Department of Computer Science & Application

Institute of Engineering &Technology

Mini Project - II REPORT

HouseHive - Your One Destination For Buying & Selling
(A Buying & Selling Platform)

SUBMITTED TO:-

Mr. Manoj Varshney
(Technical Trainer)

SUBMITTED BY:-

Himanshu Tenguriya(201500297)
Divyansh Sharma(201500232)
Garima Yadav(201500245)
Amit Lavania (201500082

DECLARATION

We, the undersigned members declare that the group project submitted is our original work, and that we have acknowledged all the sources used in its preparation in accordance with the guidelines provided. We affirm that this project has not been submitted elsewhere for any other purpose, and that it does not infringe on the intellectual property rights of any third party. We also certify that all the information provided in this project is true and accurate to the best of our knowledge. We understand that any misrepresentation or falsification of information may result in the rejection of our submission or the revocation of any award or recognition that may be conferred upon us. By submitting this project, we agree to abide by all the rules and regulations governing here.

Name of candidate:

Himanshu Tenguriya (201500297)

Divyansh Sharma (201500232)

Garima Yadav (201500245)

Amit Lavania (201500082)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to every one who contributed to the successful completion of this web development group project. Firstly, we would like to thank our project supervisor, Mr. Manoj Varshney for providing us with invaluable guidance and support throughout the project. Your expertise in web development and constructive feedback helped us to refine our ideas and approach. We would also like to extend our gratitude to our team members, for their dedication and hard work. Each team member brought their unique set of skills and expertise, and the collaboration and support through out the project were instrumental in its success. Finally, we would like to acknowledge the vast resources and online communities that provided us with valuable information and insight into web development, which helped us to overcome challenges and achieve our project goals. Thank you all for your contribution and support.

Himanshu Tenguriya (201500297)

Divyansh Sharma (201500232)

Garima Yadav (201500245)

Amit Lavania (201500082)

CERTIFICATE

This is to certify that this group has successfully completed the group project synopsis on a selling and buying platform (HouseHive). The group has demonstrated a high level of knowledge, creativity, and innovation in the field of social media platform development. The project involved the design and development of a social media platform that enables users to buy and sell their own property on the platform. The group members worked collaboratively to complete the project within the given timeframe and to the satisfaction of the project supervisor. The group demonstrated excellent communication, collaboration, and problem-solving skills, which were essential for the successful completion of the project. We hope that the knowledge and experience gained from this project will help the group members to excel in their future endeavors. We wish them all the best in their future endeavors.

Date: Sign:

INDEX

INTRODUCTION

Welcome to our house selling and renting website, where we strive to make the process of buying, selling, or renting a home as seamless and stress-free as possible. With a vast selection of properties to choose from and a user-friendly platform, we are your one-stop destination for all your real estate needs. Our team of experienced professionals is dedicated to providing personalized service and guidance every step of the way. Whether you're a first-time buyer, a seasoned investor, or simply searching for your dream home, we're here to help you unlock the door to a brighter future. At our house selling and renting website, we understand that finding the right home is one of the most important decisions you'll make in life. That's why we've created a platform that offers a wide range of properties to buy, sell, or rent, along with a suite of tools and resources to make the process easier. Whether you're searching for a cozy apartment or a spacious family home, we have something for everyone. Our team of experts is committed to helping you navigate the complex world of real estate with confidence, so you can find the perfect place to call home. Join us today and start your journey towards a brighter future. Looking for your dream home can be a daunting task, but with our house selling and renting website, it doesn't have to be. We provide a simple and effective platform that connects buyers, sellers, and renters with ease. With a wide range of properties available at your fingertips, you can browse, compare, and choose the perfect home for your needs. Our team of experts is here to support you every. Step of the way, from initial inquiry to final closing. Whether you're looking for your first home, a vacation rental, or a real estate investment, our website provides you with everything you need to make an informed decision. With powerful search filters, detailed property descriptions, and a range of multimedia content, you can explore each property as if you were there in person. Plus, our team of experienced professionals is always available to answer any questions you may have and offer personalized guidance. We believe that buying, selling, or renting a home should be a positive and empowering experience, and we're committed to making that a reality for every client who comes through our virtual doors. Join us today and discover the home of your dreams.

Title of the Project

- HouseHive

A house selling and renting website, where we strive to make the process of buying, selling, or renting a home as seamless and stress-free as possible.

Objective

The main objective of this project is to develop a website that will provide a platform for people to buy, sell, and rent houses. The website will provide users with an easy-to-use interface that will allow them to search for properties based on their preferences. The objective of a house selling and renting website is to provide a platform for buyers, sellers, and renters of real estate properties to connect with each other. This objective can be achieved through a variety of features, such as property listings, user authentication, direct communication between users, and database management. By building a user-friendly and practical website, the main objective is to provide a reliable and convenient solution for people looking to buy, sell, or rent real estate properties. Additionally, the website can aim to provide a secure and efficient way for users to manage their property listings and communicate with potential buyers or renters. Ultimately, the main objective is to create a successful and functional website that satisfies the needs of its users and contributes to the development of the real estate industry.

Why This Project?

A house selling and renting website can be a valuable resource for people who are in the market for a new home. Such a website can help connect sellers and renters with potential buyers and tenants, providing a platform for them to showcase their properties and find interested parties. Additionally, a house selling and renting website can offer various tools and resources to help users make informed decisions about their transactions. This can include features like property listings, virtual tours, neighborhood guides, and financial calculators, among others. Overall, a house selling and renting website has the potential to simplify the process of buying or renting a home, making it more accessible and transparent for all parties involved.

Features of The Project

- Search Filters: A search filter is essential to make it easier for users to find properties that match their requirements. It could include filters for location, price range, number of rooms, property type, and other important criteria.- Social Networking Features: The platform enables users to follow other users, engage with content through likes, comments, and shares, and discover new content and users through a robust search and discovery system.

- Property Listings: A website for buying and renting houses should have a property listing section that displays all available properties for sale or rent.

- High-Quality Images: High-quality images of the properties for sale or rent are a must-have feature. These images should be clear, visually appealing, and should give users a good idea of what the property looks like.- Database Management: The platform was built using MongoDB, a NoSQL database used for storing and managing large volumes of data.

- Property Details: A detailed description of the property is necessary to give users an accurate understanding of what they are buying or renting. This should include the number of bedrooms, bathrooms, square footage, property age, and any other relevant details.

- Contact Form: A contact form is necessary for users to get in touch with the real estate agent or property owner to ask questions or arrange a viewing.

Application

A house selling and renting website can be a useful tool for both buyers and sellers/landlords. Here are some potential applications of such a website:

- Property Management: The website can provide property management services to landlords who need help managing their properties. This can include services such as tenant screening, rent collection, maintenance and repairs, and lease preparation.

- Customer Support: The website can provide customer support services to both buyers and sellers/landlords. This can include answering questions about the properties listed on the site, providing guidance on the buying/selling process, and addressing any issues that may arise during the transaction.

- Property Listings: The website can feature property listings for sale or rent. Buyers can browse through these listings to find properties that match their requirements, while sellers/landlords can list their properties on the website to attract potential buyers/tenants.

- Real-time Notifications: The website can provide real-time notifications to both buyers and sellers/landlords about new properties that have been listed or that have been sold/rented. This can help buyers stay informed about new properties that may meet their needs, while sellers/landlords can stay up-to-date on the current market conditions.

Overall, a house selling and renting website can be a valuable tool for anyone involved in the real estate market. It can help streamline the buying and selling process, provide access to a wider range of properties, and provide valuable resources and support to both buyers and sellers/landlords.

Module In The Project

1. User Management: This module will allow users to create an account, log in, and manage their profile information. It will also allow users to view their saved property listings, search history, and other relevant details.
2. Property Listing Management: This module will allow property owners to create and manage their property listings, including adding property details, uploading photos, setting prices, and managing inquiries from potential buyers or renters.
3. Property Valuation: This module will allow users to get an estimated value for their property based on various factors such as location, property type, and market trends. It may also provide users with tips for improving the value of their property.
4. Communication: This module will allow users to communicate with each other regarding property listings, including sending messages, making inquiries, and scheduling viewings.
5. Property Management: This module will allow property owners to manage their rental properties, including features such as tenant screening, lease management, and rent collection. It may also include tools for managing maintenance requests and property inspections.
6. Analytics and Reporting: This module will provide data analytics and reporting features to help users understand market trends, property demand, and other relevant metrics.
7. Admin Management: This module will allow the website administrator to manage user accounts, property listings, and other relevant details, as well as perform site maintenance tasks.

Overall, these modules work together to provide a comprehensive and user-friendly house selling and renting website that meets the needs of both property owners and potential buyers or renters.

SYSTEM REQUIREMENTS

Hardware Requirements :

1. Processor: Intel Corei5orhigher
2. RAM: 8GB or higher
3. Storage: 256GB or higher
4. Internet Connection: High-speed internet connection

Software Requirements :

1. Operating System: Windows10ormacOS High Sierra or later
2. Integrated Development Environment (IDE): Visual Studio Code or any other suitable IDE for React JS development.
3. Node.js: The latest stable version of Node.js and npm package manager
4. MongoDB : The latest stable version of MongoDB database management system
5. Git : The latest stable version of Git version control system.
6. Web Browser : Google Chrome or any other modern web browser

In addition to the above requirements, developers may also need to use additional libraries , frameworks, and tools depending on the project's requirements and complexity. It is recommended to use the latest versions of all the required software to ensure compatibility and stability.

Theoretical Aspects of Web Development Project

Web development refers to building website and deploying on the web. Web development requires use of scripting languages both at the server end as well as at client end.

Before developing a web site once should keep several aspects in mind like:

- What to put on the web site?
- Who will host it?
- How to make it interactive?
- How to code it?
- How to create search engine friendly web site?

Web Development Process :

Web development process includes all the steps that are good to take to build an attractive, effective and responsive website.

Web development tools :

Web development tools helps the developer to test and debug the web sites. Now a days the web development tool come with the web browsers as add-ons. All web browsers have built in tools for this purpose.

These tools allow the web developer to use HTML, CSS and JavaScript etc. These are accessed by hovering over an item on a web page and selecting the “Inspect Element” from the context menu.

Scope of the Project

The scope of a house selling and renting website can be significant in the future for society. As technology advances and more people rely on online platforms for various activities, a website that connects buyers, sellers, and renters of real estate properties can become increasingly relevant and useful. Some potential benefits for society include:

- Convenience: A house selling and renting website can provide a convenient and accessible way for people to buy, sell or rent real estate properties. By having a platform that is available 24/7, users can browse listings, communicate with potential buyers or renters, and manage their property listings from the comfort of their own home.

- Efficiency: A website that automates processes such as property listing and communication between users can increase efficiency and save time for all parties involved. This can benefit the real estate industry by making it more streamlined and less time-consuming.

- Transparency: By providing a platform for direct communication between buyers and sellers, a house selling and renting website can increase transparency in the real estate market. This can lead to more informed decision-making and potentially more fair and equitable transactions.

- Accessibility: A house selling and renting website can provide accessibility to people who may not have had access to the real estate market otherwise, such as people who live in remote areas or who have mobility issues.

- Increased competition: A house selling and renting website can lead to increased competition in the real estate market, which can benefit both buyers and sellers. By having more options available, buyers can find better deals and sellers can potentially sell their properties at a higher price.

- Improved data analysis: By collecting and analyzing data on property listings, user behaviour , and market trends, a house selling and renting website can provide valuable insights into the real estate market. This can be useful for buyers, sellers, and real estate professionals in making informed decisions.

Overall, the scope of a house selling and renting website in the future for society can be significant in terms of providing convenience, efficiency, transparency, and accessibility to the real estate market. By building a functional and user-friendly website, the potential benefits for society can be numerous.

Implementation

Implementation is the stage in the project where the theoretical design is turned into the working system and is giving confidence to the new system for the users i.e. will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of method to achieve the changeover, an evaluation, of change over methods Define the goals and features of house selling and renting site: Before you start building your house selling and renting site, you need to define what your site is going to do, who your target audience is, and what features you want to include. Some features might include user profiles, a search filters, property details, and property listings. Choose a programming language and framework: There are many programming languages and frameworks you can use to build a house selling and renting site. Build the database: Your house selling and renting site will need a database to store user information and other data. You can use a variety of database management systems, such as MySQL ,to build your database. Develop the user interface: The user interface is what your users will interact with, so it's important to make sure it's easy to use and visually appealing. You can use HTML, CSS, and JavaScript to develop your user interface. Implement user authentication and security features: User authentication is essential for any house selling and renting site to ensure that only authorized users can access the site. You'll also want to implement security features to protect user data and prevent unauthorized access to the site. Add functionality: Once we have the basic framework in place, we can start adding functionality to our social media site. This might include features like , a search filters, property details, property listings, and High-Quality Images. Test the website: Once the website is developed, you need to test it thoroughly. This includes testing the functionality of the website, checking for bugs and errors, and ensuring that the website works on all devices and browsers.

SOURCE CODE

Home Page

```
import React from "react";

import { useNavigate } from "react-router-dom";

import Slider from "../components/Slider";

import Layout from "./../components/Layout/Layout";

import "../styles/homepage.css";



const HomePage = () => {

  const navigate = useNavigate();

  const img1 =

    "https://images.unsplash.com/photo-1600585154340-be6161a56a0c?ixlib=rb-1.2.1&ixid=MnwxMjA3fDB8MHxzZWFyY2h8Mnx8cHJvcGVydHl8ZW58MHx8MHx8&auto=format&fit=crop&w=500&q=60";

  const img2 =

    "https://images.unsplash.com/photo-1626178793926-22b28830aa30?ixlib=rb-1.2.1&ixid=MnwxMjA3fDB8MHxzZWFyY2h8M3x8cHJvcGVydHl8ZW58MHx8MHx8&auto=format&fit=crop&w=500&q=60";

  return (

    <Layout>

      <Slider />

      <div className="home-cat row d-flex align-items-center justify-content-center">
```

```
        <h1>Category</h1>
        <div className="col-md-5 ">
          <div className="Imagecontainer">
            <img src={img1} alt="Rent" style={{ width: "100%" }} />
            <button className="btn" onClick={() => navigate("/category/rent")}>
              FOR RENT
            </button>
          </div>
        </div>
        <div className="col-md-5">
          <div className="Imagecontainer">
            <img src={img2} alt="Rent" style={{ width: "100%" }} />
            <button className="btn" onClick={() => navigate("/category/sale")}>
              FOR SALE
            </button>
          </div>
        </div>
      </div>
    </Layout>
  );
};

export default HomePage;
```

COMPONENT

```
import React from "react";

import { useLocation, useNavigate } from "react-router-dom";

import { getAuth, signInWithPopup, GoogleAuthProvider } from
"firebase/auth";

import { doc, setDoc, getDoc, serverTimestamp } from "firebase/firestore";

import { db } from "../firebase.config";

import { toast } from "react-toastify";

import { FcGoogle } from "react-icons/fc";


const OAuth = () => {

  const navigate = useNavigate();

  const location = useLocation();


  const onGoolgleAuthHandler = async () => {

    try {

      const auth = getAuth();

      const provider = new GoogleAuthProvider();

      const result = await signInWithPopup(auth, provider);

      const user = result.user;

      const docRef = doc(db, "users", user.uid);

      const docSnap = await getDoc(docRef);

      if (!docSnap.exists()) {

        await setDoc(doc(db, "users", user.uid), {
```

```
        name: user.displayName,
        email: user.email,
        timestamp: serverTimestamp(),
      });
    }
    navigate("/");
  } catch (error) {
    toast.error("Problem With Google Auth ");
  }
};

return (
  <div>
    <h3 className="mt-4 text-center ">
      Sign {location.pathname === "/signup" ? "Up" : "in"} With
      <button
        onClick={onGoolgleAuthHandler}
        style={{
          outline: "none",
          backgroundColor: "transparent",
          border: "none",
          borderBottom: "1px solid black",
        }}
```

```
          >
            <span>
              <FcGoogle />
              oogle
            </span>
          </button>
        </h3>
      </div>
    );
  };

  export default OAuth;
```

Listing Items

```
import React from "react";
import { Link } from "react-router-dom";
import { FaBed, FaBath } from "react-icons/fa";
import { GiTakeMyMoney } from "react-icons/gi";
import "../styles/listingitem.css";
const ListingItem = ({ listing, id, onDelete, onEdit }) => {
  return (
    <>
      <div className="card-item-parent d-flex align-items-center justify-
content-center">
        <div className="item-card category-link mb-2 w-75  ">
```

```
          <Link to={`/category/${listing.type}/${id}`}>
            <div className="row  p-2">
              <div className="col-md-5 item-card-continer1">
                <img src={listing.imgUrls[0]} alt={listing.name} />
              </div>
              <div className="col-md-5 item-card-continer2">
                <h2>{listing.name}</h2>
                <p>{listing.location}</p>
                <p>
                  <GiTakeMyMoney /> RS :{" "}
                  {listing.offer
                    ? listing.discountedPrice
                    : listing.regularPrice}{" "}
                  {listing.type === "rent" && " / Month"}
                </p>
                <p>
                  <FaBed /> &nbsp;
                  {listing.bedrooms > 1
                    ? `${listing.bedrooms} Bedrooms`
                    : "1 Bedroom"}
                </p>
                <p>
                  <FaBath /> &nbsp;
            {listing.bathrooms > 1
              ? `${listing.bathrooms} Bathrooms`
              : "1 Bathroom"}
          </p>
        </div>
      </div>
    </Link>
    <div className="m-2 p-3">
      {onDelete && (
        <button
          className="btn btn-danger"
          onClick={() => onDelete(listing.id)}
        >
          Delete Listing
        </button>
      )}
      {onEdit && (
        <button
          className="btn btn-info ms-3"
          onClick={() => onEdit(listing.id)}
        >
          Edit Listing
        </button>
```

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

export default ListingItem;
```

Slider

```
import React, { useState, useEffect } from "react";
import { ImLocation2 } from "react-icons/im";
import { db } from "../firebase.config";
import "../styles/slider.css";
import {
  collection,
  getDoc,
  query,
  orderBy,
  limit,
  getDocs,
} from "firebase/firestore";
import { useNavigate } from "react-router-dom";
import SwipeCore, { EffectCoverflow, Navigation, Pagination } from
"swiper";

import { Swiper, SwiperSlide } from "swiper/react";

import "swiper/swiper-bundle.min.css";

import "swiper/swiper.min.css";

import Spinner from "./Spinner";


//config

SwipeCore.use([EffectCoverflow, Pagination]);


const Slider = () => {

  const [listings, setListings] = useState(null);

  const [loading, setLoading] = useState(true);

  const navigat = useNavigate();

  const userPic =

    "https://openclipart.org/download/247319/abstract-user-flat-3.svg";


  useEffect(() => {

    const fetchListings = async () => {

      const listingRef = collection(db, "listings");

      const q = query(listingRef, orderBy("timestamp", "desc"), limit(5));

      const querySnap = await getDocs(q);

      let listings = [];

      querySnap.forEach((doc) => {
        return listings.push({
          id: doc.id,
          data: doc.data(),
        });
      });
      setLoading(false);
      setListings(listings);
    };
    fetchListings();
    console.log(listings === null ? "loading" : listings);
    // eslint-disable-next-line
  }, []);

  if (loading) {
    return <Spinner />;
  }
  return (
    <>
      <div style={{ width: "100%" }}>
        {listings === null ? (
          <Spinner />
        ) : (
          <Swiper
  effect={"coverflow"}
  grabCursor={true}
  centeredSlides={true}
  slidesPerView={1}
  coverflowEffect={{
    rotate: 50,
    stretch: 0,
    depth: 100,
    modifier: 1,
    slideShadows: true,
  }}
  pagination={true}
  className="mySwipe"
>
  {listings.map(({ data, id }) => (
    <SwiperSlide
      key={id}
      onClick={() => {
        navigat(`/category/${data.type}/${id}`);
      }}
    >
      <img
        src={data.imgUrls[0]}
```

```
                  alt={data.name}

                  className="slider-img"

                />

                <h4 className=" text-light p-4 m-0 ">

                  {/* <img alt="user pic" src={userPic} height={35} width={35}
/> */}

                  <ImLocation2 size={20} className="ms-2" /> Recently Added
:{" "}

                  <br />

                  <span className="ms-4 mt-2"> {data.name}</span>

                  <span className="ms-2">

                    | Price ( Rs {data.regularPrice} )

                  </span>

                </h4>

              </SwiperSlide>

            ))}

          </Swiper>

        )}

      </div>

    </>

  );

};


export default Slider;
```

SIGN IN

```
import React, { useState, useEffect } from "react";
import { toast } from "react-toastify";
import { useNavigate, Link } from "react-router-dom";
import Layout from "../components/Layout/Layout";
import { getAuth, updateProfile } from "firebase/auth";
import { db } from "../firebase.config";
import { FaEdit, FaArrowAltCircleRight } from "react-icons/fa";
import { MdDoneOutline } from "react-icons/md";
import {
  doc,
  updateDoc,
  collection,
  getDocs,
  query,
  where,
  orderBy,
  deleteDoc,
} from "firebase/firestore";
import ListingItem from "../components/ListingItem";
import "../styles/profile.css";

const Profile = () => {
```

```
const auth = getAuth();
const navigate = useNavigate();
// eslint-disable-next-line
const [loading, setLoading] = useState(true);
const [listings, setListings] = useState(null);

//useeffect for getting data
useEffect(() => {
  const fetchUserListings = async () => {
    const listingRef = collection(db, "listings");
    const q = query(
      listingRef,
      where("useRef", "==", auth.currentUser.uid),
      orderBy("timestamp", "desc")
    );
    const querySnap = await getDocs(q);
    console.log(querySnap);
    let listings = [];
    querySnap.forEach((doc) => {
      return listings.push({
        id: doc.id,
        data: doc.data(),
      });
      });
      console.log(listings);
      setListings(listings);
      setLoading(false);
    };
    fetchUserListings();
  }, [auth.currentUser.uid]);
  const [changeDetails, setChangeDetails] = useState(false);
  const [formData, setFormData] = useState({
    name: auth.currentUser.displayName,
    email: auth.currentUser.email,
  });
  const { name, email } = formData;

  const logoutHandler = () => {
    auth.signOut();
    toast.success("Successfully Logout");
    navigate("/signin");
  };

  //onChange
  const onChange = (e) => {
    setFormData((prevState) => ({
```

```
    ...prevState,
    [e.target.id]: e.target.value,
  }));
};
//submit handler
const onSubmit = async () => {
  try {
    if (auth.currentUser.displayName !== name) {
      await updateProfile(auth.currentUser, {
        displayName: name,
      });
      const userRef = doc(db, "users", auth.currentUser.uid);
      await updateDoc(userRef, { name });
      toast.success("User Updated!");
    }
  } catch (error) {
    console.log(error);
    toast("Something Went Wrong");
  }
};

//delete handler
const onDelete = async (listingId) => {
    if (window.confirm("Are You Sure  want to delete ?")) {
      // await deleteDoc(doc, (db, "listings", listingId));
      await deleteDoc(doc(db, "listings", listingId));
      const updatedListings = listings.filter(
        (listing) => listing.id !== listingId
      );
      setListings(updatedListings);
      toast.success("Listing Deleted Successfully");
    }
  };

  //edit handler
  const onEdit = (listingId) => {
    navigate(`/editlisting/${listingId}`);
  };
  return (
    <Layout>
      <div className="row profile-container">
        <div className="col-md-6 profile-container-col1">
          <img src="./assets/profile.svg" alt="profile" />
        </div>
        <div className="col-md-6 profile-container-col2">
          <div className="container mt-4  d-flex justify-content-
between">
```

```
            <h2>Profile Details</h2>

            <button className="btn btn-danger"
      onClick={logoutHandler}>

             Logout

            </button>

           </div>

           <div className="   mt-4 card">

            <div className="card-header">

             <div className="d-flex justify-content-between ">

              <p>Your Personal Details </p>

              <span

               style={{ cursor: "pointer" }}

               onClick={() => {

                changeDetails && onSubmit();

                setChangeDetails((prevState) => !prevState);

               }}

              >

               {changeDetails ? (

                <MdDoneOutline color="green" />

               ) : (

                <FaEdit color="red" />

               )}

              </span>

             </div>
```

```
          </div>
          <div className="card-body">
            <form>
              <div className="mb-3">
                <label htmlFor="exampleInputPassword1"
className="form-label">
                  Name
                </label>
                <input
                  type="text"
                  className="form-control"
                  id="name"
                  value={name}
                  onChange={onChange}
                  disabled={!changeDetails}
                />
              </div>
              <div className="mb-3">
                <label htmlFor="exampleInputEmail1" className="form-
label">
                  Email address
                </label>
                <input
                  type="email"
                  value={email}
                  className="form-control"
                  id="email"
                  aria-describedby="emailHelp"
                  onChange={onChange}
                  disabled={!changeDetails}
                />
              </div>
            </form>
          </div>
        </div>
        <div className="mt-3 create-listing">
          <Link to="/create-listing">
            <FaArrowAltCircleRight color="primary" /> &nbsp; Sell or
Rent Your
            Home
          </Link>
        </div>
      </div>
    </div>


    <div className="container-fluid mt-4 your-listings">
      {listings && listings?.length > 0 && (
        <>
```

```
            <h3 className="mt-4">Your Listings</h3>
            <div>
              {listings.map((listing) => (
                <ListingItem
                  className="profile-listing"
                  key={listing.id}
                  listing={listing.data}
                  id={listing.id}
                  onDelete={() => onDelete(listing.id)}
                  onEdit={() => onEdit(listing.id)}
                />
              ))}
            </div>
          </>
        )}
      </div>
    </Layout>
  );
};

export default Profile;
```

SIGN UP

```
import React, { useState } from "react";
import { Link, useNavigate } from "react-router-dom";
import Layout from "./../components/Layout/Layout";
import { toast } from "react-toastify";
import { BsFillEyeFill } from "react-icons/bs";
import {
  getAuth,
  createUserWithEmailAndPassword,
  updateProfile,
} from "firebase/auth";
import { db } from "../firebase.config";
import { doc, setDoc, serverTimestamp } from "firebase/firestore";
import OAuth from "../components/OAuth";
import "../styles/signup.css";

const Signup = () => {
  const [showPassword, setShowPassword] = useState(false);
  const [formData, setFormData] = useState({
    email: "",
    name: "",
    password: "",
  });
```

```
const { name, email, password } = formData;
const navigate = useNavigate();

const onChange = (e) => {
  setFormData((prevState) => ({
    ...prevState,
    [e.target.id]: e.target.value,
  }));
};

const onSubmitHndler = async (e) => {
  e.preventDefault();
  try {
    const auth = getAuth();
    const userCredential = await createUserWithEmailAndPassword(
      auth,
      email,
      password
    );
    const user = userCredential.user;
    updateProfile(auth.currentUser, { displayName: name });
    const formDataCopy = { ...formData };
    delete formDataCopy.password;
    formDataCopy.timestamp = serverTimestamp();
    await setDoc(doc(db, "users", user.uid), formDataCopy);
    toast.success("Signup Successfully !");
    navigate("/");
  } catch (error) {
    console.log(error);
    toast.error("Something Went Wrong");
  }
};
return (
  <Layout title="signup - house marketplace">
    <div className="row signup-container">
      <div className="col-md-6 signup-container-col-1">
        <img src="./assets/signup.svg" alt="welcome" />
      </div>
      <div className="col-md-6 signup-container-col-2">
        <form onSubmit={onSubmitHndler}>
          <h3 className=" mt-2 text-center ">Sign Up </h3>
          <div className="mb-3">
            <label htmlFor="exampleInputEmail1" className="form-label">
              Your Name
            </label>
            <input
```

```
              type="text"
              value={name}
              className="form-control"
              id="name"
              onChange={onChange}
              aria-describedby="nameHelp"
            />
          </div>
          <div className="mb-3">
            <label htmlFor="exampleInputEmail1" className="form-label">
              Email address
            </label>
            <input
              type="email"
              value={email}
              onChange={onChange}
              className="form-control"
              id="email"
              aria-describedby="emailHelp"
            />
          </div>
          <div className="mb-3">
            <label htmlFor="exampleInputPassword1" className="form-
label">
```

```
    Password
  </label>
  <input
    type={showPassword ? "text" : "password"}
    value={password}
    onChange={onChange}
    className="form-control"
    id="password"
  />
</div>
<div className="mb-3">
  show password
  <BsFillEyeFill
    className="text-danger ms-2  "
    style={{ cursor: "pointer" }}
    onClick={() => {
      setShowPassword((prevState) => !prevState);
    }}
  />
</div>
<button type="submit" className="btn signup-button">
  Sign up
</button>
```

```
            <span className="ms-4">Already User</span>{" "}
            <Link to="/signin">Login</Link>
            <div className="mt-3">
              <OAuth />
            </div>
          </form>
        </div>
      </div>
    </Layout>
  );
};

export default Signup;
```

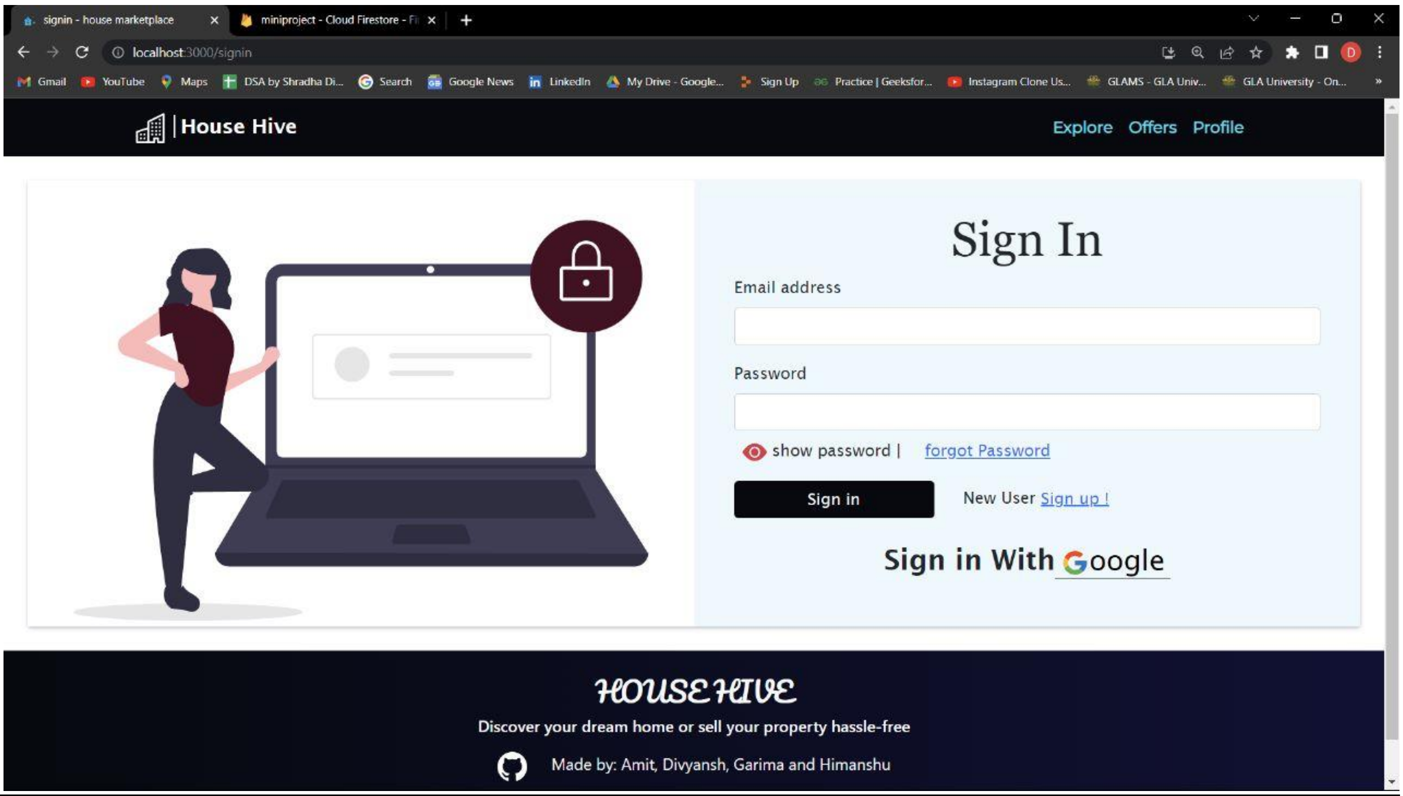
Forgot Password

```
import React, { useState } from "react";
import Layout from "./../components/Layout/Layout";
import { Link, useNavigate } from "react-router-dom";
import { getAuth, sendPasswordResetEmail } from "firebase/auth";
import { toast } from "react-toastify";
import "../styles/forgotpassword.css";

const ForgotPassword = () => {
  const [email, setEmail] = useState("");
const navigate = useNavigate();


const onSubmitHandler = async (e) => {
  e.preventDefault();
  try {
    const auth = getAuth();
    await sendPasswordResetEmail(auth, email);
    toast.success("Email was sent");
    navigate("/signin");
  } catch (error) {
    toast.error("Somthing went wrong");
  }
};
return (
  <Layout title="forgot password page">
    <div className="row forgot-password-container">
      <div className="col-md-7 forgot-password-col1">
        <img src="./assets/forgot-password.svg" alt="forgot-img" />
      </div>
      <div className="col-md-5 forgot-password-col2">
        <h1>Reset Your Password</h1>
        <form onSubmit={onSubmitHandler}>
          <div className=" mb-3">
```

```
   <label htmlFor="exampleInputEmail1" className="form-label">
     Email address
   </label>
   <input
     type="email"
     value={email}
     onChange={(e) => setEmail(e.target.value)}
     className="form-control"
     id="exampleInputEmail1"
     aria-describedby="emailHelp"
   />
   <div id="emailHelp" className="form-text">
     reset email will sent to this email
   </div>
 </div>
 <div className="d-flex justify-content-between btn-goup">
   <button type="submit" className="btn ">
     Reset Password
   </button>
   <button
     type="button"
     className="btn signin"
     onClick={() => navigate("/signin")}
```

```
              >
                Sing In
              </button>
            </div>
          </form>
        </div>
      </div>
    </Layout>
  );
};

export default ForgotPassword;
```

SNAPSHOTS


signin - house marketplace
miniproject - Cloud Firestore - Fi
localhost:3000/signin
Gmail YouTube Maps DSA by Shradha Di... Search Google News LinkedIn My Drive - Google... Sign Up Practice | Geeksfor... Instagram Clone Us... GLAMS - GLA Univ... GLA University - On...
House Hive
Explore
Offers
Profile

Sign In

Email address

Password

show password | forgot Password

Sign in

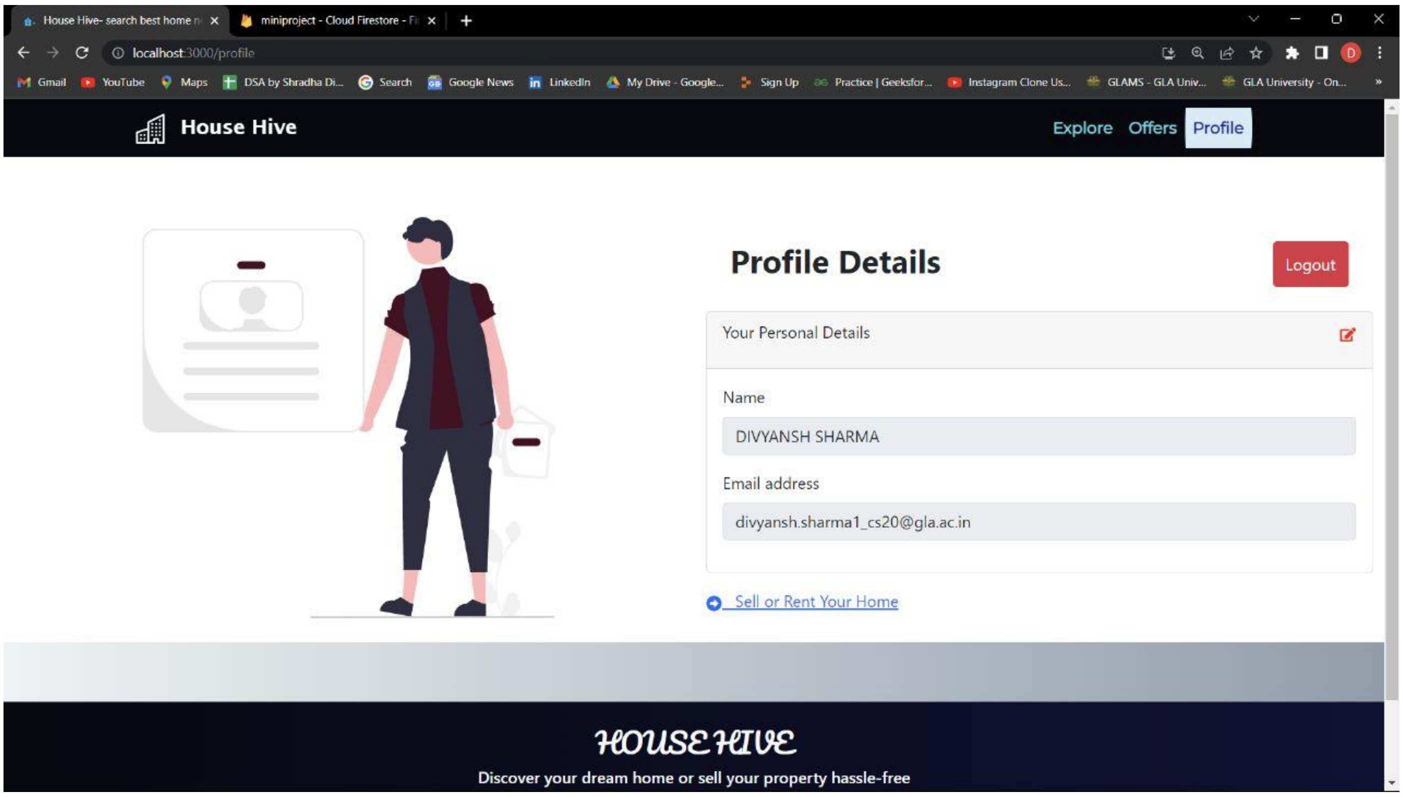
New User Sign up !

Sign in With Google

HOUSE HIVE

Discover your dream home or sell your property hassle-free

Made by: Amit, Divyansh, Garima and Himanshu


House Hive- search best home
miniproject - Cloud Firestore - Fi
localhost:3000/profile
Gmail YouTube Maps DSA by Shradha Di... Search Google News LinkedIn My Drive - Google... Sign Up Practice | Geeksfor... Instagram Clone Us... GLAMS - GLA Univ... GLA University - On...
House Hive
Explore
Offers
Profile

Profile Details

Logout

Your Personal Details

Name

DIVYANSH SHARMA

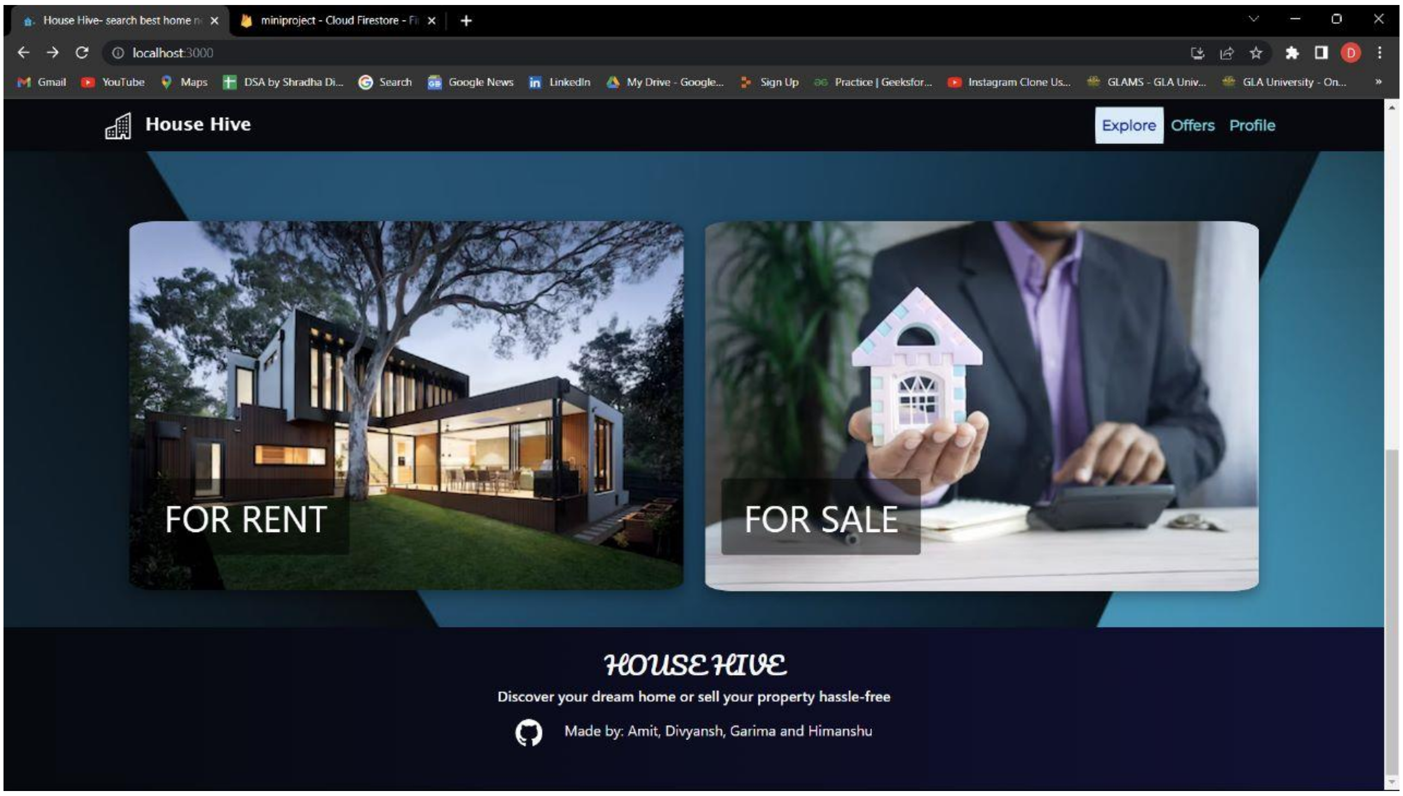
Email address

divyansh.sharma1_cs20@gla.ac.in

Sell or Rent Your Home

HOUSE HIVE

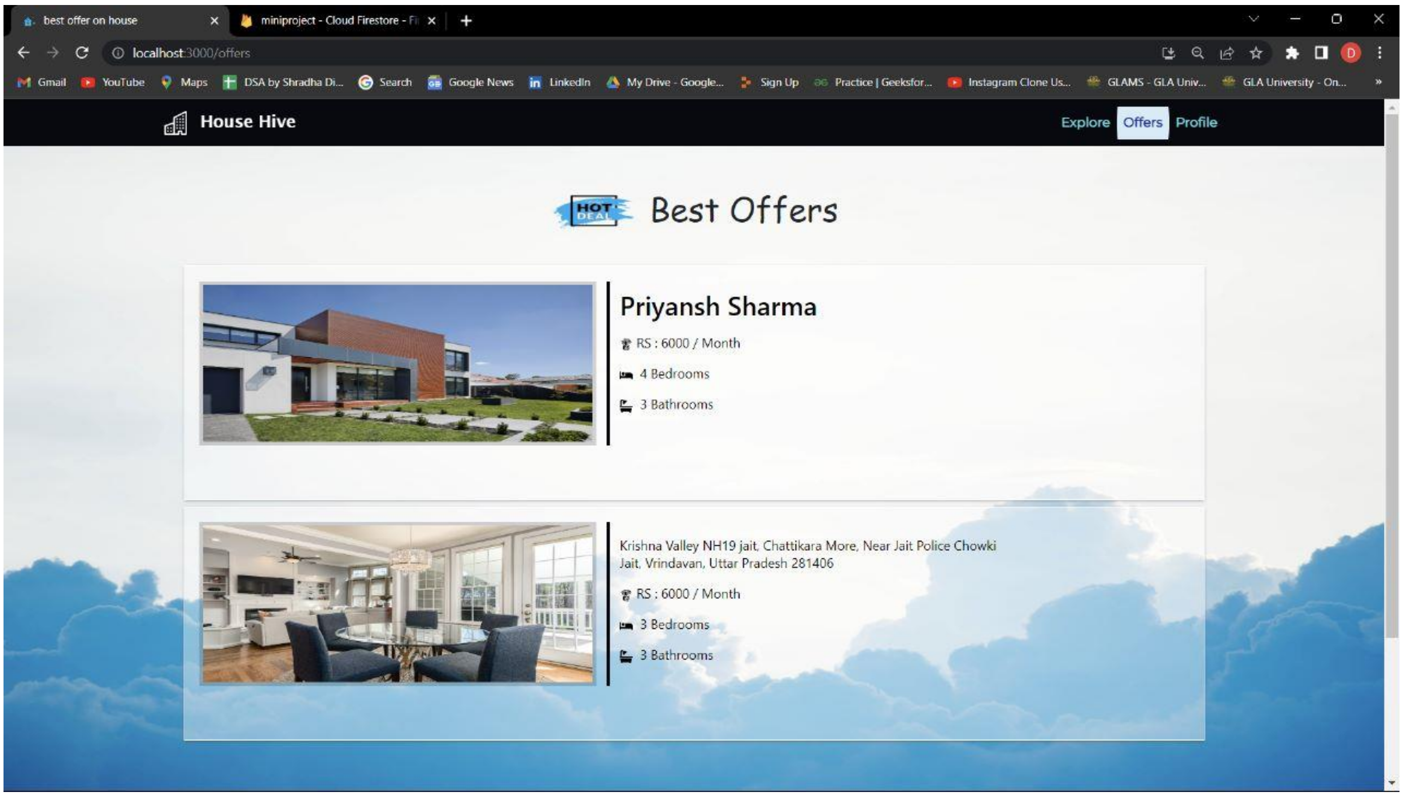
Discover your dream home or sell your property hassle-free



HOUSE HIVE

Discover your dream home or sell your property hassle-free

Made by: Amit, Divyansh, Garima and Himanshu



Best Offers

Priyansh Sharma

RS : 6000 / Month

4 Bedrooms

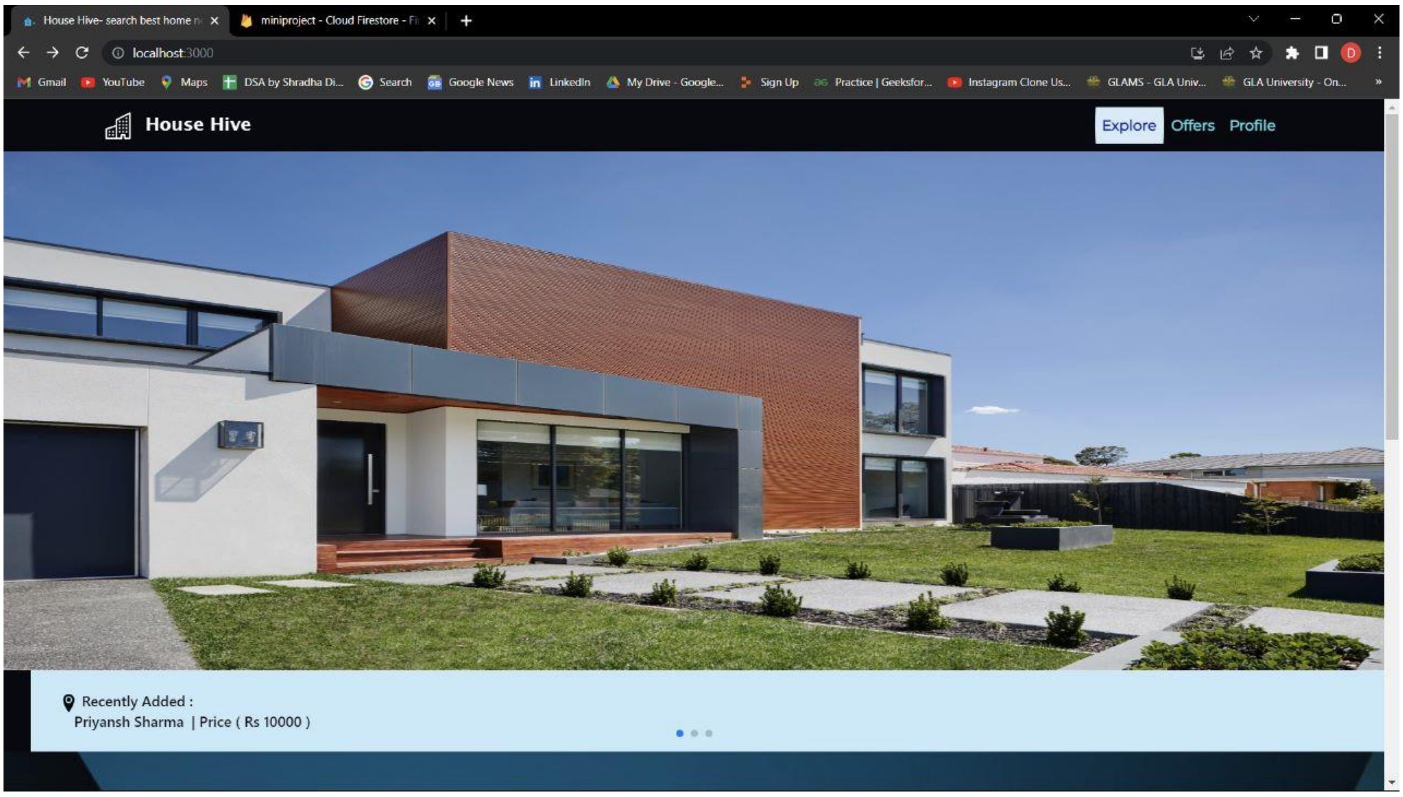
3 Bathrooms

Krishna Valley NH19 jait, Chattikara More, Near Jait Police Chowki Jait, Vrindavan, Uttar Pradesh 281406

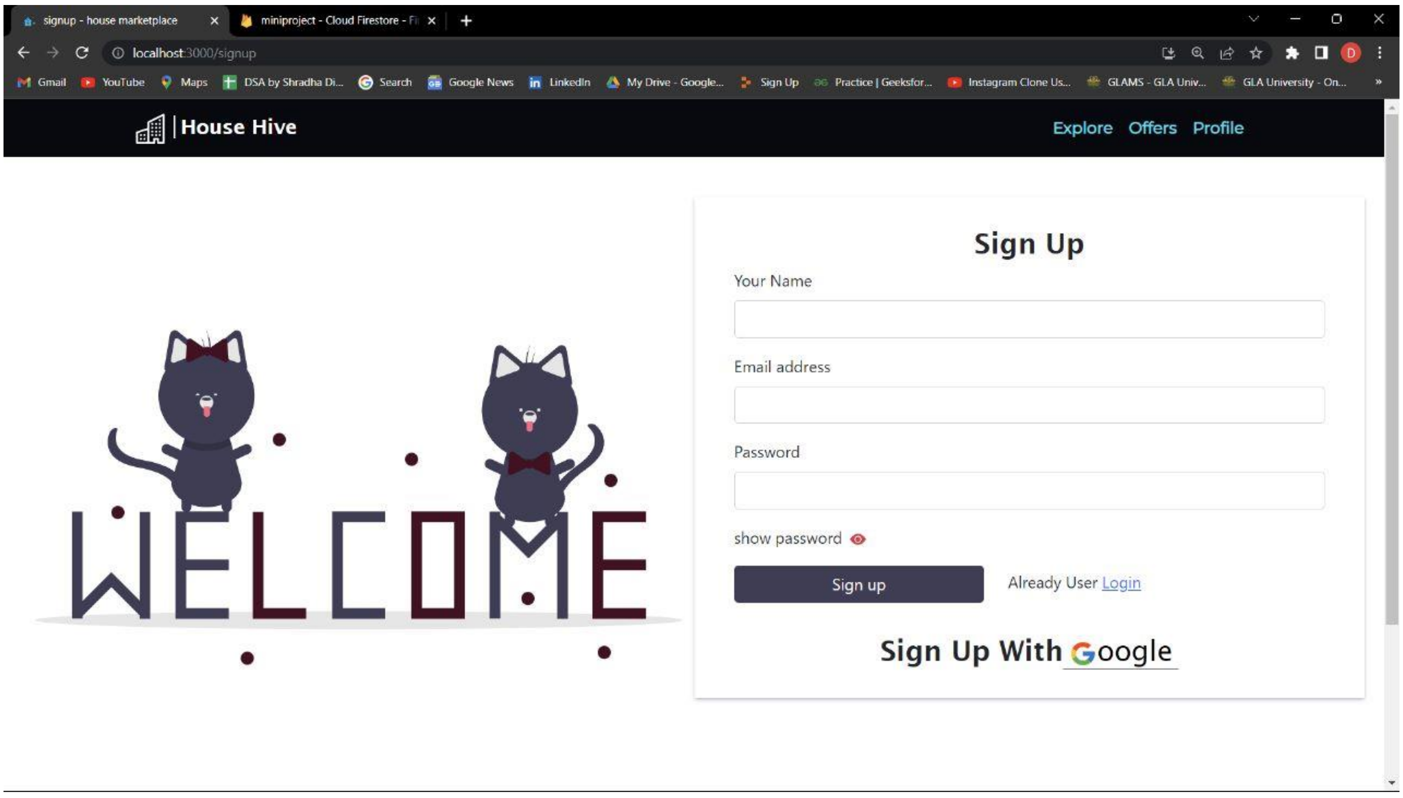
RS : 6000 / Month

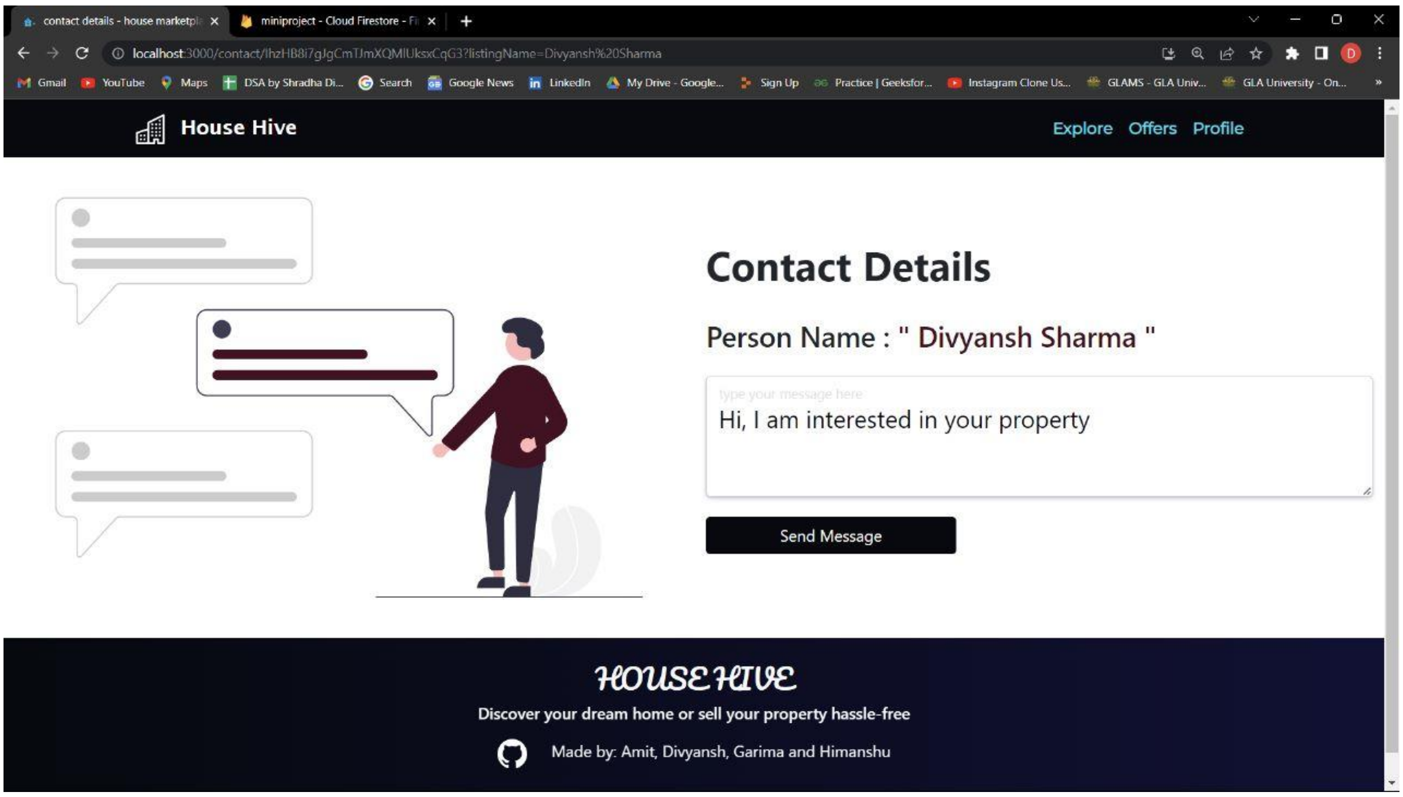
3 Bedrooms

3 Bathrooms



Recently Added :
Priyansh Sharma | Price (Rs 10000)





Contact Details

Person Name : " Divyansh Sharma "

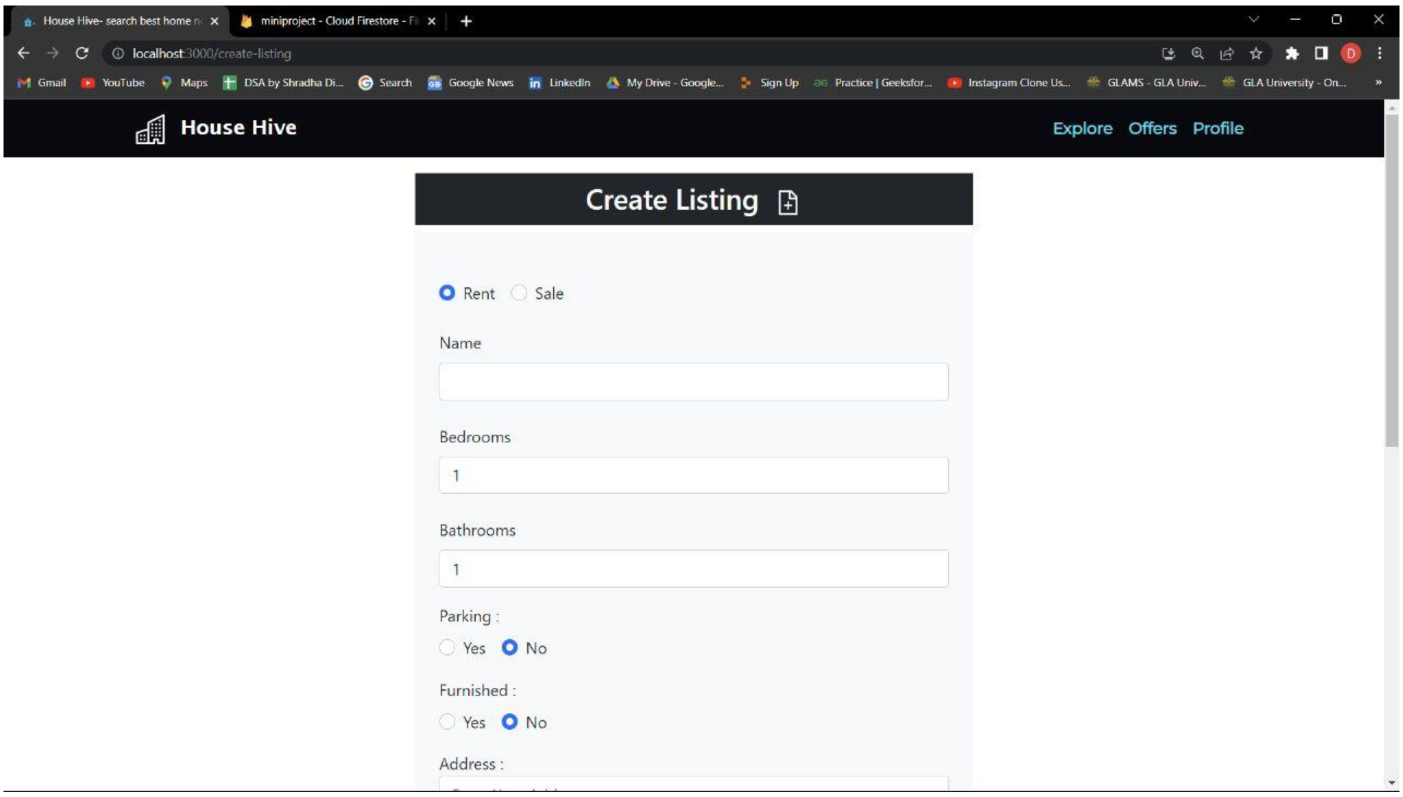
type your message here

Hi, I am interested in your property

HOUSE HIVE

Discover your dream home or sell your property hassle-free

Made by: Amit, Divyansh, Garima and Himanshu



Create Listing

[✓ Rent] Sale

Name

Bedrooms

1

Bathrooms

1

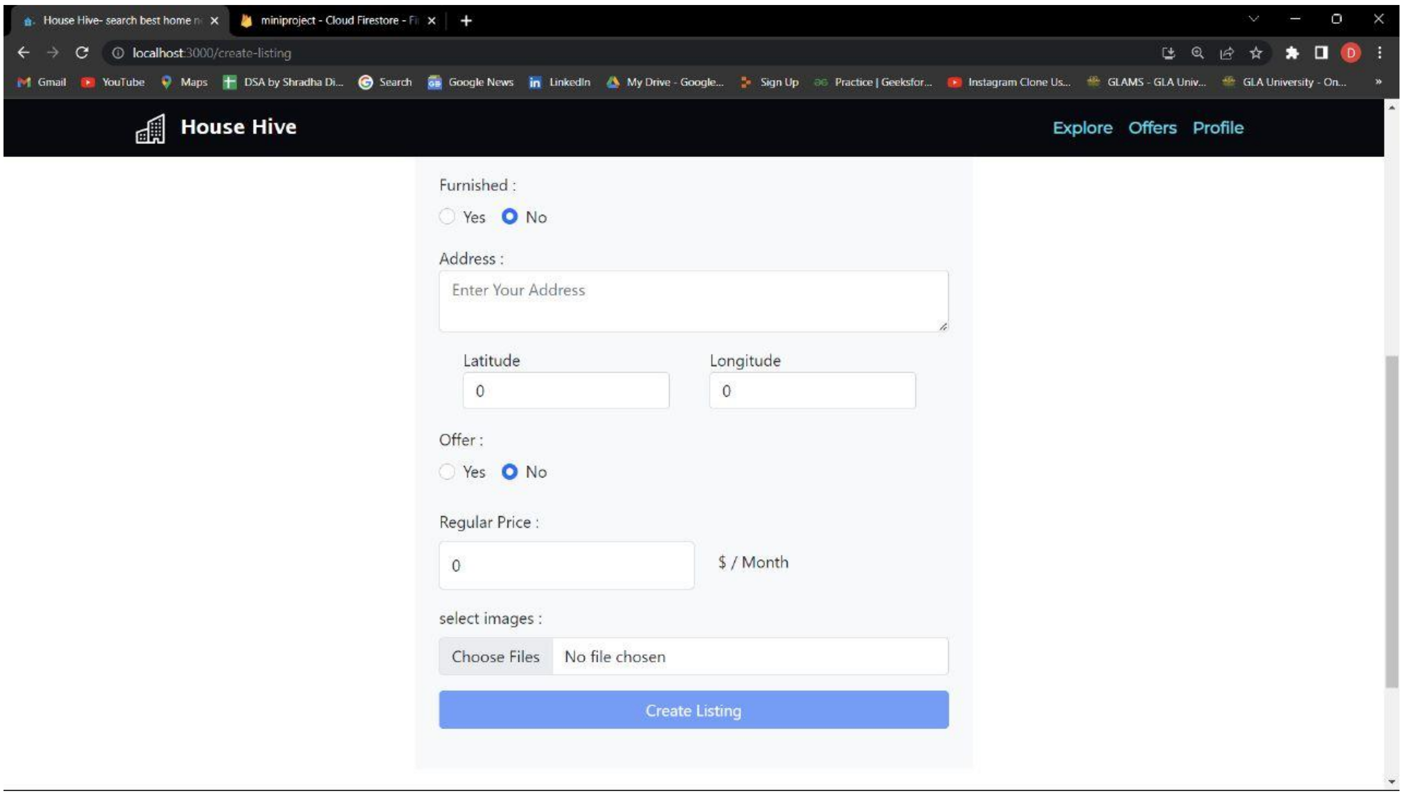
Parking :

Yes [✓ No]

Furnished :

Yes [✓ No]

Address :



Furnished :
Yes [✓ No]

Address :
Enter Your Address

Latitude
0

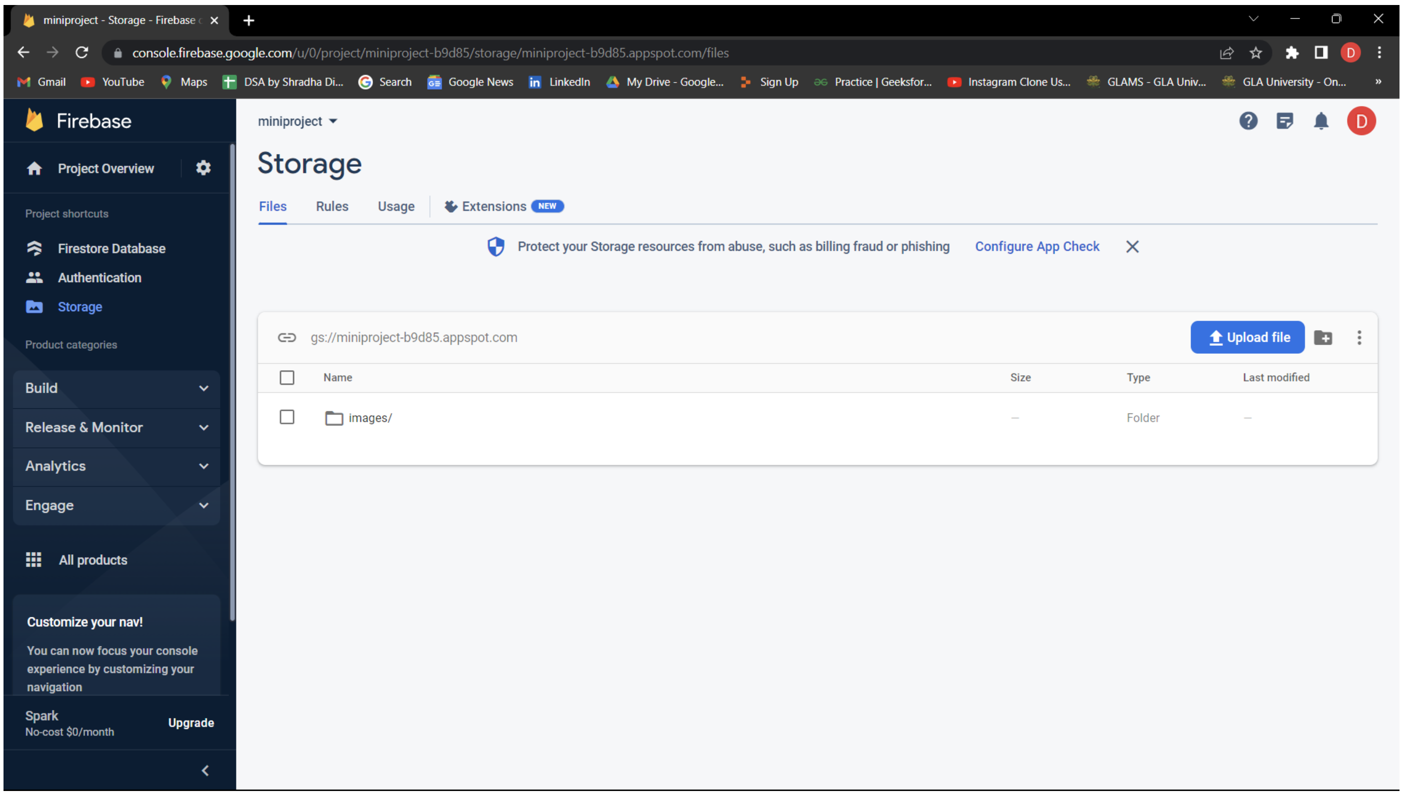
Longitude
0

Offer :
Yes [✓ No]

Regular Price :
0
$ / Month

select images :
Choose Files No file chosen

Create Listing



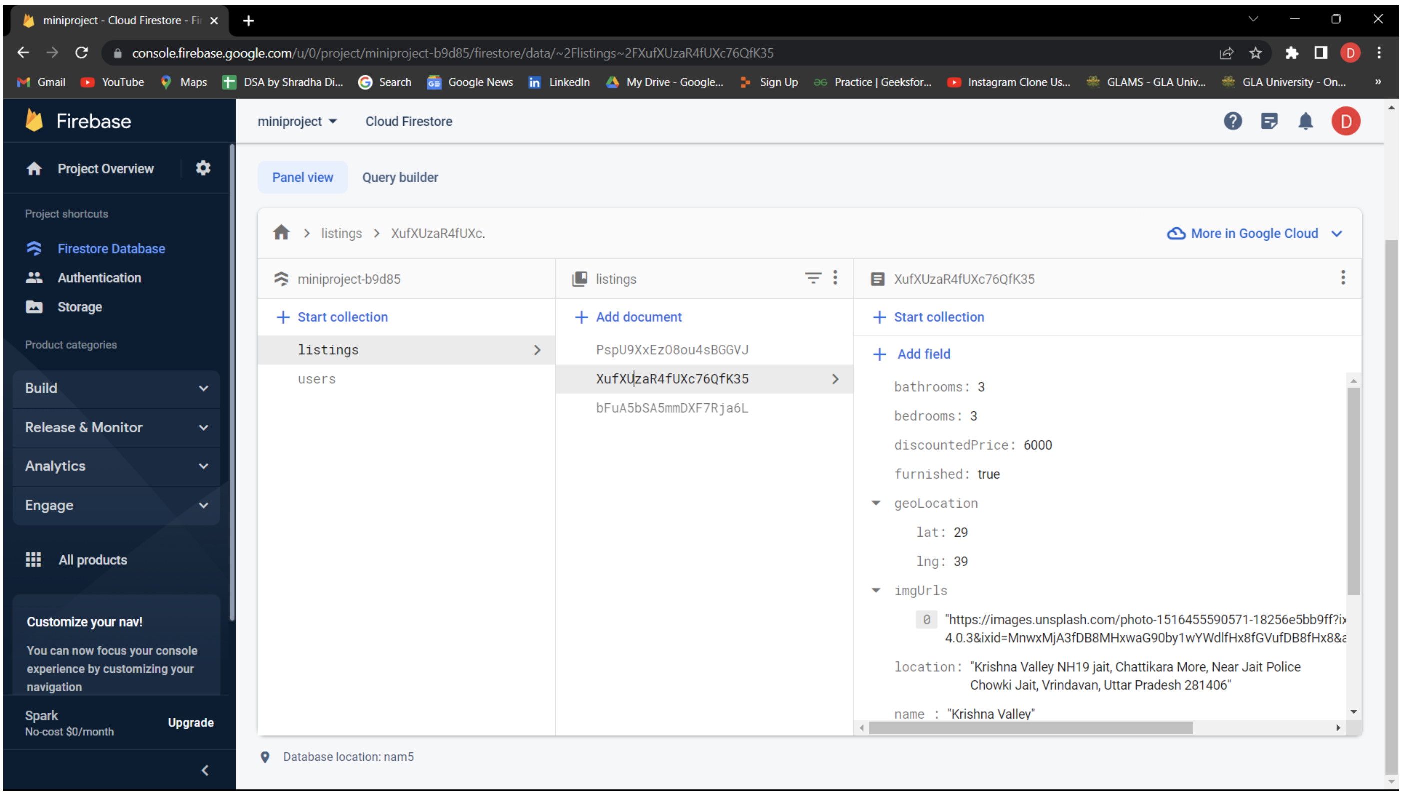
console.firebase.google.com/u/0/project/miniproject-b9d85/storage/miniproject-b9d85.appspot.com/files

Storage

Protect your Storage resources from abuse, such as billing fraud or phishing

gs://miniproject-b9d85.appspot.com

Name	Size	Type	Last modified
images/	—	Folder	—



```
listings
PspU9XxEz08ou4sBGGVJ
XufXUzaR4fUXc76QfK35
bFuA5bSA5mmDXF7Rja6L

XufXUzaR4fUXc76QfK35
bathrooms: 3
bedrooms: 3
discountedPrice: 6000
furnished: true
geoLocation
    lat: 29
    lng: 39
imgUrls
    0 "https://images.unsplash.com/photo-1516455590571-18256e5bb9ff?ix
      4.0.3&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&a
location: "Krishna Valley NH19 jait, Chattikara More, Near Jait Police
          Chowki Jait, Vrindavan, Uttar Pradesh 281406"
name : "Krishna Valley"
```

Github Link:

https://github.com/Dev4821/house-hive

Conclusion

The house selling and renting website will provide a user-friendly platform for property owners and real estate agents to advertise their properties for sale or rent, and for potential buyers or renters to search and view available properties. The website will be developed using modern web development technologies and will be designed to be responsive, mobile-friendly, and optimized for search engines. The project was developed using HTML, CSS, JavaScript, react ,and Node Js programming languages, resulting in a responsive and reliable website. The website has been tested extensively to ensure that it meets the needs of its users. Overall, the project was a valuable experience in website development and provided an opportunity to design and develop a platform that meets the needs of its users. With its user-friendly features, the website will help users find their dream properties easily and efficiently. The resulting website is a fully functional platform that allows users to buy, sell or rent properties online. The website's features include search filters for location, property type, price range, and other relevant information. Users can also create accounts to save their favourite listings, receive notifications on new properties, and communicate with property owners or agents. The payment gateway is secure and easy to use, allowing users to make transactions with confidence. The website's user-friendly interface makes it easy to navigate and find the information users need. The website has been tested extensively to ensure that it is responsive, fast, and reliable.