MAJOR PROJECT PRESENTATION

AMAZON CLONE

INTRODUCTION

This Amazon Clone is built using React JS library with CSS, JS functionality and Firebase.

This app includes Firebase authentication system, Real-time database and Stripe Payment Gateway Integration.

TECHNOLOGIES USED

- Language: CSS, Modern JavaScript (React JS)
- Framework: JavaScript (React-16.14), Material UI

- IDE/Editors/Debuggers: VS Code and Chrome Debugger
- Firebase

What is React ?

- React is an open-source, front-end, JavaScript library for building user interfaces or UI components.
- It is maintained by Facebook.
- React can be used as a base in the development of single-page or mobile applications.

Why React ?

- Easy creation of dynamic applications.
- Loads the website very fast.
- Works very well with search engine optimization.
- Improves performance
- Reusable components.

Firebase

- Firebase is a mobile and web application development platform developed by Firebase, in 2011. Later, it was acquired by Google in 2014.
- It is a Backend-as-a-service which helps you to build better apps, websites, or games regardless of server-side coding, API or backend data storage.
- This includes things like analytics, authentication, databases, configuration, file storage, push messaging, and the list goes on.

Landing Page

App.js

src > App.js > ...

```
1  import React, { useEffect } from "react";
2  import "./App.css";
3  import Header from "./Header";
4  import Home from "./Home";
5  import { BrowserRouter as Router, Switch, Route } from "react-router-dom";
6  import Checkout from "./Checkout";
7  import Login from "./Login";
8  import Payment from "./Payment";
9  import Orders from "./Orders";
10 import { auth } from "./firebase";
11 import { useStateValue } from "./StateProvider";
12 import { loadStripe } from "@stripe/stripe-js";
13 import { Elements } from "@stripe/react-stripe-js";
14
15 const promise = loadStripe(
16   "pk_test_51J4v9LSI9BQADIiBwa31btxfBBpIY3LBrghmQJHsEk16TZsDGVieEq1eKULZv9Snm47HBdIul13ZStNPxNmPszO3007VJKNGCs"
17 );
18
19 function App() {
20   const [{}, dispatch] = useStateValue();
21
22   useEffect(() => {
23     // will only run once when the app component loads...
24
25     auth.onAuthStateChanged((authUser) => {
26       console.log("THE USER IS >>> ", authUser);
27
28       if (authUser) {
29         // the user just logged in / the user was logged in
30
31         dispatch({
```

Rendering Components

index.js

src > index.js

```
1  import React from "react";
2  import ReactDOM from "react-dom";
3  import "./index.css";
4  import App from "./App";
5  import * as serviceWorker from "./serviceWorker";
6  import reducer, { initialState } from "./reducer";
7  import { StateProvider } from "./StateProvider";
8
9  ReactDOM.render(
10   <React.StrictMode>
11     <StateProvider initialState={initialState} reducer={reducer}>
12       <App />
13     </StateProvider>
14   </React.StrictMode>,
15   document.getElementById("root")
16 );
17
18 // If you want your app to work offline and load faster, you can change
19 // unregister() to register() below. Note this comes with some pitfalls.
20 // Learn more about service workers: https://bit.ly/CRA-PWA
21 serviceWorker.unregister();
22
```

Products Build

Product.js

src > Product.js > ...

```
1  import React from "react";
2  import "./Product.css";
3  import { useStateValue } from "./StateProvider";
4
5  function Product({ id, title, image, price, rating }) {
6    const [{ basket }, dispatch] = useStateValue();
7
8    const addToBasket = () => {
9      // dispatch the item into the data layer
10     dispatch({
11       type: "ADD_TO_BASKET",
12       item: {
13         id: id,
14         title: title,
15         image: image,
16         price: price,
17         rating: rating,
18       },
19     });
20   };
21
22   return (
23     <div className="product">
24       <div className="product__info">
25         <p>{title}</p>
26         <p className="product__price">
27           <small>$</small>
28           <strong>{price}</strong>
29         </p>
30         <div className="product__rating">
31           {Array(rating)
```

Payment Build

Payment.js

src > Payment.js > ...

```
1  import React, { useState, useEffect } from 'react';
2  import './Payment.css';
3  import { useStateValue } from "./StateProvider";
4  import CheckoutProduct from "./CheckoutProduct";
5  import { Link, useHistory } from "react-router-dom";
6  import { CardElement, useStripe, useElements } from "@stripe/react-stripe-js";
7  import CurrencyFormat from "react-currency-format";
8  import { getBasketTotal } from "./reducer";
9  import axios from './axios';
10 import { db } from "./firebase";
11
12 function Payment() {
13     const [{ basket, user }, dispatch] = useStateValue();
14     const history = useHistory();
15
16     const stripe = useStripe();
17     const elements = useElements();
18
19     const [succeeded, setSucceeded] = useState(false);
20     const [processing, setProcessing] = useState("");
21     const [error, setError] = useState(null);
22     const [disabled, setDisabled] = useState(true);
23     const [clientSecret, setClientSecret] = useState(true);
24
25     useEffect(() => {
26         // generate the special stripe secret which allows us to charge a customer
27         const getClientSecret = async () => {
28             const response = await axios({
29                 method: 'post',
30                 // Stripe expects the total in a currencies subunits
31                 url: `/payments/create?total=${getBasketTotal(basket) * 100}`
```

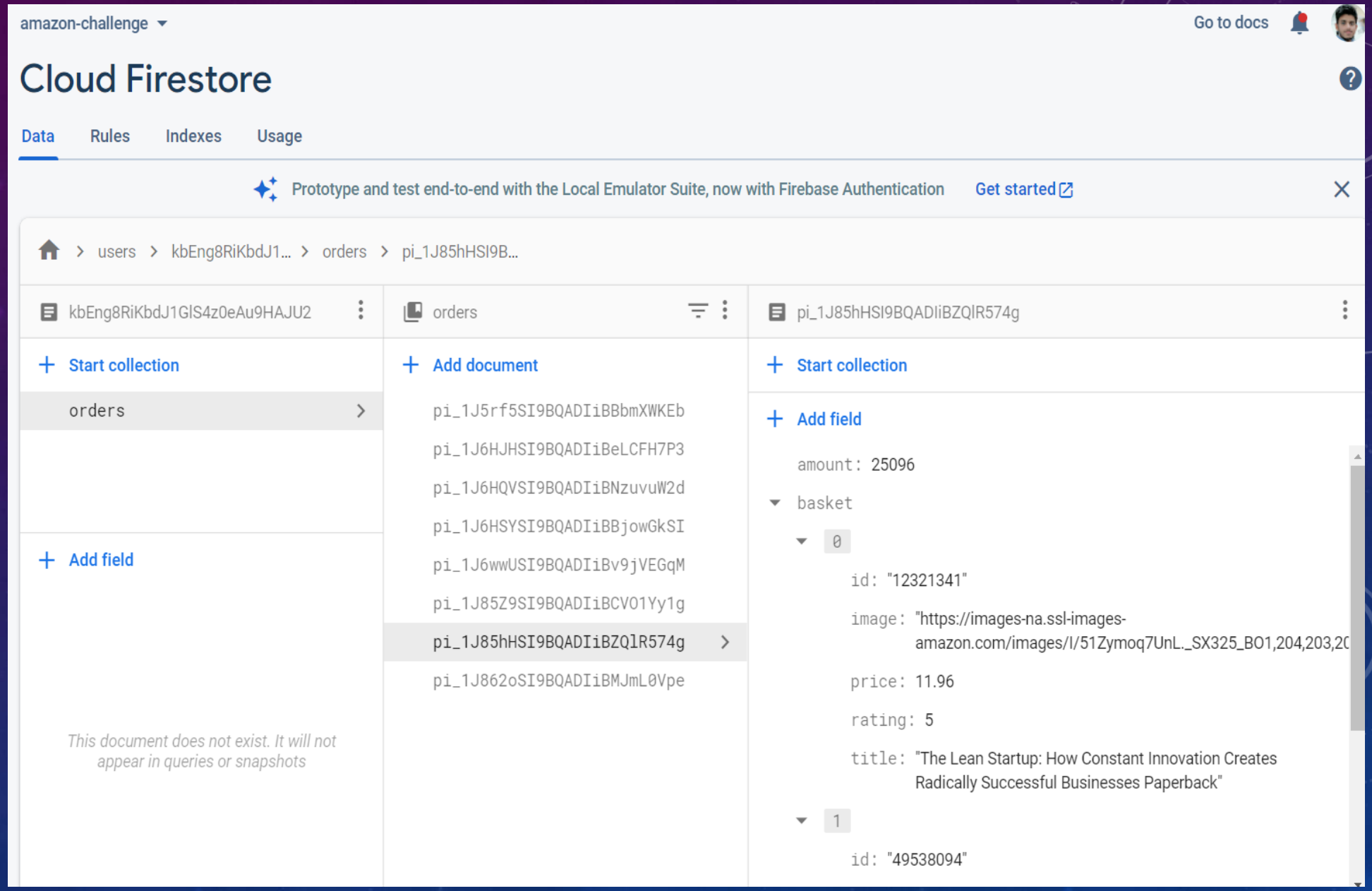
Order Build

Orders.js

src > Orders.js > ...

```
1  import React, { useState, useEffect } from 'react';
2  import { db } from "./firebase";
3  import './Orders.css'
4  import { useStateValue } from "./StateProvider";
5  import Order from './Order'
6
7  function Orders() {
8    const [{ basket, user }, dispatch] = useStateValue();
9    const [orders, setOrders] = useState([]);
10
11   useEffect(() => {
12     if(user) {
13         db
14         .collection('users')
15         .doc(user?.uid)
16         .collection('orders')
17         .orderBy('created', 'desc')
18         .onSnapshot(snapshot => (
19             setOrders(snapshot.docs.map(doc => ({
20                 id: doc.id,
21                 data: doc.data()
22             })))
23         ))
24     } else {
25         setOrders([])
26     }
27
28   }, [user])
29
30     return (
31         <div className='orders'>
```

Cloud Firestore



Stripe Payment Gateway

- Stripe Payments is a payment processing platform. It allows you to transfer money from a customer's bank account into your business's account by way of a credit or debit card transaction.
- Stripe has a reputation for taking security very seriously, and it appears to be well-earned. Stripe is a certified PCI Service Provider Level 1, which means it meets the most stringent security standards in the industry.
- Stripe plays nicely with popular server-side languages/frameworks, with particular care given to Ruby, Python, PHP, Java, Node.js, Go, and .NET. The minimal setup up for Stripe is actually pretty simple.

Conclusion

- I have learnt the implementation and technology used for creating this Amazon Clone.
- Concluding this project, of course there is always scope for future development and adding to it.
- I would wish to add search functionality and description page for products.
- I would wish to host this project, upon getting a cheap hosting service.

Source

- https://www.udemy.com/course/react-the-complete-guide
- https://reactjs.org/
- https://cleverprogrammer.teachable.com/courses
- https://www.google.com/
- https://stripe.com/docs
- https://firebase.google.com/

Thank You !!