SE/WT PROJECT REPORT SUBMITTED
On

ORDER BY US

Project Report submitted in partial fulfilment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By:

Student Name: Padidala Santhosh Kumar

H.T.No.: B171777

Student Name: Kamidi Vijayasimha Reddy

H.T.No.: B171414

Under the guidance and supervision of

SE/WT Project Coordinators

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge and Technologies

Basar, Nirmal, Telangana, INDIA. Pin-code : 504107

Website: www.rgukt.ac.in, AY: 2021-2022

Rajiv Gandhi University of Knowledge and Technologies

Basar, Nirmal, Telangana, INDIA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the **"Project Report**" entitled **" ORDER BY US"** submitted by **SANTHOSH KUMAR PADIDALA, (H.T.No.: B171777)** is work done by him and submitted during 2021 – 2022 academic year, in partial fulfilment of the requirements for the award of the degree of **Bachelor Of Technology** in **Computer Science And Engineering** to **Rajiv Gandhi University Of Knowledge And Technologies, Basar** is a record of bonafide work carried out by them under my guidance and supervision from Jan-2022 to June-2022.

The results presented in this project have been verified and found to be satisfactory.

Signature of the SE/WT Project In-charge
Mr.K.RaviKanth
Asst.Prof., CSE,RGUKT

Signature of the SE/WT Project In-charge
Mr.B.Venkat Raman
Asst.Prof., CSE,RGUKT

Signature of the Head of Department
Ms.G.Srujana
Asst.Prof., CSE,RGUKT

Rajiv Gandhi University of Knowledge and Technologies

Basar, Nirmal, Telangana, INDIA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the **"Project Report**" entitled **" ORDER BY US"** submitted by **VIJAYASIMHA REDDY KAMIDI, (H.T.No.: B171414)** is work done by him and submitted during 2021 – 2022 academic year, in partial fulfilment of the requirements for the award of the degree of **Bachelor Of Technology** in **Computer Science And Engineering** to **Rajiv Gandhi University Of Knowledge And Technologies, Basar** is a record of bonafide work carried out by them under my guidance and supervision from Jan-2022 to June-2022.

The results presented in this project have been verified and found to be satisfactory.

Signature of the SE/WT Project In-charge
Mr.K.RaviKanth
Asst.Prof., CSE,RGUKT

Signature of the SE/WT Project In-charge
Mr.B.Venkat Raman
Asst.Prof., CSE,RGUKT

Signature of the Head of Department
Ms.G.Srujana
Asst.Prof., CSE,RGUKT

ACKNOWLEDGEMENT

First I Would like to thank management of RGUKT-Basar for giving me the opportunity to do this project work within the organisation.

I also would like to thank all the people who worked along with me RGUKT-Basar, with their patience and openness they created an enjoyable working environment.

It is indeed with a great sense of pleasure and immense of gratitude that I acknowledge the help of these individuals.

I am highly indebted to Vice-Chancellor and Administrative Officer, for the facilities provided to accomplish this project work.

I would like to thank my Head of the Department **,CSE** for her constructive criticism throughout my project work.

I would like to thank my supervisors **Mr.K.Ravikanth & Mr.B.Venkat Raman, Assistant Professors ,CSE** for their constructive criticism throughout my project work.

I would like to thank our department PRC Team Members, CSE for their support and advices to get and complete project within the given guidelines .

I am extremely great full to my department staff members, family members and friends who helped me in successful completion of this project work.

Santhosh Kumar Padidala, (B171777) **Signature**

Vijayasimha Reddy Kamidi, (B171414) **Signature**

ABSTRACT

We the students of Computer Science Engineering RGUKT Basar are developing a website named OrderByUs.For students in universities similar to RGUKT, who has no accessibility to an online food ordering facility, OrderByUs is a platform for those students. Where they can order food besides delivering food employing they can get benefits. So this is like a platform to facilitate students saving their time and get their food at their door step furthermore making students financially independent in college. Our task is to join the users who want food and who want money. Whoever wants food can place an order on our website, and agents who are willing to deliver can take up the order and deliver to the user receiving reasonable delivery charges. Provides a good and healthy interface between users who want food parcels at their doorsteps and users who are willing to deliver food to get money. Avoiding fraud agents, users, and data. OrderByUs is fraud-free from users, agents, and data. Any illegal activity can be cleared in a very short span of time. We exchange details of user and agent whenever any agent takes up the order placed by any user. So that they can have communication regarding their mutual benefits. As the details are exchanged there is no chance of deceptiveness. Order can be placed only at specific timings and orders which are not taken by any agent will not last after the end timings of the day. We finally come by want we expected, now we are ready with our website to help a student who is not having an online food ordering facility. This website will help many students who are studying in rural area colleges/universities. We are providing a platform to students without expecting any benefits, OrderByUs is a completely free platform, anyone can access it, and not only in colleges, we can implement this anywhere. All this site need is a user who wants things delivered at their doorstep and a delivery agent who is ready to deliver the things specified by the user. OrderByUs helped a lot in improving our business analysis skills. As the

OrderByUs project related to customer relationship management it made us think about every nook and corner in overcoming loopholes. This project gave us an eminent experience and furthermore dragged our curiosity on the road to business. We used HTML5, CSS3, Bootstrap5, JavaScript, Nextjs, and Firebase technologies to build this project.

Page left Intentionally..

TABLE OF CONTENTS

LIST OF FIGURES

Page left Intentionally..

CHAPTER-1

INTRODUCTION

1.1 MOTIVATION :

As we know that necessity is the mother of invention. For students like us

who study in rural areas, universities doesn't have an online food ordering facility.

To help those students we are developing this website.

1.2 and 1.3 PROBLEM DEFINITION and OBJECTIVE OF THE PROJECT:

Nowadays students in universities like RGUKT, it became a hectic task to get the food from campus food courts. As the food courts are usually far from the hostels, most people are not willing to walk up to the food courts and get their food. So we are developing a web page which helps people in delivering their food at their doorstep by the students themselves, whoever deliver the food get paid by the client. It will be like a part-time job for needy students, as well as helps the customers in saving their energy.

CHAPTER-2

SYSTEM ANALYSIS

2.1.EXISTING SYSTEM:

ZOMATO:

Zomato is an Indian multinational restaurant aggregator and food delivery company founded by Deepinder Goyal and Pankaj Chaddah in 2008. Zomato provides information, menus and user reviews of restaurants as well as food delivery options from partner restaurants in select cities.As of 2019, the service is available in 24 countries and more than 10,000 cities.

SWIGGY:

Swiggy is an Indian online food ordering and delivery platform. Founded in July 2014, Swiggy is based in Bangalore, and operates in 500 Indian cities, as of September 2021. Apart from food delivery, Swiggy also provides on-demand grocery deliveries under the name Instamart and an instant package delivery service called "Swiggy Genie". Swiggy is operated by Bundl Technologies Private Limited.

2.2.PROPOSED SYSTEM

ORDER BY US is an Indian online food ordering and delivery platform. Founded in June 2022, ORDER BY US is based in RGUKT Basar and operates in the RGUKT Basar campus. Order by us provides information on food menus in the campus food courts as well as food delivery options within the campus. Students who deliver food are called delivery agents. Order by us facilitates students with part-time job opportunities where students can get paid by the clients (who order food). This site is entirely free, only users get benefits from the site. ORDER BY US founded by Padidala Santhosh Kumar and Kamidi Vijayasimha reddy B17 batch of RGUKT Basar.

2.3.SOFTWARE REQUIREMENT SPECIFICATION

2.3.1 FUNCTIONAL REQUIREMENT

— CLIENT RELATED FUNCTIONAL REQUIREMENTS

SignUp :

Before placing the order client need accessing credentials for the website , so for the first time client need to signup by entering email id and password.After that he must confirm the mail , then the client account is created now client is well and good to proceed with the website functionalities.

Login :

After creating an account now client can access all the functions which are provided by order by us. He just need to login.Username and password will be required for logging in to the website. Now client can access website full wholly.

Active orders:

In active orders section clients can view all active orders, and clients can place order by clicking on plus symbol provided at the bottom of the page .

Ongoing orders :

After placing order in active section , clients waits for their order to be taken up by any of the agents.Soon after thier order is taken up by agent their order will move to ongoing order which is personal to clients.Clients who placed the order can see their respective ongoing orders.

Your orders :

All successful orders of clients will be shown to respective clients.This is personal to clients.

All orders :

In all orders section , all orders successfully delivered on that day will be displayed clients.

PLACE_ORDER :

In side bar when user clicks on active orders there he can see plus symbol ,clients can place orders by clicking that sysmbol.For placeing the order clients must enter their mobile number and list of items.

— AGETN RELATED FUNCTIONAL REQUIREMENTS

SignUp :

Before taking up the order agent need accessing credentials for the website , so for the first time agent need to signup by entering email id and password.After that he must confirm the mail , then the agent account is created now agent is well and good to proceed with the website functionalities.

Login :

After creating an account now agent can access all the functions which are provided by order by us. He just need to login.Username and password will be required for logging in to the website. Now agent can access website full entirely.

Active orders:

In active orders section agents can view all active orders,

and can take-up order by clicking take-up order.

Taken up orders :

Soon after the agent takes up an order from the active section that order

comes into agents personal data. where only agent can see the data and information

of agent will be sent to client soon after some agent takes up the clients order.

All orders :

In all orders section, all orders successfully delivered on that day

will be displayed to agents.

2.4.SOFTWARE TOOL USED

Operating System : Windows/ Linux / Mac

Next JS framework

HTML, CSS, Bootstrap

Node JS

Back-end : FIREBASE

2.5.HARDWARE USED

4 GB RAM

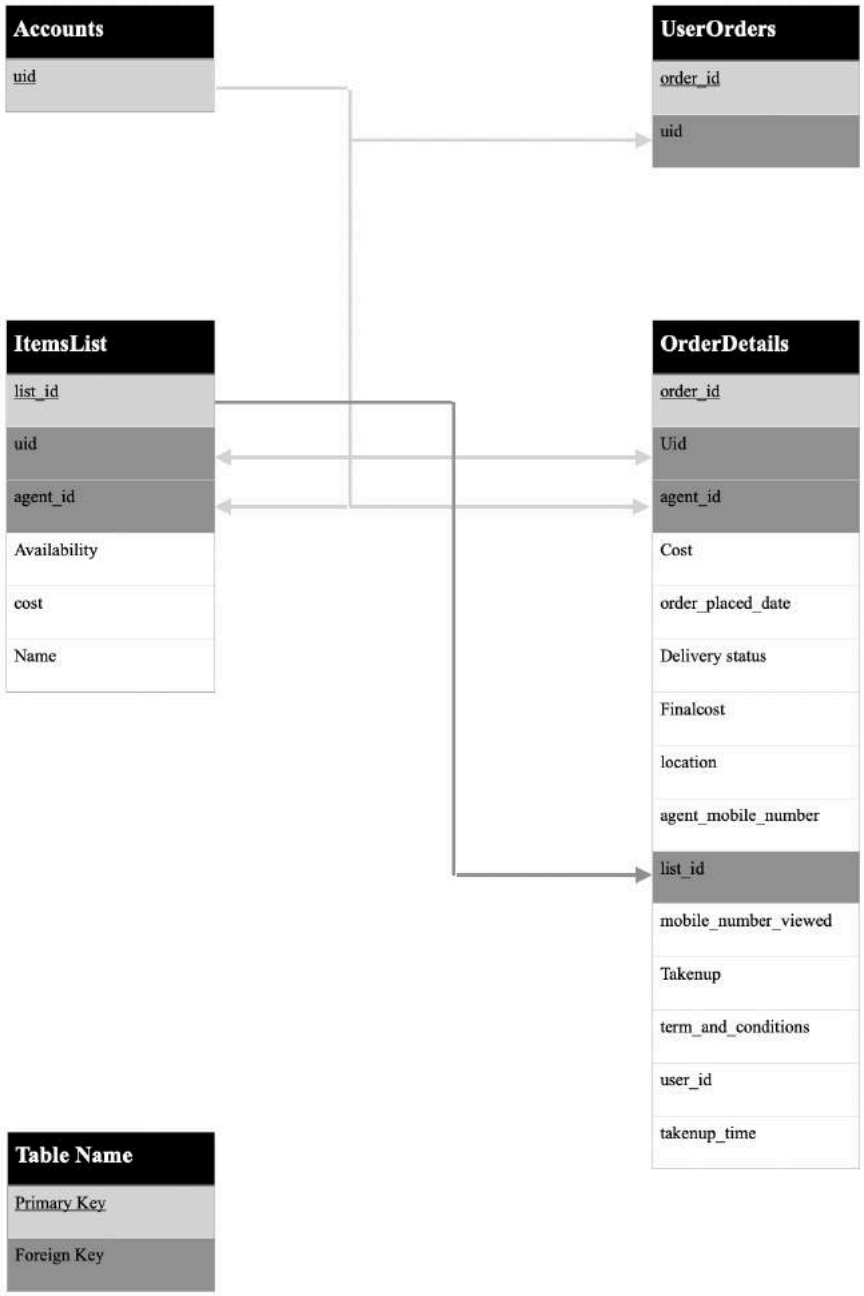
Intel Core i3 processor

CHAPTER-3

SYSTEM DESIGN

3.1.TABLE DESIGN

Database schema for OrderByUs website:



Tables :

Fig:3.1.1 Database schema

Database Tables and type

TABLE ACCOUNTS

Column	Null?	Type
U_ID	NOT NULL	VARCHAR2(50)

TABLE USERORDERS

Column	Null?	Type
ORDER_ID	NOT NULL	VARCHAR2(50)
U_ID	-	VARCHAR2(50)

TABLE ITEMSLIST

Column	Null?	Type
LIST_ID	NOT NULL	VARCHAR2(50)
U_ID	-	VARCHAR2(50)
AGENT_ID	-	VARCHAR2(50)
AVAILABILITY_	-	NUMBER(1,0)
COST_	-	NUMBER(10,0)
NAME_	-	VARCHAR2(50)

TABLE ORDERDETAILS

Column	Null?	Type
ORDER_ID	NOT NULL	VARCHAR2(50)
U_ID	-	VARCHAR2(50)
AGENT_ID	-	VARCHAR2(50)
COST_	-	NUMBER(10,0)
PLACE_DATE	-	VARCHAR2(6)
DELIVERY_STATUS	-	NUMBER(1,0)
FINALCOST	-	NUMBER(10,0)
LOCATION_	-	VARCHAR2(50)
AGENT_MOBILE	-	NUMBER(10,0)
LIST_ID	-	VARCHAR2(50)
VIEWED	-	NUMBER(1,0)
TAKENUP	-	NUMBER(1,0)
TERMS_AND_CONDITIONS	-	NUMBER(1,0)
TAKENUP_TIME	-	VARCHAR2(10)
USER_ID	-	VARCHAR2(50)

Fig:3.1.2 Database Tables

3.2.DATA FLOW DIAGRAMS

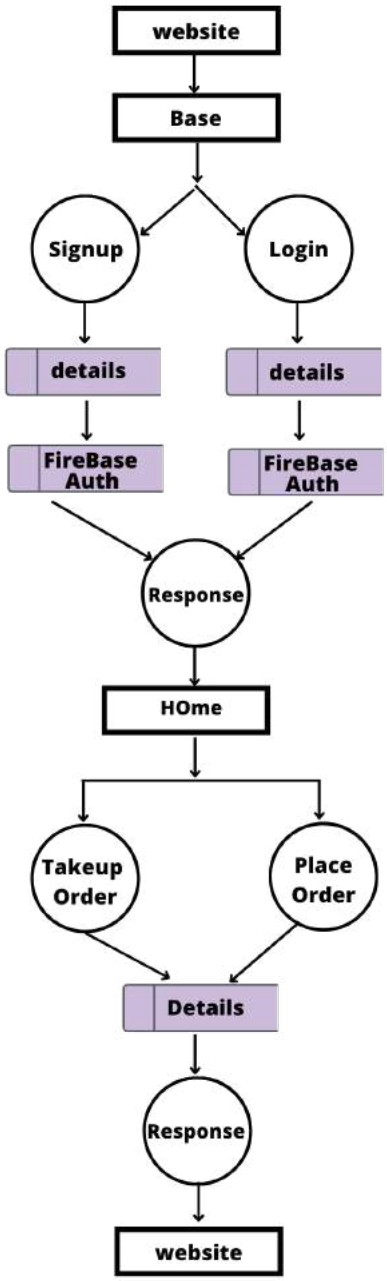


Fig:3.2.1 DataFlow Diagram

3.3.UML DIAGRAMS

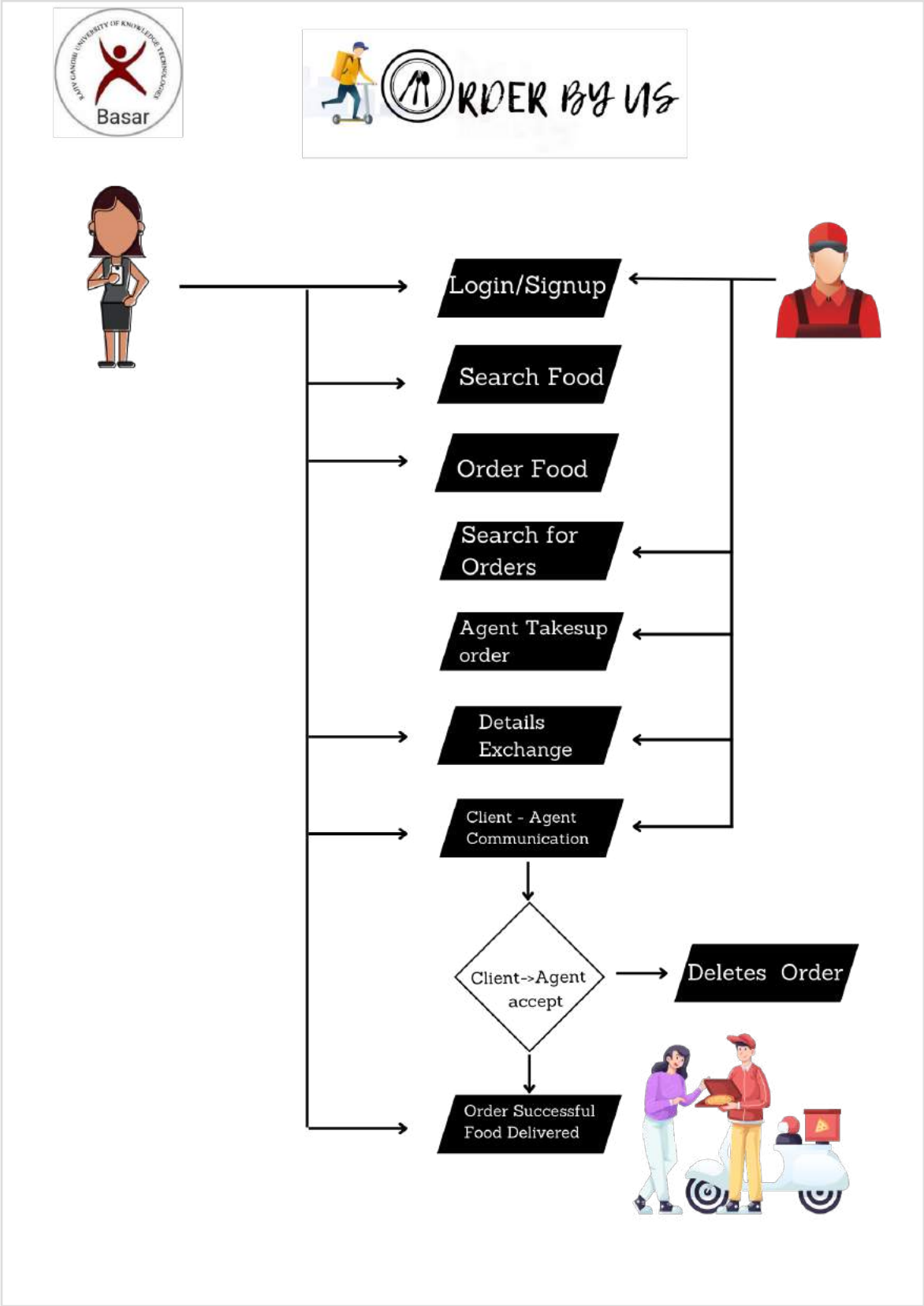


Fig:3.3.1 UML Diagram

CHAPTER-4

SYSTEM IMPLEMENTATION

4.1.INTRODUCTION TO TECHNOLOGIES USED

HTML5 :

HTML stands for Hyper Text Markup Language. It is used to design web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages.

- **Web development**. Developers use HTML code to design how a browser displays web page elements, such as text, hyperlinks, and media files.
- **Internet navigation**. Users can easily navigate and insert links between related pages and websites as HTML is heavily used to embed hyperlinks.
- **Web documentation**. HTML makes it possible to organize and format documents, similarly to Microsoft Word.

CSS3 :

Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page. It describes how a webpage should look: it prescribes colors, fonts, spacing, and much more. In short, you can make your website look however you want. CSS lets developers and designers define how it behaves, including how elements are positioned in the browser.

BOOTSTRAP:

Bootstrap, the world's most popular framework for building responsive, mobile-first sites, with jsDelivr and a template starter page.

Bootstrap is developed mobile first, a strategy in which we optimize code for mobile devices first and then scale up components as necessary using

CSS media queries. To ensure proper rendering and touch zooming for all devices

JAVASCRIPT :

JavaScript is a very powerful **client-side scripting language**. JavaScript is used mainly for enhancing the interaction of a user with the webpage. In other words, you can make your webpage more lively and interactive, with the help of JavaScript. JavaScript is also being used widely in game development and Mobile application development.

NEXTJS : Next.js is based on react, webpack and babel. It is an awesome tool for creating web application and famous for server-side rendering. Next.js is build by Zeit.

NextJs :

The Next.js is React Based framework with server side rendering capability. It is very fast and SEO friendly. Using Next.js, you can create robust react based application quite easily and test them.

FIREBASE :

Firebase is a product of Google which helps developers to build, manage, and grow their apps easily. It helps developers to build their apps faster and in a more secure way. No programming is required on the firebase side which makes it easy to use its features more efficiently. It provides services to android, ios, web, and unity. It provides cloud storage. It uses NoSQL for the database for the storage of data.

4.2.MODULE DESCRIPTION

— NAVBAR :

What is Navbar in HTML?

A navigation bar (also called a Navbar) is a user interface element within a webpage that contains links to other sections of the website. In most cases.

Order by us navbar contais of 6 links

- Home
- About
- Team
- Contact
- Login
- Signup

— SignUp -Login :

- SignUp:

Firstly user need to create an account by using email id and password in Signup page user can create an account , where the page will be asking for a valid email , password and cornfirming the password next to that user have to verify the email.Thats it an accout will be created and based on the credentials.

- Login:

After creating an account user now can access all the functions which are provided by order by us. He just need to login.Username and password will be required for logging in to the website.

— SIDE BAR:

Side bar consists of Active orders , Taken Up orders , Ongoing orders , Your orders and All orders

- Active orders :

In active orders section you can see all active or current orders and also

user can place order by clicking on plus symbol provided at the bottom.

Every one can view this section that is both clients (who want to place orders)

and agents (who are willing to deliver food).

- Taken up orders :

Taken up orders is personal to agents (who are willing to deliver food) , soon after the agent takes up an order from the active section that order comes into agents personal data. where only agent can see the data and information of agent will be sent to client soon after some agent takes up the clients order.

- Ongoing orders :

Afterplacing order in active section , clients waits for their order to be taken up by any of the agents.Soon after thier order is taken up by agent their order will move to ongoing order which is personal to clients.Clients who placed the order can see their respective ongoing orders.

- Your orders :

All successfull orders of clients will be shown to respective clients.This is personal to clients.

- All orders :

In all orders section , all orders successfully delivered on that day will be displayed.

— PLACE_ORDER :

In side bar when user clicks on active orders there he can see plus symbol , clients can place orders by clicking that sysmbol.For placeing the order clients must enter their mobile number and list of items.

4.3.SAMPLE CODE

```
activeOrders.js ×
pages > activeOrders.js > ActiveOrders > closeSheet
140     // setLoader(false);
141   }, []);
142
143   const openSheet = () => {
144     document.getElementById("popup").style.cssText = `visibility:visible;`;
145     // document.getElementById("list").style.cssText = `filter:blur(10px);`;
146     // alert("hii");
147   };
148   const closeSheet = () => {
149     document.getElementById("popup").style.cssText = `visibility:hidden;`;
150     // document.getElementById("list").style.cssText = `filter:blur(0px);`;
151
152   };
153
154   return (
155
156     <Layout loginn="true" >
157       <div>Active orders</div>
158
159       {loader?<div className={styles.loader}>
160        Fetching data...
161       </div>:<>
162       <div className={styles.list} id="list">
163
164
165           {list.length != 0 ?list.map(
166         ({
167           index,
168           order_id,
169           location,
170           user_id,
171           finalCost,
172           orderList,
173           takenUp,
174           date,
175           time,
176           uid,
177           delivered
```

master* ⊗ 0 △ 0 Live Share 12 selections Spaces: 2 UTF-8 LF {} JavaScript Prettier

Fig:4.3.1 Active Orders code

```
allOrders.js
pages > allOrders.js > AllOrders > useEffect() callback > fetchData
31          setLoader(true);
32          ll = true;
33          const fetchData = async () => {
34              const accessToken = userAccessToken();
35              if (!accessToken)
36              {
37                  alert("login to enter orderspage")
38                  return router.push("/home");
39              }
40              const [userInfo] = fetchUser();
41              const uidd = userInfo?.uid;
42
43              setUser(userInfo);
44
45              try {
46                  var l = [];
47                  const querySnapshot = await getDocs(collection(database, "database"));
48
49                  if (querySnapshot.size)
50                      querySnapshot.forEach(async (doc) => {
51                          l.push(doc.id);
52                          // doc.data() is never undefined for query doc snapshots
53                          // var q = await getDocs(collection(database,doc.id));
54                          //
55
56
57
58                      });
59
60                  //
61                  l.map(async (uid) => {
62                      const qSS = await getDocs(collection(database, uid));
63                      if (qSS.size) {
64                          qSS.forEach(async (d) => {
65
66                              const docRef = doc(database, uid, d.id);
67                              const docSnap = await getDoc(docRef);
68                              if (docSnap.exists()) {
```

master* ⊗ 0 ⚠ 0 Live Share 14 selections Spaces: 4 UTF-8 LF {} JavaScript Prettier

Fig: :4.3.2 All Orders code

```
import { React, useState, useEffect } from "react";
import { userAccessToken, fetchUser } from "../utils/fetchDetails";
import NavBar from "../design/NavBar";
import Body from "../design/Body";
import { useRouter } from "next/router";
import HOME from "../design/HOME";
import Image from "next/image";
import loaderGif from "../public/images/loader.gif";
import styles from "../design/HOME.module.css"

const Home = () => {
  const router = useRouter();
  const [user, setUser] = useState(null);
  const [loader, setLoader] = useState(false);

  useEffect(() => {
    const accessToken = userAccessToken();
    if (accessToken) return router.push("/activeOrders");
    setLoader(true);
    setInterval(change, 4000);


    function change() {

      location.href = "/home.html";
      setLoader(false);
      clearInterval(change);

    }



  }, []);
```

Fig: :4.3.3 Home.js code

```
index.js
pages > index.js > Index > useEffect() callback > fetchData
12  const index = () => {
13    //login Route
14
15    const router = useRouter();
16    const [user, setUser] = useState(null);
17    useEffect( () => {
18      const fetchData = async()=>{
19        const accessToken = userAccessToken();
20        if (!accessToken) return router.push("/home");
21        if (accessToken) return router.push("/activeOrders");
22        const [userInfo] = fetchUser();
23
24        setUser(userInfo);
25      }
26      fetchData();
27
28      //   //getting database values
29    }, []);
30
31    const logout = () => {
32      localStorage.clear();
33      router.push("/login");
34      console.clear();
35    };
36
37    return (
38      <>
39        {/* <div>
40          <WriteToCloudFirestore />
41        </div> */}
42
43        {/* <div className="w-screen h-screen flex justify-center items-center bg-slate-100">
44          <div className="w-1/3 h-auto p-4 bg-white shadow-md rounded-md flex justify-start items-center relative">
45            <IoLogOut
46              fontSize={25}
47              className="absolute top-3 right-3 cursor-pointer text-gray-600"
48              onClick={logout}
49            />
master*  0 0  Live Share   Ln 23, Col 7  Spaces: 2  UTF-8  LF  {} JavaScript  Prettier
```

Fig:4.3.2.4 Index code

```
const [e1, sete1] = useState('password');
const [e2, sete2] = useState("password");
const [page, setPage] = useState("");

const [email, setEmail] = useState("");
const [password, setPassword] = useState("");
const [confirmPassword, setConfirmPassword] = useState("");
//for sign in page
const [emailS, setEmailS] = useState("");
const [passwordS, setPasswordS] = useState("");
const [loading, setLoading] = useState(false);

useEffect(() => {
    //

    const p = router.asPath.substring(9, router.asPath.length)
    //
    if (p) {
        setPage(p);
    }
    else {
        setPage("register");
    }

    const accessToken = userAccessToken();
    if (!accessToken) return router.push("/login");
    else return router.push("/activeOrders");
    const [userInfo] = fetchUser();

}, [])



const f2 = () => {
    setPage("signin");
```

Fig:4.3.2.5 Login code

```
ongoingOrders.js ×
pages > ongoingOrders.js > OngoingOrders > useEffect() callback > showDiv1 > change
14      getDoc,
15      getDocs,
16      collection,
17  } from "firebase/firestore";
18  import { database } from "../firebase-config";
19  function OngoingOrders() {
20      const [loader, setLoader] = useState(false);
21      const [list, setlist] = useState([]);
22
23      const router = useRouter();
24
25
26      const [user, setUser] = useState(null);
27      var ll = false;
28      useEffect( () => {
29          document.getElementById("network").style.visibility = "hidden";
30
31          function showDiv1() {
32              if (ll == true) {
33                  document.getElementById("network").style.visibility = "visible";
34
35                  function change() {
36                      if (ll == false) {
37                          document.getElementById("network").style.visibility = "hidden";
38
39                          clearInterval(inst);
40                      }
41                  }
42                  var inst = setInterval(change, 1000);
43              }
44
45
46
47          }
48          setTimeout(showDiv1, 15000);
49          setLoader(true);
50          ll = true;
master*  ⊗ 0 ⚠ 0  Live Share   13 selections  Spaces: 4  UTF-8  LF  {} JavaScript  Prettier
```

Fig:4.3.2.6 Ongoing Orders code

```
takenUpOrders.js ×
pages > takenUpOrders.js > TakenUpOrders > useEffect() callback > fetchData
//      location.reload();
// }, 5000);
const [user, setUser] = useState(null);
  var ll = false;
useEffect(() => {
    document.getElementById("network").style.visibility = "hidden";

    function showDiv1() {
        if (ll == true) {
            document.getElementById("network").style.visibility = "visible";

            function change() {
                if (ll == false) {
                    document.getElementById("network").style.visibility = "hidden";

                    clearInterval(inst);
                }
            }
            var inst = setInterval(change, 1000);
        }



    }
    setTimeout(showDiv1, 15000);
    setLoader(true);
    ll = true;
    const fetchData = async ()=>{
      const accessToken = userAccessToken();
        if (!accessToken) {
            alert("login to enter orderspage")
            return router.push("/home");
        }
      const [userInfo] = fetchUser();
      const uidd = userInfo?.uid;

      setUser(userInfo);
master*  ⊗ 0 △ 0  Live Share   17 selections  Spaces: 4  UTF-8  LF  {} JavaScript  Prettier
```

Fig:4.3.2.7 TakenUp Orders code

```
const [user, setUser] = useState(null);
var ll = false;
useEffect(() => {
    document.getElementById("network").style.visibility = "hidden";

    function showDiv1() {
        if (ll == true) {
            document.getElementById("network").style.visibility = "visible";

            function change() {
                if (ll == false) {
                    document.getElementById("network").style.visibility = "hidden";

                    clearInterval(inst);
                }
            }
            var inst = setInterval(change, 1000);
        }



    }
    setTimeout(showDiv1, 15000);
    setLoader(true);
    ll = true;
    const fetchData = async()=>{
        const accessToken = userAccessToken();
        if (!accessToken) {
            alert("login to enter orderspage")
            return router.push("/home");
        }
        const [userInfo] = fetchUser();
        const uidd = userInfo?.uid;

        setUser(userInfo);

        try {
```

Fig:4.3.2.8 Your Orders code

```javascript
import React from 'react'
import styles from "../design/mail.module.css";
import { useState } from 'react';

function Mail() {
    const [name, setName] = useState('')
    const [email, setEmail] = useState('')
    const [message, setMessage] = useState('')
    const [submitted, setSubmitted] = useState(false)
    const handleSubmit = (e) => {
        e.preventDefault()

        let data = {
            name,
            email,
            message
        }
        fetch('/api/contact', {
            method: 'POST',
            headers: {
                'Accept': 'application/json, text/plain, */*',
                'Content-Type': 'application/json'
            },
            body: JSON.stringify(data)
        }).then((res) => {

            if (res.status === 200) {

                setSubmitted(true)
                setName('')
                setEmail('')
                setBody('')
            }
        })
    }

 return (
```

Fig:4.3.2.9 Email code

```
Tile.js
design > Tile.js > Tile > fun
74      }
75
76
77
78
79    }
80
81    const sendmail = ()=>{
82      let name = props.orderid;
83      let email = props.uid;
84      let message = `Your Order ${props.orderid} of cost ₹${cost} has been taken up by ${user?.uid}.
85            click on orderbyus.herokuapp.com/ongoingOrders for more details
86
87            Thank You..
88            `
89      let data = {
90        name,
91        email,
92        message
93      }
94      fetch('/api/contact', {
95        method: 'POST',
96        headers: {
97          'Accept': 'application/json, text/plain, */*',
98          'Content-Type': 'application/json'
99        },
100       body: JSON.stringify(data)
101     }).then((res) => {
102
103     })
104   }
105
106   let t = 0;
107   const fun = () => {
108     if (t % 2 == 0) {
109       document.getElementById("toggle" + props.orderid).style.cssText =
110         " transform: rotate(-180deg) ;transition:0.5s ";
master*  ⊗ 0 ⚠ 0  Live Share   34 selections  Spaces: 2  UTF-8  LF  {} JavaScript  Prettier
```

Fig:4.3.2.10 Sending Email code

```
const [user, setUser] = useState(null);
const [disable, setDisable] = useState(true);
const [verified, setVerified] = useState(false);
const disableControl = (e) => {
  if (disable) {
    setDisable(false);
  }
  else {
    setExpandForm(true);
    requestOTP(e);
  }
}
const countryCode = "+91";
const [OTP, setOTP] = useState('');

const generateRecaptcha = () => {
  window.recaptchaVerifier = new RecaptchaVerifier('recaptcha-container', {
    'size': 'invisible',
    'callback': (response) => {
      // reCAPTCHA solved, allow signInWithPhoneNumber.
      // onSignInSubmit();

    }
  }, authentication);
}



const requestOTP = (e) => {
  let phoneNumber = number;
  if(phoneNumber.length == 10){
    phoneNumber = "+91"+ phoneNumber;
  }

  e.preventDefault();
```

Fig:4.3.2.11 Generating OTP code

```
Tile.js
design > Tile.js > Tile > fun
311   return (
312     <div className={styles.card2} id={"card" + props.orderid}>
313       <div className={styles.content} onClick={fun}>
314         <div className={styles.details}>
315           <div id={styles.table}>
316             <div>
317               <p>Order_ID</p>
318               <p style={{ "fontSize": "15px" }}> Location</p>
319               <p >User_ID</p>
320             </div>
321             <div className={styles.values}>
322               <p>{props.orderid}</p>
323               <p style={{ "fontSize":"15px" }}>{props.location}</p>
324               <p >{props.uid == user?.uid ? "You" : props.user_id}</p>
325             </div>
326           </div>
327         </div>
328         <div className={styles.right}>
329           <div className={styles.hovercost} >{"₹" + props.cost}</div>
330           <div>{props.date}</div>
331           <div>{props.time}</div>
332         </div>
333
334       </div>
335       <div className={styles.line}></div>
336       <div className={styles.dropdown}>
337         <div className={styles.bottom}>
338           <div>
339             <p>Item_name</p>
340             {props.orderlist.map(({ name }) => (
341               <p className={styles.p1}>{name}</p>
342             ))}
343           </div>
344           <div className={styles.p2}>
345             <p>Cost</p>
346             {props.orderlist.map(({ cost }) => (
347               <p className={styles.p1}>{"₹" + cost}</p>
```

Fig:4.3.2.12 Generating Order Tiles code

```
var md5 = require("md5");

//sending data to firebase
try {
  // await setDoc(doc(database,"database", uid), {
  //     uid:uid,
  //     number:null,
  // });
  await setDoc(doc(database, uid, order_id), {
    order_id: order_id,
    location: location,
    agent_id: agent_id,
    user_id: user_id,
    finalCost: finalCost,
    orderList: orderList,
    date: date,
    time: time,
    cost: 0,
    takenUp: takenUp,
    delivered: delivered,
    uid: uid,
    mobile: number,
    paid: false,
    terms_and_conditions: false,
    agent_accept: false,
  });

  // await updateDoc(doc(database,title,"L"),{
  //     name:"Hell"
  // })
  // await deleteDoc(doc(database,title,"L"),{

  // })
  props.props();
  alert("success");
  router.reload(window.location.pathname);
```

Fig:4.3.2.13 PopUp order code

contact.js

pages > api > contact.js > default > mailData

```javascript
export default function (req, res) {


    let nodemailer = require('nodemailer')


    const transporter = nodemailer.createTransport({
        service: 'gmail',
        port: 465,
        secure: true,
        auth: {
            user: 'b[redacted]@rgukt.ac.in',
            pass: "[redacted]",
        },
    });
    const mailData = {
        from: 'b171777@rgukt.ac.in',
        to: req.body.email,
        subject: `Your Order ${req.body.name} has been takenUp`,
        text: req.body.message + " | Sent from: " + "orderbyus.herokuapp.com",

        html: `<div>${req.body.message}</div><p>Sent from:
    "orderbyus.herokuapp.com"</p>`
    }

    transporter.sendMail(mailData, function (err, info) {
        if (err)
            console.log(err, "here");
        else
            console.log(info, "yes")
    })
    res.status(200)
}
```

master* ⊗ 0 △ 0 Live Share Ln 16, Col 23 Spaces: 4 UTF-8 LF {} JavaScript Prettier

Fig:4.3.2.14 API calls

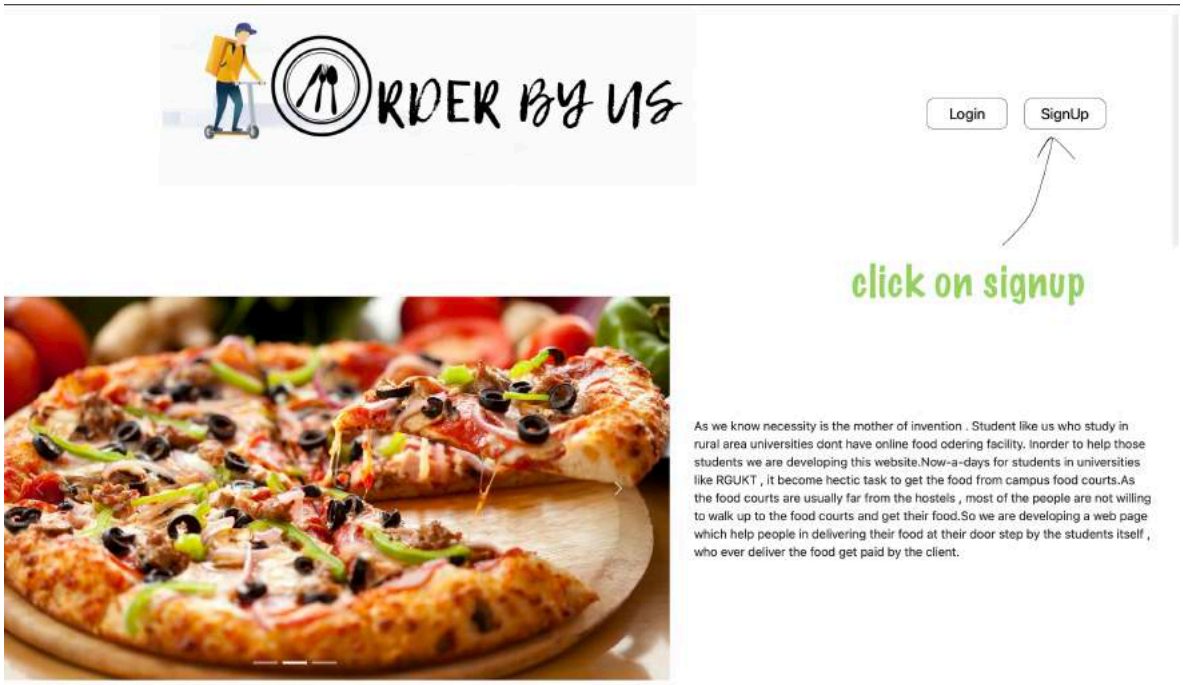
4.4.SCREEN SHOTS

User documentation :

— For both agent and client:

Navigate to https://orderbyus.herokuapp.com/

Step - 1:



Click on Login/ Signup

Fig:4.4.1 Home Page

Step - 2 :

Enter your email and create a password.

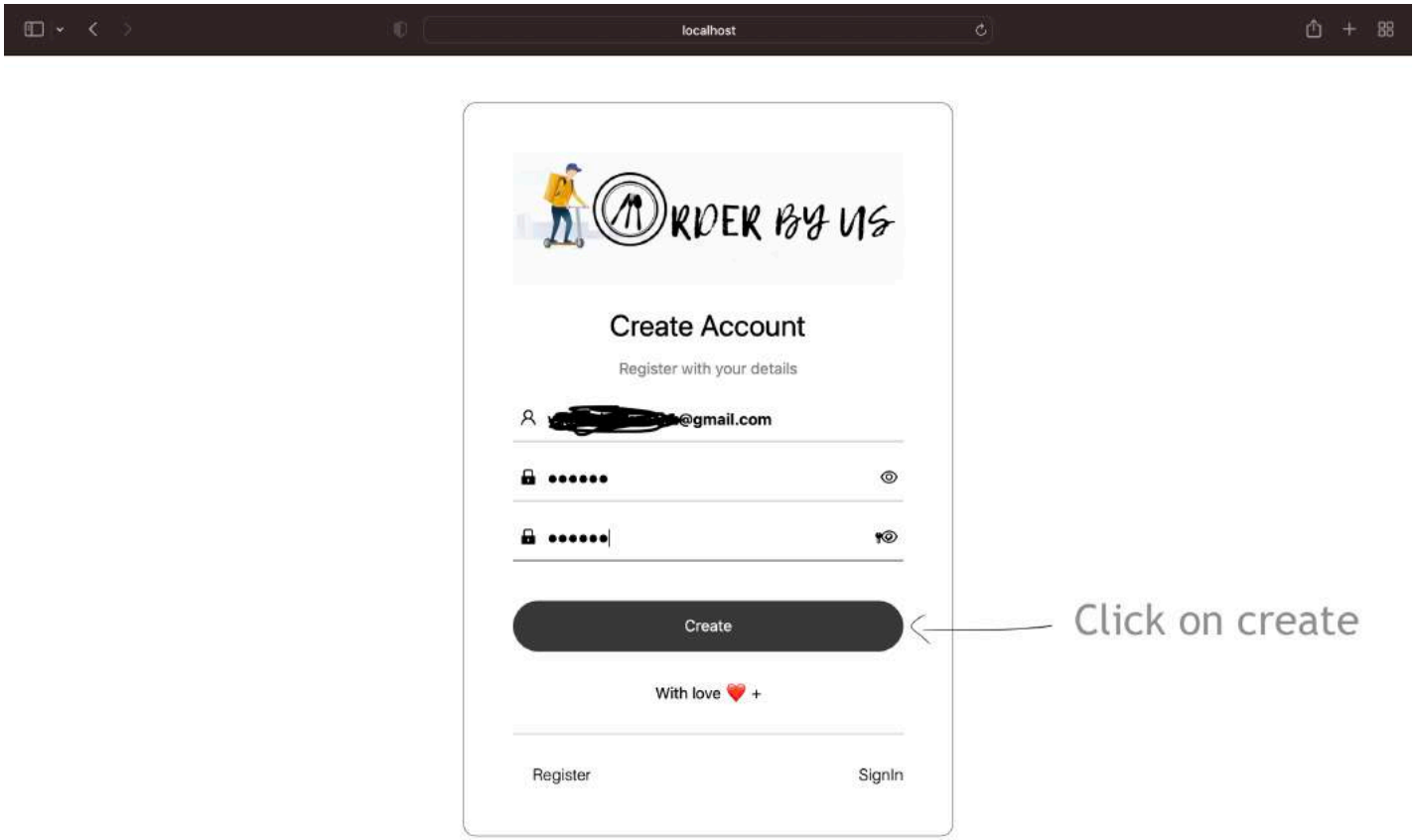


Fig:4.4.2 SignUp Page

To login to your account you need to verify your email

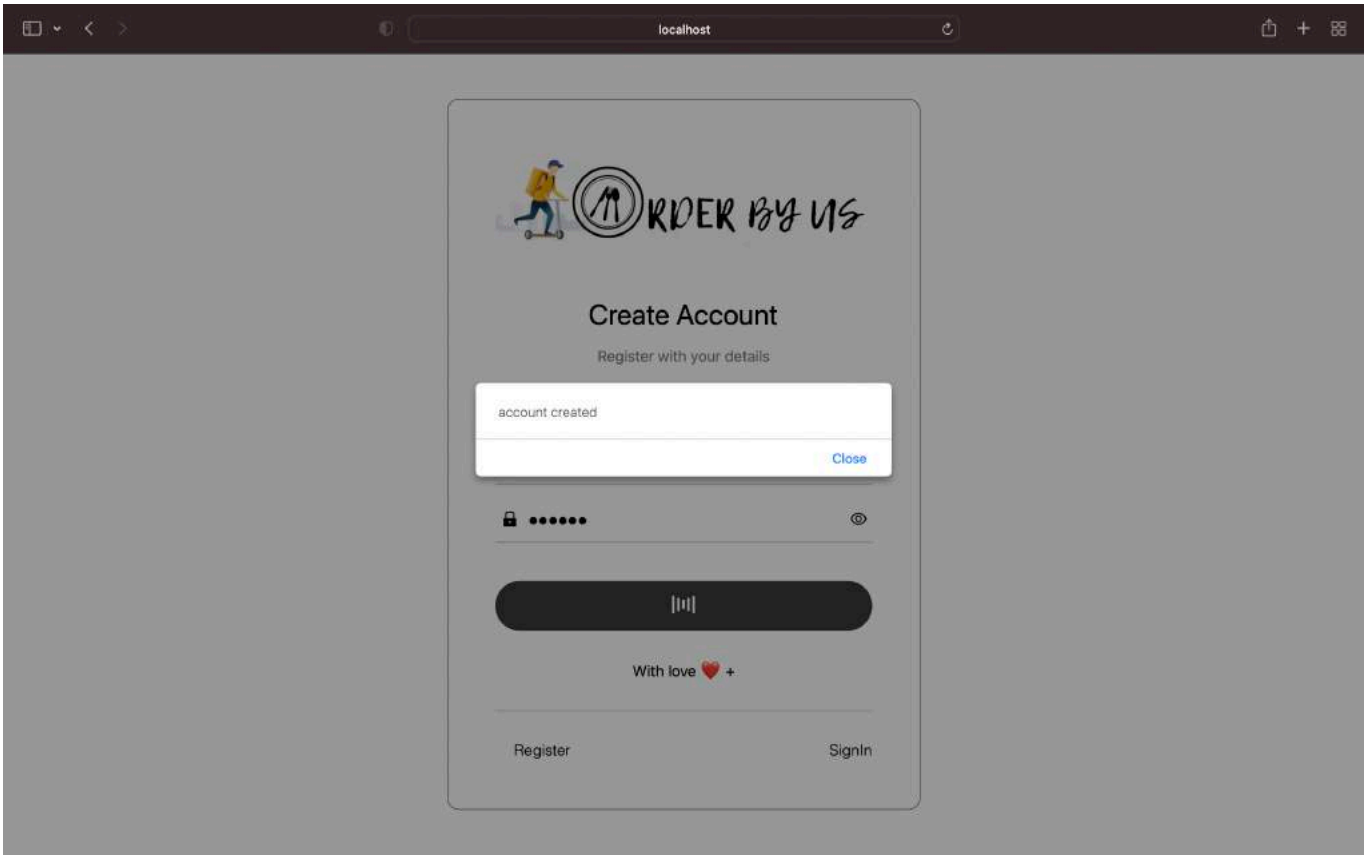


Fig:4.4.2.1 Verification email sent page

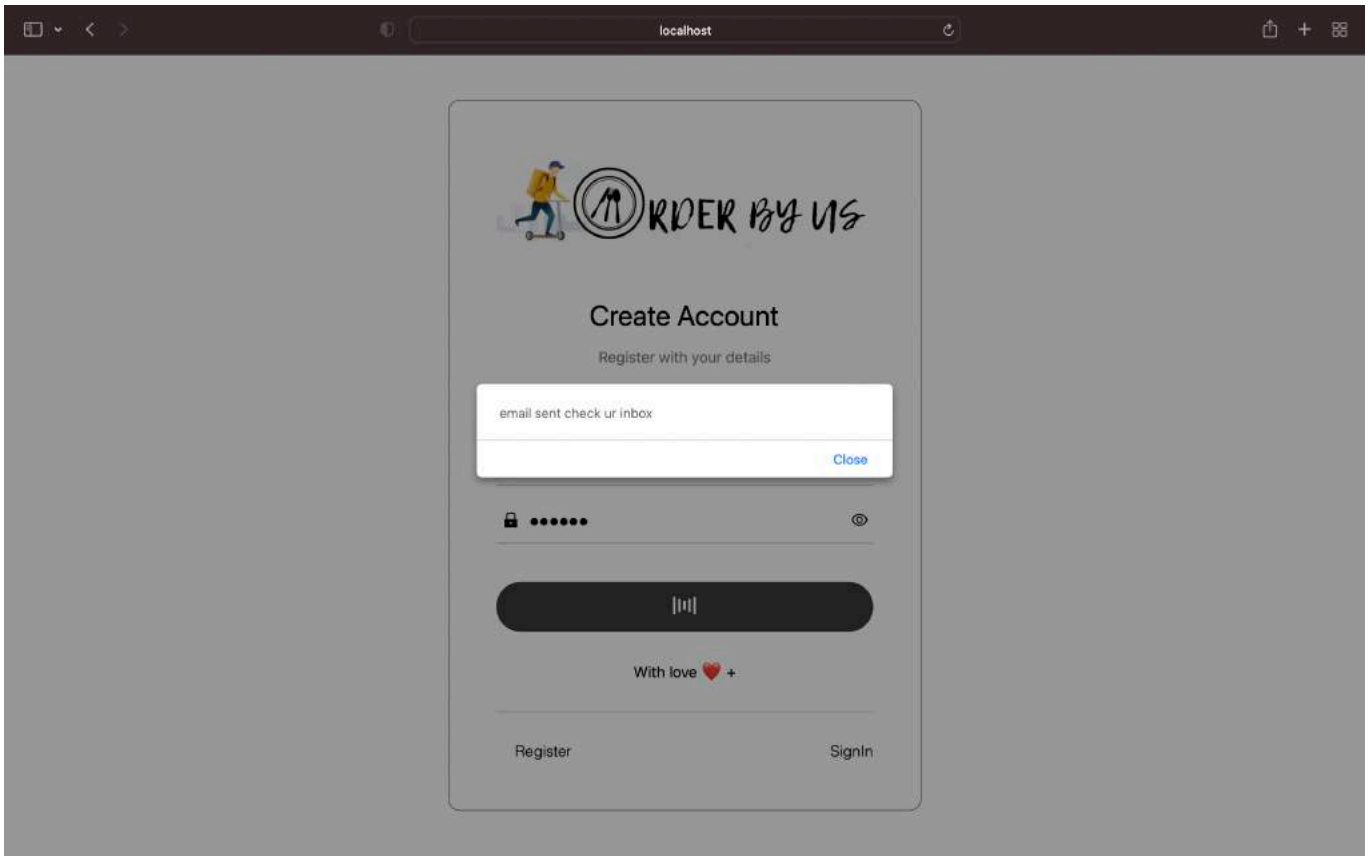
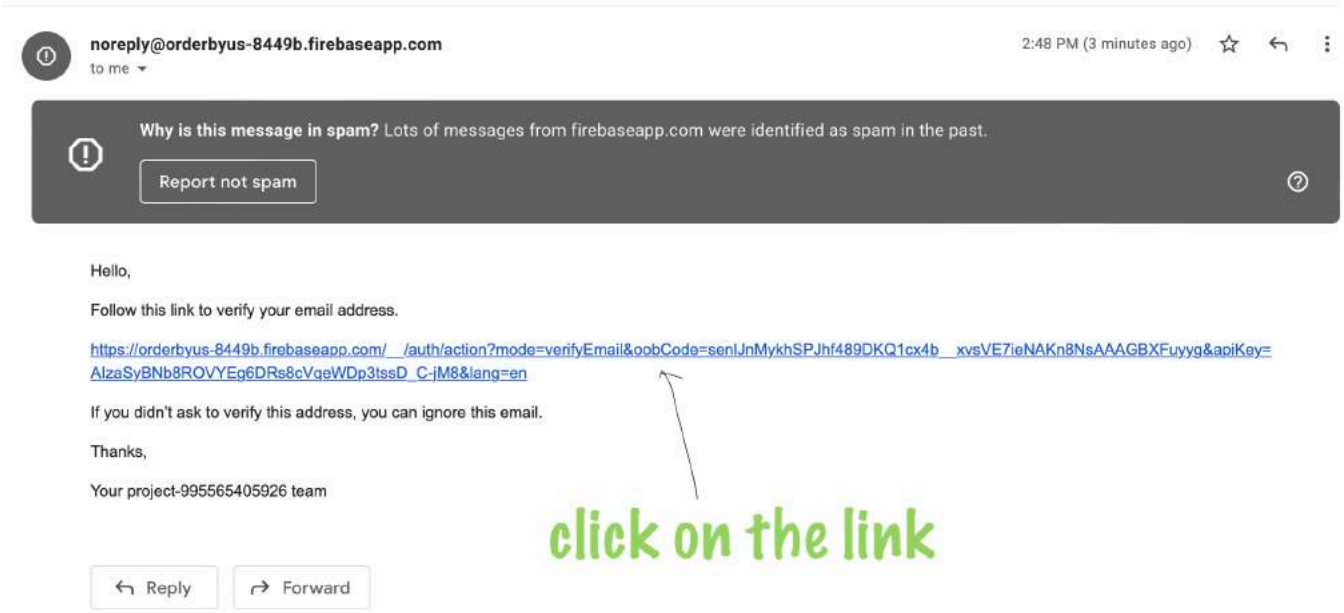


Fig:4.4.2.2 Account Created page

Step - 3:

Check in spam folder and click on the link.

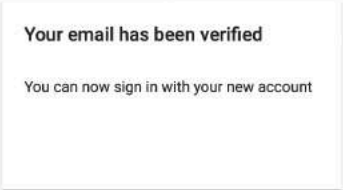
Fig:4.4.3 Email verification Link



After clicking the link …

Step-4 :

Go back to login page and login with your credentials



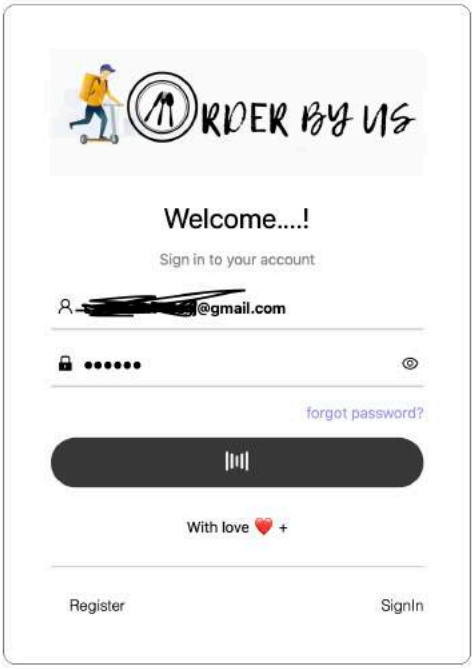


Fig:4.4.4 Login Page

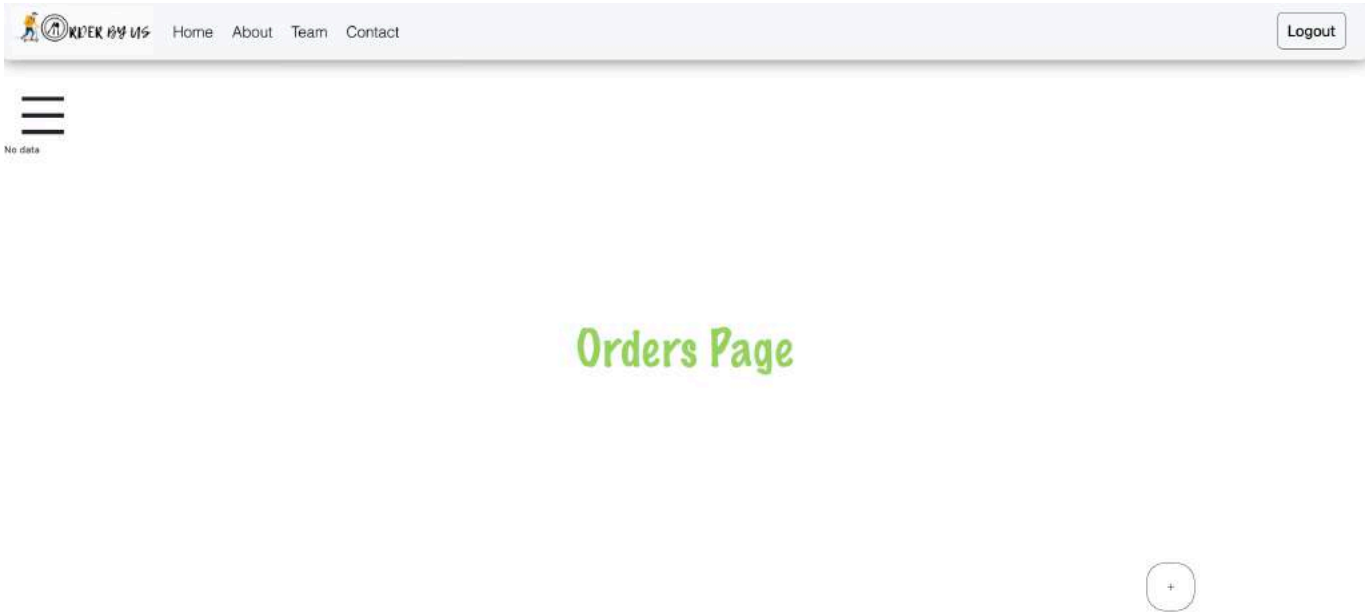


Fig:4.4.5 Active Orders Page

Client :

Step-5 :

If you want to place order

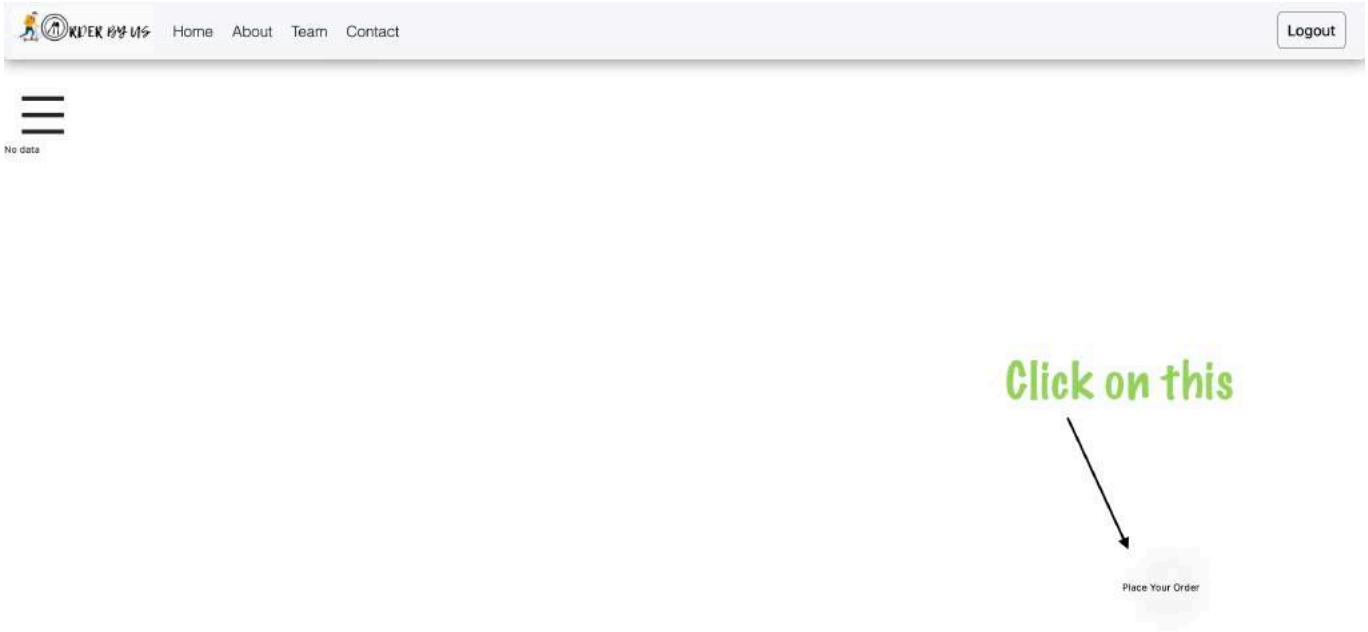


Fig:4.4.6 PlaceOrder Page

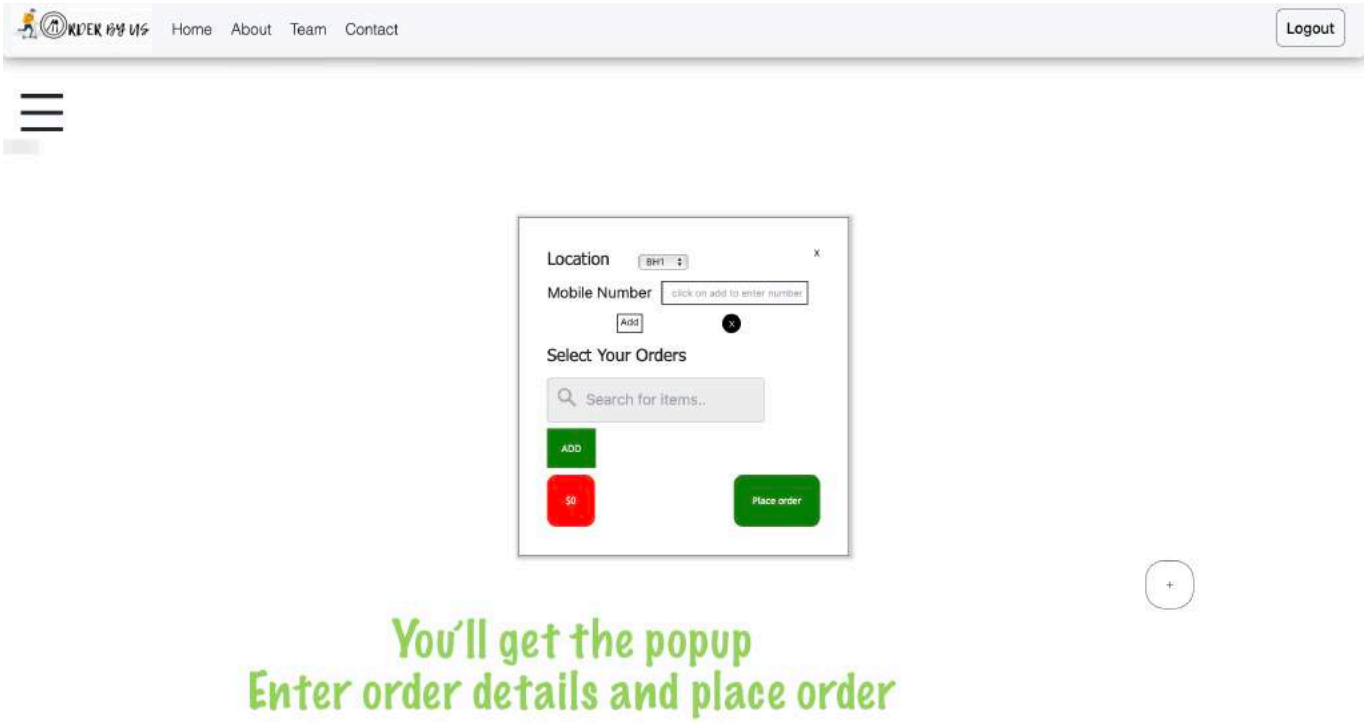


Fig:4.4.7 Popup Page

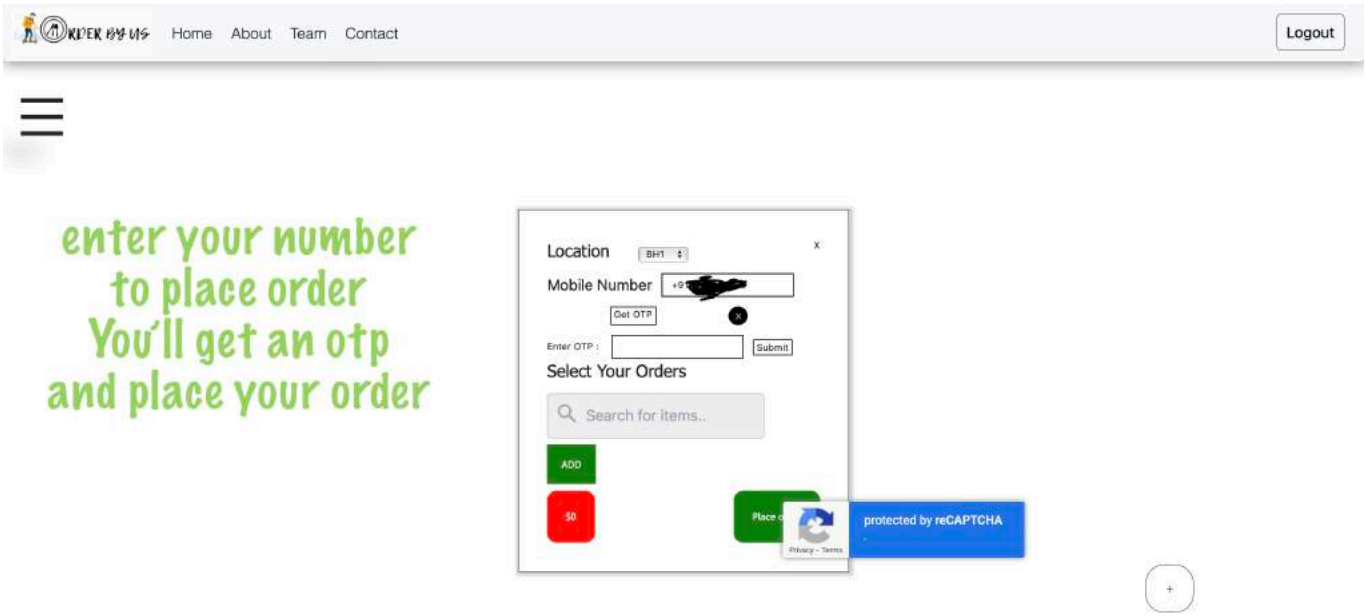
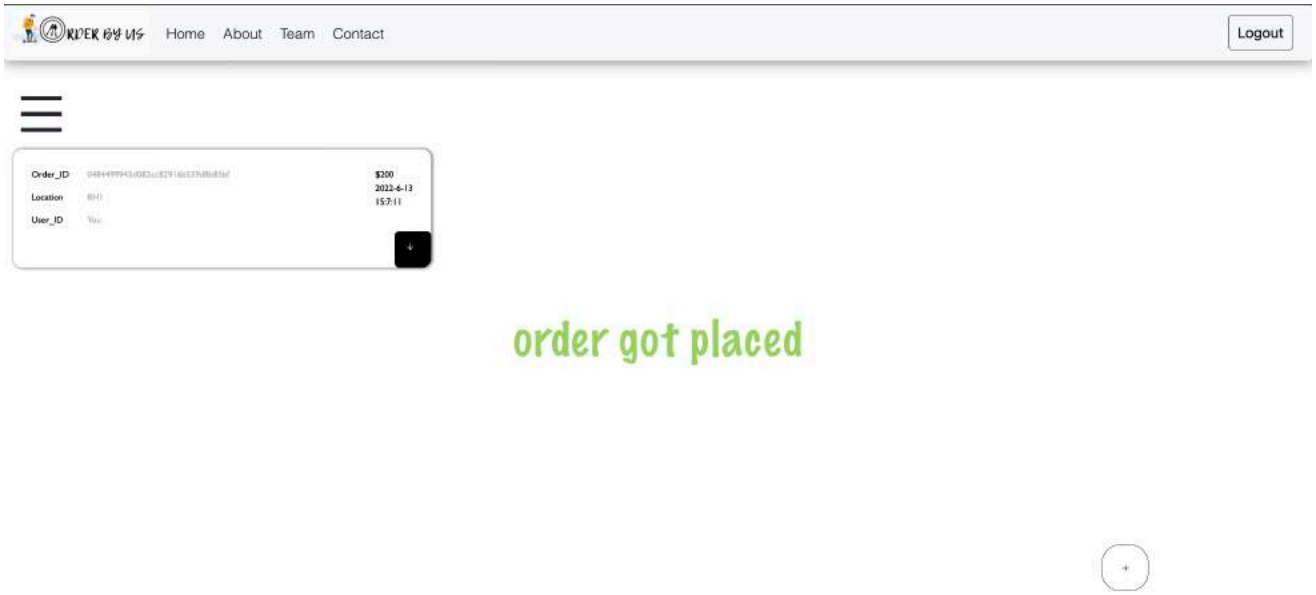


Fig:4.4.8 OTP Page

Agent can choose orders and take-up them.

Fig:4.4.9 Orders Page



CHAPTER - 5

SYSTEM TESTING

5.1.UNIT TESTING &

5.2.INTEGRATION TESTING

Nextjs+Firebase Integration

Fig:5.2.1 Firebase Integration Code

```
firebase-config.js
firebase-config.js > ...
import { getApps, initializeApp } from "firebase/app";
import { getAuth } from "firebase/auth";

import { getFirestore } from "firebase/firestore";
// import { apps } from "firebase-admin";

const firebaseConfig = {
  apiKey: [redacted]
};

export const firebaseApp = initializeApp(firebaseConfig);
export const database = getFirestore(firebaseApp);


//for mobile

export const authentication = getAuth(firebaseApp);


// export const firebaseApp = ()=>{
//     if(!apps.length)
//     {
//         initializeApp(firebaseConfig);
//     }
// }

// if (!firebase.apps.length) {
//     firebase.initializeApp(clientCredentials);
// }
```

— created a nextjs app

— run this command

npm i firebase firestore

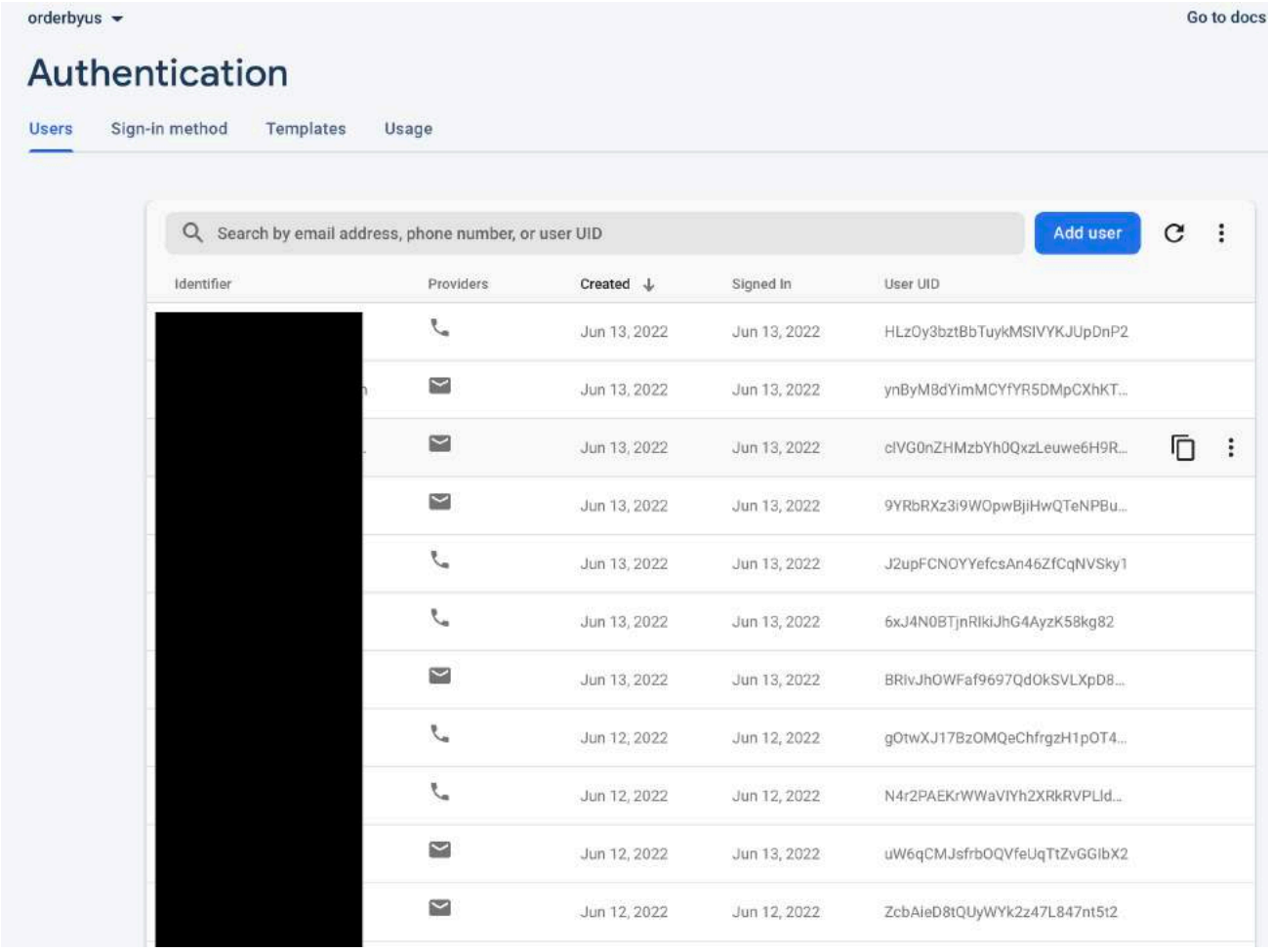
— added firebase configuration keys

— so we can use database, authentication services from firebase.

5.3TEST CASES

Authentication testing:

Fig:5.3.1 Authenticated users list in Firebase



Database Testing :

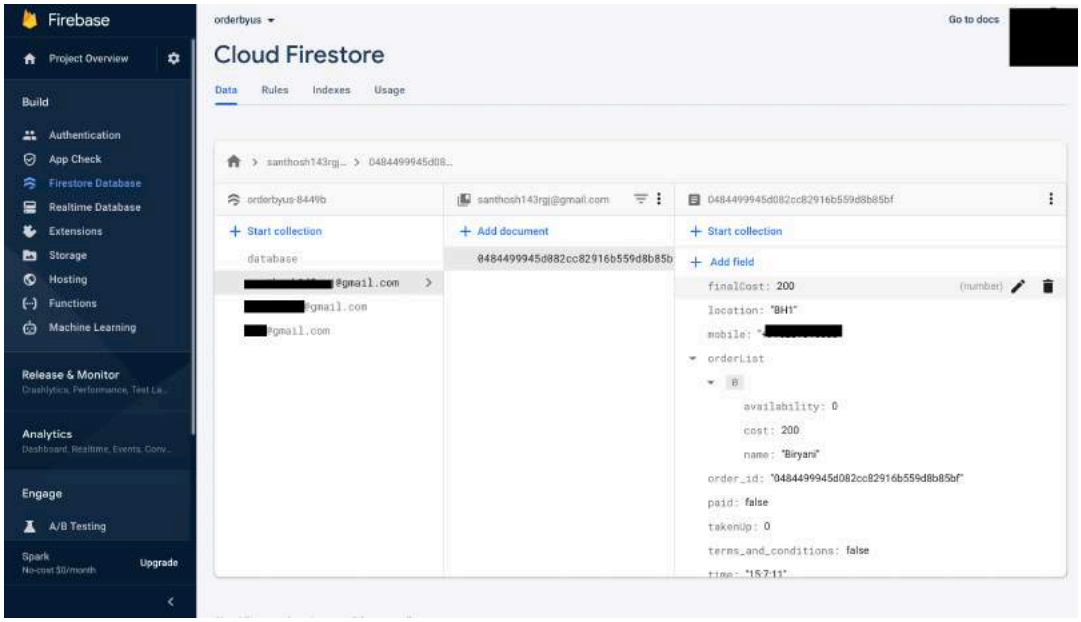


Fig:5.3.2 Data Storage in Firebase

Successfully stored data into Firestore Database.

CHAPTER-6

CONCLUSION AND FUTURE SCOPE

It was a wonderful learning experience for me while working on this project. This project took me through the various phases of project development and gave me real insight into the world of software engineering. The joy of working and the thrill involved while tackling the various problems and challenges gave me a feel of the developers' industry. It was due to this project I came to know how professional websites is designed.

Time and Money are two superior things everyone desire for.Order by us helps users in saving time and earning money without charging users single penny.We strongly believe in our work and definitely we see this website growing in less span of time.Currently we are implementing this in our campus , we will definitely expand and introduce this website in other campuses ,our ultimate goal is to help students like us.

CHAPTER-7

REFERENCES

https://nextjs.org/learn/foundations/about-nextjs

https://firebase.google.com/docs?gclsrc=ds&gclsrc=ds&gclid=CN35qY_YqvgCFY4PjgodD4QG5g

Page left intentionally…

Rajiv Gandhi University of Knowledge Technologies, Basar
Department of Computer Science and Engineering

A BRIEF PRESENTATION
ON

TITLE OF THE PROJECT :ORDER BY US

By

STUDENT NAME : SANTHOSH KUMAR PADIDALA
ID:B171777
STUDENT NAME : VIJAYASIMHA REDDY KAMIDI
ID:B171414

Under the Guidance of

SE/WT Project In-charge
Mr.K.RaviKanth
Asst.Prof., CSE,RGUKT

SE/WT Project In-charge
Mr.B.Venkat Raman
Asst.Prof., CSE,RGUKT

ABSTRACT

We the students of Computer Science Engineering RGUKT Basar are developing a website named OrderByUs.For students in universities similar to RGUKT, who has no accessibility to an online food ordering facility, OrderByUs is a platform for those students. Where they can order food besides delivering food employing they can get benefits. So this is like a platform to facilitate students saving their time and get their food at their door step furthermore making students financially independent in college. Our task is to join the users who want food and who want money. Whoever wants food can place an order on our website, and agents who are willing to deliver can take up the order and deliver to the user receiving reasonable delivery charges. Provides a good and healthy interface between users who want food parcels at their doorsteps and users who are willing to deliver food to get money. Avoiding fraud agents, users, and data.

Introduction

Motivation

As we know that necessity is the mother of invention. For students like us who study in rural areas, universities doesn't have an online food ordering facility. To help those students we are developing this website.

Problem Statement

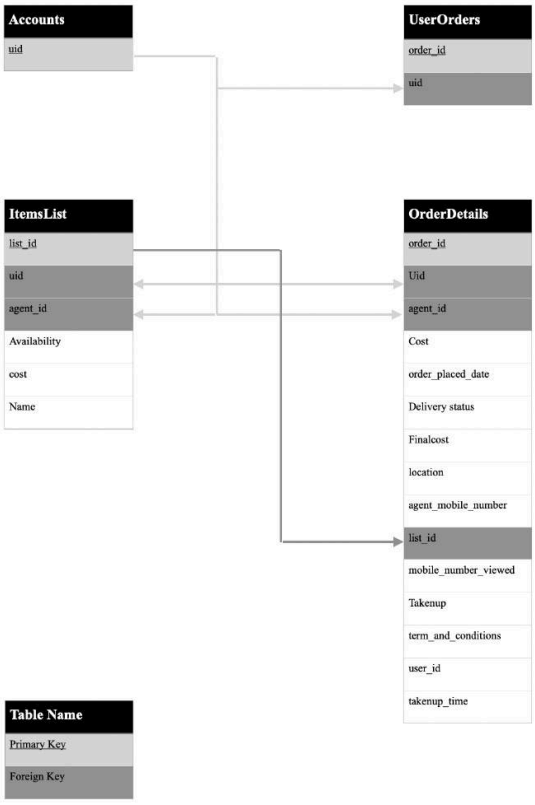
Now-a-days students in universities like RGUKT, it became a hectic task to get the food from campus food courts. As the food courts are usually far from the hostels, most people are not willing to walk up to the food courts and get their food. So we are developing a web page which helps people in delivering their food at their doorstep by the students themselves, whoever deliver the food get paid by the client. It will be like a part-time job for needy students, as well as helps the customers in saving their energy.

Objective and Scope of the Project

Helping students of rural universities by providing a online food ordering platform

DESIGN MODULES

Database schema for OrderByUs website:



DESIGN MODULES

Database Tables and type

TABLE ACCOUNTS

Column	Null?	Type
U_ID	NOT NULL	VARCHAR2(50)

TABLE USERORDERS

Column	Null?	Type
ORDER_ID	NOT NULL	VARCHAR2(50)
U_ID	-	VARCHAR2(50)

TABLE ITEMSLIST

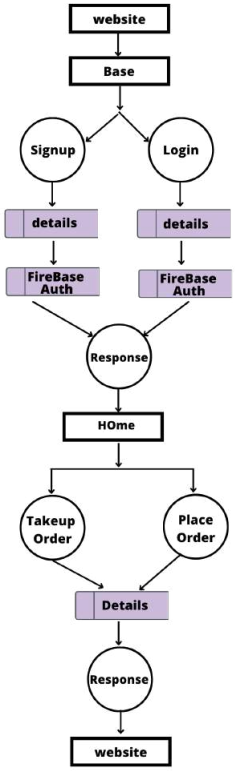
Column	Null?	Type
LIST_ID	NOT NULL	VARCHAR2(50)
U_ID	-	VARCHAR2(50)
AGENT_ID	-	VARCHAR2(50)
AVAILABILITY_	-	NUMBER(1,0)
COST_	-	NUMBER(10,0)
NAME_	-	VARCHAR2(50)

TABLE ORDERDETAILS

Column	Null?	Type
ORDER_ID	NOT NULL	VARCHAR2(50)
U_ID	-	VARCHAR2(50)
AGENT_ID	-	VARCHAR2(50)
COST_	-	NUMBER(10,0)
PLACE_DATE	-	VARCHAR2(6)
DELIVERY_STATUS	-	NUMBER(1,0)
FINALCOST	-	NUMBER(10,0)
LOCATION_	-	VARCHAR2(50)
AGENT_MOBILE	-	NUMBER(10,0)
LIST_ID	-	VARCHAR2(50)
VIEWED	-	NUMBER(1,0)
TAKENUP	-	NUMBER(1,0)
TERMS_AND_CONDITIONS	-	NUMBER(1,0)
TAKENUP_TIME	-	VARCHAR2(10)
USER_ID	-	VARCHAR2(50)

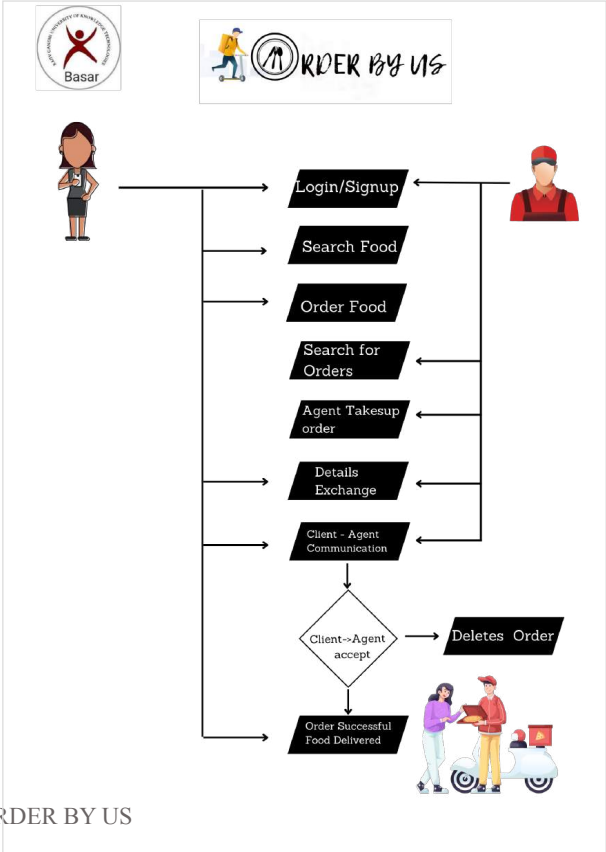
DESIGN MODULES

DATA FLOW DIAGRAMS



DESIGN MODULES

UML DIAGRAMS



Tools used

- HTML5
- CSS3
- JAVA SCRIPT
- BOOTSTRAP 5
- NEXT JS
- FIREBASE

SOFTWARRE & HARDWARE REQUIREMENT SPECIFICATION

SOFTWARE TOOL USED

Operating System : Windows/ Linux / Mac

Node JS framework

HTML, CSS, Bootstrap

NEXT JS

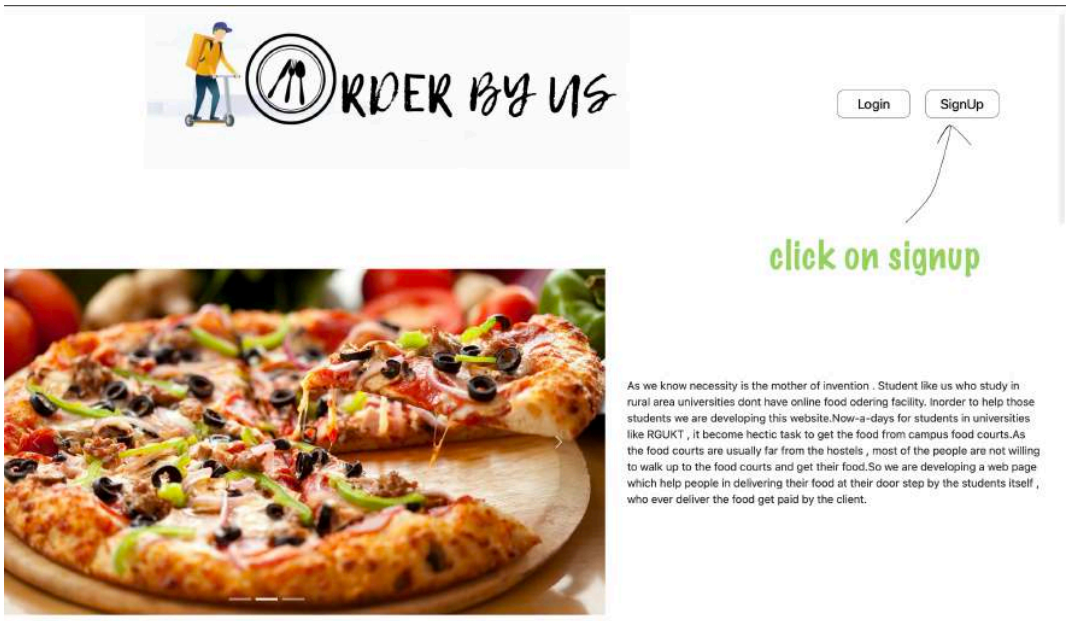
Back-end : FIREBASE

2.5.HARDWARE USED

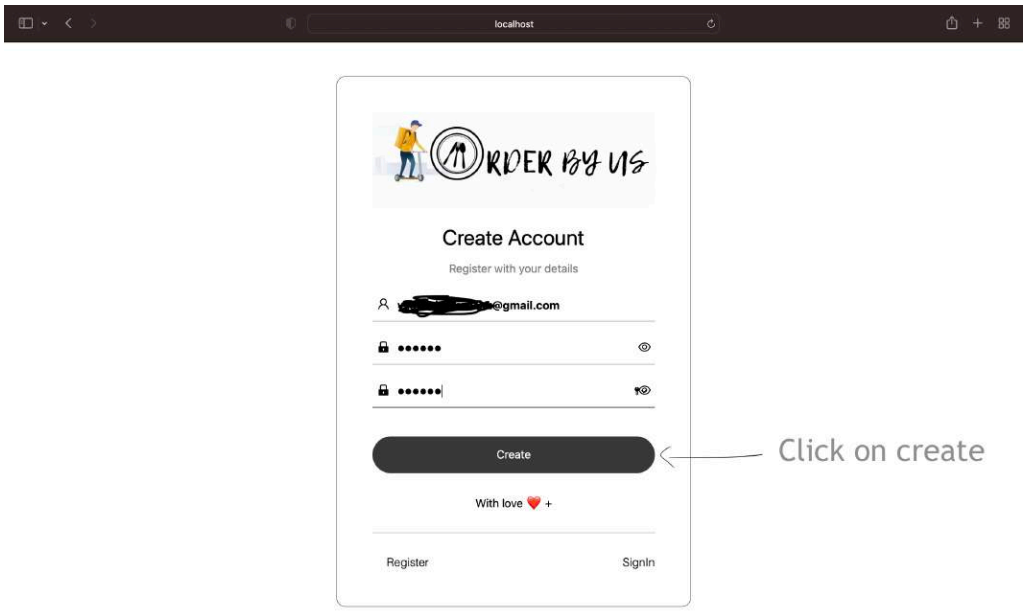
4 GB RAM

Intel Core i3 processor

Results/ Screen shots

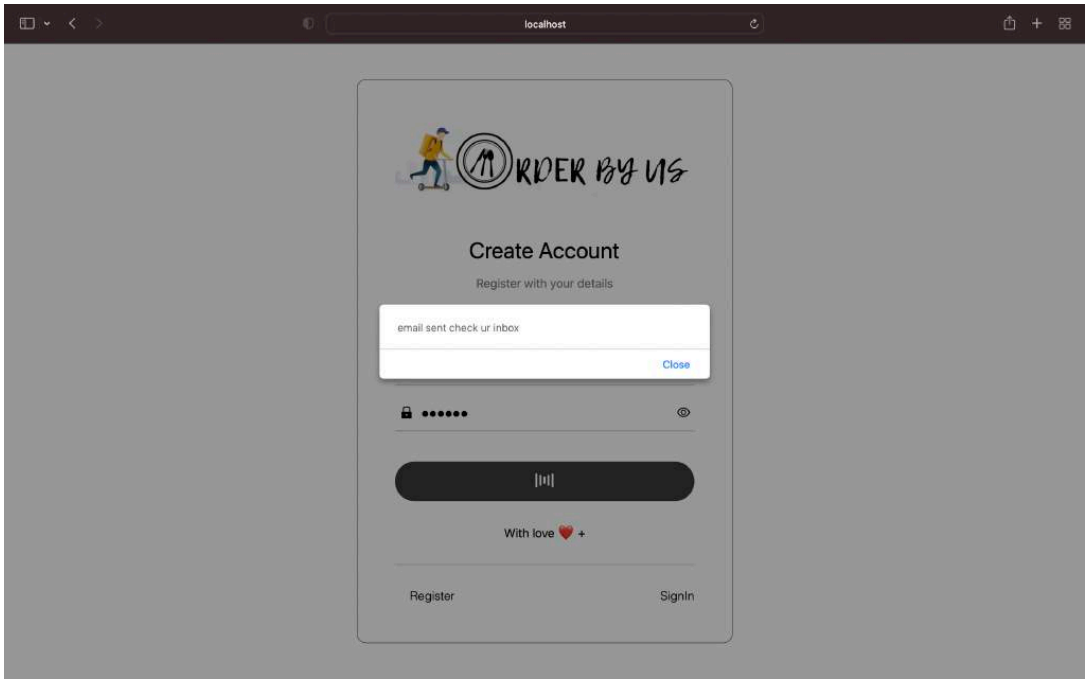


Results/ Screen shots



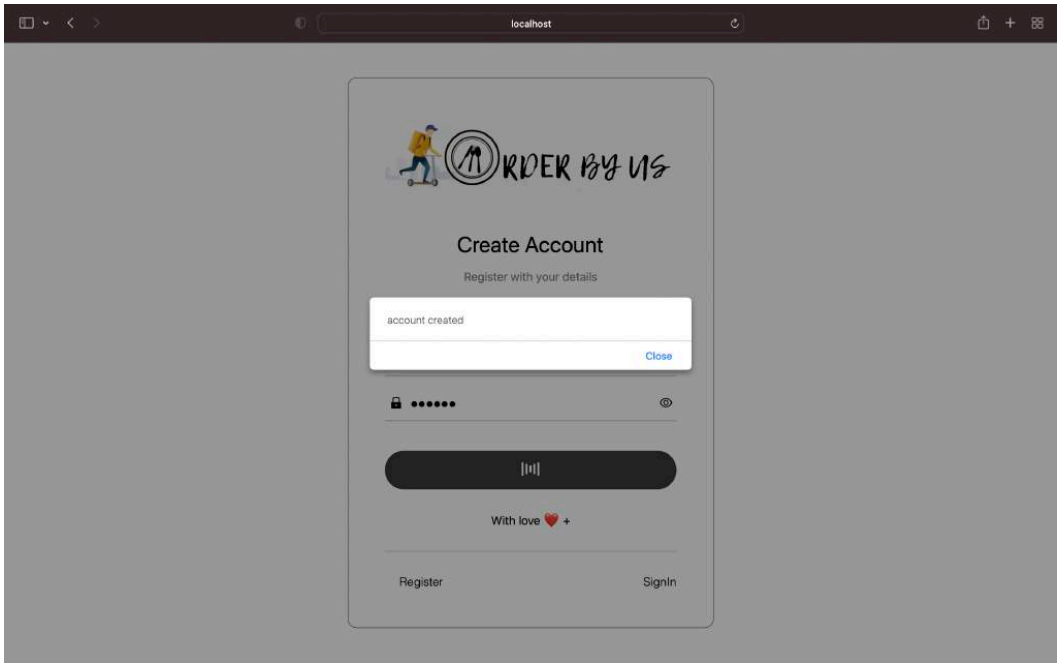
Results/ Screen shots

Verification email sent page



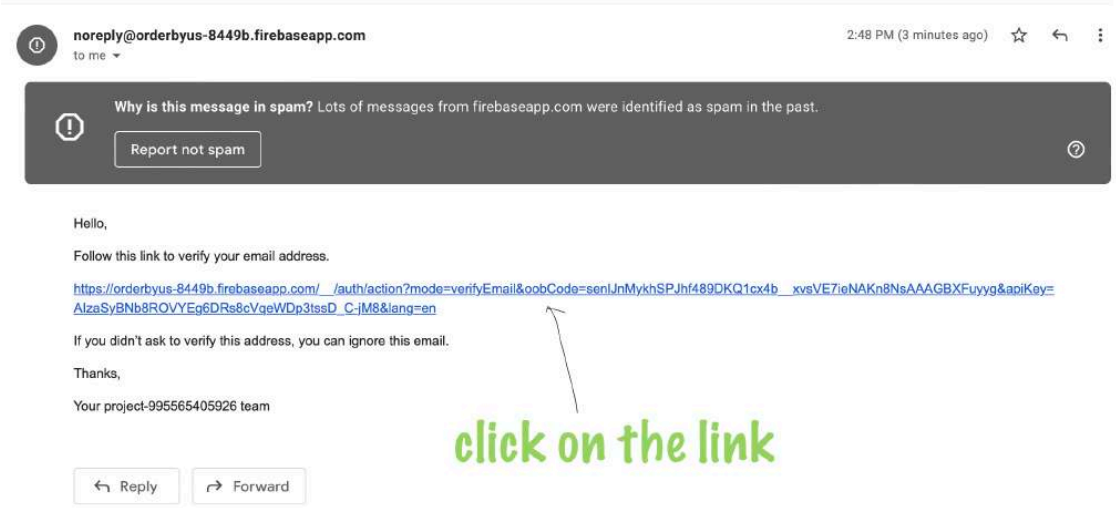
Results/ Screen shots

Account Created page



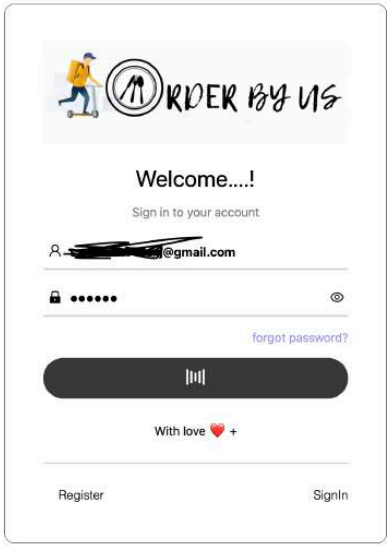
Results/ Screen shots

Email verification Link

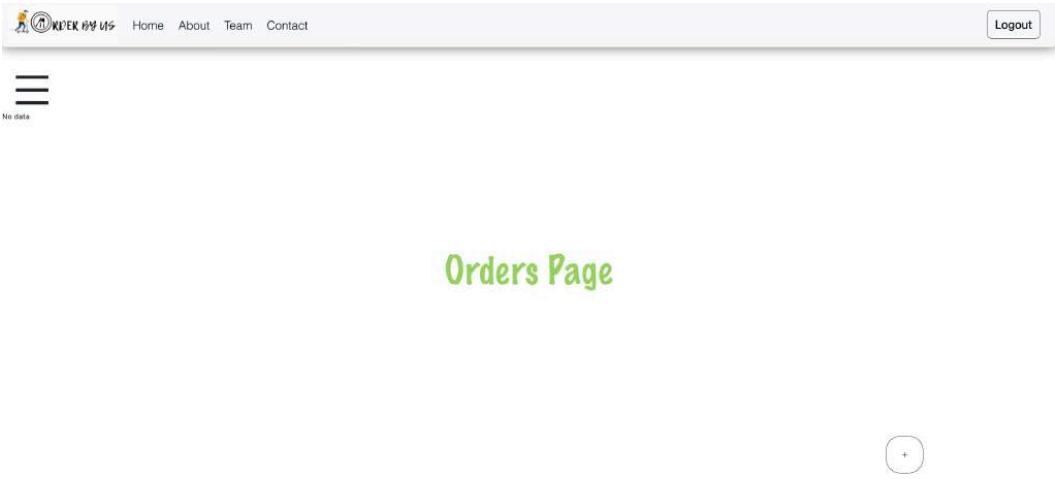


Results/ Screen shots

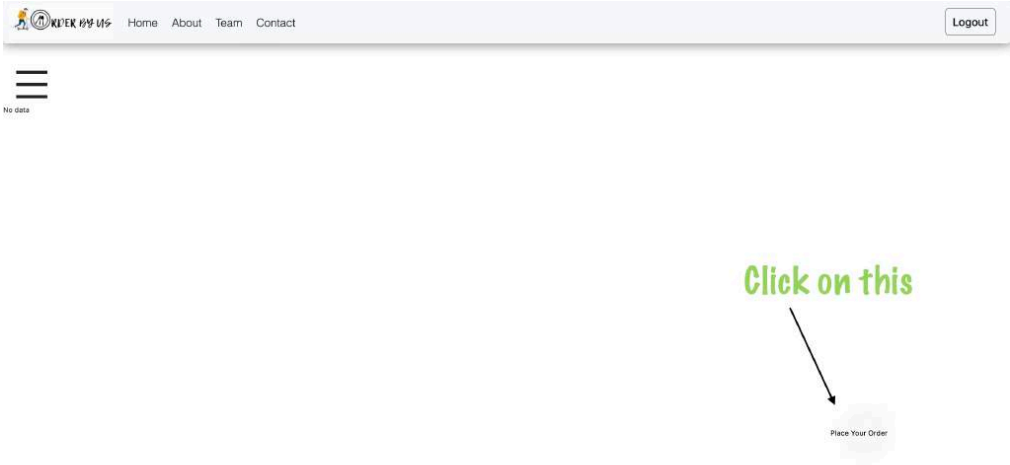
Login Page



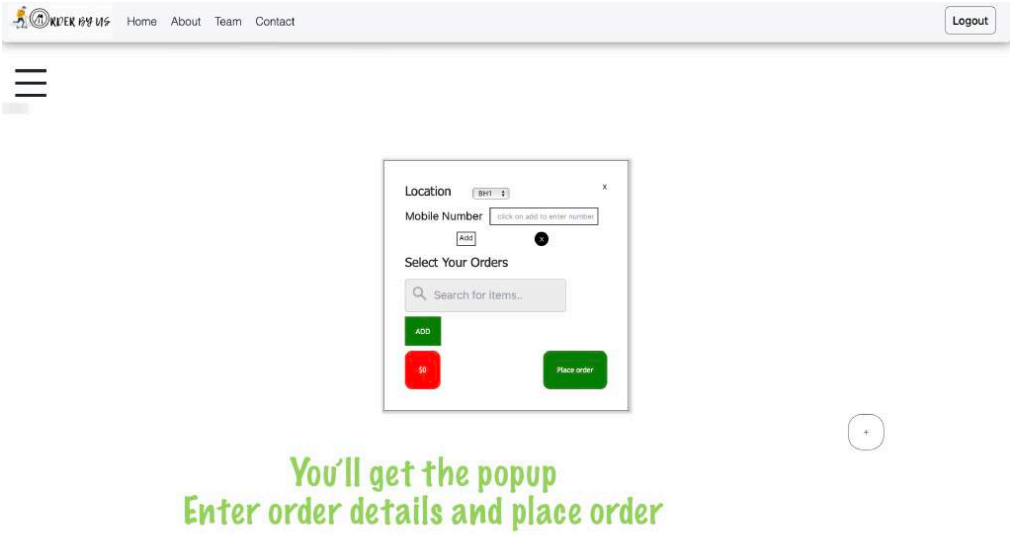
Results/ Screen shots



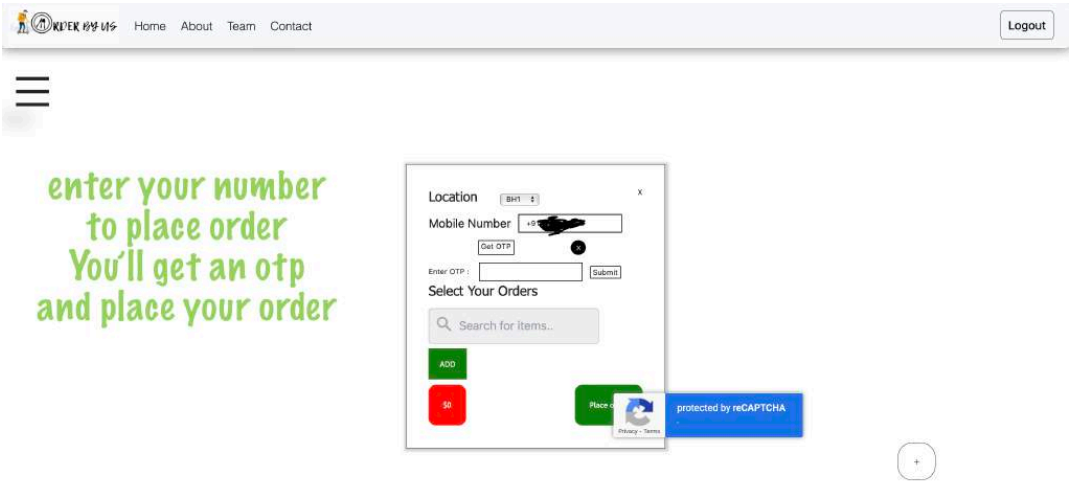
Results/ Screen shots



Results/ Screen shots



Results/ Screen shots



Results/ Screen shots

Conclusion & Future Scope

It was a wonderful learning experience for me while working on this project. This project took me through the various phases of project development and gave me real insight into the world of software engineering. The joy of working and the thrill involved while tackling the various problems and challenges gave me a feel of the developers' industry. It was due to this project I came to know how professional websites is designed.

Time and Money are two superior things everyone desire for.Order by us helps users in saving time and earning money without charging users single penny.We strongly believe in our work and definitely we see this website growing in less span of time.Currently we are implementing this in our campus , we will definitely expand and introduce this website in other campuses ,our ultimate goal is to help students like us.

References

https://nextjs.org/learn/foundations/about-nextjs

https://firebase.google.com/docs?gclsrc=ds&gclsrc=ds&gclid=CN35qY_YqvgCFY4PjgodD4QG5g

ACKNOWLEDGEMENT

First I Would like to thank management of RGUKT-Basar for giving me the opportunity to do this project work within the organisation.

I also would like to thank all the people who worked along with me RGUKT-Basar, with their patience and openness they created an enjoyable working environment.

It is indeed with a great sense of pleasure and immense of gratitude that I acknowledge the help of these individuals.

I am highly indebted to Vice-Chancellor and Administrative Officer, for the facilities provided to accomplish this project work.

I would like to thank my Head of the Department **,CSE** for her constructive criticism throughout my project work.

I would like to thank my supervisors **Mr.K.Ravikanth & Mr.B.Venkat Raman, Assistant Professors ,CSE** for their constructive criticism throughout my project work.

I would like to thank our department PRC Team Members, CSE for their support and advices to get and complete project within the given guidelines .

I am extremely great full to my department staff members, family members and friends who helped me in successful completion of this project work.

Any Queries

Thank you