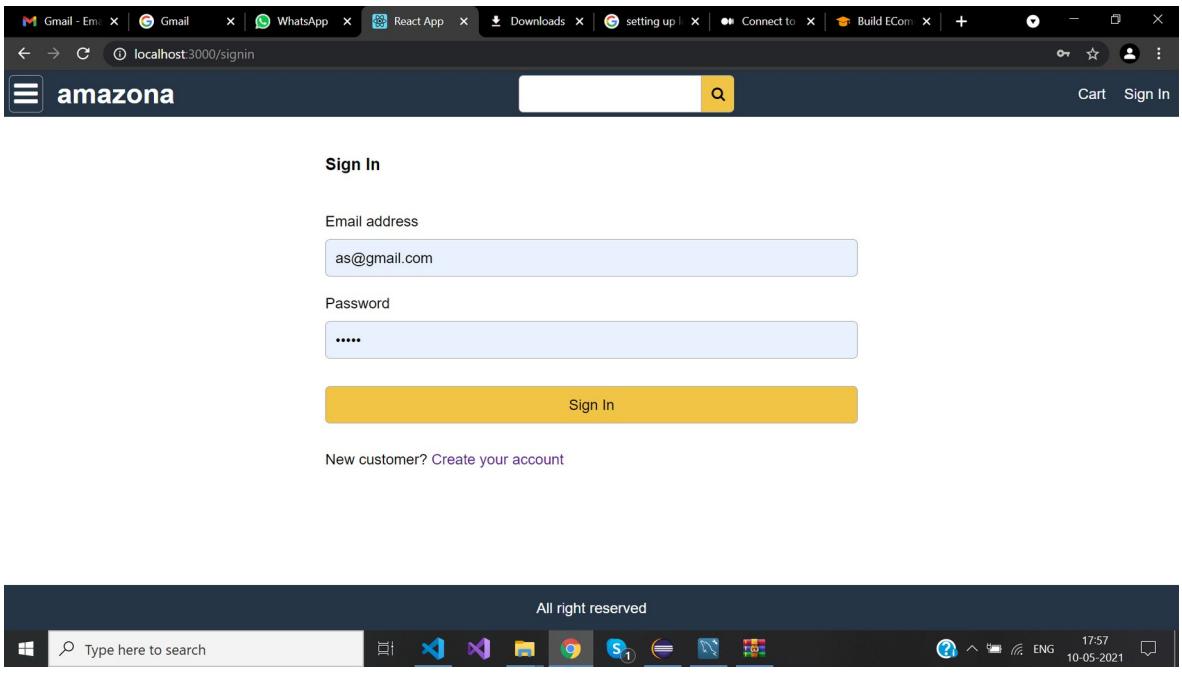
   if (user exists already)

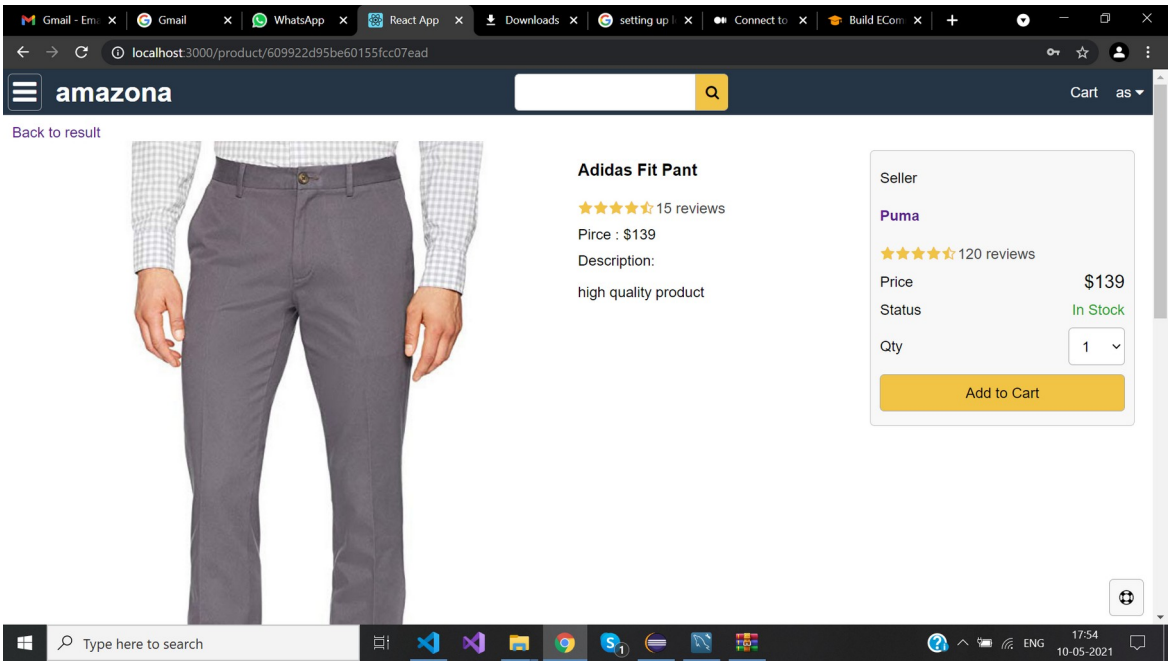
   Generate err window(“User already exists.”);

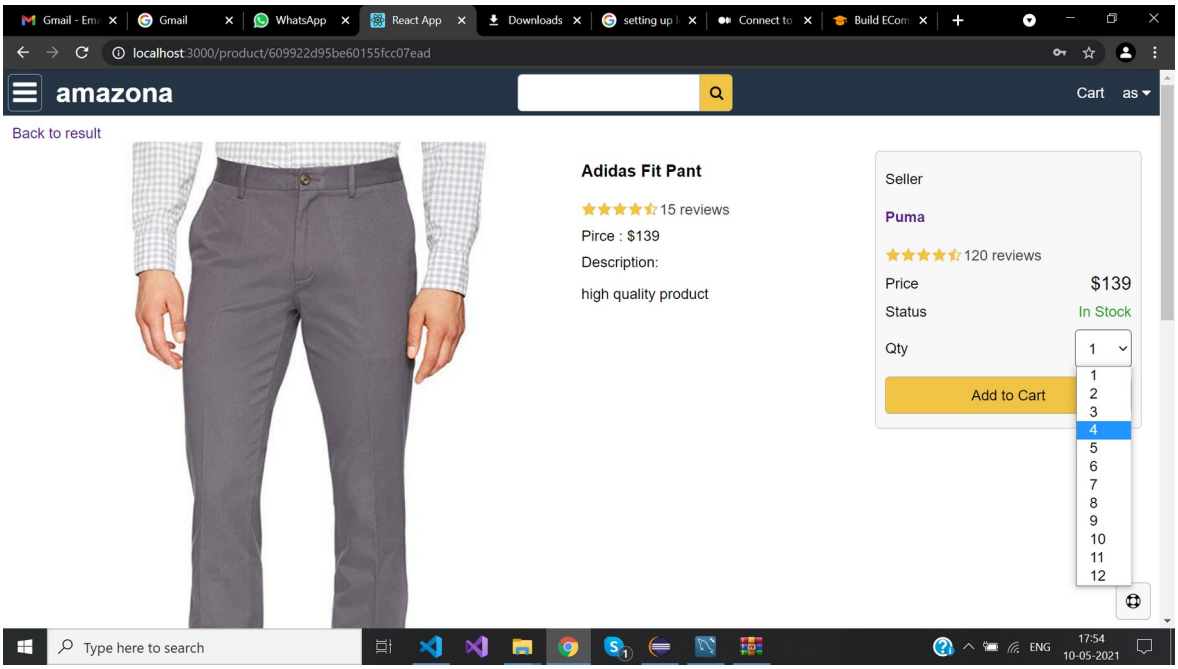
   else

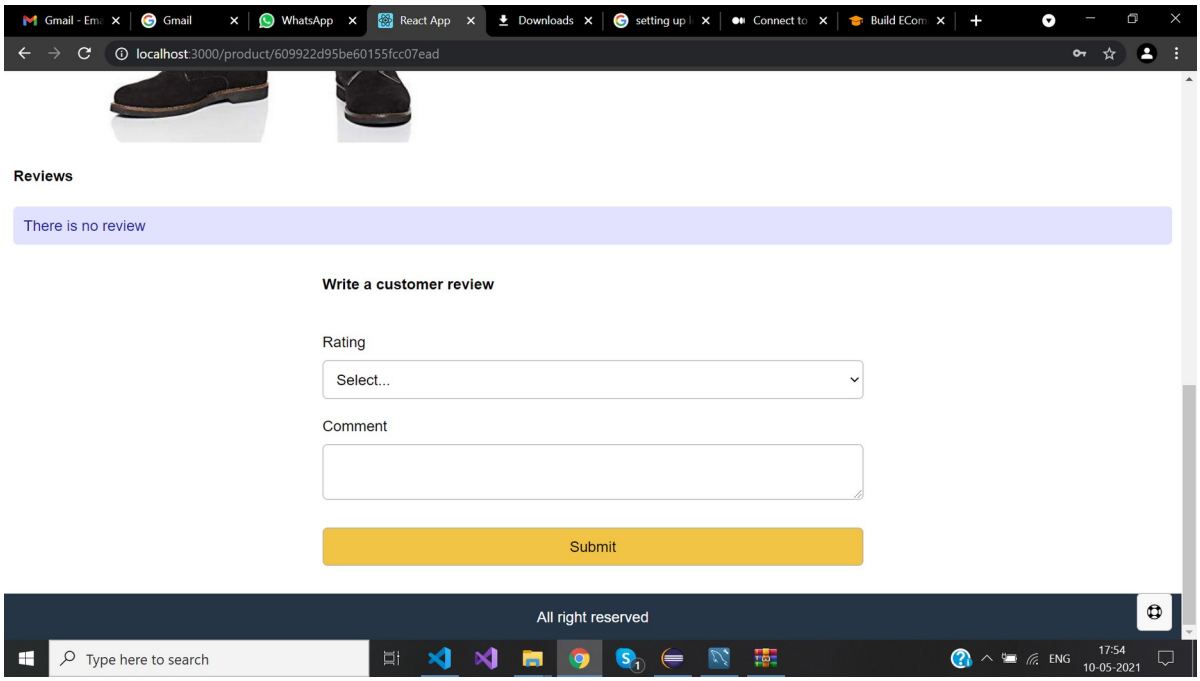
   Sign Up to Website.

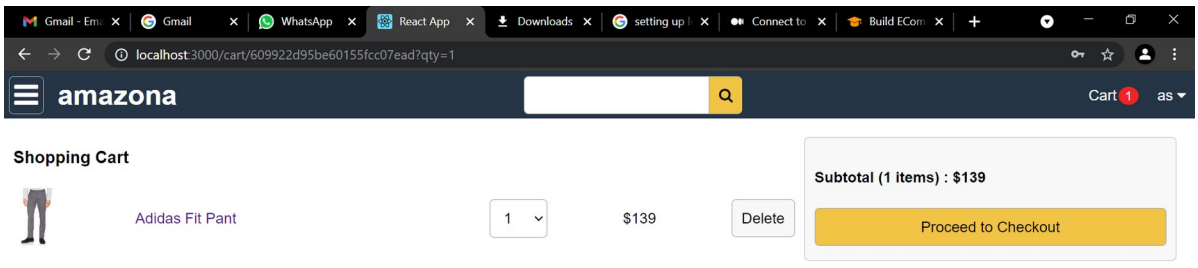

localhost:3000/profile
amazona
Cart
as
User Profile
Name
as
Email
as@gmail.com
Password
confirm Password
Enter confirm password
Update
Type here to search
ENG
17:56
10-05-2021

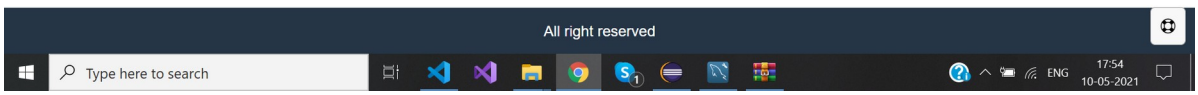

localhost:3000/signin
amazona
Cart
Sign In
Sign In
Email address
as@gmail.com
Password
Sign In
New customer? Create your account
All right reserved
Type here to search
ENG
17:57
10-05-2021

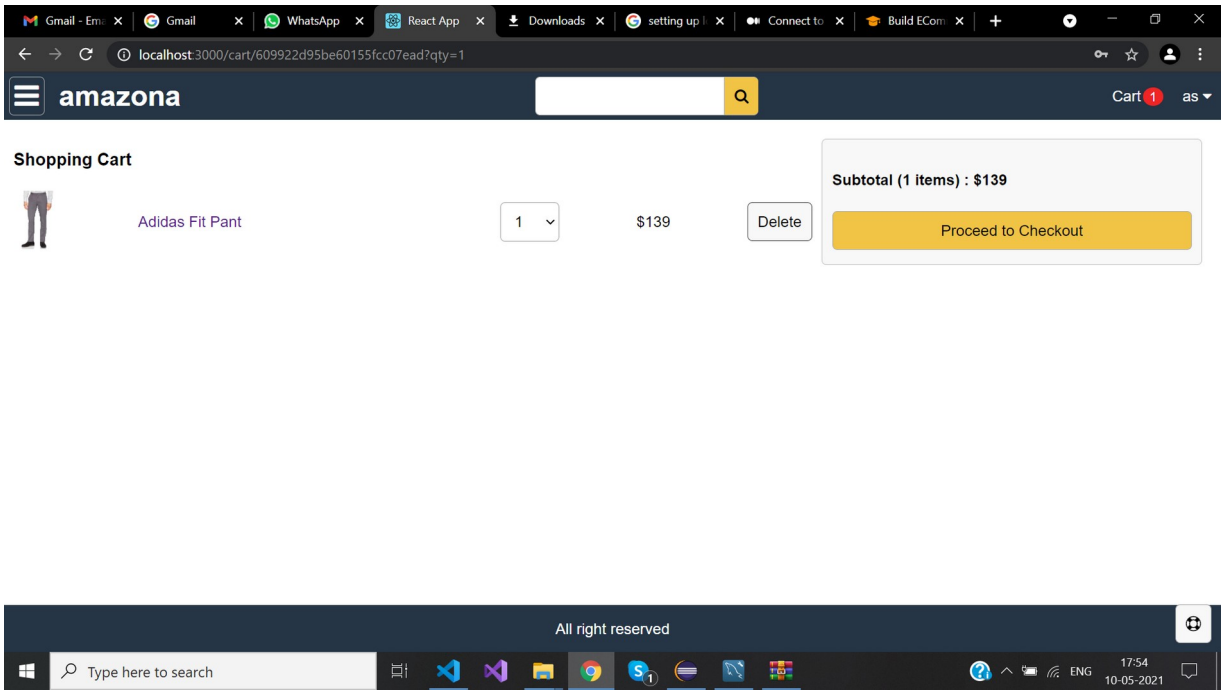

localhost:3000/product/609922d95be60155fcc07ead
amazona
Cart
as
Back to result
Adidas Fit Pant
15 reviews
Pirce : $139
Description:
high quality product
Seller
Puma
120 reviews
Price
$139
Status
In Stock
Qty
1
Add to Cart
Type here to search
ENG
17:54
10-05-2021

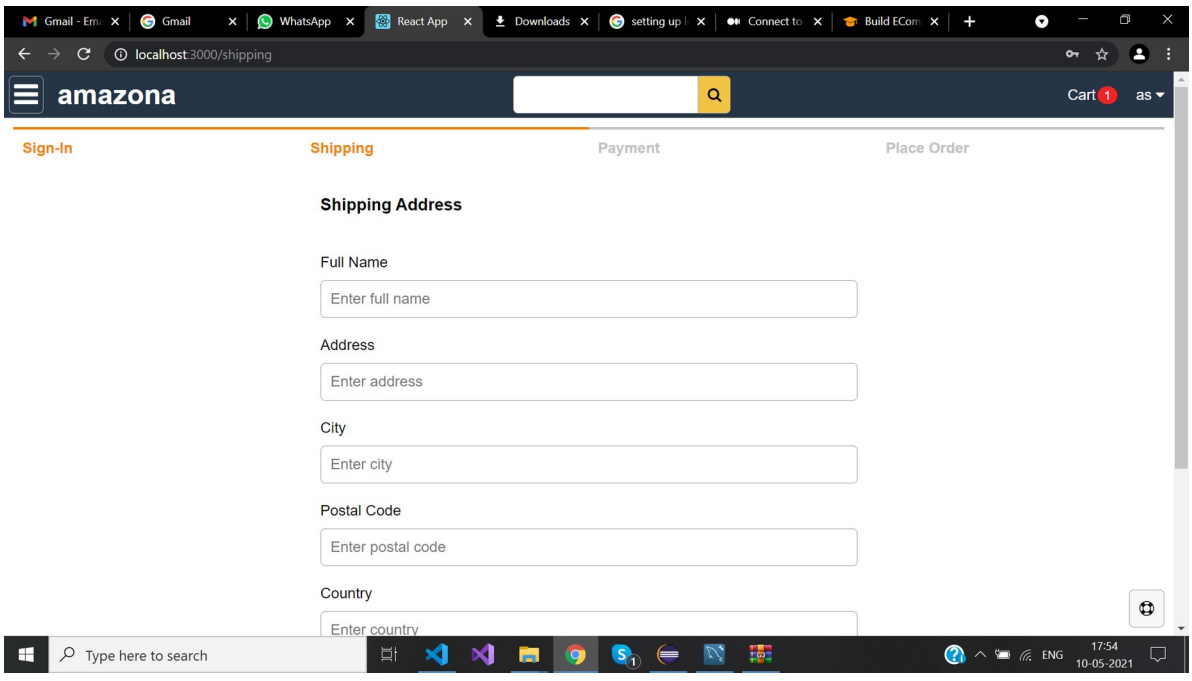

localhost:3000/product/609922d95be60155fcc07ead
amazona
Cart
as
Back to result
Adidas Fit Pant
15 reviews
Pirce : $139
Description:
high quality product
Seller
Puma
120 reviews
Price
$139
Status
In Stock
Qty
1
Add to Cart
1
2
3
4
5
6
7
8
9
10
11
12
Type here to search
ENG
17:54
10-05-2021

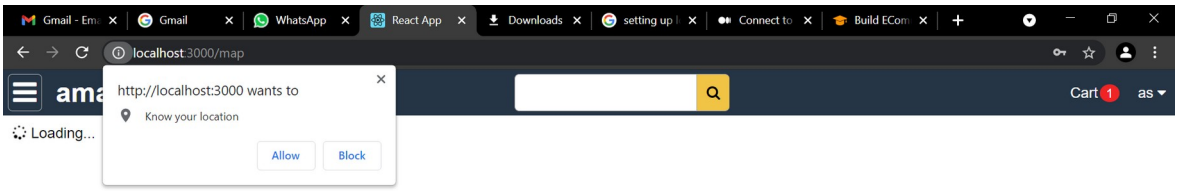

localhost:3000/product/609922d95be60155fcc07ead
Reviews
There is no review
Write a customer review
Rating
Select...
Comment
Submit
All right reserved
Type here to search
ENG
17:54
10-05-2021

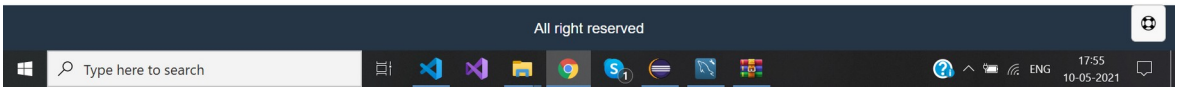

localhost:3000/cart/609922d95be60155fcc07ead?qty=1
amazona
Cart 1
as
Shopping Cart
Adidas Fit Pant
1
$139
Delete
Subtotal (1 items) : $139
Proceed to Checkout

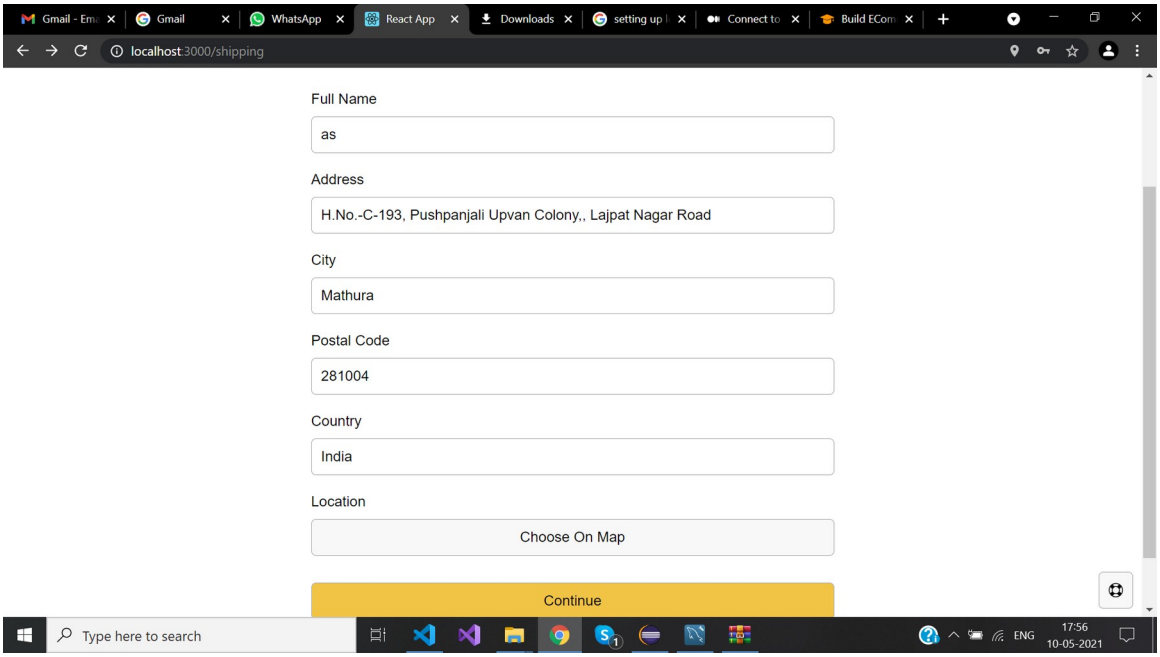

All right reserved
Type here to search
ENG
17:54
10-05-2021

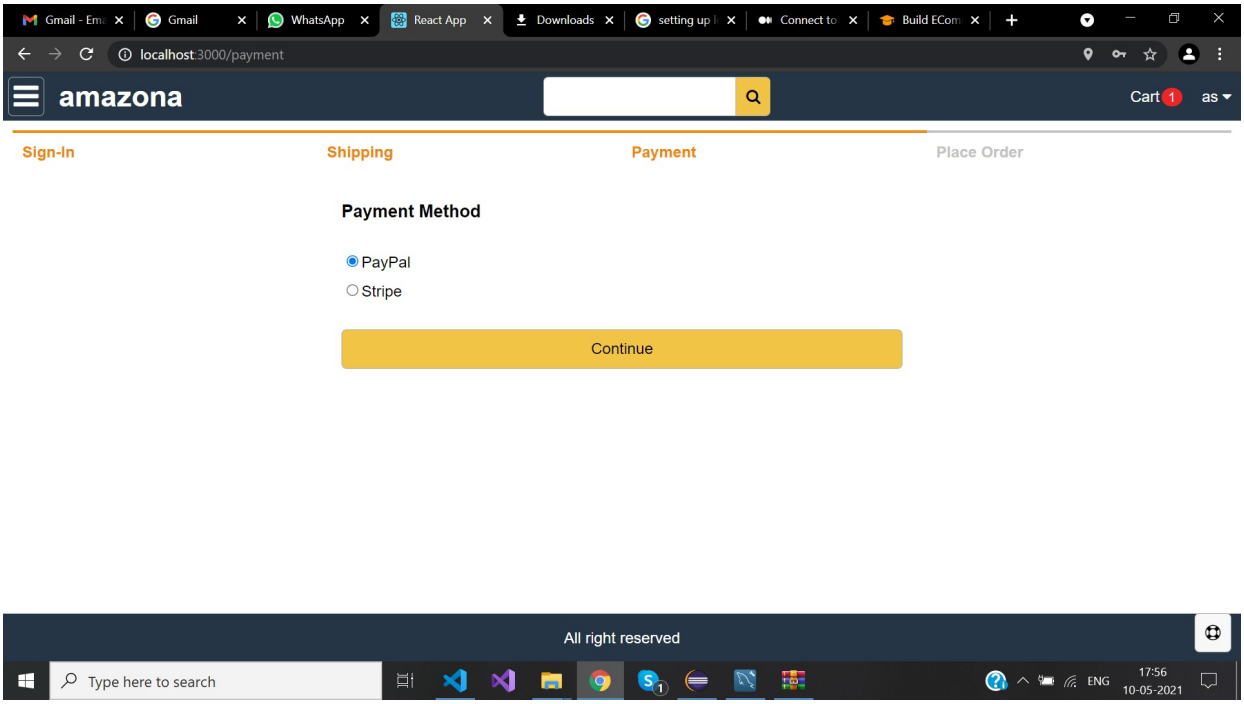

localhost:3000/cart/609922d95be60155fcc07ead?qty=1
amazona
Cart 1
as
Shopping Cart
Adidas Fit Pant
1
$139
Delete
Subtotal (1 items) : $139
Proceed to Checkout
All right reserved
Type here to search
ENG
17:54
10-05-2021

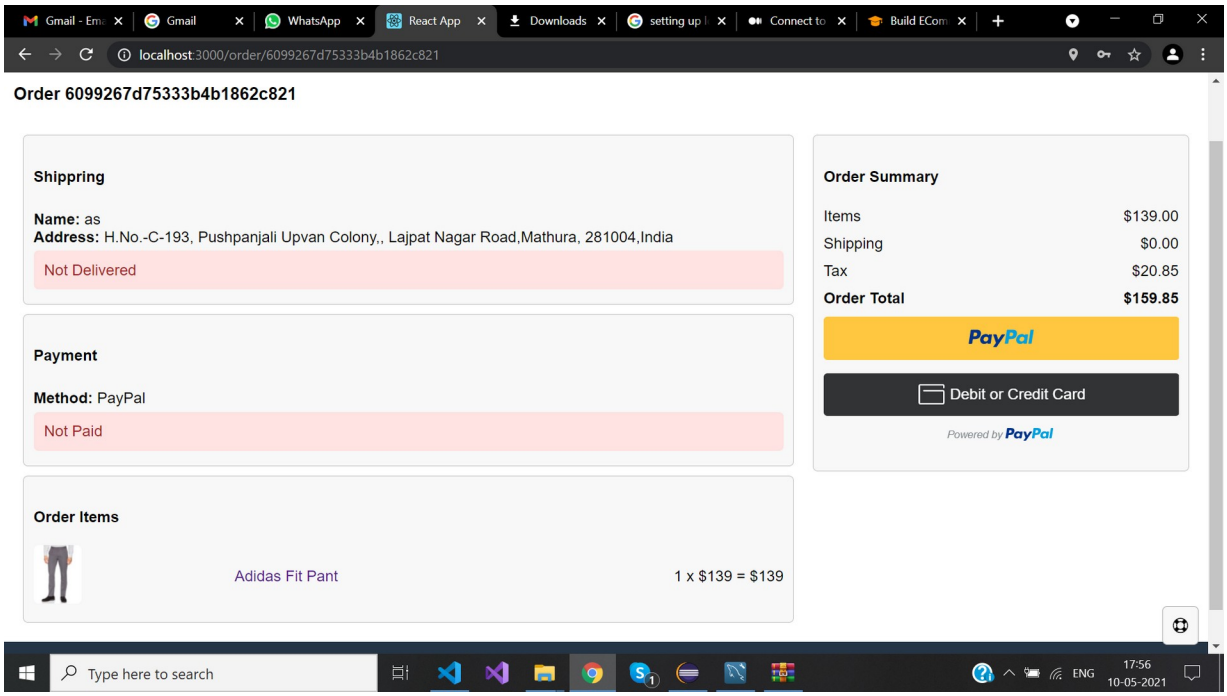

localhost:3000/shipping
amazona
Cart 1
as
Sign-In
Shipping
Payment
Place Order
Shipping Address
Full Name
Enter full name
Address
Enter address
City
Enter city
Postal Code
Enter postal code
Country
Enter country
Type here to search
ENG
17:54
10-05-2021

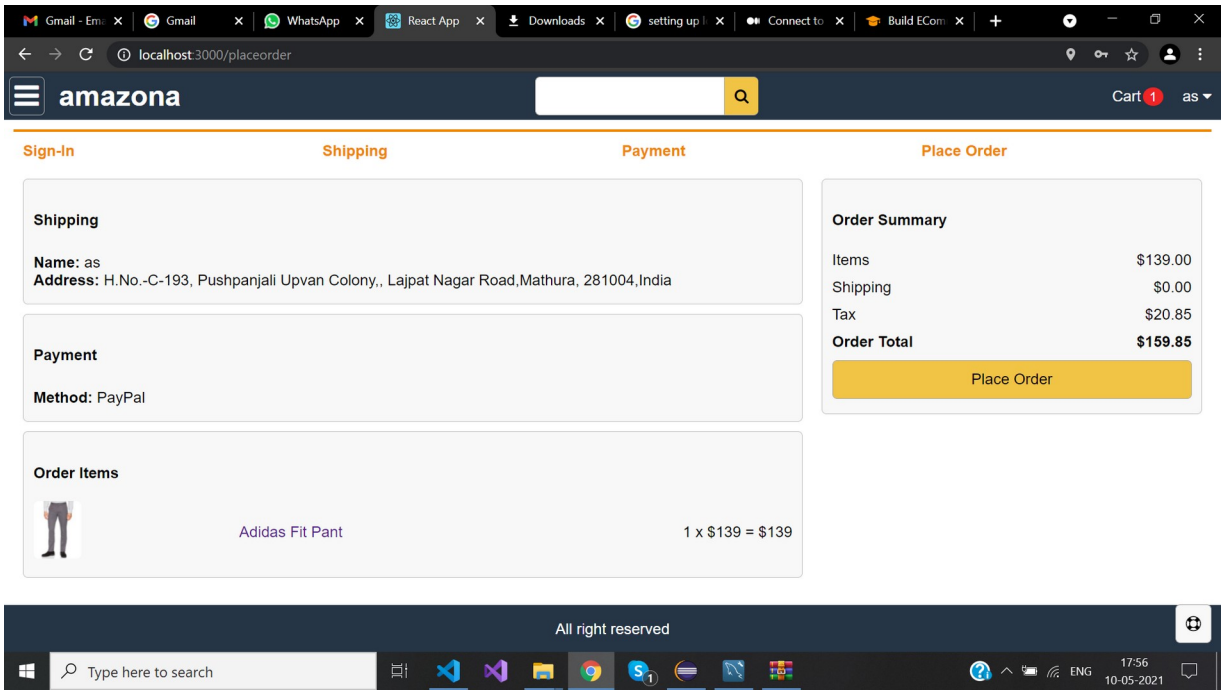

localhost:3000/map
http://localhost:3000 wants to
Know your location
Allow
Block
Loading...
Cart 1
as

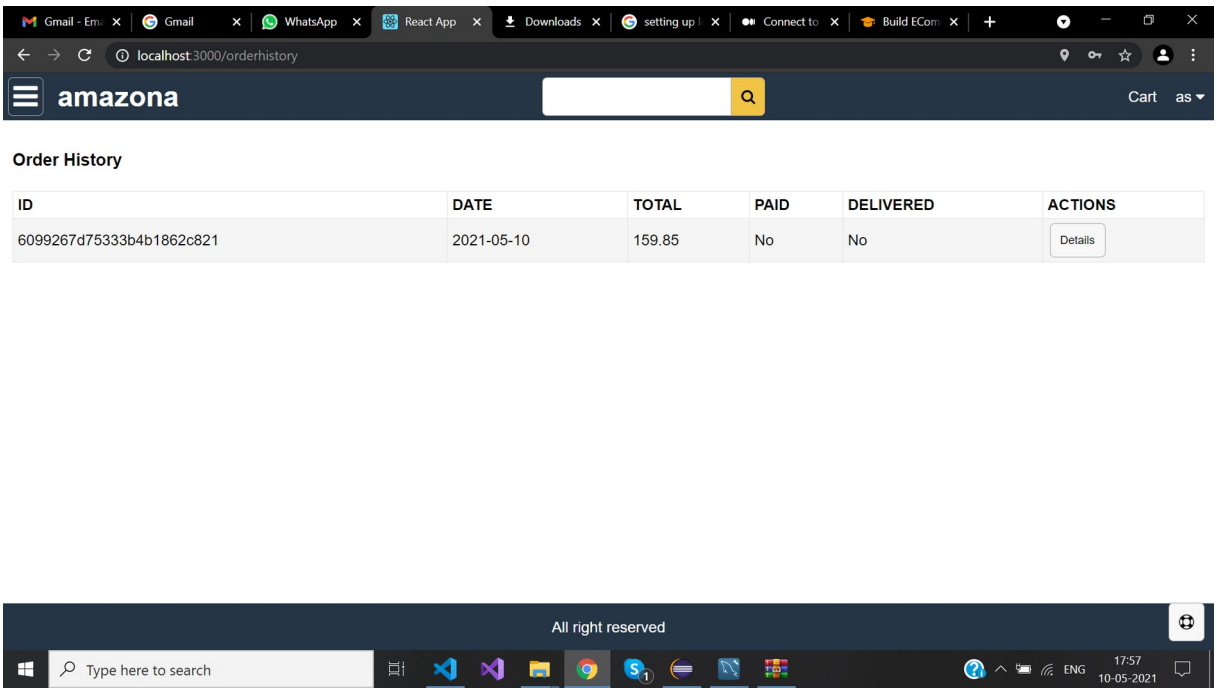

All right reserved
Type here to search
ENG
17:55
10-05-2021

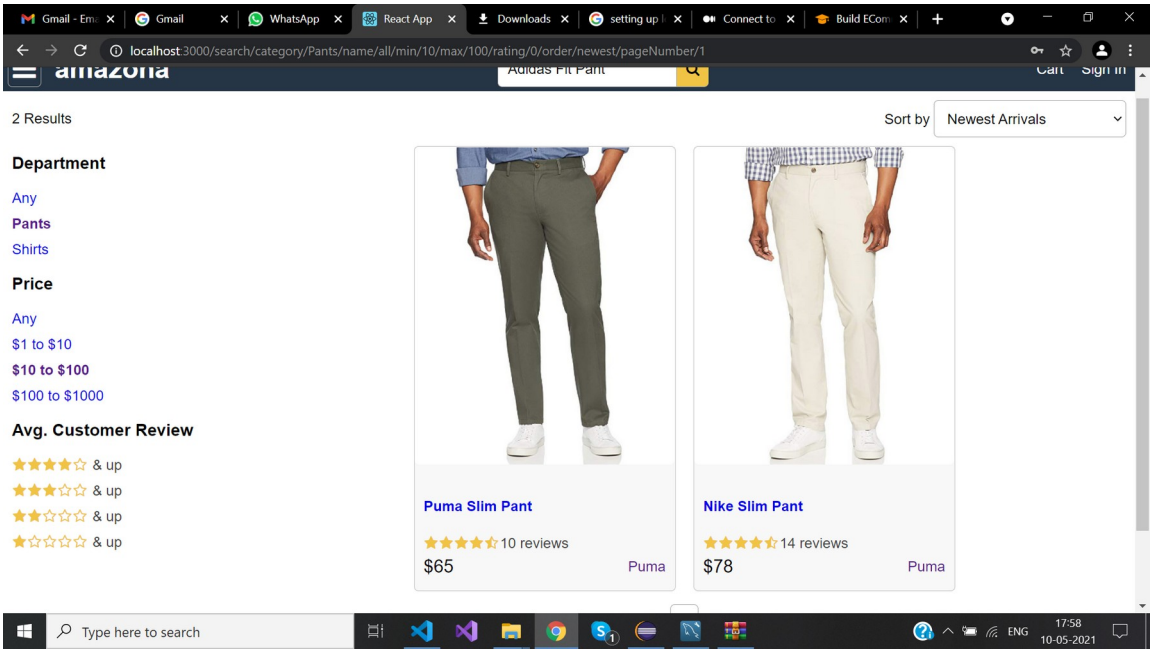

localhost:3000/shipping
Full Name
as
Address
H.No.-C-193, Pushpanjali Upvan Colony,, Lajpat Nagar Road
City
Mathura
Postal Code
281004
Country
India
Location
Choose On Map
Continue
Type here to search
ENG
17:56
10-05-2021

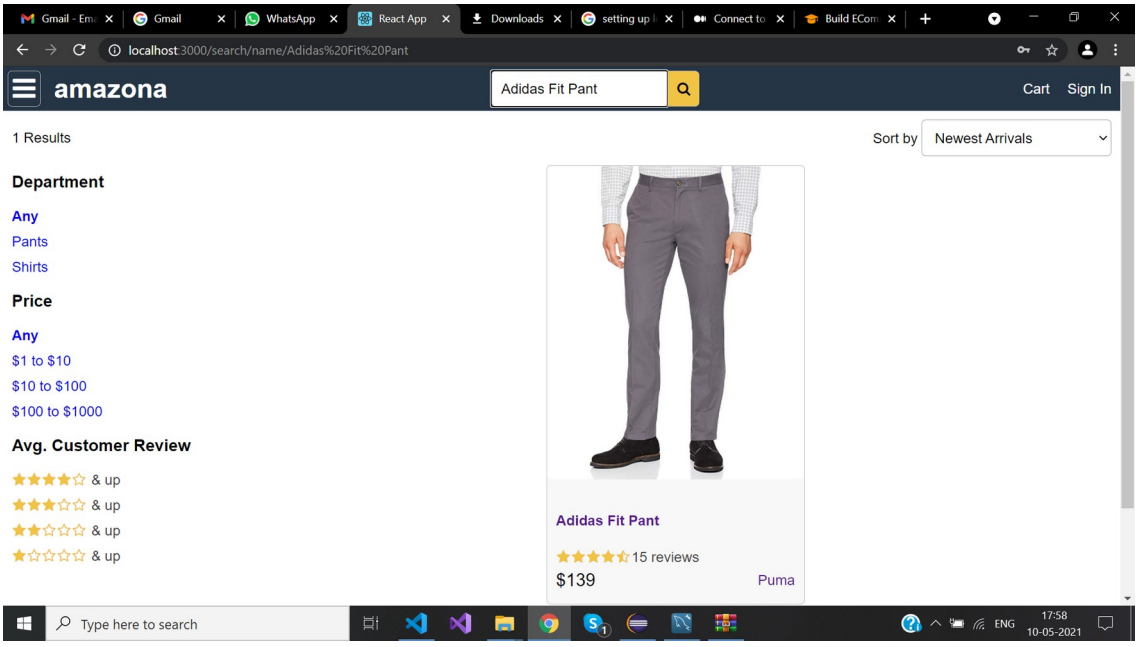

localhost:3000/payment
amazona
Cart 1
as
Sign-In
Shipping
Payment
Place Order
Payment Method
PayPal
Stripe
Continue
All right reserved
Type here to search
ENG
17:56
10-05-2021


localhost:3000/order/6099267d75333b4b1862c821
Order 6099267d75333b4b1862c821
Shippring
Name: as
Address: H.No.-C-193, Pushpanjali Upvan Colony,, Lajpat Nagar Road,Mathura, 281004,India
Not Delivered
Payment
Method: PayPal
Not Paid
Order Items
Adidas Fit Pant
1 x $139 = $139
Order Summary
Items $139.00
Shipping $0.00
Tax $20.85
Order Total $159.85
PayPal
Debit or Credit Card
Powered by PayPal
Type here to search
ENG
17:56
10-05-2021


localhost:3000/placeorder
amazona
Cart 1
as
Sign-In
Shipping
Payment
Place Order
Shipping
Name: as
Address: H.No.-C-193, Pushpanjali Upvan Colony,, Lajpat Nagar Road,Mathura, 281004,India
Payment
Method: PayPal
Order Items
Adidas Fit Pant
1 x $139 = $139
Order Summary
Items $139.00
Shipping $0.00
Tax $20.85
Order Total $159.85
Place Order
All right reserved
Type here to search
ENG
17:56
10-05-2021


localhost:3000/orderhistory
amazona
Cart
as
Order History
ID
DATE
TOTAL
PAID
DELIVERED
ACTIONS
6099267d75333b4b1862c821
2021-05-10
159.85
No
No
Details
All right reserved
Type here to search
ENG
17:57
10-05-2021


localhost:3000/search/category/Pants/name/all/min/10/max/100/rating/0/order/newest/pageNumber/1
amazona
Adidas Fit Pant
Cart
Sign In
2 Results
Sort by
Newest Arrivals
Department
Any
Pants
Shirts
Price
Any
$1 to $10
$10 to $100
$100 to $1000
Avg. Customer Review
& up
& up
& up
& up
Puma Slim Pant
10 reviews
$65
Puma
Nike Slim Pant
14 reviews
$78
Puma
Type here to search
ENG
17:58
10-05-2021


localhost:3000/search/name/Adidas%20Fit%20Pant
amazona
Adidas Fit Pant
Cart
Sign In
1 Results
Sort by
Newest Arrivals
Department
Any
Pants
Shirts
Price
Any
$1 to $10
$10 to $100
$100 to $1000
Avg. Customer Review
& up
& up
& up
& up
Adidas Fit Pant
15 reviews
$139
Puma
Type here to search
ENG
17:58
10-05-2021

Chapter 7. References

1. www.react.io
2. www.w3schools.com
3. www.nodejs.org
4. www.monogodb.com
5. www.youtube.com
6. www.npmjs.com
7. www.bootstrap.com
8. www.stackoverflow.com
9. www.googlefonts.com