

MINI PROJECT -II

(2021-22)

“G-MAIL CLONE”

Project Report



Institute of Engineering & Technology

Submitted By -

Raman (191500634)

Vaibhav Choudhary(191500885)

Abhas Raghuvanshi(191500008)

Rajeshwar Gupta(191500630)

Rajat Pratap Singh(191500628)

Under the Supervision Of

Mr. Abhishek Tiwari

Technical Trainer

Department of Training & Development



Department of Computer Engineering and Applications

GLA University, Mathura

17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuhan

Declaration

We hereby declare that the work which is being presented in the Bachelor of Technology, "GMAIL CLONE", in partial fulfillment of the requirements for mini- Project viva voce, is an authentic record of our own work carried by the team members under the supervision of our mentor Mr. Mayank Saxena.

Group Members : Raman(191500634)

Rajeshwar Gupta(191500630)
Abhas Raghuvanshi(191500008)
Vaibhav Choudhary(191500885)
Rajat Pratap Singh(191500628)

Course : B.Tech (Computer Science and Engineering)

Year : III

Semester : VIth

Supervised To :

Mr. Abhishek Tiwari, Technical Trainer,

Training & Development Department , GLA University

Certificate

This is to certify that the above statements made by the candidates are correct to the best of my/our knowledge and belief.

Supervisor

Mr Abhishek Tiwari

Technical Trainer

T&D Department , GLA University

Project Coordinator

(Mr. Mandeep Singh)

Program Coordinator

(Mr. Shashi Shekhar)

Contents

1. Introduction:

About the project

Acknowledgment.....

Abstract...

2. Technologies Used:

- REACT JS
- FIREBASE
- MATERIAL UI

3. List of Figures.....

- Code
- Output

4. Software Testing.....

5. Conclusion....

6. Bibliography...

Introduction

About the Project

Website cloning is the process of creating a replica of your existing website design or content to create a new website with ease. Website cloning lets developers and designers create blueprints, test compatibility, and perform updates safely before implementing the changes on your live website.

In this guide, I will cover a few of the use cases where website cloning can solve the problem and then discuss several ways to clone a WordPress website.

Here are a few use-cases when you might need to clone a website.

- Compatibility Test
- Move Your Website to a New Server
- Back up Your Website
- Clone Website for a Similar Project
- Compatibility Test

There are many reasons you might want to clone websites on your cPanel server. Because the copies are identical to an existing website, cloning helps deploy testing and staging sites. It's also a quick way to set up a new site using the old one as a baseline configuration. Site owners often use cloned sites to give designers and developers access to a working environment that behaves like the live environment.

Acknowledgement

It gives us a great sense of pleasure to present the synopsis of the B.Tech Mini Project- "GMAIL CLONE" undertaken during B.Tech IIIrd year.

This Mini Project Itself is going to be an acknowledgment to the inspiration, drive and technical assistance will be contributed to it by many individuals. I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to "Mr. Abhishek Tiwari(Technical Trainer)" for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards my parents & member of GLA University for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons forgiving me such attention and time.

My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

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 co- operation.

Raman
Rajeshwar Gupta
Abhas Raghuvanshi
Vaibhav Choudhary
Rajat Pratap Singh

(191500634)
(191500630)
(191500008)
(191500885)
(191500628)

GMAIL CLONE

Abstract

Website cloning refers to the copying or modification of an existing website design or script to create a new website. Website cloning allows designers to create websites without the need to write scripts from scratch.

In this era of digital connectivity, it is difficult to ensure that the original content you place on your website is not distributed or used elsewhere without your knowledge.

As part of an effort to combat this form of copyright violation, some websites feature "protected" content. The term is used to describe websites where it is impossible to right-click text or select on-page content for the purpose of copying and pasting it.

If you need to copy text from a protected website, you have to take a creative approach.

First is to develop a website from scratch and other is to use clone scripts. Clones make it possible to replicate the concept of a well-known website and add improved design and functionality as per your requirement. In recent times, many entrepreneurs are using this idea to start their businesses with ease.

Online businesses choose to get a clone when they want to build an application that is already popular among users. This enables them to leverage the benefits offered by such platforms at an affordable price.

Technologies Used

REACT JS

React (also known as **React.js** or **ReactJS**) is a free and open-source front-end JavaScript library for building user interfaces based on UI components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. React can be used as a base in the development of single-page, mobile, or server-rendered applications with frameworks like Next.js. However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality

Declarative

React adheres to the declarative programming paradigm. Developers design views for each state of an application, and React updates and renders components when data changes. This is in contrast with imperative programming.

Components

React code is made of entities called components. These components are reusable and must be formed in the SRC folder following the Pascal Case as its naming convention (capitalize camelCase). Components can be rendered to a particular element in the DOM using the React

DOM library. When rendering a component, one can pass the values between components through "props":

-Functional components

Function components are declared with a function that then returns some JSX.

-Class-based components

Class-based components are declared using ES6 classes.

Virtual DOM

Another notable feature is the use of a virtual Document Object Model, or virtual DOM. React creates an in-memory data-structure cache, computes the resulting differences, and then updates the browser's displayed DOM efficiently. This process is called **reconciliation**. This allows the programmer to write code as if the entire page is rendered on each change, while the React libraries only render subcomponents that actually change. This selective rendering provides a major performance boost. It saves the effort of recalculating the CSS style, layout for the page and rendering for the entire page.

JSX

JSX, or JavaScript Syntax Extension, is an extension to the JavaScript language syntax. Similar in appearance to HTML, JSX provides a way to structure component rendering

using syntax familiar to many developers. React components are typically written using JSX, although they do not have to be (components may also be written in pure JavaScript). JSX is similar to another extension syntax created by Facebook for [PHP](#) called [XHP](#).

An example of JSX code:

```
class App extends React.Component {  
  render() {  
    return (  
      <div>  
        <p>Header</p>  
        <p>Content</p>  
        <p>Footer</p>  
      </div>  
    );  
  }  
}
```

Common Idioms

React does not attempt to provide a complete "application library". It is designed specifically for building user interfaces^[3] and therefore does not include many of the tools some developers might consider necessary to build an application. This allows the choice of whichever libraries the developer prefers to accomplish tasks such as performing network access or local data storage. Common patterns of usage have emerged as the library matures.

React was created by Jordan Walke, a software engineer at Facebook, who released an early prototype of React called "FaxJS". He was influenced by XHP, an HTML component library for PHP. It was first deployed on Facebook's News Feed in 2011 and later on Instagram in 2012. It was open-sourced at JSConf US in May 2013.

React Native, which enables native Android, iOS, and UWP development with React, was announced at Facebook's React Conf in February 2015 and open-sourced in March 2015.

On April 18, 2017, Facebook announced React Fiber, a new set of internal algorithms for rendering, as opposed to React's old rendering algorithm, Stack. React Fiber was to become the foundation of any future improvements and feature development of the React library. The actual syntax for programming with React does not change; only the way that the syntax is executed has changed. React's old rendering system, Stack, was developed at a time when the focus of the system on dynamic change was not understood. Stack was slow to draw complex animation, for example, trying to accomplish all of it in one chunk. Fiber breaks down animation into segments that can be spread out over multiple frames. Likewise, the structure of a page can be broken into segments that may be maintained and updated separately.

JavaScript functions and virtual DOM objects are called "fibers", and each can be operated and updated separately, allowing for smoother on-screen rendering.

On September 26, 2017, React 16.0 was released to the public.

On February 16, 2019, React 16.8 was released to the public. The release introduced React Hooks.

On August 10, 2020, the React team announced the first release candidate for React v17.0, notable as the first major release without major changes to the React developer-facing API.

Project status can be tracked via the core team discussion forum. However, major changes to React go through the Future of React repository issues and [pull requests](#). This enables the React community to provide feedback on new potential features, experimental APIs and JavaScript syntax improvements.

FIREBASE

Firestore is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in 2011. In 2014, Google acquired the platform and it is now their flagship offering for app development.

Firestore evolved from Envolv, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolv provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that were not chat messages. Developers were using Envolv to sync application data such as game state in real time across their users. Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it. They founded Firestore as a separate company in 2011 and it launched to the public in April 2012.

Firestore's first product was the Firestore Realtime Database, an API that synchronizes application data across iOS, Android, and Web devices, and stores it on Firestore's cloud. The product assists software developers in building real-time, collaborative applications.

In May 2012, a month after the beta launch, Firestore raised \$1.1 million in seed funding from venture capitalists Flybridge Capital Partners, Greylock Partners, Founder

Collective, and New Enterprise Associates. In June 2013, the company further raised \$5.6 million in Series A funding from Union Square Ventures and Flybridge Capital Partners.

In 2014, Firebase launched two products. Firebase Hosting and Firebase Authentication.^[7] This positioned the company as a mobile backend as a service.

In October 2014, Firebase was acquired by Google. A year later, in October 2015, Google acquired Divshot, an HTML5 web-hosting platform, to merge it with the Firebase team.

In May 2016, at Google I/O, the company's annual developer conference, Firebase introduced Firebase Analytics and announced that it was expanding its services to become a unified backend-as-a-service (BaaS) platform for mobile developers. Firebase now integrates with various other Google services, including Google Cloud Platform, AdMob, and Google Ads to offer broader products and scale for developers. Google Cloud Messaging, the Google service to send push notifications to Android devices, was superseded by a Firebase product, Firebase Cloud Messaging, which added the functionality to deliver push notifications to both iOS and web devices.

In July 2016, Google announced that it was acquiring the mobile developer platform LaunchKit, which specialized in app developer marketing, and would be folding it into the Firebase Growth Tools team. In January 2017, Google acquired Fabric and Crashlytics from Twitter to add those services to Firebase.

In October 2017, Firebase launched Cloud Firestore, a real-time document database as the successor product to the original Firebase Realtime Database.

Firebase has been claimed to be used by Google to track users without their knowledge. On July 14, 2020, a lawsuit was filed accusing Google of violating federal wire tap law and California privacy law. It stated that through Firebase, Google collected and stored user data, logging what the user was looking at in many types of apps, despite the user following Google's own instructions to turn off the web and app activity collected by the company. The lawsuit was dismissed on January of 2022.

The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync data between your users in realtime. NEW: Cloud Firestore enables you to store, sync and query app data at global scale.

7 Benefits of Using Firebase for your Mobile Application Development

- Cloud Messaging for Cross-Platform Apps. ...
- Higher Website Traffic with App Indexing. ...
- Swift and Secured Web Hosting. ...
- Higher Accessibility to Machine Learning APIs. ...
- Crash Reporting for Swift Bugs Fixing. ...
- Optimized App Performance.

Material UI

Material Design (codenamed **Quantum Paper**) is a design language developed by Google in 2014. Expanding on the "cards" that debuted in Google Now, Material Design uses more grid-based layouts, responsive animations and transitions, padding, and depth effects such as lighting and shadows. Google announced Material Design on June 25, 2014, at the 2014 Google I/O conference.

The main purpose of Material Design is the creation of a new visual language that combines principles of good design with technical and scientific innovation. Designer Matías Duarte explained that, "unlike real paper, our digital material can expand and reform intelligently. Material has physical surfaces and edges. Seams and shadows provide meaning about what you can touch." Google states that their new design language is based on paper and ink but implementation takes place in an advanced manner.

In 2018, Google detailed a revamp of the language, with a focus on providing more flexibility for designers to create custom "themes" with varying geometry, colors, and typography.

Google released Material Theme Editor exclusively for the macOS design application Sketch.

Material Design was to be gradually extended throughout Google's array of web and mobile products, providing a consistent experience across all platforms and applications. Google has also released application programming interfaces (APIs) for third-party developers to incorporate the design language into their applications.

The canonical implementation of Material Design for web application user interfaces is called Polymer. It consists of the Polymer library, a shim that provides a Web Components API for browsers that do not implement the standard natively, and an elements catalog, including the "paper elements collection" that features visual elements of the Material Design. Google also has created an accompanying icon set licensed under the Apache 2.0 license.¹

After the 2018 revamp, Google began redesigning most of their apps into a customized and adapted version of Material Design called the Google Material Theme, also dubbed "Material Design 2", which heavily emphasizes white space, rounded corners, colorful icons, bottom navigation bars, and utilizes a special size-condensed version of Google's proprietary Product Sans font called Google Sans.

At Google I/O in May 2021, Google announced a new concept on Android 12 known as "Material You" (also known as "Material Design 3"), emphasizing increased animation, larger buttons, and the ability for custom UI themes to be generated from the user's

wallpaper. Material You was gradually rolled out to various Google apps on older Android versions in the following months, and acted as a major focus on the Pixel 6 and Pixel 6 Pro smartphone series.

Application Programming Interface (API)

An **application programming interface (API)** is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software. A document or standard that describes how to build or use such a connection or interface is called an *API specification*. A computer system that meets this standard is said to *implement* or *expose* an API. The term API may refer either to the specification or to the implementation.

In contrast to a user interface, which connects a computer to a person, an application programming interface connects computers or pieces of software to each other. It is not intended to be used directly by a person (the end user) other than a computer programmer who is incorporating it into the software. An API is often made up of different parts which act as tools or services that are available to the programmer. A program or a programmer that uses one of these parts is said to *call* that portion of the API. The calls that make up the API are also known as subroutines, methods, requests, or endpoints. An API specification *defines* these calls, meaning that it explains how to use or implement them.

One purpose of APIs is to hide the internal details of how a system works, exposing only those parts a programmer will find useful and keeping them consistent even if the internal details later change. An API may be custom-built for a particular pair of systems, or it may be a shared standard allowing interoperability among many systems.

The term API is often used to refer to web APIs, which allow communication between computers that are joined by the internet. There are also APIs for programming languages, software libraries, computer operating systems, and computer hardware. APIs originated in the 1940s, though the term did not emerge until the 1960s and 1970s. Recent developments in utilizing APIs have led to the rise in popularity of microservices, which are ultimately loosely coupled services accessed through public APIs.

In building applications, an API simplifies programming by abstracting the underlying implementation and only exposing objects or actions the developer needs. While a graphical interface for an email client might provide a user with a button that performs all the steps for fetching and highlighting new emails, an API for file input/output might give the developer a function that copies a file from one location to another without requiring that the developer understand the file system operations occurring behind the scenes.

Web API

Web APIs are a service accessed from client devices (Mobile Phones, Laptop, etc.) to a web server using the Hypertext Transfer Protocol (HTTP). Client devices send a request in the form of an HTTP request, and are met with a response message usually in JavaScript Object Notation (JSON) or Extensible Markup Language (XML) format. Developers typically use Web APIs to query a server for a specific set of data from that server.

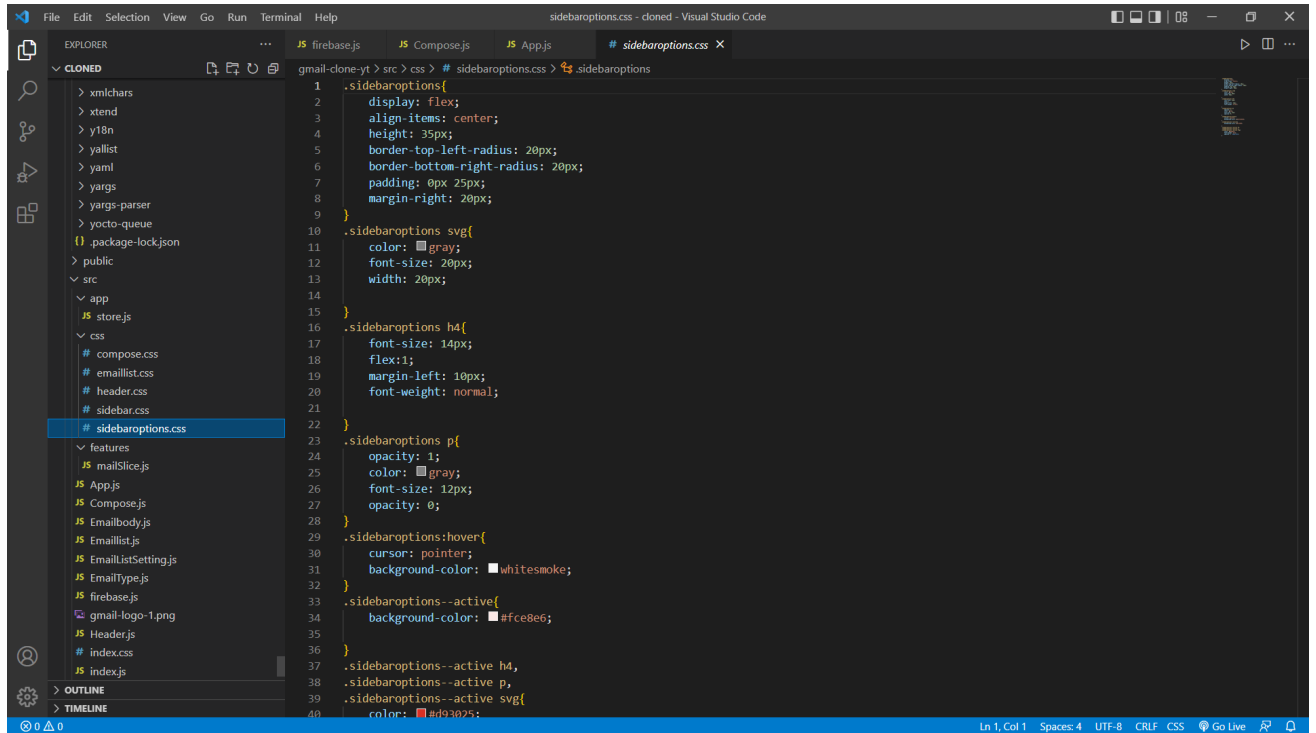
An example might be a shipping company API that can be added to an eCommerce-focused website to facilitate ordering shipping services and automatically include current shipping rates, without the site developer having to enter the shipper's rate table into a web database. While "web API" historically has been virtually synonymous with web service, the recent trend (so-called Web 2.0) has been moving away from Simple Object Access Protocol (SOAP) based web services and service-oriented architecture (SOA) towards more direct representational state transfer (REST) style web resources and resource-oriented architecture (ROA).^[33] Part of this trend is related to the Semantic Web movement toward Resource Description Framework (RDF), a concept to promote web-based ontology engineering technologies. Web APIs allow the combination of multiple APIs into new applications known as mashups. In the social media space, web APIs have allowed web communities to facilitate sharing content and

data between communities and applications. In this way, content that is created in one place dynamically can be posted and updated to multiple locations on the web. For example, Twitter's REST API allows developers to access core Twitter data and the Search API provides methods for developers to interact with Twitter Search and trends data.

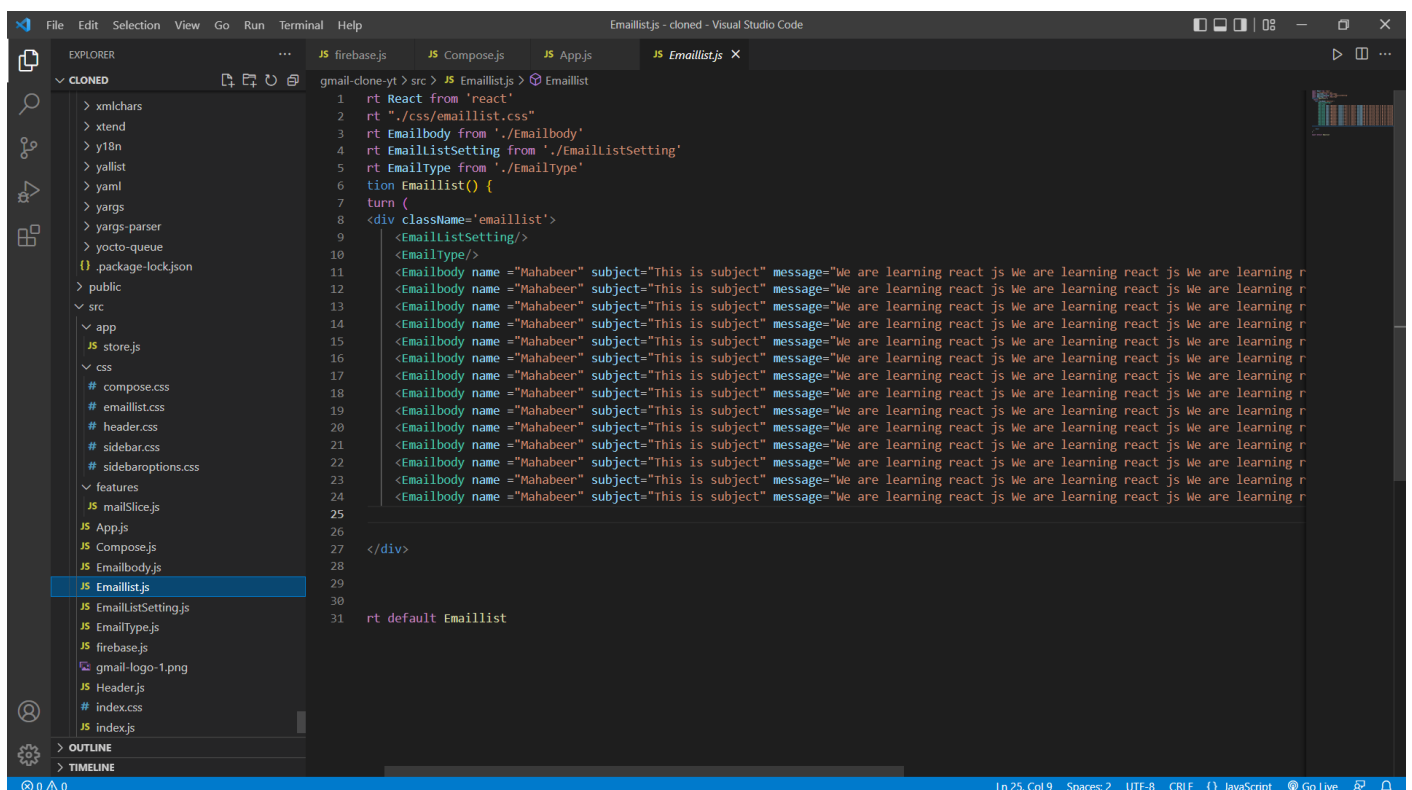
API security is very critical when developing a public facing API. Common threats include SQL injection, Denial-of-service attack (DoS), broken authentication, and exposing sensitive data. Without ensuring proper security practices bad actors can get access to information they should not have or even gain privileges to make changes to your server. Some common security practices include proper connection security using HTTPS, content security to mitigate data injection attacks, having an updated Robots exclusion standard, and requiring an API key to use your service. Many public facing API services require you to use an assigned API key, and will refuse to serve data without sending the key with your request.

List Of Figures

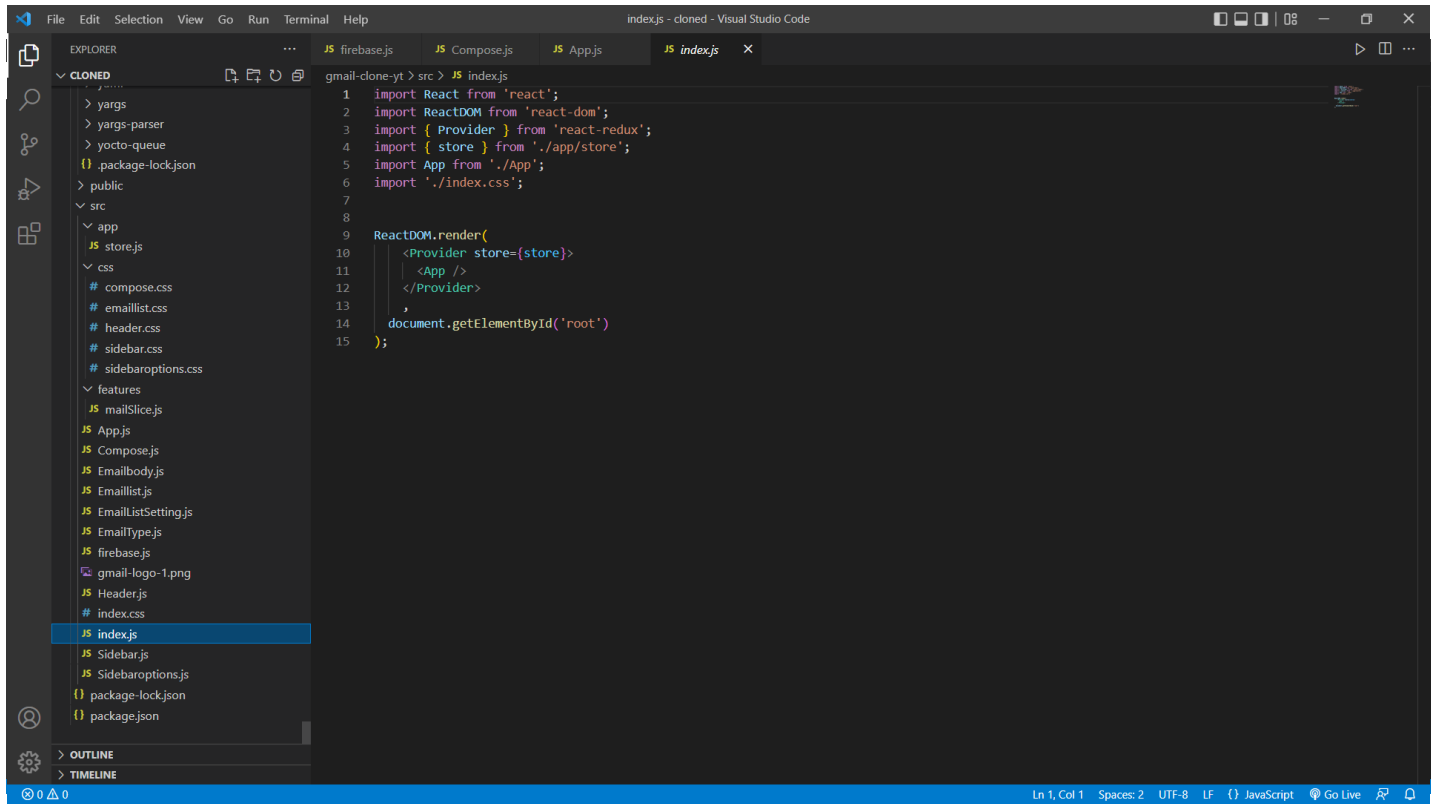
1-Sidebar options



2-Email list



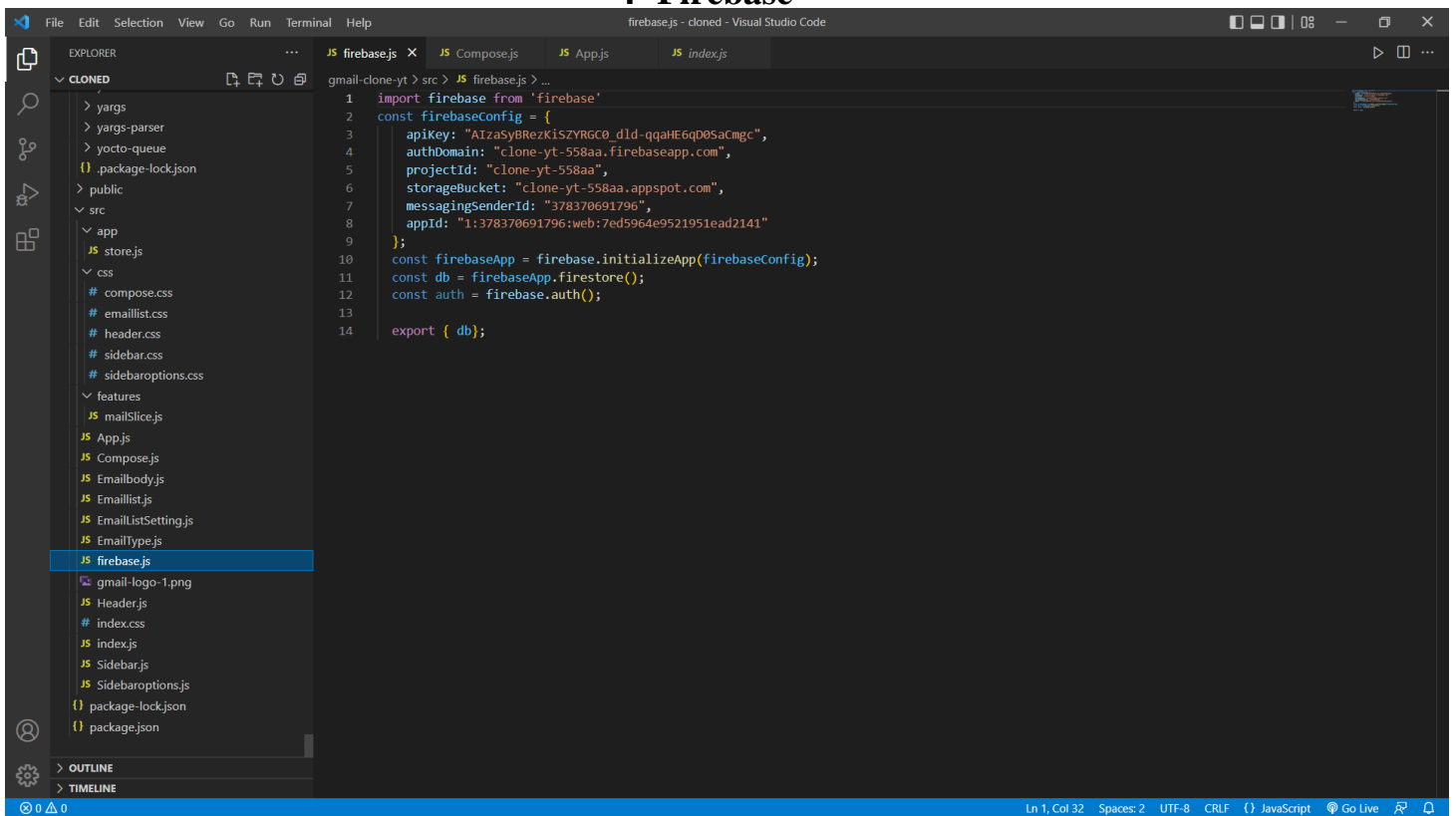
3-Index



The screenshot shows the Visual Studio Code interface with the Explorer panel on the left displaying the file structure of a project named 'gmail-clone-yt'. The file 'index.js' is selected in the 'src' directory. The main editor displays the content of 'index.js', which imports React, ReactDOM, Provider, store, App, and index.css, and renders the App component within a Provider.

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import { Provider } from 'react-redux';
4 import { store } from './app/store';
5 import App from './App';
6 import './index.css';
7
8
9 ReactDOM.render(
10   <Provider store={store}>
11     <App />
12   </Provider>
13   ,
14   document.getElementById('root')
15 );
```

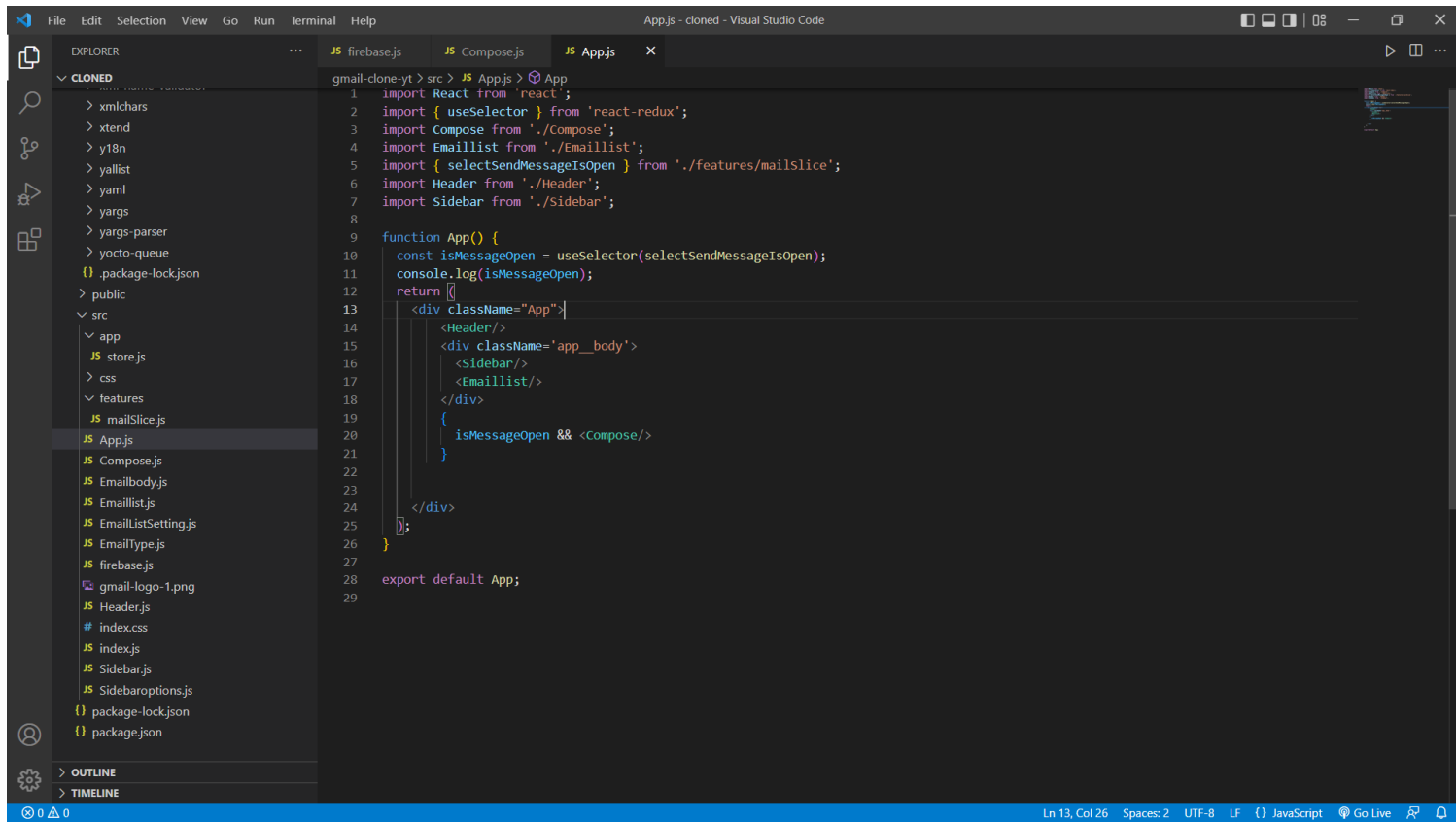
4- Firebase



The screenshot shows the Visual Studio Code interface with the Explorer panel on the left displaying the file structure of a project named 'firebasejs - cloned'. The file 'firebase.js' is selected in the 'src' directory. The main editor displays the content of 'firebase.js', which imports firebase, initializes the firebase app with a config object, and exports the db instance.

```
1 import firebase from 'firebase'
2 const firebaseConfig = {
3   apiKey: "AIzaSyB8RezKiSZVRGCO_dld-qqaHE6qD0SaCmgc",
4   authDomain: "clone-yt-558aa.firebaseio.com",
5   projectId: "clone-yt-558aa",
6   storageBucket: "clone-yt-558aa.appspot.com",
7   messagingSenderId: "378370691796",
8   appId: "1:378370691796:web:7ed5964e9521951ead2141"
9 };
10 const firebaseApp = firebase.initializeApp(firebaseConfig);
11 const db = firebaseApp.firestore();
12 const auth = firebase.auth();
13
14 export { db};
```

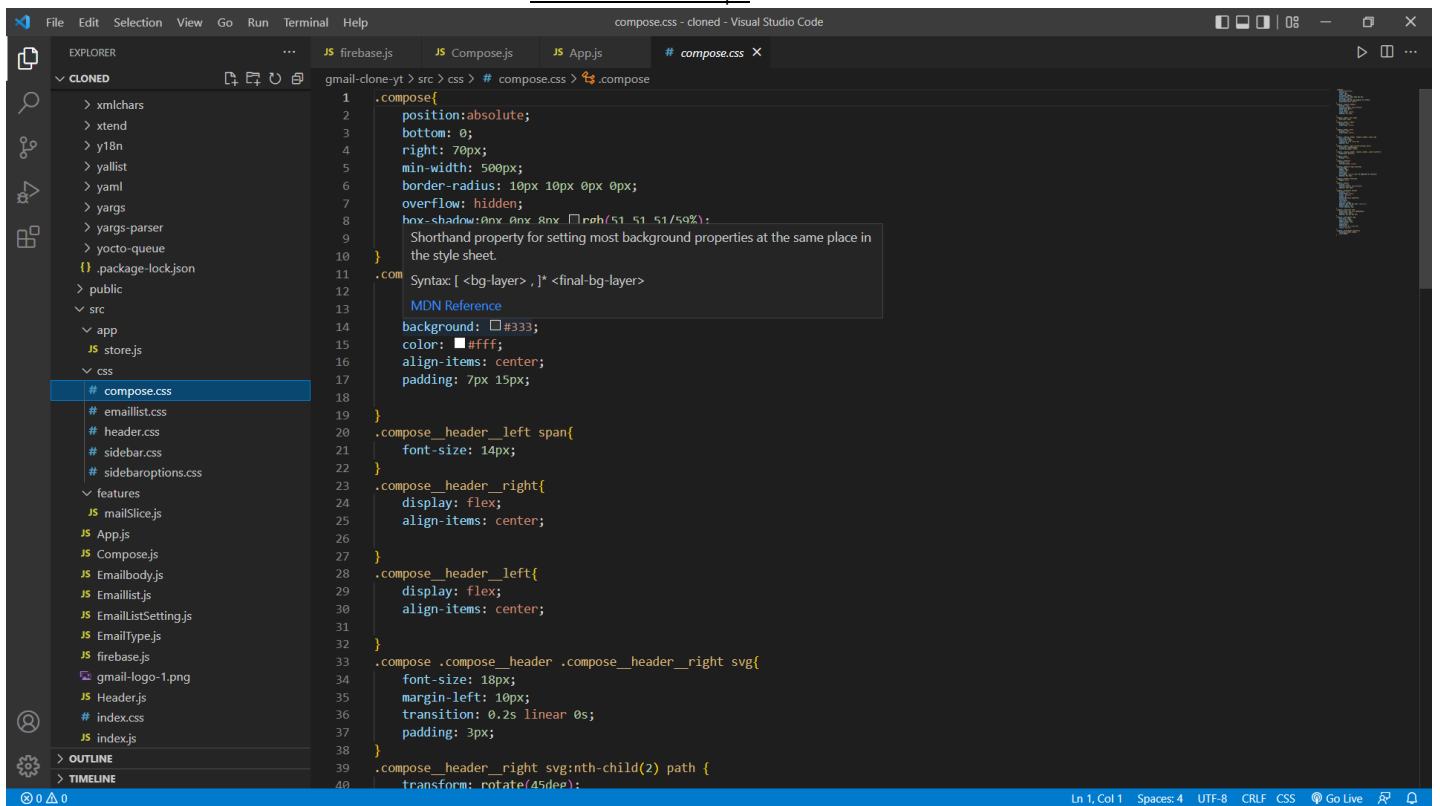

5-App.js



The screenshot shows the Visual Studio Code editor with the 'Appjs - cloned' project open. The Explorer panel on the left shows the file structure, with 'App.js' selected under the 'src' directory. The main editor displays the content of 'App.js', which is a React component. The code imports 'React' from 'react', 'useSelector' from 'react-redux', and various components from the project's 'Compose.js', 'Emaillist.js', 'features/maillSlice.js', 'Header.js', and 'Sidebar.js'. The 'App' function defines a stateful component that uses 'useSelector' to manage the 'isMessageOpen' state and returns a JSX element with a 'div' containing a 'Header', a 'div' with 'app_body' containing a 'Sidebar' and 'Emaillist', and a 'Compose' component. The 'Compose' component is rendered conditionally based on the 'isMessageOpen' state. The 'App' function is exported as the default export.

```
1 import React from 'react';
2 import { useSelector } from 'react-redux';
3 import Compose from './Compose';
4 import Emaillist from './Emaillist';
5 import { selectSendMessageIsOpen } from './features/maillSlice';
6 import Header from './Header';
7 import Sidebar from './Sidebar';
8
9 function App() {
10   const isMessageOpen = useSelector(selectSendMessageIsOpen);
11   console.log(isMessageOpen);
12   return (
13     <div className="App">
14       <Header/>
15       <div className='app_body'>
16         <Sidebar/>
17         <Emaillist/>
18       </div>
19       {
20         isMessageOpen && <Compose/>
21       }
22     </div>
23   );
24 }
25
26 export default App;
```

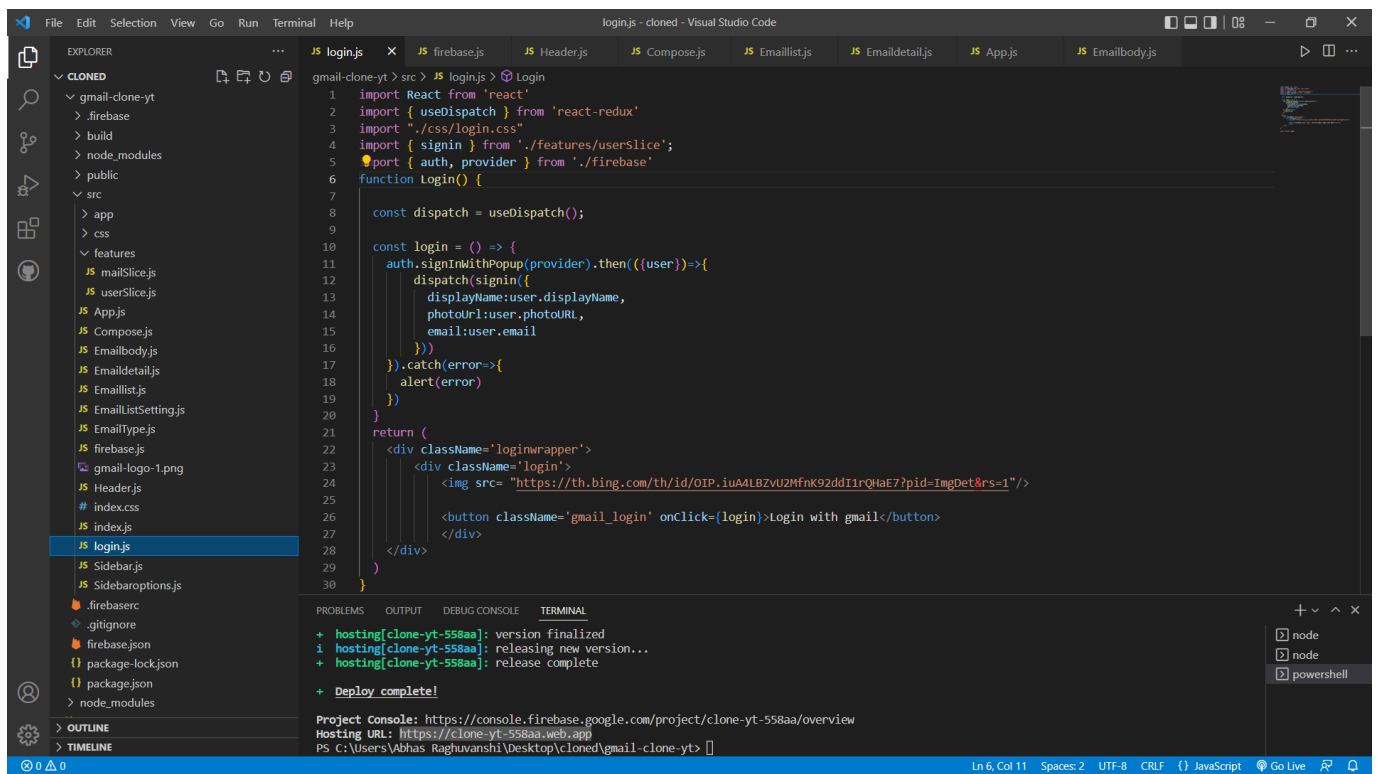
6-Email compose



The screenshot shows the Visual Studio Code editor with the 'compose.css - cloned' project open. The Explorer panel on the left shows the file structure, with 'compose.css' selected under the 'css' directory. The main editor displays the content of 'compose.css', which defines styles for the email compose interface. The styles include a base style for the '.compose' class, which sets absolute positioning, dimensions, border-radius, overflow, and box-shadow. It also defines styles for the '.compose_header_left span', '.compose_header_right', '.compose_header_left', and '.compose_header_right svg' elements. The '.compose_header_right svg' element is styled with a font-size of 18px, a margin-left of 10px, a transition of 0.2s linear 0s, and a padding of 3px. The '.compose_header_right svg:nth-child(2) path' element is styled with a transform of rotate(45deg).

```
1 .compose{
2   position:absolute;
3   bottom: 0;
4   right: 70px;
5   min-width: 500px;
6   border-radius: 10px 10px 0px 0px;
7   overflow: hidden;
8   box-shadow:0px 0px 8px 0px #ccc(51.51%);
9 }
10
11 .compose_header_left span{
12   font-size: 14px;
13 }
14
15 .compose_header_right{
16   display: flex;
17   align-items: center;
18 }
19
20 .compose_header_left{
21   display: flex;
22   align-items: center;
23 }
24
25 .compose .compose_header .compose_header_right svg{
26   font-size: 18px;
27   margin-left: 10px;
28   transition: 0.2s linear 0s;
29   padding: 3px;
30 }
31
32 .compose_header_right svg:nth-child(2) path {
33   transform: rotate(45deg);
34 }
```

7-Login Page



```
1 import React from 'react'
2 import { useDispatch } from 'react-redux'
3 import './css/login.css'
4 import { signin } from './features/userSlice';
5 import { auth, provider } from './firebase'
6 function Login() {
7
8   const dispatch = useDispatch();
9
10  const login = () => {
11    auth.signInWithPopup(provider).then(({user})=>{
12      dispatch(signin({
13        displayName:user.displayName,
14        photoUrl:user.photoURL,
15        email:user.email
16      })))
17    }).catch(error=>{
18      alert(error)
19    })
20  }
21  return (
22    <div className='loginwrapper'>
23      <div className='login'>
24        <img src= "https://th.bing.com/th/id/OIP.iuA4LBZVU2MfnK92ddI1rQHaE7?pid=ImpDet&rs=1"/>
25
26        <button className='gmail_login' onClick={login}>Login with gmail</button>
27      </div>
28    </div>
29  )
30 }
```

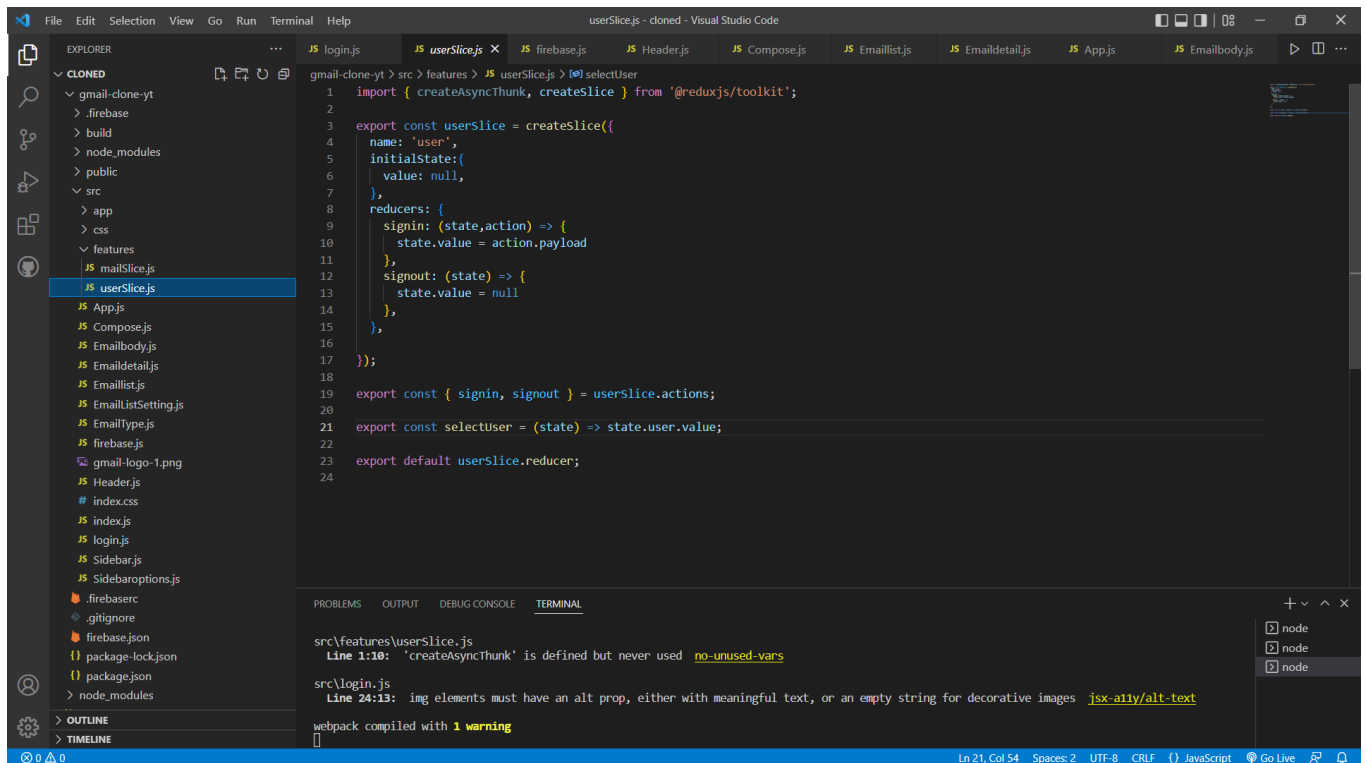
Problems OUTPUT DEBUG CONSOLE TERMINAL

+ hosting[clone-yt-558aa]: version finalized
i hosting[clone-yt-558aa]: releasing new version...
+ hosting[clone-yt-558aa]: release complete

+ Deploy complete!

Project Console: <https://console.firebase.google.com/project/clone-yt-558aa/overview>
Hosting URL: <https://clone-yt-558aa.web.app>
PS C:\Users\Abhas Raghuvanshi\Desktop\cloned\gmail-clone-yt>

8-User



```
1 import { createAsyncThunk, createSlice } from '@reduxjs/toolkit';
2
3 export const userSlice = createSlice({
4   name: 'user',
5   initialState:{
6     value: null,
7   },
8   reducers: {
9     signin: (state,action) => {
10       state.value = action.payload
11     },
12     signout: (state) => {
13       state.value = null
14     },
15   },
16 });
17
18 export const { signin, signout } = userSlice.actions;
19
20 export const selectUser = (state) => state.user.value;
21
22 export default userSlice.reducer;
```

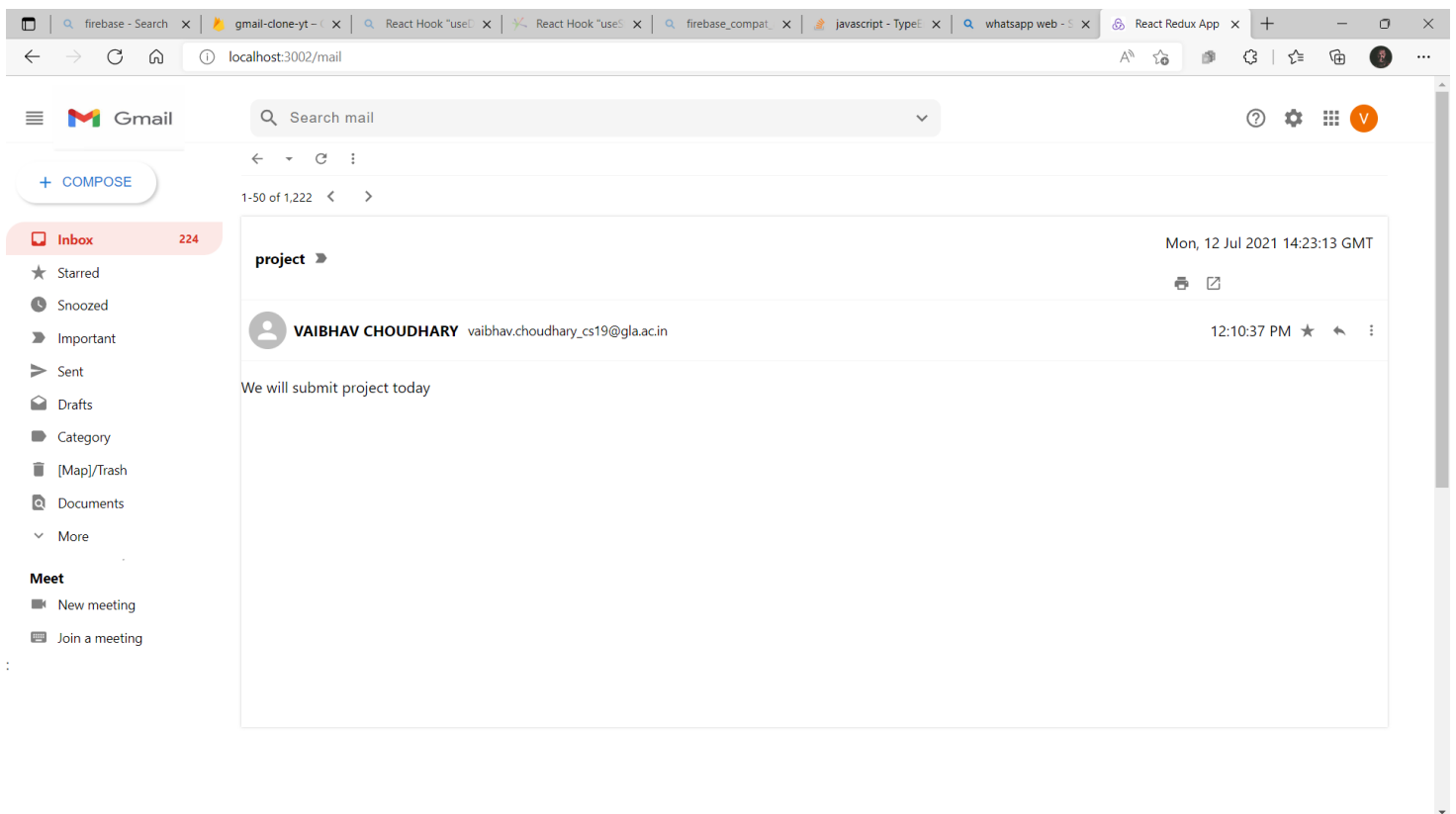
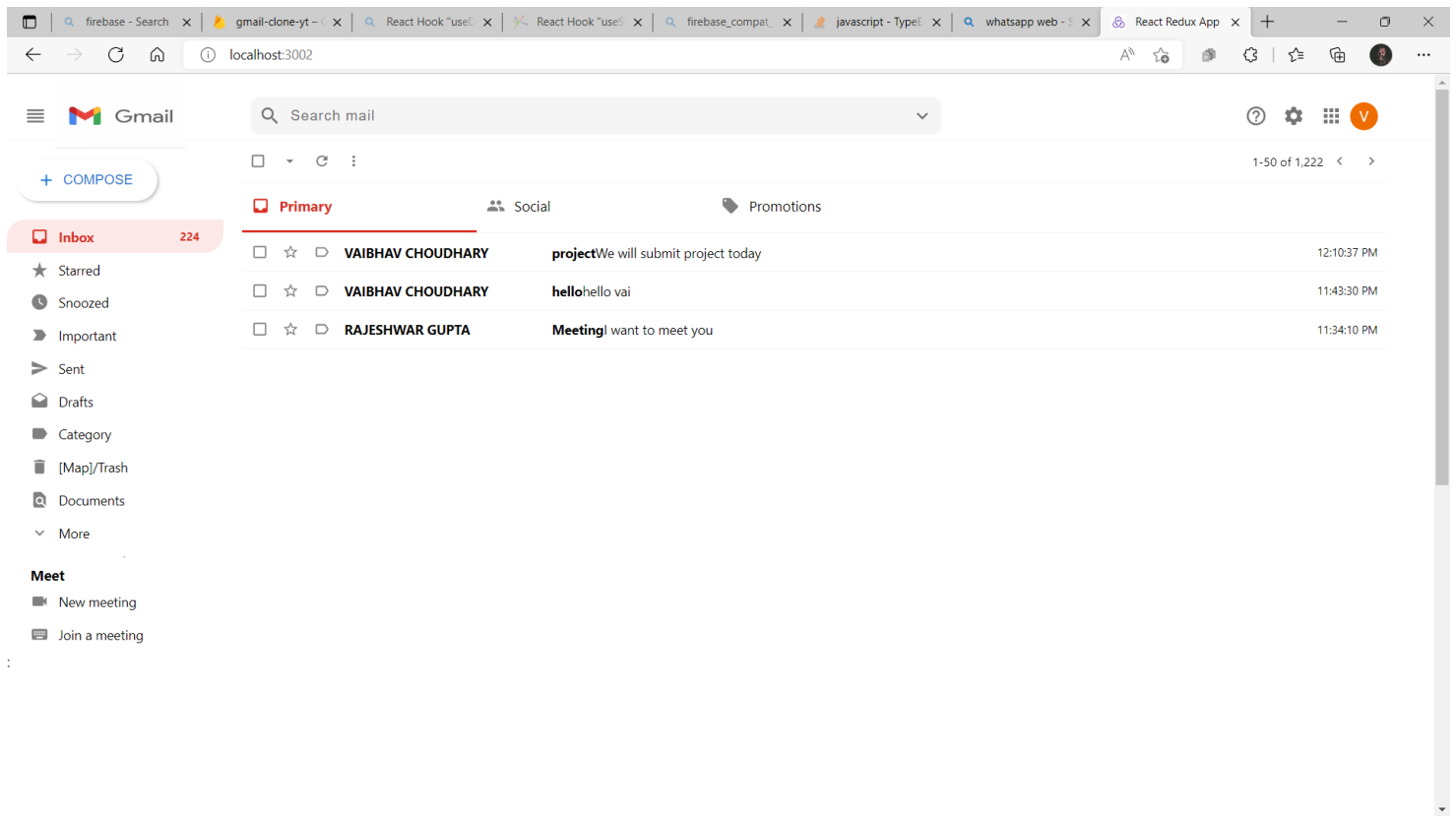
Problems OUTPUT DEBUG CONSOLE TERMINAL

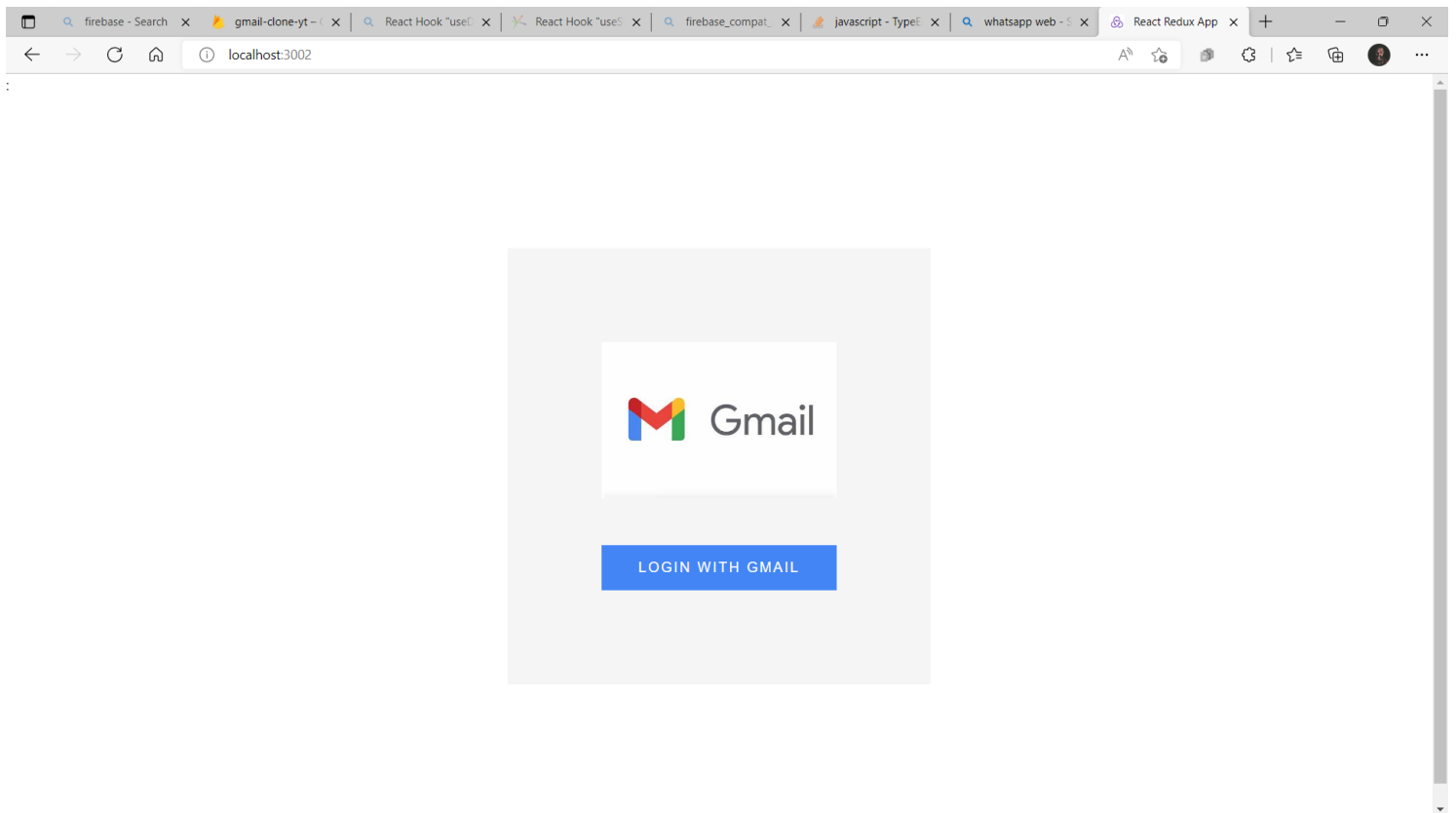
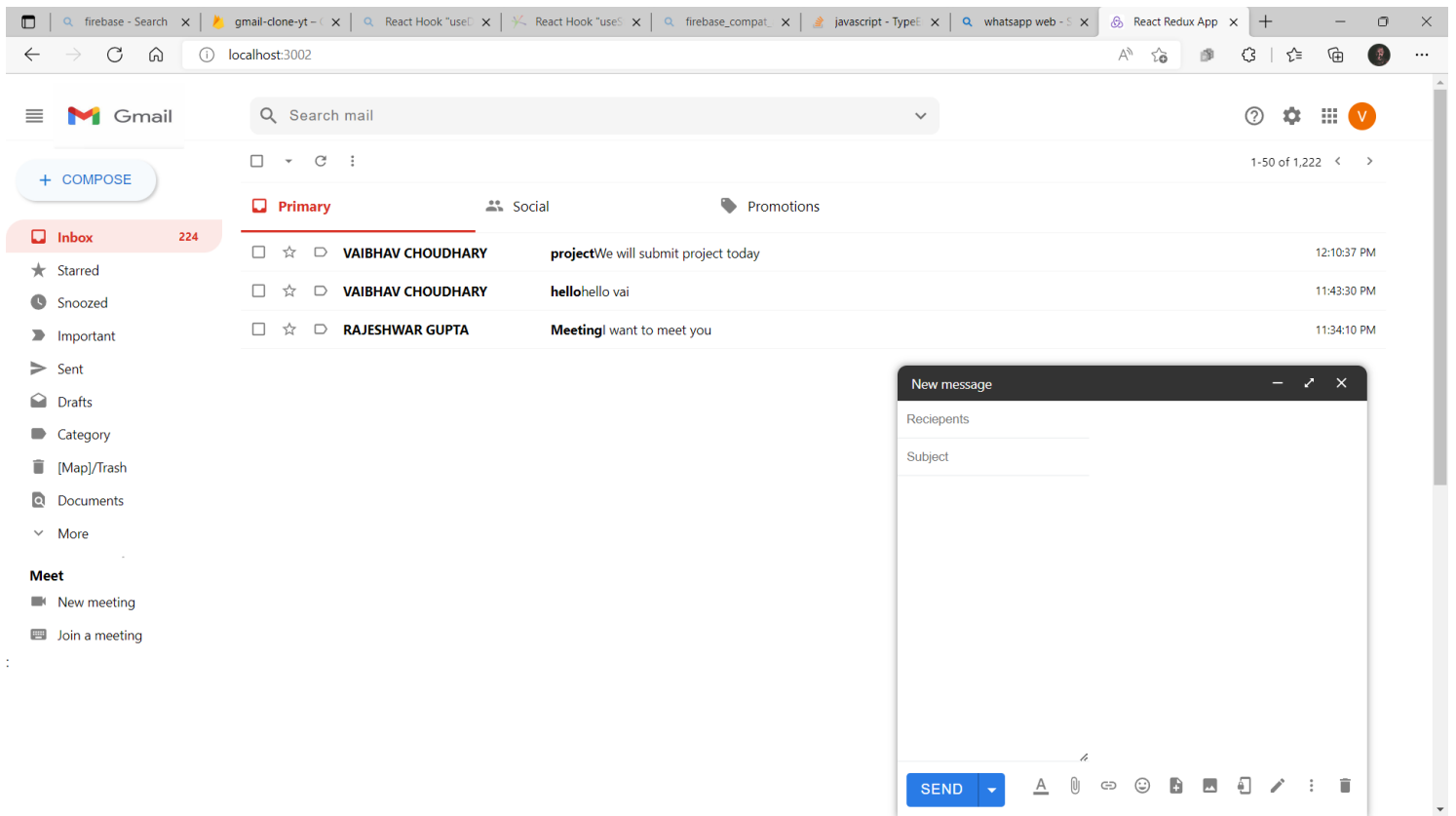
src\features\userSlice.js
Line 1:10: 'createAsyncThunk' is defined but never used no-unused-vars

src\login.js
Line 24:13: img elements must have an alt prop, either with meaningful text, or an empty string for decorative images jsx-attr/alt-text

webpack compiled with 1 warning

CLONED GMAIL





Software Testing

Once source code has been generated, software must be tested to uncover as many errors as possible before delivery. It is very important to work the system successfully and achieve high quality of software. Testing include designing a series of test cases that have a high likelihood of finding errors by applying software-testing techniques. System testing makes logical assumptions that if all the parts of the system are correct, the goal will be successfully achieved. The system should be checked logically. Validations and cross checks should be there. Avoid duplications of record that cause redundancy of data. In other words,

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. It is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

The preliminary goal of implementation is to write source code and internal documentation so that conformance of the code to its specifications can be easily

verified, and so that debugging, testing and modifications are eased. This goal can be achieved by making the source code as clear and straightforward as possible.

Simplicity, clarity and elegance are the hallmark of good programs, obscurity, cleverness, and complexity are indications of inadequate design and misdirected thinking.

Source code clarity is enhanced by structured coding techniques, by good coding style, by, appropriate supporting documents, by good internal comments, and by feature provided in modern programming languages. The implementation team should be provided with a well-defined set of software requirement, an architectural design specification, and a detailed design description. Each team member must understand the objectives of implementation.

TERMINOLOGY

Error The term error is used in two ways. It refers to the difference between the actual output of software and the correct output, in this interpretation, error is essentially a measure of the difference between actual and ideal. Error is also used to refer to human action that results in software containing a defect or fault.

Fault is a condition that causes to fail in performing its required function. A fault is a basic reason for software malfunction and is synonymous with the commonly used term Bug.

Failure is the inability of a system or component to perform a required function according to its specifications. A software failure occurs if the behavior of the software is different from the specified behavior. Failure may be caused due to functional or performance reasons.

TYPES OF TESTING

Unit Testing The term unit testing comprises the sets of tests performed by an individual programmer prior to integration of the unit into a larger system. A program unit is usually small enough that the programmer who developed it can test it in great detail,

and certainly in greater detail than will be possible when the unit is integrated into an evolving software product. In the unit testing the programs are tested separately, independent of each other. Since the check is done at the program level, it is also called program testing.

a. Module Testing : A module encapsulates related component. So it can be tested without other system module.

b. Subsystem Testing Subsystem testing may be independently design and implemented common problems are sub-system interface mistake in this checking we concentrate on it. There are four categories of tests that a programmer will typically perform on a program unit.

i Functional test

ii Performance

test iii Stress test

iv Structure test

Functional Test Functional test cases involve exercising the code with Nominal input values for which expected results are known; as well as boundary values

(minimum values, maximum values and values on and just outside the functional boundaries) and special values.

Performance Test Performance testing determines the amount of execution time spent in various parts of the unit, program throughput, response time, and device utilization by the program unit. A certain amount of avoid expending too much effort on fine-tuning of a program unit that contributes little to the overall performance of the entire system. Performance testing is most productive at the subsystem and system levels.

Stress Test Stress test are those designed to intentionally break the unit. A great deal can be learned about the strengths and limitations of a program by examining the manner in which a program unit breaks.

Structure Test Structure tests are concerned with exercising the internal logic of a program and traversing particular execution paths. Some authors refer collectively to functional performance and stress testing as "black box" testing. While structure testing is referred to as "white box" or "glass box" testing. The major activities in structural testing are deciding which path to exercise,

deriving test data to exercise those paths, determining the test coverage criterion to be used, executing the test, and measuring the test coverage achieved when the test cases are exercised.

Conclusion

We have completed our project within time limit with the coordination of our team members under the supervision of our mentor Mr. Abhishek Tiwari

Our project repository is available at:

- <https://github.com/VaibhavChoudhary282/Mini-Project-2>

Bibliography

www.google.com

www.github.com

www.youtube.com