Additional Packages used

1. Material UI core and Material UI Icons – for Icons
2. react-router-dom – used for navigating to other pages
3. react-currency-format – used to format curreny

Snippets:

1. To print a component multiple time,

{Array(count).fill().map(() => ( <p>⭐</p> ))}

1. Click on image and redirect to a page (using {Link} from react-router-dom)

<Link to='/checkout'>

<img className='header\_\_logo' src={logo} />

</Link>

1. Navigation in the Header Component using react-router-dom

import { BrowserRouter as Router, Route, Routes } from "react-router-dom";

<Router>

        <Routes>

          <Route path="/login" element={<Login />} />

          <Route path="/checkout" element={<Checkout />} />

          <Route path="/" element={<Home />} />

        </Routes>

</Router>

Using Context API / Redux:

* When the “Add to basket” is click the product data will pushed to “data layer”
* This data layer is React context API / Redux
* From this data layer we can get the info to other components (i.e., Cart)

Refer StateProvider.js and reducer.js for Context API Code

**StateProvider.js** is a template which created the data layer and provides access to data layer

**reducer.js** is used to manipulate the data layer

**reducer.js** will contain the contain for adding the item to cart and delete the item from cart

**Firebase Setup:**

1. Create new Project in firebase and create a new web app in the project
2. Navigate to Project Settings and scroll down to **SDK setup and configuration**
3. Click on **Config** and copy the configuration object shown
4. Create a new file **firebase.js** in project folder and paste it
5. Install firebase in the project folder using the commands

For the first time use this to install firebase globally:

**(npm install -g firebase-tools)**

Next, In the project directory,

**npm i firebase**

**Check this link for user register and sign in, sign out process using firebase**

<https://firebase.google.com/docs/auth/web/start>

Deploy app in firebase:

firebase login

firebase init

* Select **Hosting**
* use existing directory
* public directory? **build**
* single page app? **y**

npm run build

(if any changes are made afterwards, then **npm run build** should run again)

firebase deploy

**PART-3**

Stripe functionality

Build Checkout/Payment Page with stripe frontend

Deploy Cloud functions to process stripe payment (serverless architecture)

Build “My Orders” Page

Store orders for logged in users

Firebase for real-time database (Firestore)

**Stripe**

npm i @stripe/stripe-js

npm i @stripe/react-stripe-js

refer stripe documentation for creating an stripe object and use of useStripe() and useElement() Hooks

need to install axios to get the **clientSecret** from stripe

axios popular is a fetching library. We can fetch, Post request , allows to interact with API

npm i axios

setup cloud functions using firebase

<https://www.youtube.com/watch?v=RDV3Z1KCBvo&t=23630s>

functions folder will be created where complete backend will be set up

After that an express app is developed and will be hosted on cloud functions

Install backend dependencies inside functions folder

1. express – npm i express
2. cors – npm i cors
3. stripe – npm i stripe

To setup API we need 4 things

1. App Configuration

2. Middlewares

3. API Routes

4. Listen Command

**Once the payment is success, navigate to order page**

We need to push the order details into database with respect to login user

Steps to setup **firestore**:

* Go to Firebase Console
* Navigate to **FireStore Database** and click on Create Database, **Start in Test mode** and select any location and Enable

Once it’s done we are gonna have a **collection of users** and **each user** will have **collection of orders**

In **firebase.js** we have 2 instances of **db** and **auth,**

import **db** in Payment.js and after each successful payment push the order details into the order collection

db

  .collection('users')

  .doc(user?.id)

  .collection('orders')

  .doc(paymentIntent.id)

  .set({

  basket: basket,

      amount: paymentIntent.amount,

      created: paymentIntent.created

  })

This is how we reach into the data base

1. Accessing **user** collection
2. Reaching specific user using **user id**
3. Accessing **orders** collection
4. Reaching specific order using **payment id**
5. then setting data like **basket, amount, created date**

if there is no existing order it will create a new one using payment id as reference

**Order Page rendering with order details**

Once payment is successful, it will navigate to orders page

In the orders page using useEffect() hook data is queried from firestore db and updated in a local orders state

Then, using a separate Order Component, all the order will be displayed based on logged in user

Dependency for frontend component

Moment – its library for parsing any time stamps

npm install moment