Date: 16/11/2021

DELHI TECHNOLOGICAL UNIVERSITY, (DTU) DELHI

COLLEGE SOCIAL MEDIA WEBSITE USING REACT.JS & FIREBASE

Submitted by: ***Sonu K. Kushwaha & Ayush Karn***

Roll: 2K19/CO/383 & 2K19/CO/454

Course: ***Software Engineering (CO-301)***

Submitted to: ***Mrs Sonal Gandhi***

DECLARATION

We hereby declare that project report entitled "COLLEGE SOCIAL MEDIA WEBSITE Using JAVASCRIPT / HTML / CSS / REACT.JS/ FIREBASE "submitted by (Sonu K Kushwaha & Ayush Karn) to Delhi Technological University (DTU), Delhi is a record of original work done under the guidance of Mrs. Sonal Gandhi for the course of Software Engineering. All the codes and implementations are completely written by us.

Name: Sonu K Kushwaha & Ayush Karn

Roll No: 2K19/CO/383 & 2K19/CO/454

Submitted to: Mrs. Sonal Gandhi

CERTIFICATE

This is to certify that Sonu K. Kushwaha and Ayush Karn of Computer Science and Engineering Department (COE) having Roll No 2K19/CO/383 & 2K19/CO/454 respectively have successfully completed the project work entitled "COLLEGE SOCIAL MEDIA WEBSITE Using JAVASCRIPT / HTML / CSS / REACT.JS/ FIREBASE" on Software Engineering for Fifth Semester which is to be evaluated as the Mid Term Component.

Signature:

Sonu K Kushwaha

Ayush Karn

Date: 15-11-2021

ACKNOWLEDGEMENT

We would like to express my special thanks of gratitude to teacher Mrs Sonal Gandhi as well as our college (Delhi Technological University, Delhi) which gave me the golden opportunity to do this wonderful project on the topic "COLLEGE SOCIAL MEDIA WEBSITE Using JAVASCRIPT / HTML / CSS / REACT.JS/ FIREBASE", which also helped us in doing a lot of Research and we came to know about so many new things. We are really thankful to them.

Secondly, we would also like to thank our parents and friends who helped us a lot in finalizing this project within the limited time frame.

TABLE OF CONTENT

Chapter 1

Introduction

Not only does social media play an important role in connecting people and developing relationships with key influencers and journalists covering your industry, but it also provides a great opportunity to establish customer service by gathering feedback, answering questions, and listening to their feedback.

We attempted to create a social media network for a certain set of individuals in this project, namely college students. For this website, we utilised React.JS for front-end development and Firebase for backend development.

These are the functionality we will be working on as of now.

- Sign Up
- Log In and Log Out
- Posting a Doubt
- Deleting a Doubt
- Upvoting a Doubt for more reach
- Add Caption to the Post
- Comment on a Post
- Deleting a Doubt
- Firebase Authentication
- Interactive Interface

Introduction to Language and Components Used

Firebase:

Firebase is a Google platform for developing mobile and online applications. It started off as a stand-alone business in 2011. In this project, I utilised Firebase to host my website, and I hope to use the database capability of Firebase in the future to manage the database for my app.

Our project's whole backend is based on Firebase. Everything is handled by Firebase, from authentication to database schema. Firebase is essentially a one-stop shop for everything a developer needs for his or her backend. Firebase's learning curve is also not too steep, which is one of the key reasons we picked it as our backend. Firebase also fits perfectly with React.JS making it go hand in hand with each other and making the life much easier. It also gives us a feature to host our website directly on Google's own server and we have used that to host our website on Google's server.

CSS3:

Cascading Style Sheets (CSS) is a language for specifying the appearance of a document written in a markup language like HTML. Along with HTML and JavaScript, CSS is a key component of the World Wide Web. CSS is only utilised for the front-end design in this project, but it is crucial to the user experience.

JavaScript:

JavaScript, sometimes known as JS, is a computer language that follows the ECMAScript standard. JavaScript is a multi-paradigm, high-level programming language that is frequently compiled just-in-time. Curly-bracket syntax, dynamic typing, prototype-based object orientation, and first-class functions are all features of this language.

JavaScript supports event-driven, functional, and imperative programming approaches as a multi-paradigm language. It contains APIs for working with text, dates, regular expressions, standard data structures, and the Document Object Model, among other things (DOM).

React.JS:

React is a front-end JavaScript library for creating user interfaces or UI components that is open-source. Facebook and a community of individual developers and businesses maintain it. In order to write my software, I utilised react.js. In react, we must write code in JSX format, which is then translated to JavaScript using the Babel Compiler.

React makes creating interactive UIs a breeze. Create basic views for each state of your project, and React will update and render the appropriate components as your data changes. We can create isolated components that handle their own state, then put them together to create complicated user interfaces.

HTML:

The standard markup language for texts intended to be displayed in a web browser is Hypertext Markup Language. Technologies such as Cascading Style Sheets and programming languages like JavaScript can help.

HTML elements are the components that make up HTML pages. Images and other objects, such as interactive forms, can be embedded in the produced page using HTML techniques. HTML allows you to construct organised documents by indicating structural semantics for text elements like headers, paragraphs, lists, links, quotations, and other elements.

Node.JS:

Node.js is a back-end JavaScript runtime environment that runs JavaScript code outside of a web browser. It is open-source and cross-platform. I utilised node.js to create a runtime environment for my project. Node.js allows developers to use JavaScript to create command line tools and server-side scripting, which involves running scripts on the server before sending the page to the user's web browser. As a result, Node.js symbolises a "JavaScript everywhere" paradigm, bringing online application development together around a single programming language rather than separate languages for server-side and client-side scripts.

Chapter 1(A)

Rationale of Study

We were exposed to the innovative component element of the course during our third semester of college, and this was a completely new area for us. We didn't know much about these things before, and we were definitely losing out on a lot of stuff. We were driven to study new things and explore new sectors after being introduced to the innovative component of our curriculum, and we ended up learning about Web Development, which was one of the most common yet most significant topics linked to computer science.

Every time we use Instagram or WhatsApp or any other social media we do that on web. Every time we surf internet to search for something we do that on Web. So, it should be very clear that how big web development is and it is only going to get bigger. During the 3rd semester we did a project related to E Commerce website design and that is when we got to know about React.JS and Firebase. The learning curve for React and Firebase is not that difficult and anyone with basic coding skill can catch up to it pretty quickly.

During the 4th semester we were introduced to SE and that is when we had the chance to fully utilize the firebase's backend system. In order to learn the backend implementation using firebase we had to research a bit and learn a lot. Some of the research paper that we read and articles that we read are on the next page.

Chapter 2

Literature Review

Singh H S., Singh U. "Study on Google Firebase for Website Development (The real time database)"[1] International Journal of Engineering Technology Science and Research, Volume 4, Issue 3 March 2017. From this research paper we learnt basics of firebase. Another great paper that we read on firebase was "Application of Firebase in Android App Development-A Study"[2] by Khawas C., Shah P. published in International Journal of Computer Applications Volume in June 2018

There was also some of the article that we read on firebase. One of the most impactful one was by V. Singh titled "Introduction to Firebase" [5] published in Medium.com. Firebase's own documentation was also a great help i.e., Firebase Documentation (https://firebase.google.com/docs) A YouTube channel that we learnt other things about react was B. Traversy's, "React JS Crash Course 2021" [4].

Some of the most famous companies that use React are Facebook and its sister company Instagram. Firebase is also widely used by companies like Instacart, Twitch (biggest streaming website), Alibaba Travel, Bitpanda, etc.

Chapter 3

System Design and Methodology

All the system design and methodology used in the development of the project is well included in the Software Requirement Specifications Document.

SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

Team Members:

Ayush Karn – 2k19/CO/454

Sonu Kumar Kushwaha - 2K19/CO/383

Project Name: Social Media Site, "*DTUmate*"

1. Introduction:

1.1) Category: The proposed project is a Web Application.

1.2) Purpose: The purpose of the project is to develop a social media for a targeted group of people, especially to an Engineering college. The media will bring its own particular feature of idea sharing, problem rising and communication data being anonymous to others.

1.3) Document convention: To make the document more effective and readable we used bet font style and font size and headings are bold and highlighted with attractive colors.

1.4) Intended Audience: This document is written for the researchers, project managers, programmers, designers, developers, testers, documentation writers, users involved in the website modification of the online social media website of the specific institute. This document consists of the various steps and procedures for the website update initiated with an open discussion with the company by the system analyst. The following section describes the rest of the product function, scope, and other overall descriptions; finally, with the references.

1.5) Product Scope: It will integrate the benefits of sharing knowledges and data. This website is especially beneficial for those who are shy to ask questions in front of others, and for the questions that are considered silly to be asked openly. Adding more features can enable DTU to share notices at high pace and can be a solution to resolve face notices circulation.

1.6) Reference: References that I used to create questions for the initial discussion between the client is noted below,

- ➢ Singh H S., Singh U. "Study on Google Firebase for Website Development (The real time database)" International Journal of Engineering Technology Science and Research, Volume 4, Issue 3 March 2017
- ➢ Khawas C., Shah P. "Application of Firebase in Android App Development-A Study" International Journal of Computer Applications Volume 179(46): June 2018

- Yeshwin A., Sahoo A., Shoby P. "A Research Paper on a Pet-Friendly Application using Flutter and Firebase" International Journal for Research in Applied Science & Engineering Technology (IJRASET)Volume 8 Issue XII Dec 2020

2. Overall Description:

2.1) Product Perspective: This product is to make changes in the traditional way of questioning. This product will develop a close environment to consult the doubts related subject matters, placement, interviews, coding or behavioral activities. This website will act as a better and more interactive way of studies and preparation for placements. Our approach towards programming was a highly methodical one since a Web application is created using various fields and principles.

2.2) Product Function: This product enables the official to get a detailed idea about the questioning and also to be well informed about the notice share by authority. The system includes maintaining profile, managing friends, posting doubts in form of text and images, adding like and comments to a post and to search a problem with tags.

2.3) User- classes and Characteristics: Users of the system should be able to register, login, search friend or posts, upload post, add comments and likes to a post and also manage his/her profile information. The system will support two types of user privileges, Customer, and Admin. Customers will have access to customer functions, and the Admin will have access to both customer and admin management functions. The customer should be able to do the following functions,

- Create account.
- Upload post.
- Influence posts.
- Search by tags.

The Admin should have the following functionalities:

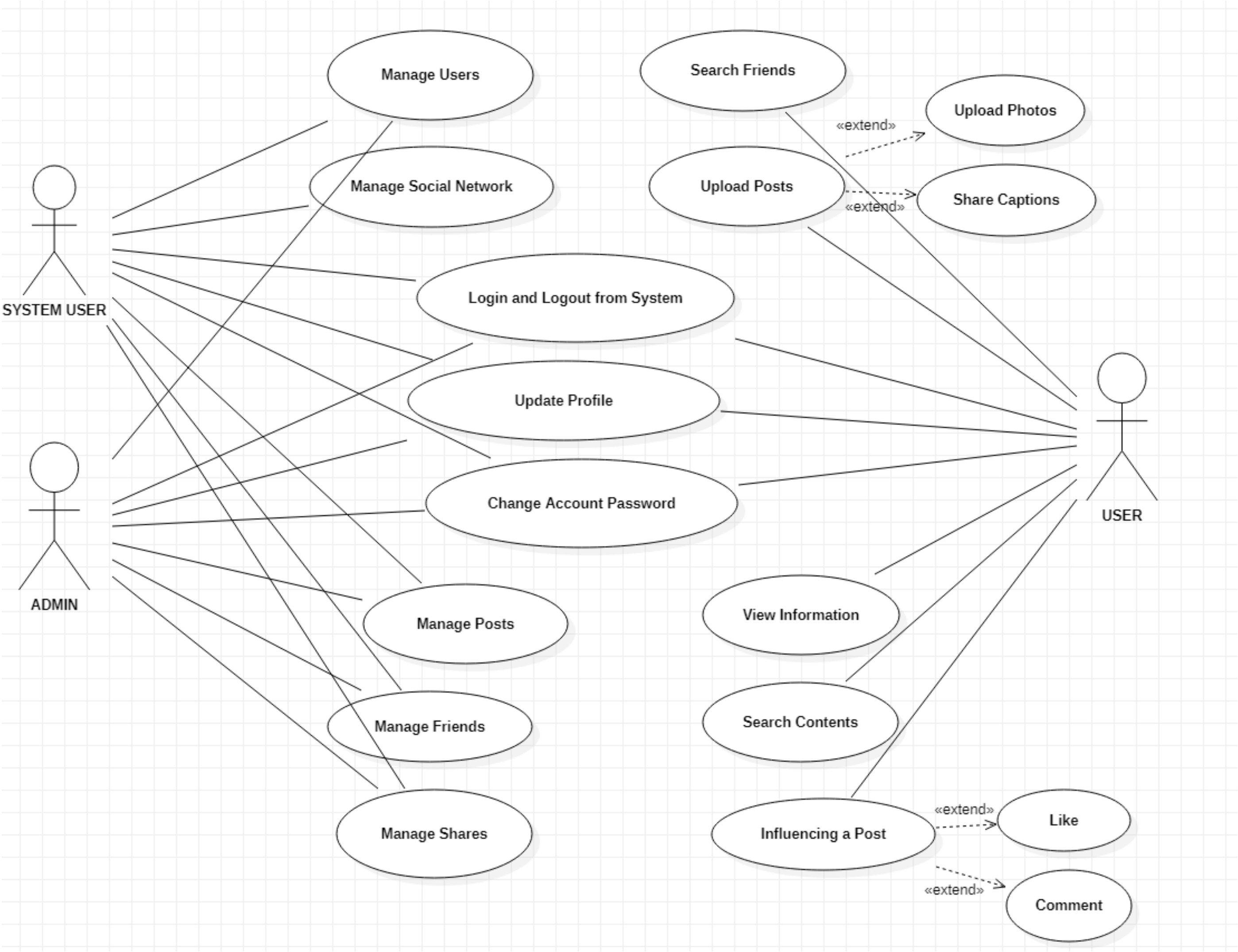
- Customer Functions:
    - Maintenance of the website in order to view track records.
    - Transfer the information of the dead posts to the archived section.
    - Maintain the confidentiality of sensitive cases.
- Administrative Functions:

- Add/Delete profiles
- Add/Delete posts
- Influence the spreading of post.
- Able to gather data
- Can make changes in website.

2.4) Operating Environment: Operating environment for the Social Media system is as listed below,

- Non-sql database
- Client/Server system
- Operating system: Window/Linux
- Database: Firebase
- Platform: HTML, CSS, JavaScript, React.js, Node.js, Firebase

2.5) Design and Implementation Constraints: The interaction between the actors and system is well shown in Use Case diagram of the product,



The Use Case Diagram has three actors, User, Admin and System User. System User can basically manage or control overall functioning of the website. User will have the functionality to register/login, create a post, influence posts, search friends or posts by names or tags. System Admin have all the functionality of User along with functionality to manipulate protected data.

2.6) User Documentation: The general characteristics of the user of our website will be anyone from the targeted audience (like here DTU students) along with professors, passouts and authorities.

2.7) Assumptions and Dependencies: Our approach towards programming was a highly methodical one since a Web application is created using various fields and principles. We also focused on backend and using different feature of Firebase by hosting.

3. External Interface Requirements:

3.1) User Interface: Front-end software: React.js version; Back-end software: Firebase

3.2) Hardware Interface: Windows and any browser which supports CGI, HTML, and JavaScript library.

3.3) Software Interfaces: Following are the software used for the social media web application,

Software Used	Description
Operating System	We have chosen Windows operating system for its best support and user-friendliness.
Database	To save the all the website data, we have chosen the FireStore (Firebase) database.
Node.js	Node Package Manager

3.4) Communication Interface: This project supports all types of web browsers. We are using simple electronic forms for interacting, browsing, sharing, etc.

4. System Features:

4.1) Description: Social media play a crucial role in connecting people and developing relationships, not only with key influencers and journalists covering your company's sector, but also provides a great opportunity to establish customer service by gathering input, answering questions and listening to their feedback.

4.2) Priority: The priority is given to,

- Enclosed environment of sharing and communication
- Interactive and anonymous sharing feature
- Easier tag searching and authority data and notice sharing.

4.3) Functional Requirements:

- NO-SQL Database: The database which don't store data in form of tables are called no-SQL database. The database of our system is stored in Firestore of Firebase in No-SQL form. The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our Apple platforms, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.
- Client/Server System: The term client/server refers primarily to architecture or logical division of responsibilities, the client is the application (also known as the front-end), and the server is the Firebase (also known as the back-end).

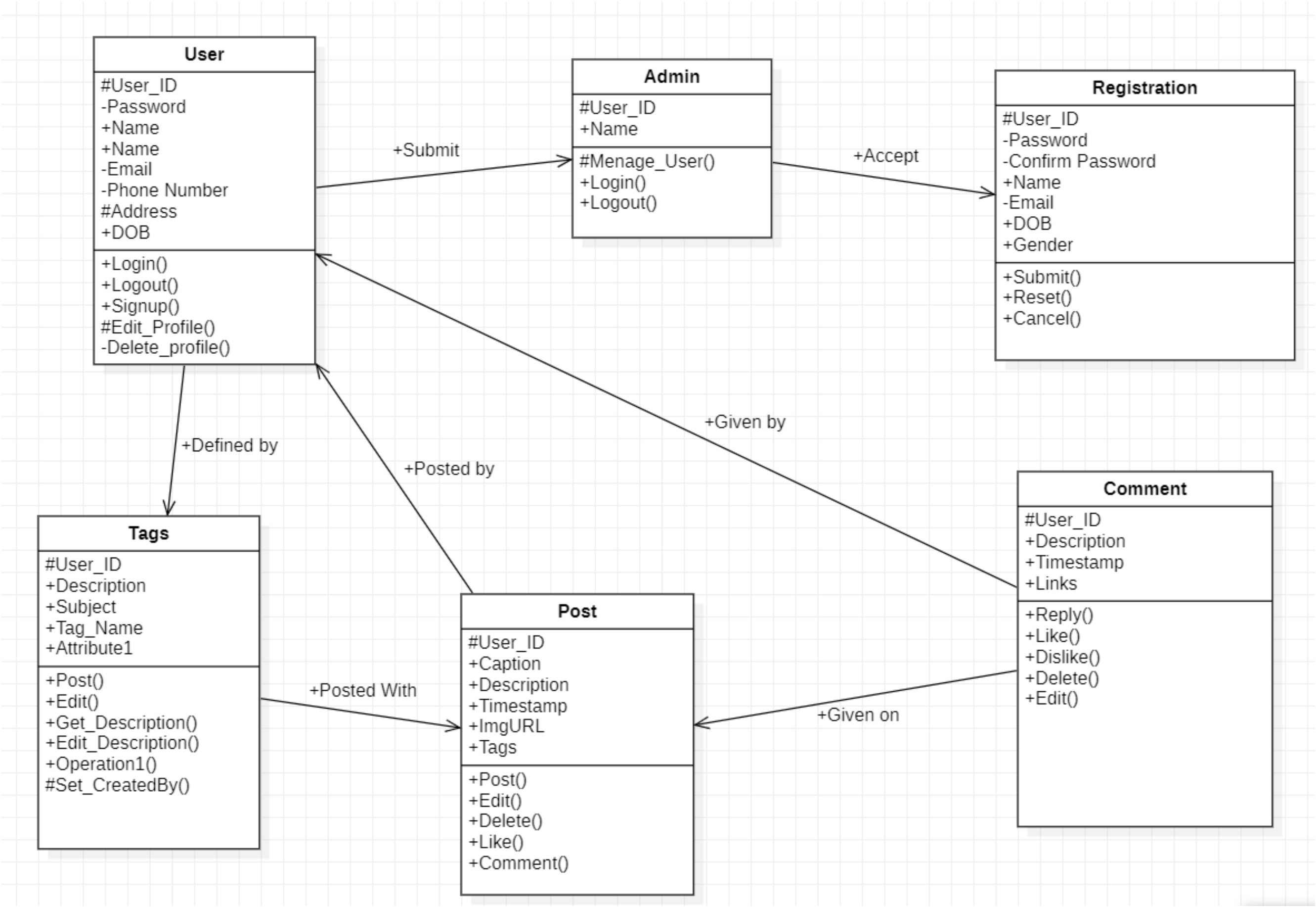
5. Non-Functional Requirements:

5.1) Performance Requirements: The steps involved to perform the implementation of the social media website system are listed below.

- Class diagram: Class diagram is a static diagram that represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system.

  The standard class diagram is composed of three sections:

  - ➔ **Upper section:** Contains the name of the class. This section is always required, whether you are talking about the classifier or an object.
  - ➔ **Middle section:** Contains the attributes of the class. Use this section to describe the qualities of the class. This is only required when describing a specific instance of a class.
  - ➔ **Bottom section:** Includes class operations (methods). Displayed in list format, each operation takes up its own line. The operations describe how a class interacts with data.



Class diagram contains 6 classes, User, Admin, Registration, Tags, Post and Comment. The User class holds the data of user also defining functionality provided to user. Registration class holds the registration firm and registered data and functionality defined for registration. Tags class holds the name and description of tags that can be used in posts for describing problem in as sort form as possible and can also be used for searching required post using the tags used in the respective post. Post and Comment classes holds data and functionality any user is provided for posting and commenting respectively.

5.2) Safety Requirements: In case there is sharing of inconvenient content action can be taken. The user can be mater of punishment for spreading fake news and notices. Research and analysis can be done to track user behaviors and feature for reporting inconvenient content can be added. User have security to share post being anonymous in any case of personal data leak.

5.3) Security Requirements: Security frameworks can be controlled by administration (DTU authority). Authority can manage users profile, posts, network connections and content searched.

5.4) Software Quality Attributes:

- **Availability:** The project is aimed at to be used by targeted audiences only, specially by Engineering college students. Availability of the website will begin from DTU, named as *DTAmate.* Including students of DTU, professors, Authorities and DTU passouts will be able t use the website. The availability will be extended to other colleges or schools by making some changes, making website specifically suitable and interactive to user according to the respective schools or college.
- **Correctness:** The main component of our project i.e. Database Management was carried out in Firebase. The database was created and maintained using Firebase is a real-time database leading to less errors in database and quite fast error detection. But querying and modifying the database through the web app required another JS library by the same name to mediate the database on the system and the web app populating the database using data from its form fields.
- **Maintainability:** The implementation of the website is carried on Firebase which secures maintainability by offering inbuilt features. The feature of real-time database improves data maintenance, feature of Google analysis helps in maintaining user interface and interactivity. Regular update will keep the system Bug-free. Here multiple checks can be implemented to ensure the integrity and correctness authentications so that the file can be maintained and not corrupted.
- **Usability:** We have presented a simple technique for sharing doubts, ideas, information, notices and data, also allowing the user to be anonymous for the purpose of security or anything else. We have additional features for administration for analysis of post circulation, attractive interactions, and fast notice circulation. These feature will add up the usability of our web application.

5.5) Business Rules: DTUmate initiatives significantly improve social media for DTU targeted audiences bringing the advance features of sharing and communication in more of interactive way while maintaining user's privacy.

As India enhancing in use of social media, this system is next step of personalized social media according to specific institute for specific targeted audience to create more usability, interactive and security.

Chapter 4

System Analysis and Design

In this project we are going to be making a social media website using React.JS/ HTML/ CSS/ JavaScript as the front-end part and we will be using Firebase for all the backend functionality for this project.

Firebase is actually a great choice instead of traditional backend such as SQL, Oracle or other similar Software because it is very new and is managed by google. Firebase is something that is being used in industry. So, learning this would be beneficial to us.

Database will be made inside Firebase's Firestore Database and Authentication will also be done using Firebase only

Firebase actually uses a mixture of SQL and NoSQL based database so it includes goods of both worlds.

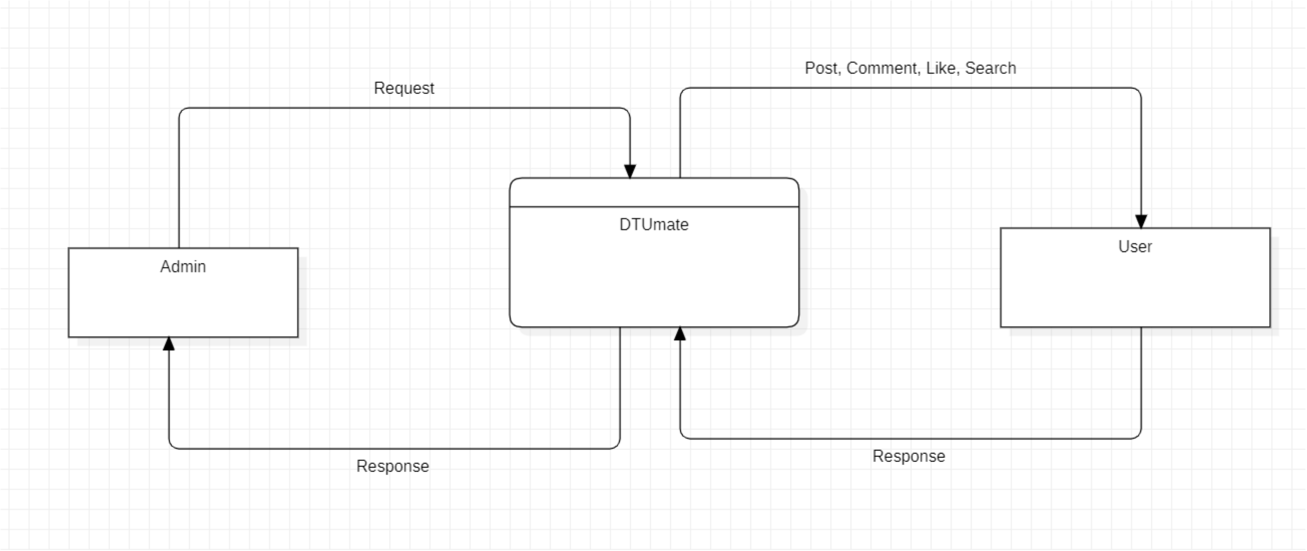
We have also hosted the website using Firebase only as Firebase also provides us with a feature to host our webpages on very fast Google's server.

DATA FLOW DIAGRAM

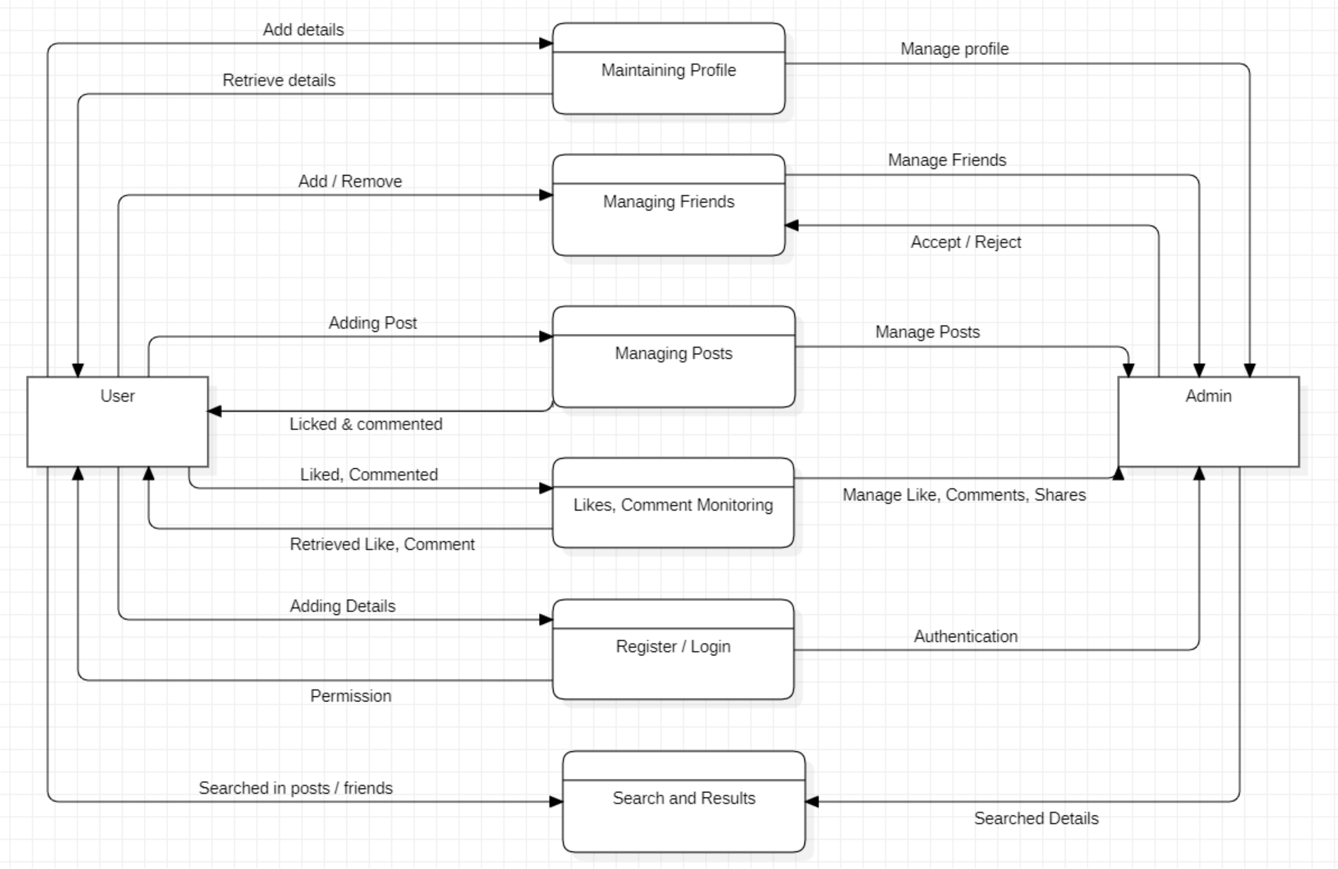
The Data Flow Diagram is graphical representation of flow of data in a system. The DFD we made includes all the functionality and flow of data through these functionalities connecting the entire system. This DFD includes the incoming data, outgoing data as well as stored data.

Here, we will see mainly 3 levels in the data flow diagram: -

I. **0-level DFD:** It is also known as a context diagram. It's designed to be an abstraction view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.

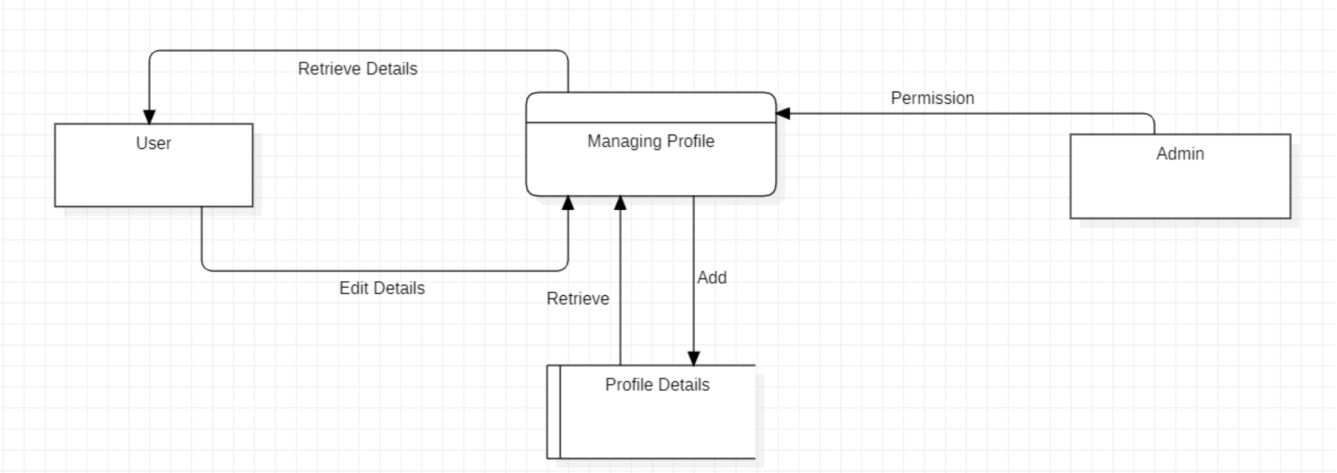


II. **1-level DFD:** In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and breakdown the high-level process of 0-level DFD into subprocesses.

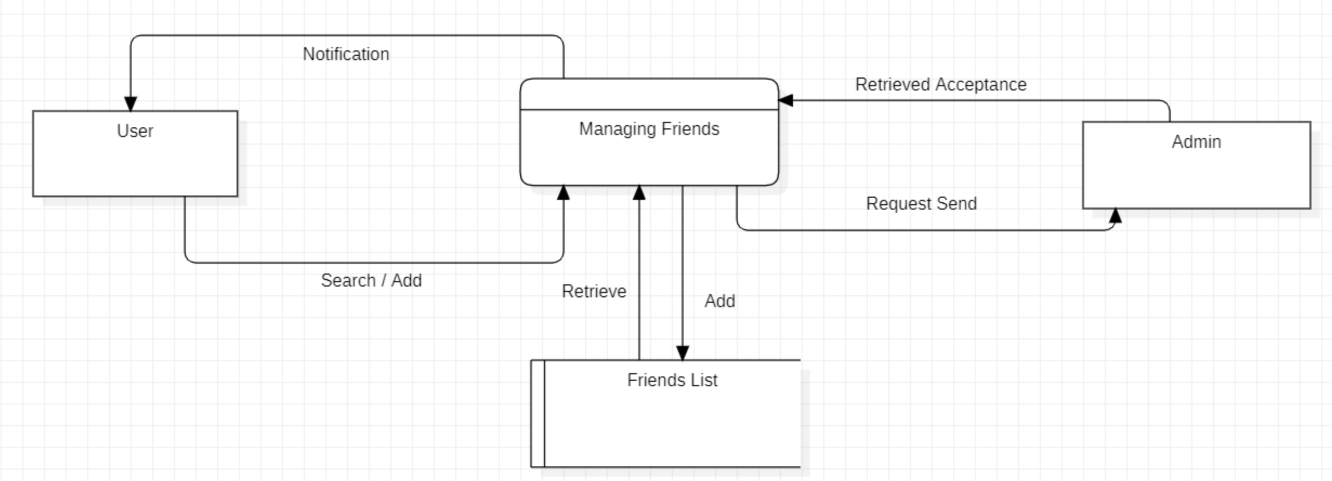


III. **2-level DFD:** 2-level DFD goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning.

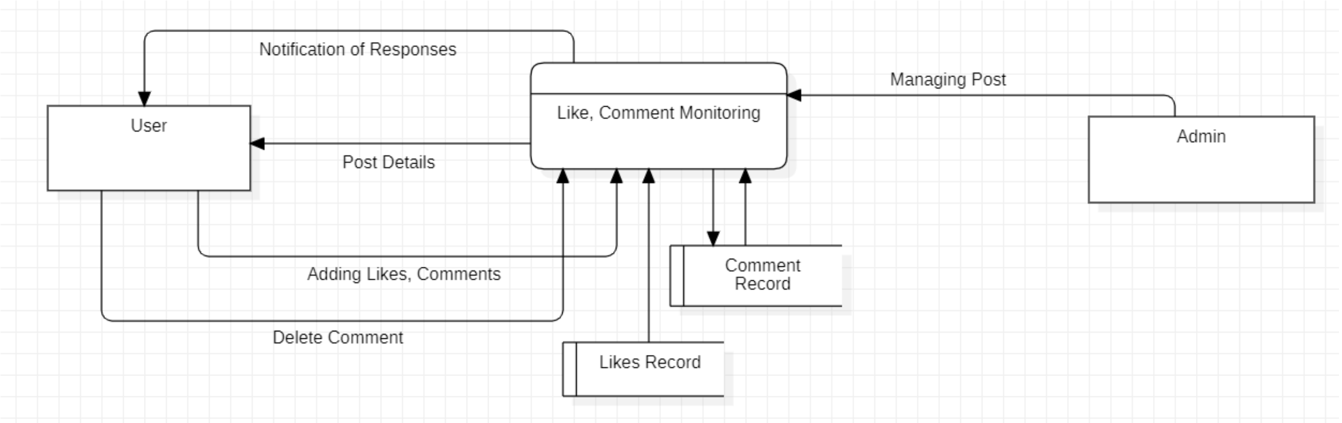
1. Managing Profile:



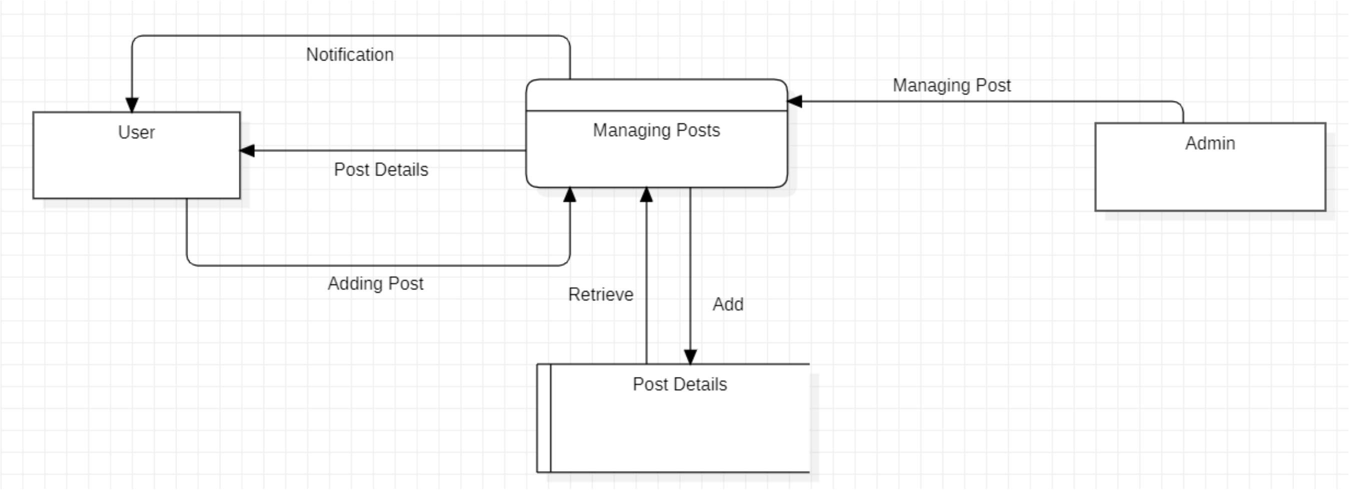
2. Managing Friends:



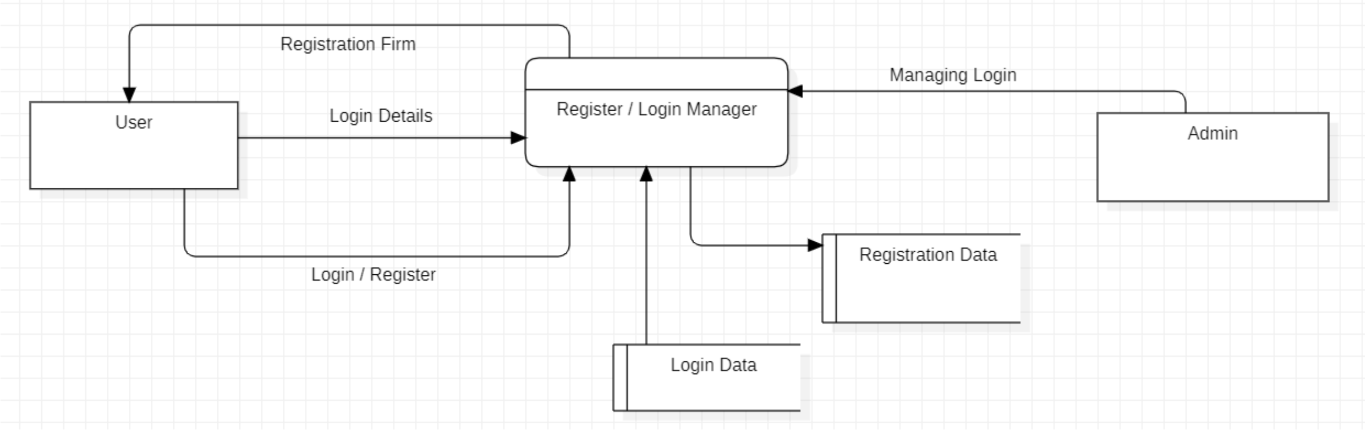
3. Likes, Comments Monitoring:



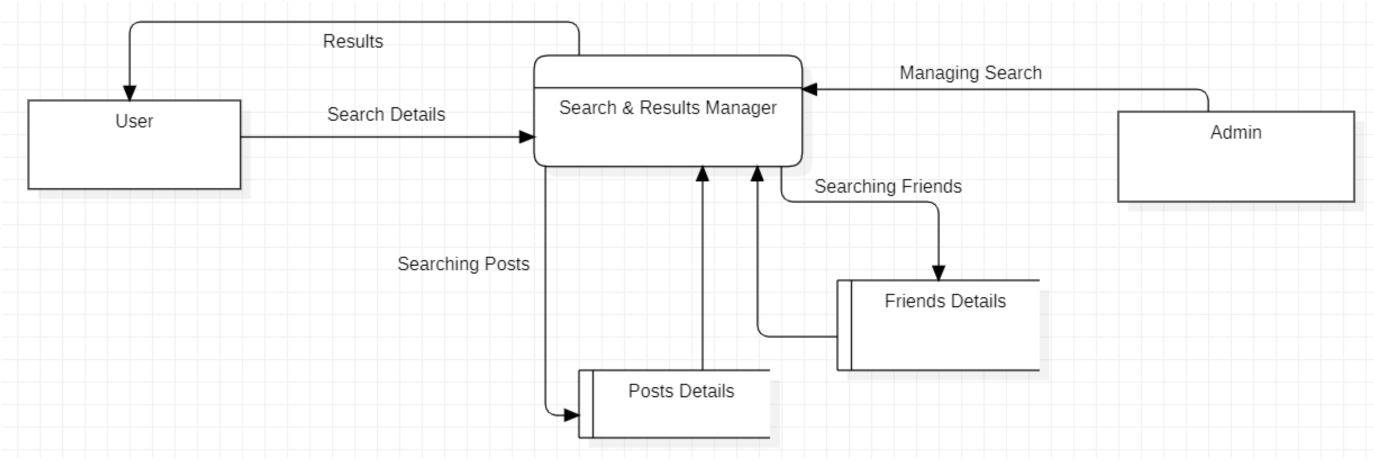
4. Managing Posts:



5. Register/Login Manager:



6. Search & Result Manager:



Chapter 5

Implementation and Code

One of the main benefits of using React.JS as front end is because each of the things can be broken down into simple components and then be used multiple times with just one line of code and changing the values according to the need.

The components that were used in this project are

<Post.js>

This component that included everything that a Social Media post holds. It included username of the uploader, Caption if there was any, timestamp of when the post was uploaded a PostID to differentiate each post kept in the database.

<ImgUpload.js>

This component that was used to upload post inside the website and was directly connected to the backend. This component was used at the bottom of the page and by using this user could upload images to the website as well as videos and then these images were uploaded to firebase's server and then firebase would generate a ImageURL which was stored in the database instead of the image. Using the image URL instead of the image helped us reduce the time taken drastically and made the website more reactive.

<App.js>

This was the main component of our entire project and it is where all other components were called and used. <Post.js> , <ImgUpload.js>, <Index.js> etc were called into this component and used multiple times as desired.

<Post.css, ImageUpload.css, App.css, etc>

All the CSS components were used just to provide better look to the website which is very important because user interface is very much useful for a website. A website with better USX will always attract more people than with a bad USX one.

Using this component-based implementation with the help of react we were able to drastically decrease the complexity of the code and it was very easier for us to debug as well as modify the code. Using component-based methodology for this project also enabled us to be flexible with the type of schema that we used for this database created inside of the firebase which we will be discussing later on in the report.

Adding Firebase to Project:

First, we need to add our project to firebase, for which we should follow the below steps and we learnt these steps from “Application of Firebase in Android App Development-A Study” [2]

Go to https://console.firebase.google.com/

Click on Add Project

Enter your Project’s name

On next page you can setup Google Firebase Analytics

On Next page select account for Google Analytics

Click Create Project

Click Create Project and Your project will be created

Go to </> Icon on the top and give your App a nickname

Check the box for setup firebase hosting

Click Next, Next and Continue to Console and your project is Finally Created

Copy the Codes from Config page into Firebase.js and you’re good to go.

Authentication

Most apps need to know the identity of a user. Knowing a user's identity allows an app to securely save user data in the cloud and provide the same personalized experience across all of the user's devices.

Firebase Authentication provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more. Firebase Authentication integrates tightly with other Firebase services.

There are basically two ways we can implement authentication either by using FirebaseUI as a complete drop-in auth solution or by using the Firebase Authentication SDK to manually integrate one or several sign-in methods into your app. We have opted for the manual sign-in integration using Authentication SDK in this project.

For other backend it requires weeks even months and 100’s of lines of difficult code in order to provide a user-based authentication, Firebase can do this in just 10 lines of code and we just have to use when and where to ask for the user to authenticate them and everything else is handled by Firebase. We just have to import the <Firebase.js> component where we want to provide the authentication feature and we are ready to go.

A part of code that handles authentication is given below

```
const signUp = (event) => {
    event.preventDefault();
    auth.createUserWithEmailAndPassword(email , password)
    .then((authUser) => {
      return authUser.user.updateProfile({
        displayName: username
      })
    })
    .catch((error) => alert(error.message));
    setOpen(false);
  }

  const signIn = (event) => {
    event.preventDefault();
    auth.signInWithEmailAndPassword(email , password)

    .catch((error) => alert(error.message));
    setOpenSignIn(false);
  }
```

Database Schema:

We have used firebase’s Firestore Database to implement the Database for our project. It is a NoSQL based Database and contains these components inside of them.

A Collection called Post that holds all the posts that are uploaded to the website.

Each Post has following documents in them:

caption: “string” this stores the caption for that post

imgUrl: “string” ImageURL is stored in this part

timestamp: “time” Firebase’s server time is stored for ordering the posts and maintaining uniformity.

username: “string” username of the uploader is stored

Each post also has another sub collection in order to store all the comments that are done on that specific post. This comment section is further divided into sub documents which are as follows:

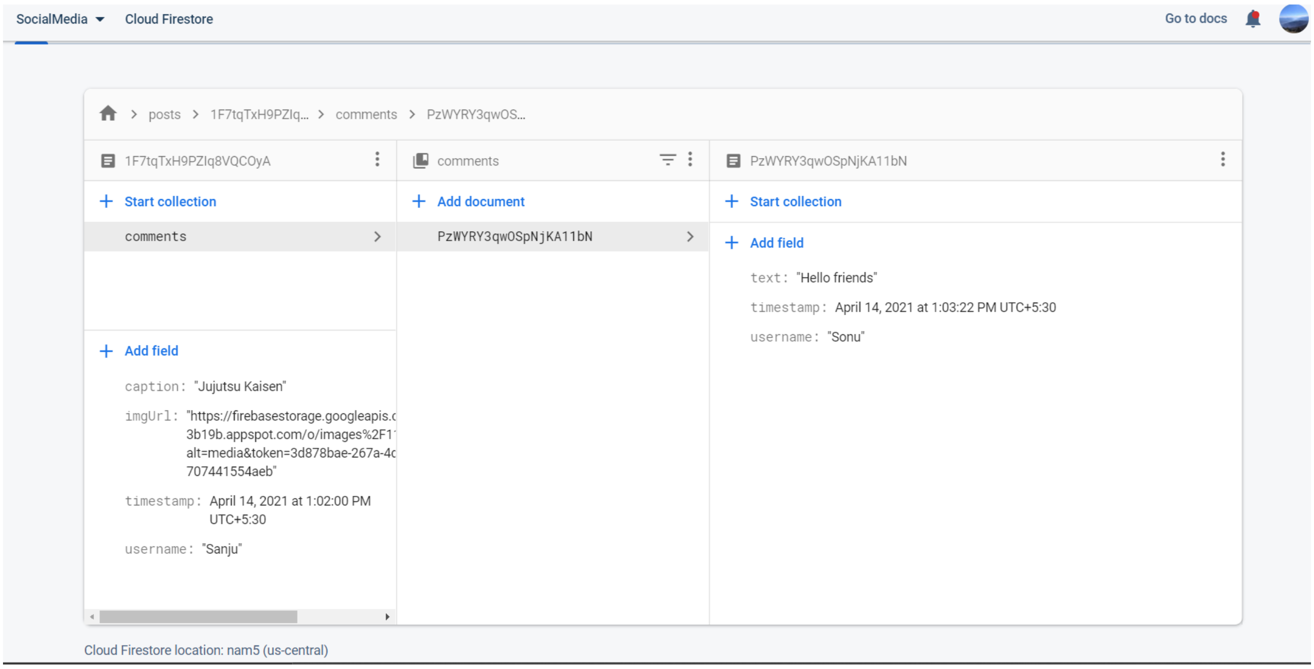
text: “string” for storing the comment on the post.

timestamp: “time” for storing the time of uploading the comment and for ordering purpose this concept was used from “A Research Paper on a Pet-Friendly Application using Flutter and Firebase” [3]

username: “string” for storing username of the person commenting on the post.

Along with all of these each of the document is automatically provided a DocumentID in order to be made unique and this DocumentID is used as a primary key for the database and also as a foreign key for relating to some other part of database.

A picture of this database is attached below for the reference.



comments

PzWYRY3qwOSpNjKA11bN

caption: "Jujutsu Kaisen"
imgUrl: "https://firebasestorage.googleapis.c
3b19b.appspot.com/o/images%2F1
alt=media&token=3d878bae-267a-4c
707441554aeb"
timestamp: April 14, 2021 at 1:02:00 PM UTC+5:30
username: "Sanju"

text: "Hello friends"
timestamp: April 14, 2021 at 1:03:22 PM UTC+5:30
username: "Sonu"

Cloud Firestore location: nam5 (us-central)

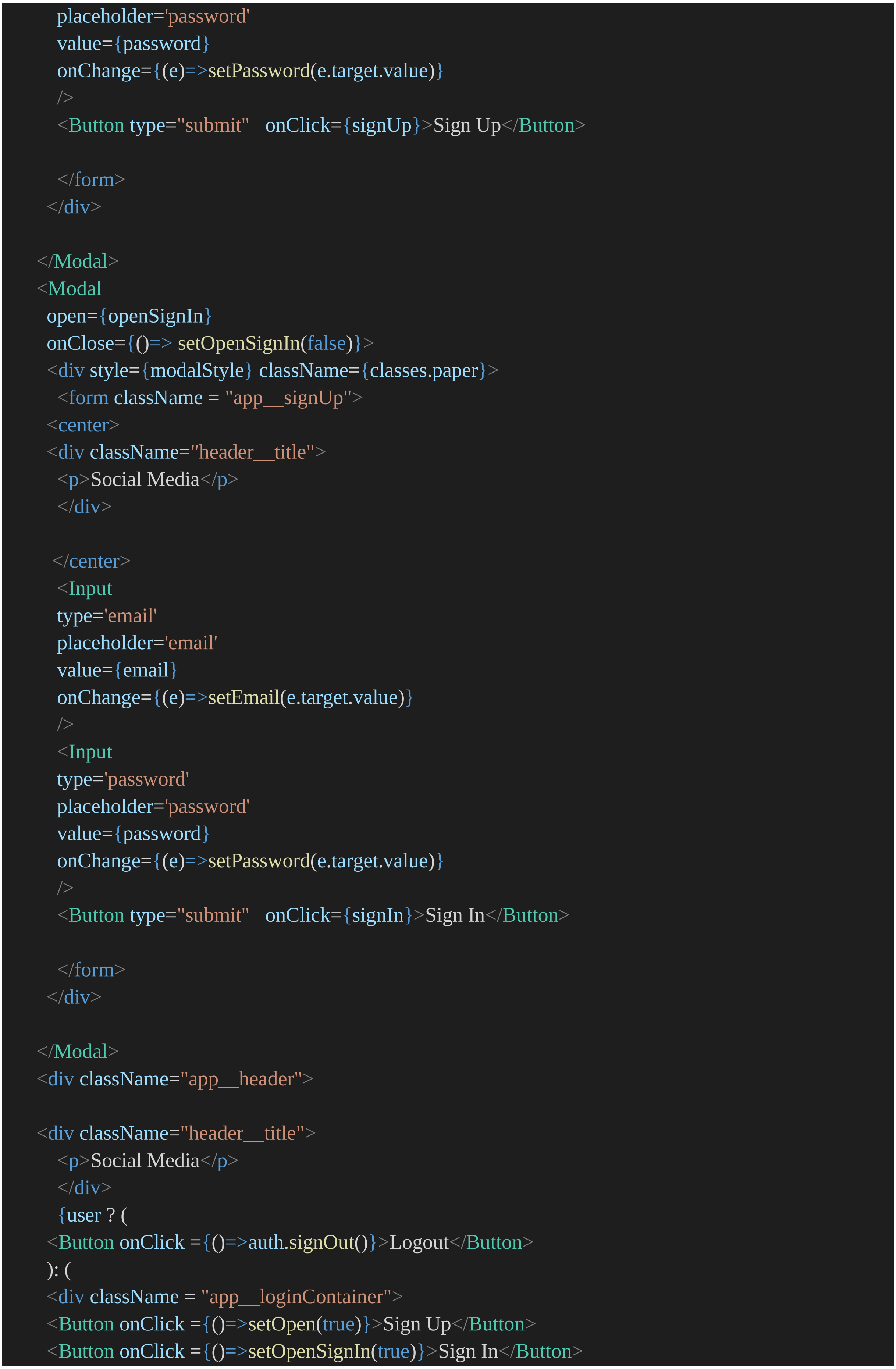
Automatically All of the code is not possible to be given in this report so I am attaching most important component's codes

Code:

<App.js>

```
import React,  {useState, useEffect}  from "react";
import './App.css';
import Post from './post.js';
import {db , auth} from './FireBase.js';
import { makeStyles } from '@material-ui/core/styles';
import Modal from '@material-ui/core/Modal';
import {Button,Input} from '@material-ui/core';
import ImgUpload from './ImgUpload';
import InstagramEmbed from 'react-instagram-embed';
//import firebase from 'firebase';

function getModalStyle() {
  const top = 50 ;
  const left = 50 ;

  return {
    top: `${top}%`,
    left: `${left}%`,
    transform: `translate(-${top}%, -${left}%)`,
  };
}
```

```
const useStyles = makeStyles((theme) => ({
  paper: {
    position: 'absolute',
    width: 400,
    backgroundColor: theme.palette.background.paper,
    border: '2px solid #000',
    boxShadow: theme.shadows[5],
    padding: theme.spacing(2, 4, 3),
  },
}));

function App() {
  const classes = useStyles();
  const [modalStyle] = useState(getModalStyle);
  const [posts, setPosts] = useState([]);
  const [open , setOpen] = useState(false);
  const [openSignIn , setOpenSignIn] = useState(false);
  const [email , setEmail] = useState('');
  const [username , setUsername] = useState('');
  const [password , setPassword] = useState('');
  const [user , setUser] = useState(null);
  useEffect(() => {
    const unsubscribe= auth.onAuthStateChanged((authUser) => {
      if(authUser){
        console.log(authUser);
        setUser(authUser);
        if(authUser.displayName){

        }else{
          return authUser.updateProfile({
            displayName: username,
          });
        }
      }else{
        setUser(null);
      }
    })
    return () => {
      unsubscribe();
    }
  },[user , username]);
   useEffect(() => {
    db.collection('posts').orderBy('timestamp','desc').onSnapshot((snapshot) => {
      setPosts(
        snapshot.docs.map(doc=>({
          id: doc.id,
          post : doc.data()
        })
      ));
    })
  }, []);

  const signUp = (event) => {
    event.preventDefault();
    auth.createUserWithEmailAndPassword(email , password)
    .then((authUser) => {
      return authUser.user.updateProfile({
        displayName: username
      })
    })
    .catch((error) => alert(error.message));
    setOpen(false);
  }

  const signIn = (event) => {
    event.preventDefault();
    auth.signInWithEmailAndPassword(email , password)

    .catch((error) => alert(error.message));
    setOpenSignIn(false);
  }

  return (
    <div className="app">

      <Modal
        open={open}
        onClose={()=> setOpen(false)}>
        <div style={modalStyle} className={classes.paper}>
          <form className = "app__signUp">
        <center>
          <div className="header__title">
          <p>Social Media</p>
          </div>
         </center>
          <Input
          type='text'
          placeholder='username'
          value={username}
          onChange={(e)=>setUsername(e.target.value)}
          />
          <Input
          type='email'
          placeholder='email'
          value={email}
          onChange={(e)=>setEmail(e.target.value)}
          />
          <Input
          type='password'
```



```
        placeholder='password'
        value={password}
        onChange={(e)=>setPassword(e.target.value)}
        />
        <Button type="submit"   onClick={signUp}>Sign Up</Button>

        </form>
      </div>

    </Modal>
    <Modal
      open={openSignIn}
      onClose={()=> setOpenSignIn(false)}>
      <div style={modalStyle} className={classes.paper}>
        <form className = "app__signUp">
      <center>
      <div className="header__title">
        <p>Social Media</p>
        </div>

       </center>
        <Input
        type='email'
        placeholder='email'
        value={email}
        onChange={(e)=>setEmail(e.target.value)}
        />
        <Input
        type='password'
        placeholder='password'
        value={password}
        onChange={(e)=>setPassword(e.target.value)}
        />
        <Button type="submit"   onClick={signIn}>Sign In</Button>

        </form>
      </div>

    </Modal>
    <div className="app__header">

    <div className="header__title">
        <p>Social Media</p>
        </div>
        {user ? (
      <Button onClick ={()=>auth.signOut()}>Logout</Button>
      ): (
      <div className = "app__loginContainer">
      <Button onClick ={()=>setOpen(true)}>Sign Up</Button>
      <Button onClick ={()=>setOpenSignIn(true)}>Sign In</Button>
```

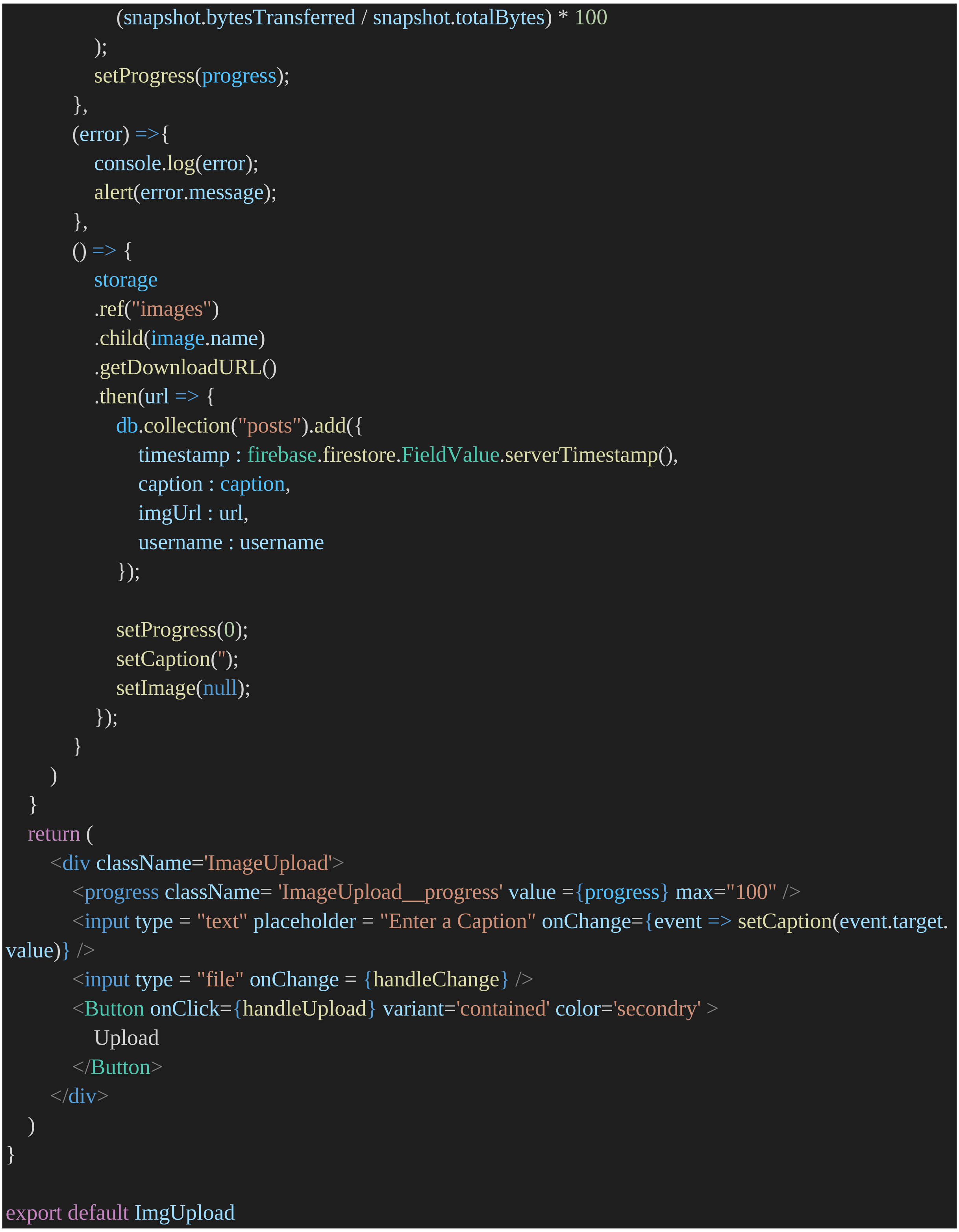
```
     </div>
    ))}
    </div>

    <div className= 'app__posts'>
      <div className = "app__postsLeft">

        {posts.map( ({id , post})  => (
          <Post
            key = {id}
            postId={id}
            user={user}
            username={post.username}
            caption={post.caption}
            imgUrl={post.imgUrl}
          />
        ))}
      </div>
      <div className = "app_postsRight">
        <InstagramEmbed
          url='https://instagr.am/p/Zw9o4/'
          maxWidth={320}
          hideCaption={false}
          containerTagName='div'
          protocol=''
          injectScript
          onLoading={() => {}}
          onSuccess={() => {}}
          onAfterRender={() => {}}
          onFailure={() => {}}
        />

      </div>
    </div>

    {user?.displayName ? (<ImgUpload username= {user.displayName} />):(<h3>Sorry you need to login !!</h3>)}
   </div>

  );
}
export default App;
```

<Firebase.js>

```
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
import firebase from 'firebase';

const FirebaseApp = firebase.initializeApp({
   apiKey: "AIzaSyDHHjcUcQYc5IzHFMdLSZ2KmJ30QT92ERU",
   authDomain: "socialmedia-3b19b.firebaseapp.com",
   projectId: "socialmedia-3b19b",
   storageBucket: "socialmedia-3b19b.appspot.com",
   messagingSenderId: "259452570231",
   appId: "1:259452570231:web:eb2d8affc7ed87a8581472",
   measurementId: "G-Y8Z78YET77"
  });

  const db = FirebaseApp.firestore();
const auth = firebase.auth();
const storage = firebase.storage();

export { db, auth, storage };
   }
```

<ImgUpload.js>

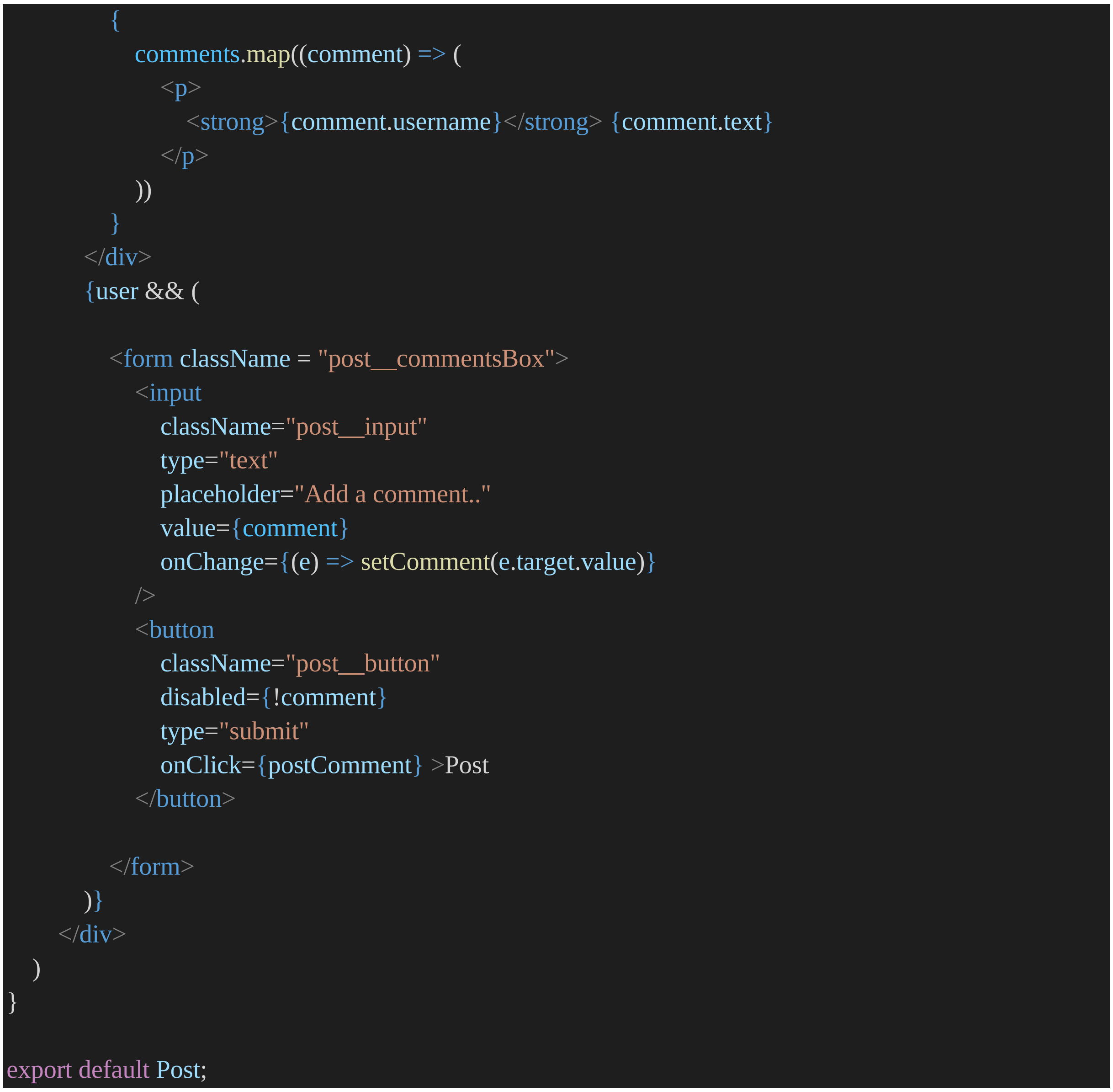
```
import React , { useState } from 'react';
import {Button} from '@material-ui/core';
import {db , storage} from './FireBase.js';
import firebase from 'firebase';
import './ImageUpload.css';

function ImgUpload({username}) {
   const [caption , setCaption] = useState('');
   const [image, setImage] = useState(null);
   //const [url, setUrl] = useState(" ");
   const [progress,setProgress] = useState(0);

   const handleChange = (e) =>{
      if(e.target.files[0]){
         setImage(e.target.files[0]);
      }
   }

   const handleUpload = () =>{
      const uploadTask = storage.ref(`images/${image.name}`).put(image);
      uploadTask.on(
         "state_changed",
         (snapshot) => {
            const progress = Math.round(
```



```
          (snapshot.bytesTransferred / snapshot.totalBytes) * 100
        );
        setProgress(progress);
      },
      (error) =>{
        console.log(error);
        alert(error.message);
      },
      () => {
        storage
        .ref("images")
        .child(image.name)
        .getDownloadURL()
        .then(url => {
          db.collection("posts").add({
            timestamp : firebase.firestore.FieldValue.serverTimestamp(),
            caption : caption,
            imgUrl : url,
            username : username
          });

          setProgress(0);
          setCaption('');
          setImage(null);
        });
      }
    )
  }
  return (
    <div className='ImageUpload'>
      <progress className= 'ImageUpload__progress' value ={progress} max="100" />
      <input type = "text" placeholder = "Enter a Caption" onChange={event => setCaption(event.target.
value)} />
      <input type = "file" onChange = {handleChange} />
      <Button onClick={handleUpload} variant='contained' color='secondry' >
        Upload
      </Button>
    </div>
  )
}

export default ImgUpload
```

<post.js>

```
import React , {useState ,useEffect} from 'react';
import './post.css';
import Avatar from "@material-ui/core/Avatar";
import {db } from './FireBase.js';
import firebase from 'firebase';

function Post({postId,user,username,caption,imgUrl}) {
    const [comments, setComments] = useState([]);
    const [comment, setComment] = useState('');
    useEffect(() => {
        let unsubscribe;
        if(postId){
            unsubscribe = db
            .collection("posts")
            .doc(postId)
            .collection("comments")
            .orderBy("timestamp","asc")
            .onSnapshot((snapshot)=>{
                setComments(snapshot.docs.map((doc)=>doc.data()))
            })
        }
        return () => {
            unsubscribe();
        }
    }, [postId]);

    const postComment = (event) =>{
        event.preventDefault();
        db.collection("posts").doc(postId).collection("comments").add({
            text: comment,
            username: user.displayName,
            timestamp: firebase.firestore.FieldValue.serverTimestamp()
        });
        setComment('');
    }

    return (
        <div className="post">
            <div className="post__header">
                <Avatar className="post__avatar"
                src="/static/images/avatar/jpg"
                alt={username}/>
                <h3>{username}</h3>
            </div>

            <img className="post__image" alt="abc" src={imgUrl}></img>
            <h4 className="post__text"><strong>{username}:</strong> {caption}</h4>
            <div className = "post__comments">
```

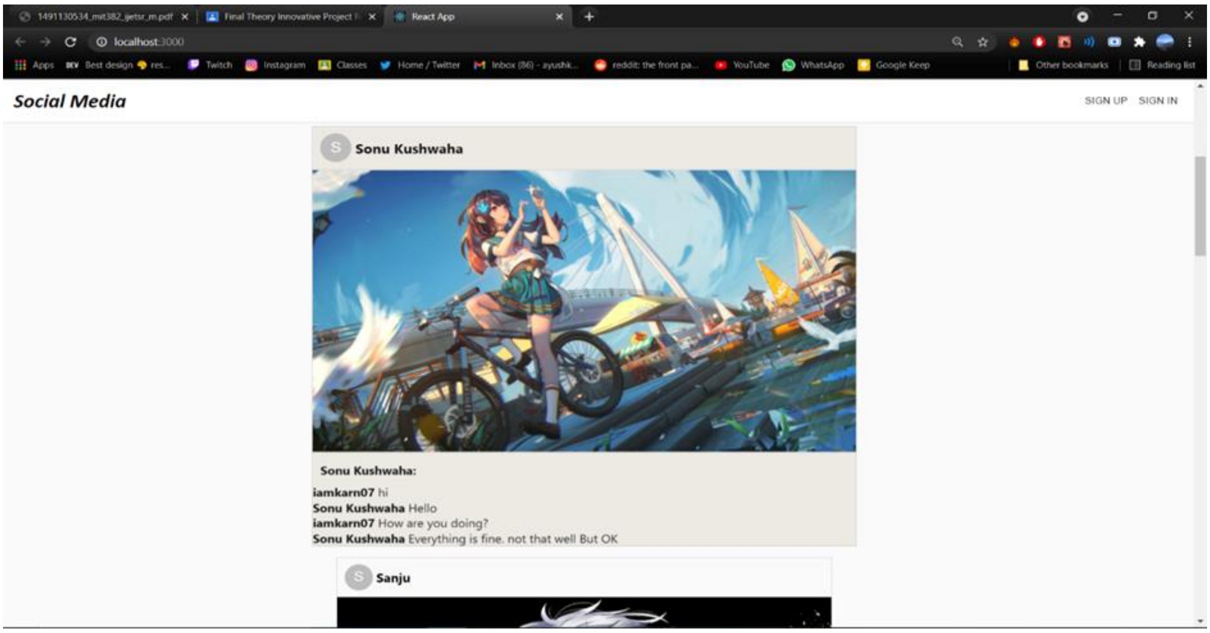


```
        {
          comments.map((comment) => (
            <p>
              <strong>{comment.username}</strong> {comment.text}
            </p>
          ))
        }
      </div>
      {user && (

        <form className = "post__commentsBox">
          <input
            className="post__input"
            type="text"
            placeholder="Add a comment.."
            value={comment}
            onChange={(e) => setComment(e.target.value)}
          />
          <button
            className="post__button"
            disabled={!comment}
            type="submit"
            onClick={postComment} >Post
          </button>

        </form>
      )}
    </div>
  )
}

export default Post;
```

Chapter 6

Result

The results that we got from learning React.JS and Firebase for just 3-4 months were very promising and we definitely could learn a lot more if we had more time up our sleeves. But because we had 5 other projects to work on we couldn't do everything that we wanted to. Some of the results screenshots are attached below. We were also able to Host our website on https://sefinaldtumate.web.app/ if you want to check this out.



Sonu Kushwaha:

iamkarn07 hi
Sonu Kushwaha Hello
iamkarn07 How are you doing?
Sonu Kushwaha Everything is fine. not that well But OK

Analysis

After analyzing what we have achieved from just 3-4 months of learning React.JS and Firebase we are pretty sure we could now work on bigger projects or we could just continue to add more features in this project. We could add chatting functionality to this project. We could add more types of post that could be posted on the website that includes Audio, Video, text-based posts, etc. We could also add more things such as following/ adding people as friends, keeping track of likes, dislikes, etc.

One innovative thing that we did in this project is that, by using react instead of other similar framework we saved a lot of data while browsing the website. For example: when a user comments on a post or when a user uploads a post with the help of react we are just refreshing that specific part of DOM and not the whole DOM. Because of this we don't have to fetch everything in the database but we could just fetch the part that is recently modified or added and we can save a lot of time and processing power.

Another innovative thing that we did in this project was using URL for image instead of the actual image inside of the database. This helped us reduce the size of the database. We also used firebase compression algorithm in order to reduce the size of the image and then stored that reduced sized image with no noticeable change in quality inside Firebase and used the URL for the same inside our database.

Chapter 7

Conclusion

From this project we would like to conclude that we have actually learnt a great deal about web development and a guy who was not at all interested in building web-based applications. I have actually come to like it and I will be making a few more projects on the web development. This innovative part that was added to our curriculum could prove to be very useful in future as we are trying to learn new things on our own and self-taught things are proven to be more effective as well as more useful for a person.

Not to forget that this Social Media website also has very good uses in today's world.

1) Can Broaden Your Brand if used properly for advertisement

2) Helps us connect with friends and family

3) Helps us meet new people.

4) Helps us stay in touch with everyone in a socially distant way and we all know how much necessary that is especially because of the pandemic we are in.

Last but not the least is that this opportunity to make something helped us learn a lot of things that we would continue to learn more about.

Chapter 8

References

[1] Singh H S., Singh U. “Study on Google Firebase for Website Development (The real time database)” International Journal of Engineering Technology Science and Research, Volume 4, Issue 3 March 2017

[2] Khawas C., Shah P. “Application of Firebase in Android App Development-A Study” International Journal of Computer Applications Volume 179(46): June 2018

[3] Yeshwin A., Sahoo A., Shoby P. “A Research Paper on a Pet-Friendly Application using Flutter and Firebase” International Journal for Research in Applied Science & Engineering Technology (IJRASET)Volume 8 Issue XII Dec 2020

[4] B. Traversy, “React JS Crash Course 2021” Jan 2021. [YouTube]

[5] V. Singh “Introduction to Firebase” Medium.com Dec 2018 [Medium.com]

[6] Firebase Documentation (https://firebase.google.com/docs)