Project Report

On

ONLINE FOOD ODER MANAGEMENT SYSTEM

Bachelor of Technology

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Sk. Mohaseen(R161052)

Under the guidance of

Mr. B. Lingamurthy
Software Engineer

Department of Computer Science and Engineering

Rajiv Gandhi University of Knowledge and Technologies(RGUKT), R.K.Valley, Kadapa, Andra Pradesh.

Rajiv Gandhi University of Knowledge Technologies
RK Valley,Kadapa (Dist), Andhra Pradesh, 516330

CERTIFICATE

This is to certify that the project work titled " **ONLINE FOOD ORDER MANAGEMENT SYSTEM**" is a bonafied projectwork submitted by Sk.Mohaseen In the department Of COMPUTER SCIENCE AND ENGINEERING in partial fulfillmentof Requirements for the award of degree of Bachelor of Technology in Computer science and engineering .For the award of degree of Bachelor of Technology in Computer science and engineering for the year 2022-2023 carried out the work under the supervision.

PROJECT INTERNAL GUIDE
Mr. B. Lingamurthy

HEAD OF THE DEPARTMENT
N Satyanandaram

Submitted on.....

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am extremely grateful to our respected Director, Prof. K. SANDHYA RANI for fostering an excellent academic climate in our institution.

I also express my sincere gratitude to our respected Head of the Department P.Harinadha BABU for his encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project.

I would like to convey thanks to our guide at college Mr. B .LINGAMURTHY for his guidance, encouragement, co-operation and kindness during the entire duration of the course and academics.

My sincere thanks to all the members who helped me directly and indirectly in the completion of project work .I express my profound gratitude to all our friends and family members for their encouragement.

INDEX

Abstract

E-commerce refers to the purchase and sale of goods and/or services via Internet. Online Food Ordering System is a part of e-commerce. ONLINE FOOD ORDERING SYSTEM is a website designed primarily for use in the food delivery industry. Through these services restaurants can sell and distribute their resources at minimal resource usage of effectively with high profits by gaining the customer trust. This Online food order system database will be helpful for the business owners to extend their business just by placing the orders online and not visiting the restaurant.

There is no confinement for placing and receiving he orders, since the order can be placed online. There will be no waiting time with the vast amount of verities at the comfortable prices. To develop this application database is the main part which will communicate through the application to retrieve the details. We will be creating the online food ordering database .Database includes Customers can place their orders from different food categories.

Introduction

It is very typical to establish a small-scale business with fewer resources to provide quality services. Now a day's people are attracted to online business. Let us assume if there is any online business where customers can order their needs and the goods will reach them at the expected delivery time.

The customers of today are not only attracted because placing an order online is very convenient but also because they have visibility into the items offered, likewise online food ordering system customers can order their favorite foods and this database will be the barrier for the customers and restaurants to provide the services.

Our solution provides ordering process for the restaurants and customers and the employees of the restaurants. The Items list and categories of the foods are available in the database so that a customer can place an order with multiple items. Once the order is placed restaurant employees process the order and deliver it to the customer at the expected delivery time.

at the end of the order customer will know about the amount how much he had to pay for the restaurant for the order. Once the Order is delivered customer can provide the feedback to the restaurant.

Purpose

The primary purpose of an online food ordering system is to allow Customers to easily place orders at a restaurant over the internet with the improvement of technology, online food ordering systems are becoming a popular topic. That does because they are serving the ever increasing demand for convince. The main purpose of an online ordering system is to provide customers for a way to place an order at a restaurant over the internet

Scope

Foods ordering app can sale Food product, preferred brands, kitchen needs, essential restaurant supplies and more, through this online, one-stop Food store. It provides you with a convenient way to sale from your Food shopping app. You can use this app as one big super market app to sale product of your store. This app make easy for user to buy product from store with easy steps and store can get easy order.

Nowadays everyone is having a busy schedule whether it is urban areas or rural. But talking specifically about the urban areas and deeply about the big cities, people out there are so busy in their life that they don't get enough time to have their meals properly. These days women are no less than men, in any field.

So, in big cities even wives are working women, therefore mostly the small families manage to have their food ordered from somewhere, as they lack time. Not only this is the case, if we talk about the children in the modern era, they like only fast food or something from the outside. But they ignore eating homemade meals.

So, the food ordering system these days has one of the fastest-growing markets, though being a new idea. In this project, we have developed something like the same to learn from and serve the nation in a much better way possible.Nowadays, people are more regular to dine-in at the restaurant for their meals.

The online food ordering system provides convenience for the customers that are nothing special but the general busy people of the society. It overcomes the demerits of the manual hotel or mess system and the old-fashioned queuing system. This system enhances the ready-made foods that people.

Therefore, this system enhances the speed of getting food on a person's plate and the quality and manner of taking the order from the customer. It provides a better communication platform. The user's details are stored using electronic media. The online food ordering system provides the menu online and the customers can easily place the order by just clicking the mouse or by touching a button on their smart phones.

Requirement Specification

Hardware Configuration

Client Side:

RAM	512MB
Hard disk	10GB
Processor	1,0GHz

Server side:

RAM	4GB
Hard disk	20GB
Processor	2.0GHz

Requirement Specification:

Frontend	HTML,CSS,JAVASCRIPT
Database Server	MYSQL
Web server	Firefox , Google Chrome or any compatible browser
Operating System	Ubuntu ,Windows or any equivalent OS
Software	VS CODE

APACHE

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards.
The Apache HTTP Server was launched in 1995 and it has been the most popular web server on the Internet since April 1996. It has celebrated its 20th birthday as a project in February 2015.

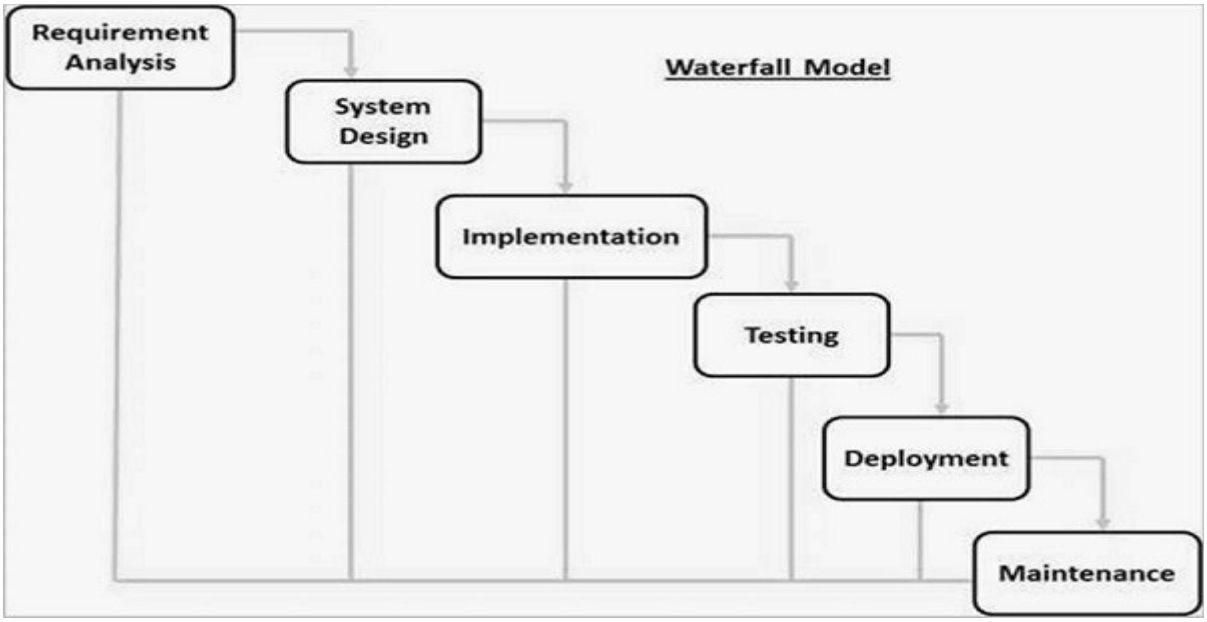
java script

JavaScript often abbreviated as JS, is an interpreted, high-level programming language. Additionally, it is a dynamic, weakly typed, prototype-based, and multi-paradigm language. One of the three fundamental technologies of the World Wide Web, together with HTML and CSS, is JavaScript. JavaScript is a crucial component of online applications because it makes web pages interactive. The vast majority of websites make use of it, and every significant web browser has an engine specifically designed to run JavaScript.

CSS
Cascading Style Sheets (CSS) is a language for creating style sheets that describe how a document produced in a markup language like HTML will look. The World Wide Web's foundational technologies, along with HTML and JavaScript, include CSS. Layout, color, and font may all be separated from content and presentation using CSS. By describing the pertinent CSS in a separate CSS file, this separation can make content more accessible, give definition of presentation features greater freedom and control, allow numerous web pages to share formatting, and reduce complexity and repetition in structural content.

MYSQL
- MYSQL is a database server
- MYSQL is ideal for both small and large applications
- MYSQL supports standard SQL
- MYSQL compiles on a number of platforms
- MYSQL is free to download and use
- How to access MySQL

Methodology Development Model:



The Waterfall model's consecutive phases are:

Requirement Gathering and analysis − During this stage, all potential system needs are gathered and outlined in a requirement specification document.

• **System Design** − The system design is created in this phase after studying the requirement specifications from the first phase. This system design aids in determining the overall system architecture as well as the hardware and system requirements.

• **Implementation** − The system is initially built in discrete programs known as units, which are then combined in the following phase, using inputs from the system design. Unit testing is the process of developing and evaluating each unit for functionality.

• **Integration and Testing** − Following the testing of each unit created during the implementation phase, the entire system is merged. The entire system is tested for errors and failures after integration.

• **Deployment of system** − Once the product has undergone functional and non-functional testing, it is either published to the market or deployed in the customer's environment.

• **Maintenance** − Various problems can arise in a client environment. Patches are published to address certain problems. Additionally, improved versions of the product are issued. To bring about these changes in the surroundings of the consumer, maintenance is performed.

Analysis and Design

The online food ordering system provides convenience for the customers that are nothing special but the general busy people of the society.

It overcomes the demerits of the manual hotel or mess system and the old fashioned queuing system. This system enhances the readymade of foods than people.

Design Introduction

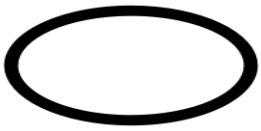
UML DIAGRAM

Actor:

A coherent set of roles that users of use cases play when interacting with the use cases.an observable result of value of an actor.

Use case:

A description of sequence of actions, including variants, which a system performs, yields an observable result of value of an actor. actor diagram is drowned in a eclipse shape.



UML stands for Unified Modeling Language. UML is a language for specifying,visualizing and documenting the system.

This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

USE CASE DIAGRAMS:

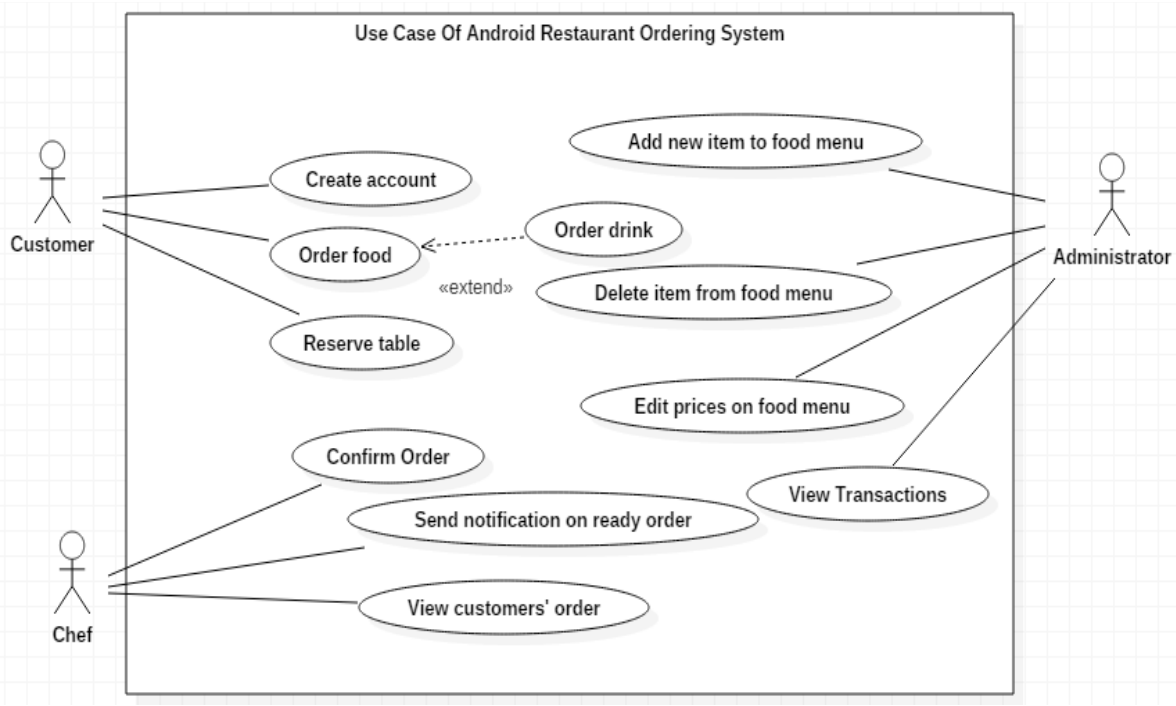
Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor. Use case diagram can be useful for getting anoverall view of the system and clarifying that can do and more importantly what they can't do. Use case diagram consists of use cases and actors and shows the interaction between the use case and actors. The purpose is to show the interactions between the use case and actor

.• To represent the system requirements from user's perspective

.• An actor could be the end-user of the system or an external system

A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object.
Primary Actor – Sender, Secondary Actor Receiver.

USE CASE DIAGRAM:



ER Diagram:

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database design for the database designer, the utility of the ER model is: It maps well to the relational model. The constructs used in the ER model can easily be

- transformed into relational tables. It is simple and easy to understand with a minimum of training. Therefore, the model can
- be used by the database designer to communicate the design to the end user. In addition, the model can be used as a design plan by the database developer to
- implement a data model in specific database management software.

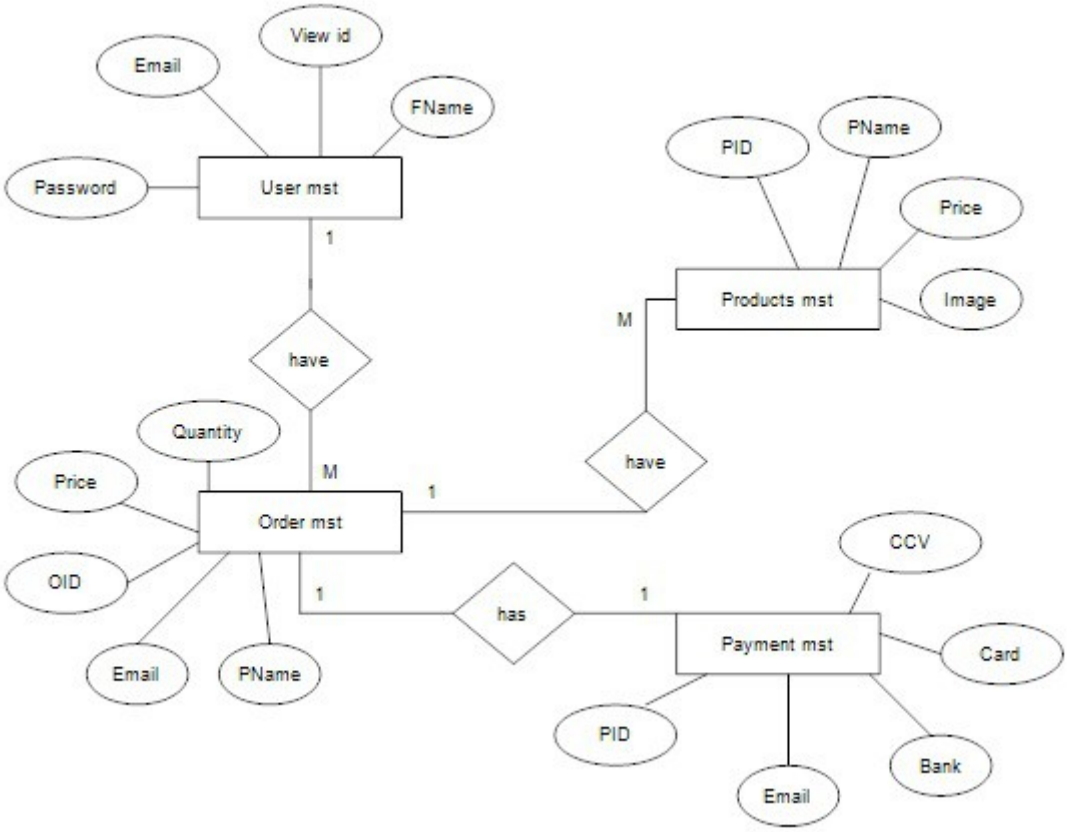
ER Natation

All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

- **Entities** are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
- **Relationships** are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs
- **Attributes,** when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
- **Cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one. Existence is represented by placing a circle or a perpendicular bar on the line.

Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required.Optional existence is shown by placing a circle next to the entity that is optional.

Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.


View id
Email
FName
Password
User mst
1
have
M
Products mst
PID
PName
Price
Image
have
1
Quantity
Price
Order mst
OID
Email
PName
1
has
1
Payment mst
CCV
Card
PID
Email
Bank

E-R Diagram

source code:

Project URL:http://localhost:3000/

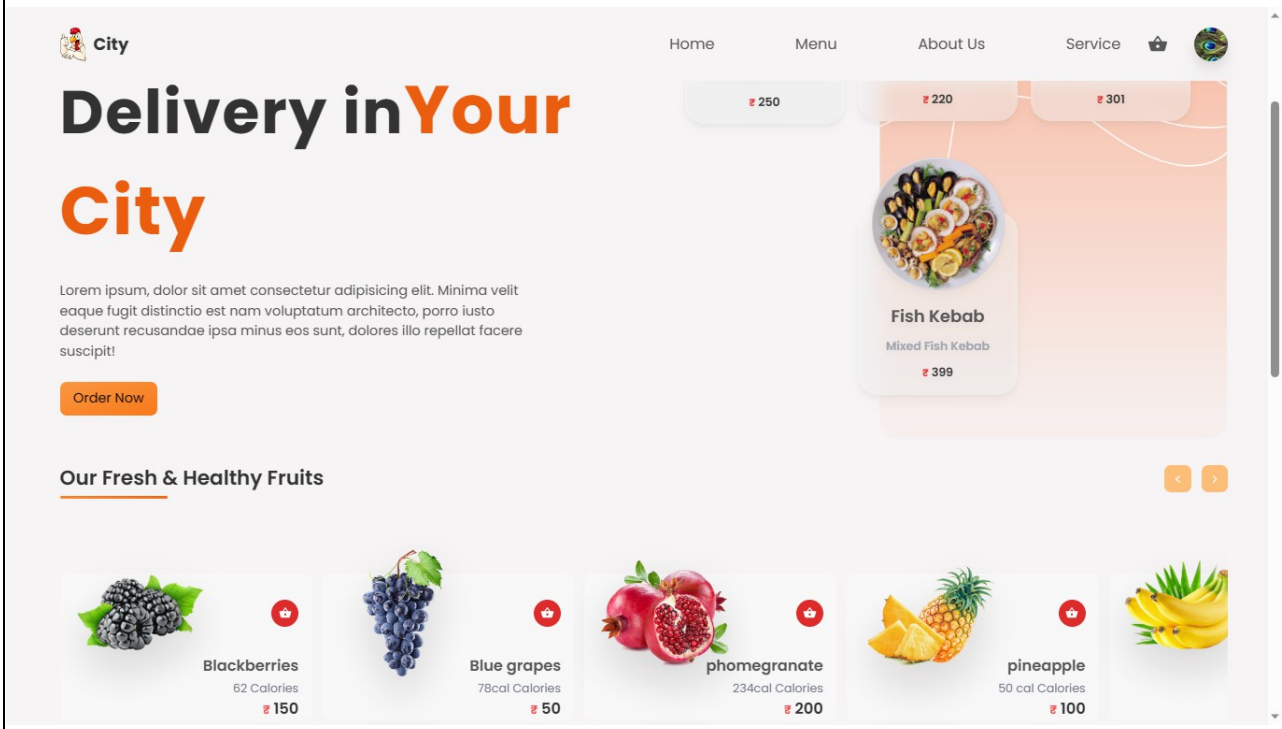
Homepage:

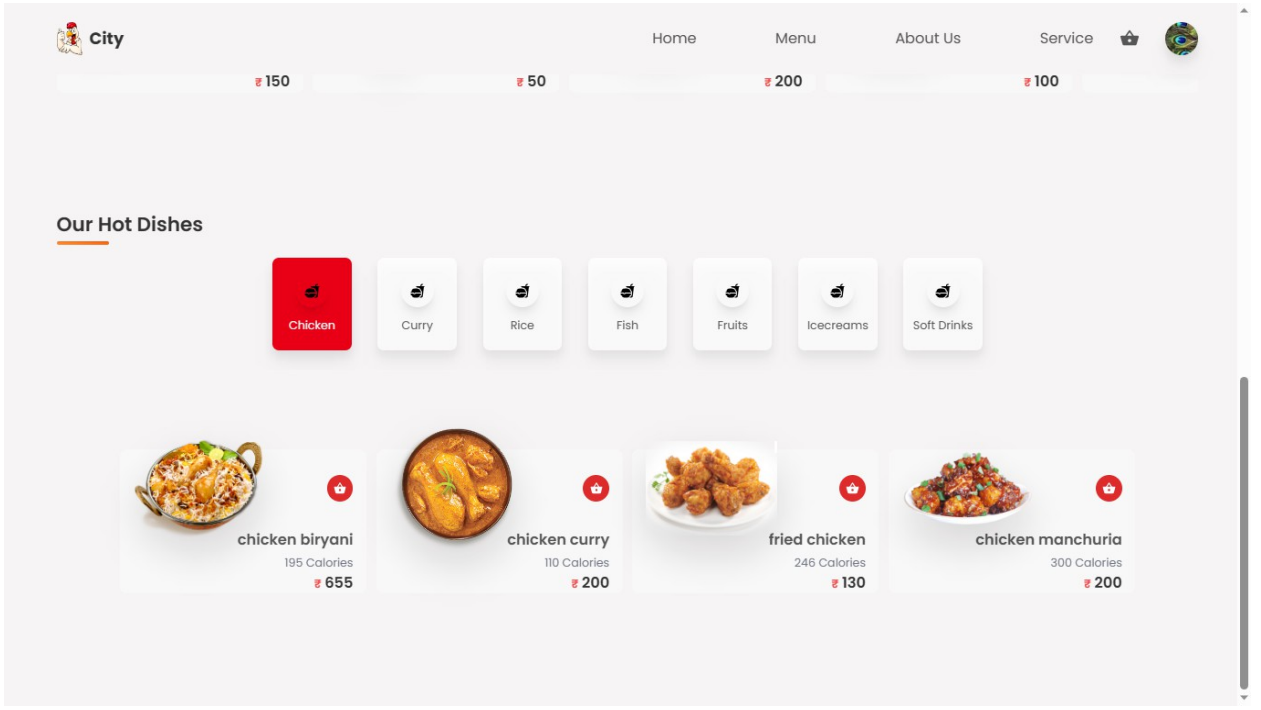
```
import React from "react";
import Delivery from "../img/delivery.png";
import HeroBg from "../img/heroBg.png";
import { heroData } from "../utils/data";

const HomeContainer = () => {
  return (
    <section
      className="grid grid-cols-1 md:grid-cols-2 gap-2 w-full "
      id="home"
    >
       <div className="py-2 flex-1 flex flex-col items-start justify-center gap-6">
         <div className="flex items-center gap-2 justify-center bg-orange-100 px-4 py-1 rounded-full">
          <p className="text-base text-orange-500 font-semibold">
           Bike Delivery
          </p>
          <div className="w-8 h-8 bg-white rounded-full overflow-hidden drop-shadow-xl">
           <img
             src={Delivery}
             className="w-full h-full object-contain"
             alt="delivery"
           />
          </div>
        </div>

         <p className="text-[2.5rem] lg:text-[4.5rem] font-bold tracking-wide text-headingColor">
          The Fastest Delivery in
          <span className="text-orange-600 text-[3rem] lg:text-[5rem]">
           Your City
          </span>
        </p>
```

```
        <p className="text-base text-textColor text-center md:text-left md:w-[80%]">
        Lorem ipsum, dolor sit amet consectetur adipisicing elit. Minima velit
        eaque fugit distinctio est nam voluptatum architecto, porro iusto
        deserunt recusandae ipsa minus eos sunt, dolores illo repellat facere
        suscipit!
       </p>

       <button
        type="button"
           className="bg-gradient-to-br from-orange-400 to-orange-500 w-full md:w-auto px-4 py-2  rounded-lg hover:shadow-lg transition-all ease-in-out duration-100"
       >
        Order Now
       </button>
     </div>
     <div className="py-2 flex-1 flex items-center relative">
      <img
       src={HeroBg}
       className=" ml-auto h-420 w-full lg:w-auto lg:h-650"
       alt="hero-bg"
      />

         <div className="w-full h-full absolute top-0 left-0 flex items-center justify-center lg:px-30  py-4 gap-4 flex-wrap">
        {heroData &&
         heroData.map((n) => (
          <div
           key={n.id}
                 className="  lg:w-190  p-4 bg-cardOverlay backdrop-blur-md rounded-3xl flex flex-col items-center justify-center drop-shadow-lg"
          >
           <img
            src={n.imageSrc}
            className="w-20 lg:w-40 -mt-10 lg:-mt-20 "
            alt="I1"
           />
            <p className="text-base lg:text-xl font-semibold text-textColor mt-2 lg:mt-4">
             {n.name}
           </p>
```

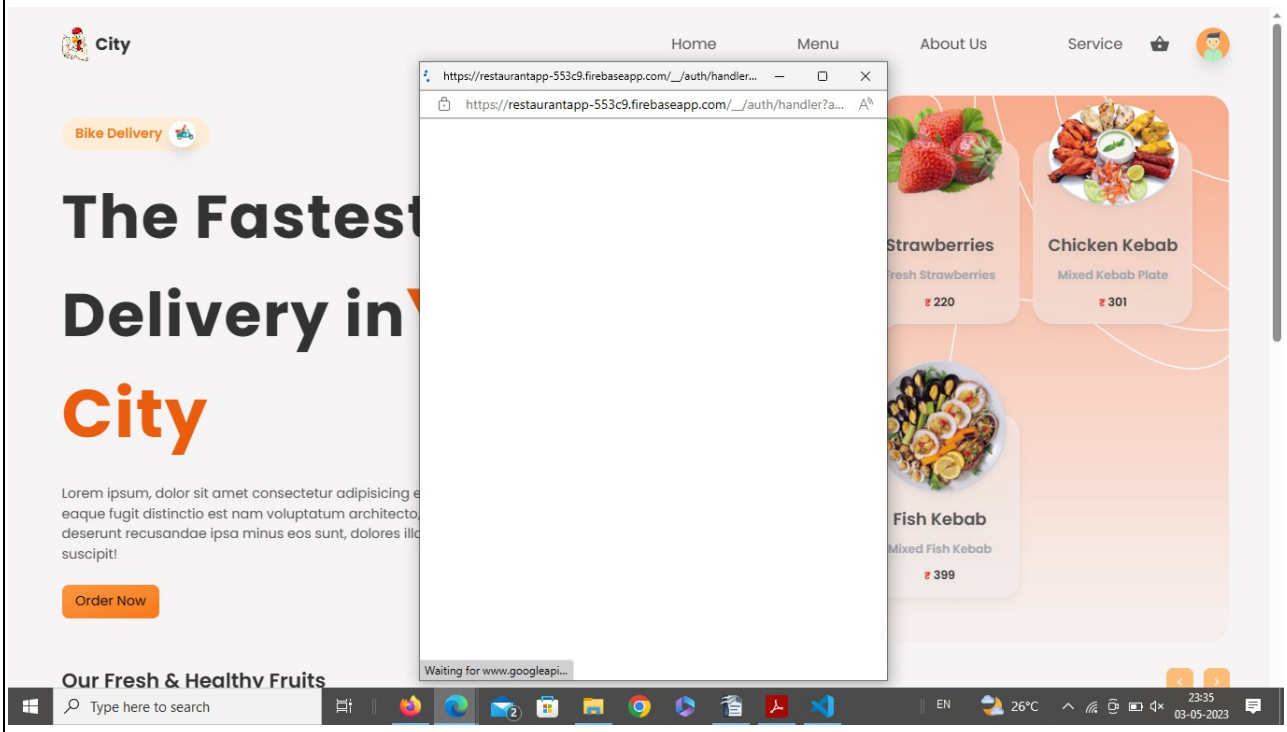
```
                <p className="text-[12px] lg:text-sm text-lighttextGray font-
semibold my-1 lg:my-3">
                  {n.decp}
                </p>

                <p className="text-sm font-semibold text-headingColor">
                  <span className="text-xs text-red-600">₹</span> {n.price}
                </p>
              </div>
            ))}
          </div>
        </div>
      </section>
    );
};

export default HomeContainer;
```

Output:


City
Home
Menu
About Us
Service
Delivery in Your City
Lorem ipsum, dolor sit amet consectetur adipisicing elit. Minima velit eaque fugit distinctio est nam voluptatum architecto, porro iusto deserunt recusandae ipsa minus eos sunt, dolores illo repellat facere suscipit!
Order Now
₹ 250
₹ 220
₹ 301
Fish Kebab
Mixed Fish Kebab
₹ 399
Our Fresh & Healthy Fruits
Blackberries
62 Calories
₹ 150
Blue grapes
78cal Calories
₹ 50
phomegranate
234cal Calories
₹ 200
pineapple
50 cal Calories
₹ 100


City
Home
Menu
About Us
Service
₹ 150
₹ 50
₹ 200
₹ 100
Our Hot Dishes
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Soft Drinks
chicken biryani
195 Calories
₹ 655
chicken curry
110 Calories
₹ 200
fried chicken
246 Calories
₹ 130
chicken manchuria
300 Calories
₹ 200

login Page:



createcontainer:

```
import React, { useState } from "react";
import { motion } from "framer-motion";
import {FaRupeeSign} from "react-icons/fa"


import {
  MdFastfood,
  MdCloudUpload,
  MdDelete,
  MdFoodBank,

} from "react-icons/md";
import { categories } from "../utils/data";
import Loader from "./Loader";
import {
  deleteObject,
  getDownloadURL,
  ref,
  uploadBytesResumable,
} from "firebase/storage";
```

```
import { storage } from "../firebase.config";
import { getAllFoodItems, saveItem } from "../utils/firebaseFunctions";
import { actionType } from "../context/reducer";
import { useStateValue } from "../context/StateProvider";

const CreateContainer = () => {
  const [title, setTitle] = useState("");
  const [calories, setCalories] = useState("");
  const [price, setPrice] = useState("");
  const [category, setCategory] = useState(null);
  const [imageAsset, setImageAsset] = useState(null);
  const [fields, setFields] = useState(false);
  const [alertStatus, setAlertStatus] = useState("danger");
  const [msg, setMsg] = useState(null);
  const [isLoading, setIsLoading] = useState(false);
  const [{ foodItems }, dispatch] = useStateValue();

  const uploadImage = (e) => {
    setIsLoading(true);
    const imageFile = e.target.files[0];
                const storageRef = ref(storage, `Images/${Date.now()}-${imageFile.name}`);
    const uploadTask = uploadBytesResumable(storageRef, imageFile);

    uploadTask.on(
      "state_changed",
      (snapshot) => {
        const uploadProgress =
          (snapshot.bytesTransferred / snapshot.totalBytes) * 100;
      },
      (error) => {
        console.log(error);
        setFields(true);
        setMsg("Error while uploading : Try AGain 🙇");
        setAlertStatus("danger");
        setTimeout(() => {
          setFields(false);
          setIsLoading(false);
        }, 4000);
      },
      () => {
```

```
      getDownloadURL(uploadTask.snapshot.ref).then((downloadURL) => {
        setImageAsset(downloadURL);
        setIsLoading(false);
        setFields(true);
        setMsg("Image uploaded successfully  ");
        setAlertStatus("success");
        setTimeout(() => {
          setFields(false);
        }, 4000);
      });
    }
  );
};

const deleteImage = () => {
  setIsLoading(true);
  const deleteRef = ref(storage, imageAsset);
  deleteObject(deleteRef).then(() => {
    setImageAsset(null);
    setIsLoading(false);
    setFields(true);
    setMsg("Image deleted successfully  ");
    setAlertStatus("success");
    setTimeout(() => {
      setFields(false);
    }, 4000);
  });
};

const saveDetails = () => {
  setIsLoading(true);
  try {
    if (!title || !calories || !imageAsset || !price || !category) {
      setFields(true);
      setMsg("Required fields can't be empty");
      setAlertStatus("danger");
      setTimeout(() => {
        setFields(false);
        setIsLoading(false);
      }, 4000);
    } else {
```

```
      const data = {
        id: `${Date.now()}`,
        title: title,
        imageURL: imageAsset,
        category: category,
        calories: calories,
        qty: 1,
        price: price,
      };
      saveItem(data);
      setIsLoading(false);
      setFields(true);
      setMsg("Data Uploaded successfully  ");
      setAlertStatus("success");
      setTimeout(() => {
        setFields(false);
      }, 4000);
      clearData();
    }
  } catch (error) {
    console.log(error);
    setFields(true);
    setMsg("Error while uploading : Try AGain  ");
    setAlertStatus("danger");
    setTimeout(() => {
      setFields(false);
      setIsLoading(false);
    }, 4000);
  }

  fetchData();
};

const clearData = () => {
  setTitle("");
  setImageAsset(null);
  setCalories("");
  setPrice("");
  setCategory("Select Category");
};
```

```
 const fetchData = async () => {
   await getAllFoodItems().then((data) => {
     dispatch({
       type: actionType.SET_FOOD_ITEMS,
       foodItems: data,
     });
   });
 };

 return (
   <div className="w-full min-h-screen flex items-center justify-center">
     <div className="w-[90%] md:w-[50%] border border-gray-300 rounded-
lg p-4 flex flex-col items-center justify-center gap-4">
       {fields && (
         <motion.p
           initial={{ opacity: 0 }}
           animate={{ opacity: 1 }}
           exit={{ opacity: 0 }}
           className={`w-full p-2 rounded-lg text-center text-lg font-semibold ${
             alertStatus === "danger"
               ? "bg-red-400 text-red-800"
               : "bg-emerald-400 text-emerald-800"
           }`}
         >
           {msg}
         </motion.p>
       )}

       <div className="w-full py-2 border-b border-gray-300 flex items-center
gap-2">
         <MdFastfood className="text-xl text-gray-700" />
         <input
           type="text"
           required
           value={title}
           onChange={(e) => setTitle(e.target.value)}
           placeholder="Give me a title..."
             className="w-full h-full text-lg bg-transparent outline-none border-
none placeholder:text-gray-400 text-textColor"
         />
       </div>
```

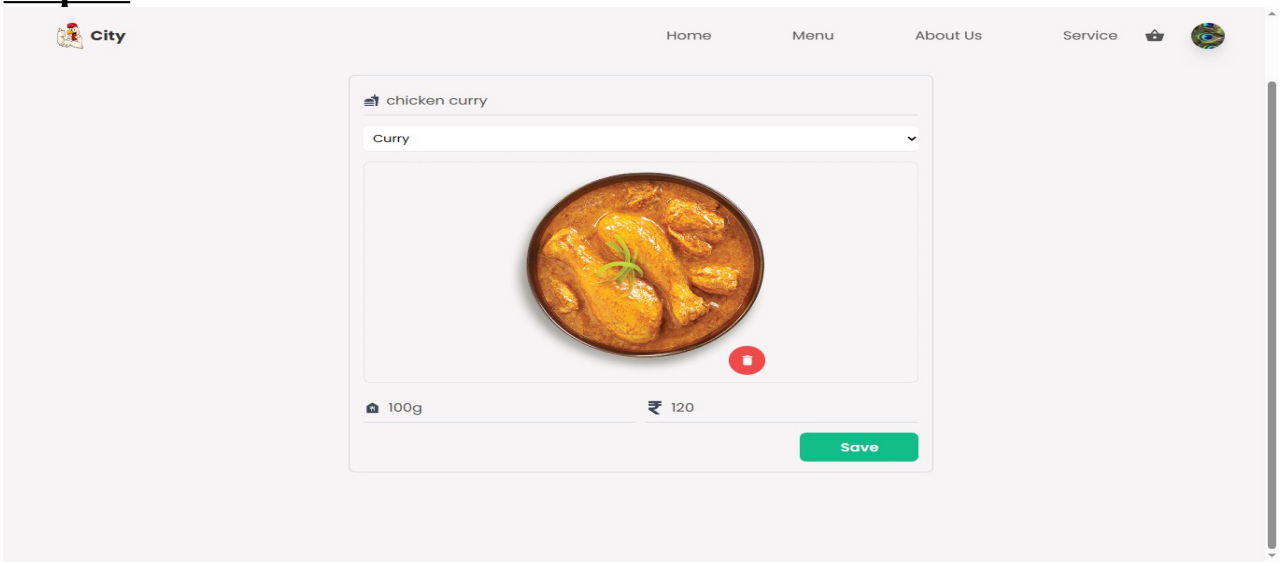
```
      <div className="w-full">
        <select
          onChange={(e) => setCategory(e.target.value)}
          className="outline-none w-full text-base border-b-2 border-gray-200
p-2 rounded-md cursor-pointer"
        >
          <option value="other" className="bg-white">
            Select Category
          </option>
          {categories &&
            categories.map((item) => (
              <option
                key={item.id}
                    className="text-base border-0 outline-none capitalize bg-white
text-headingColor"
                value={item.urlParamName}
              >
                {item.name}
              </option>
            ))}
        </select>
      </div>

       <div className="group flex justify-center items-center flex-col border-2
border-dotted border-gray-300 w-full h-225 md:h-340 cursor-pointer rounded-
lg">
        {isLoading ? (
          <Loader />
        ) : (
          <>
            {!imageAsset ? (
              <>
                <label className="w-full h-full flex flex-col items-center justify-
center cursor-pointer">
                  <div className="w-full h-full flex flex-col items-center justify-
center gap-2">
                    <MdCloudUpload className="text-gray-500 text-3xl hover:text-
gray-700" />
```

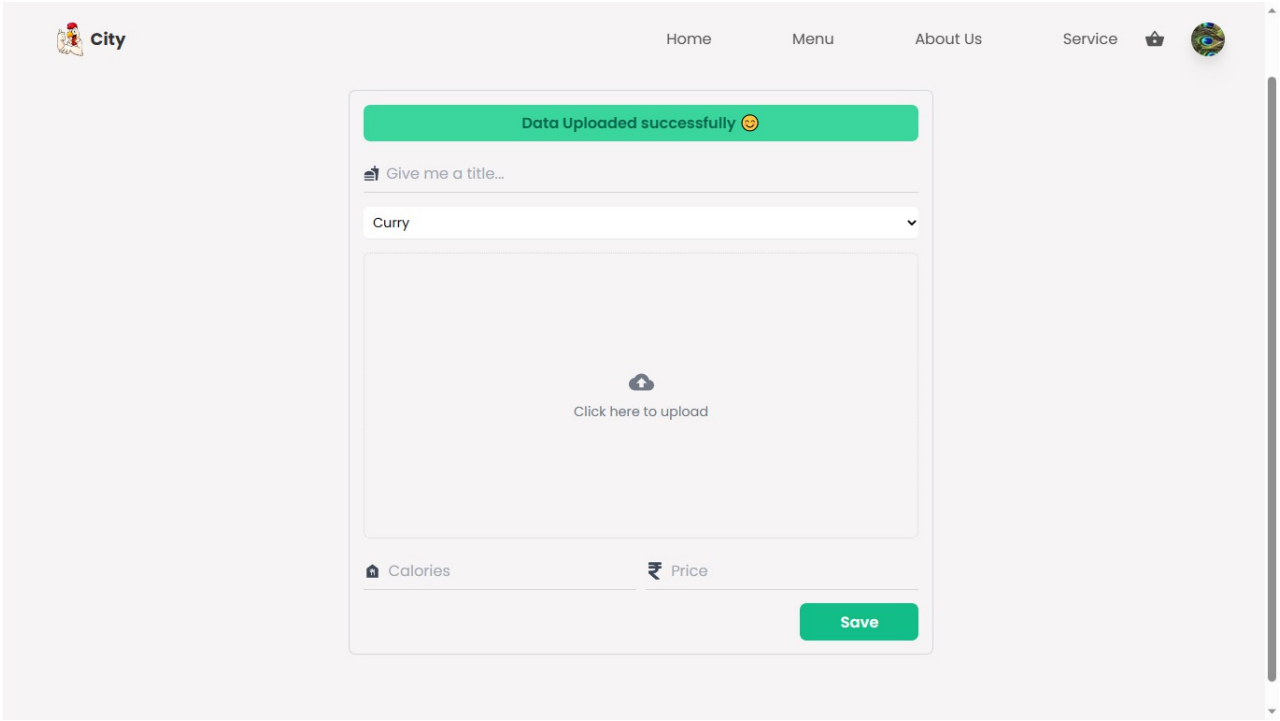
```
                <p className="text-gray-500 hover:text-gray-700">
                  Click here to upload
                </p>
              </div>
              <input
                type="file"
                name="uploadimage"
                accept="image/*"
                onChange={uploadImage}
                className="w-0 h-0"
              />
            </label>
          </>
        ) : (
          <>
            <div className="relative h-full">
              <img
                src={imageAsset}
                alt="uploaded image"
                className="w-full h-full object-cover"
              />
              <button
                type="button"
                  className="absolute bottom-3 right-3 p-3 rounded-full bg-red-
500 text-xl cursor-pointer outline-none hover:shadow-md duration-500
transition-all ease-in-out"
                onClick={deleteImage}
              >
                <MdDelete className="text-white" />
              </button>
            </div>
          </>
        )}
      </>
    )}
  </div>

  <div className="w-full flex flex-col md:flex-row items-center gap-3">
    <div className="w-full py-2 border-b border-gray-300 flex items-center
gap-2">
```

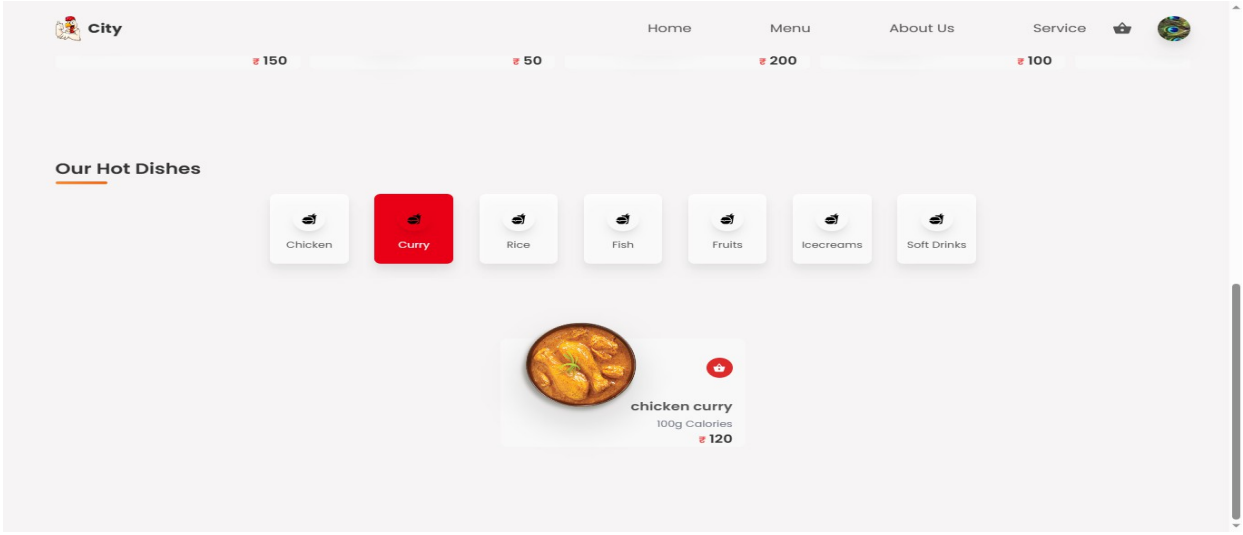
```
          <MdFoodBank className="text-gray-700 text-2xl" />
          <input
            type="text"
            required
            value={calories}
            onChange={(e) => setCalories(e.target.value)}
            placeholder="Calories"
              className="w-full h-full text-lg bg-transparent outline-none border-
none placeholder:text-gray-400 text-textColor"
          />
        </div>

        <div className="w-full py-2 border-b border-gray-300 flex items-center
gap-2">
          <FaRupeeSign className="text-gray-700 text-2xl" />
          <input
            type="text"
            required
            value={price}
            onChange={(e) => setPrice(e.target.value)}
            placeholder="Price"
              className="w-full h-full text-lg bg-transparent outline-none border-
none placeholder:text-gray-400 text-textColor"
          />
        </div>
      </div>

      <div className="flex items-center w-full">
        <button
          type="button"
            className="ml-0 md:ml-auto w-full md:w-auto border-none outline-
none bg-emerald-500 px-12 py-2 rounded-lg text-lg text-white font-semibold"
          onClick={saveDetails}
        >
          Save
        </button>
      </div>
    </div>
  </div>
 );
};
```

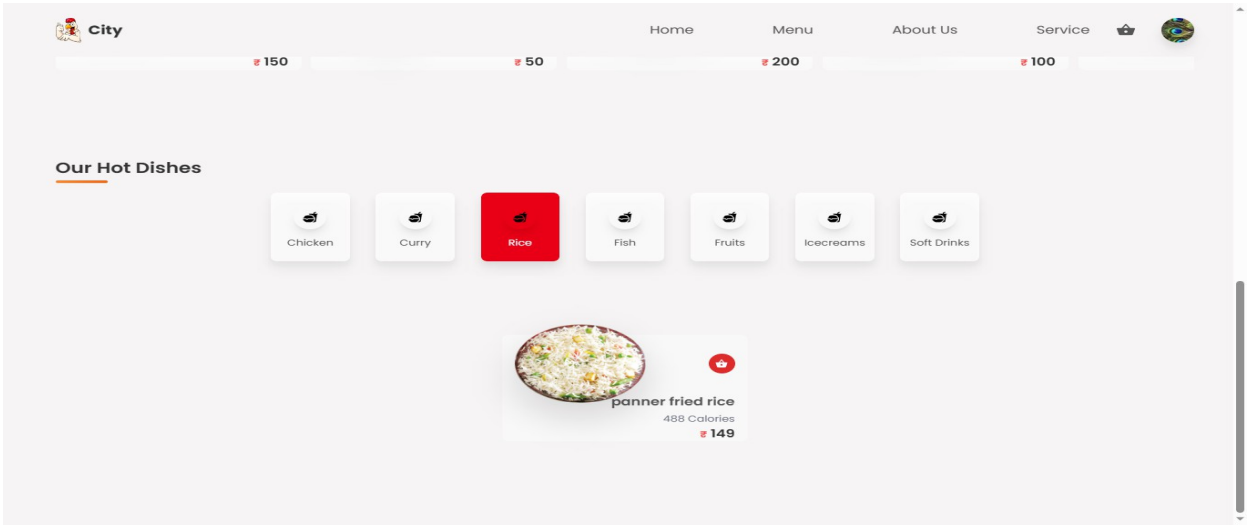
export default CreateContainer;

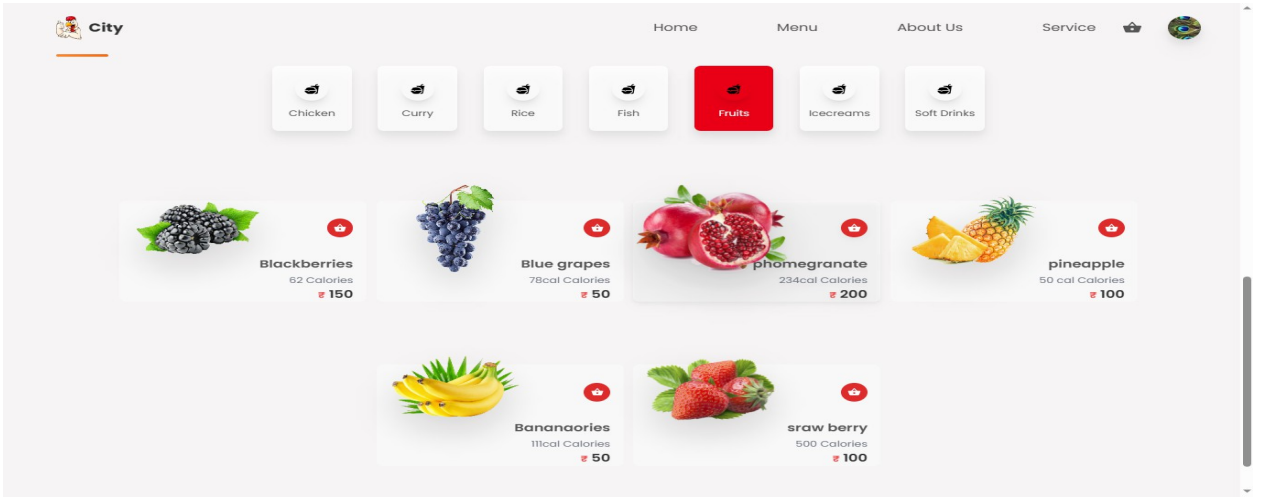
output:

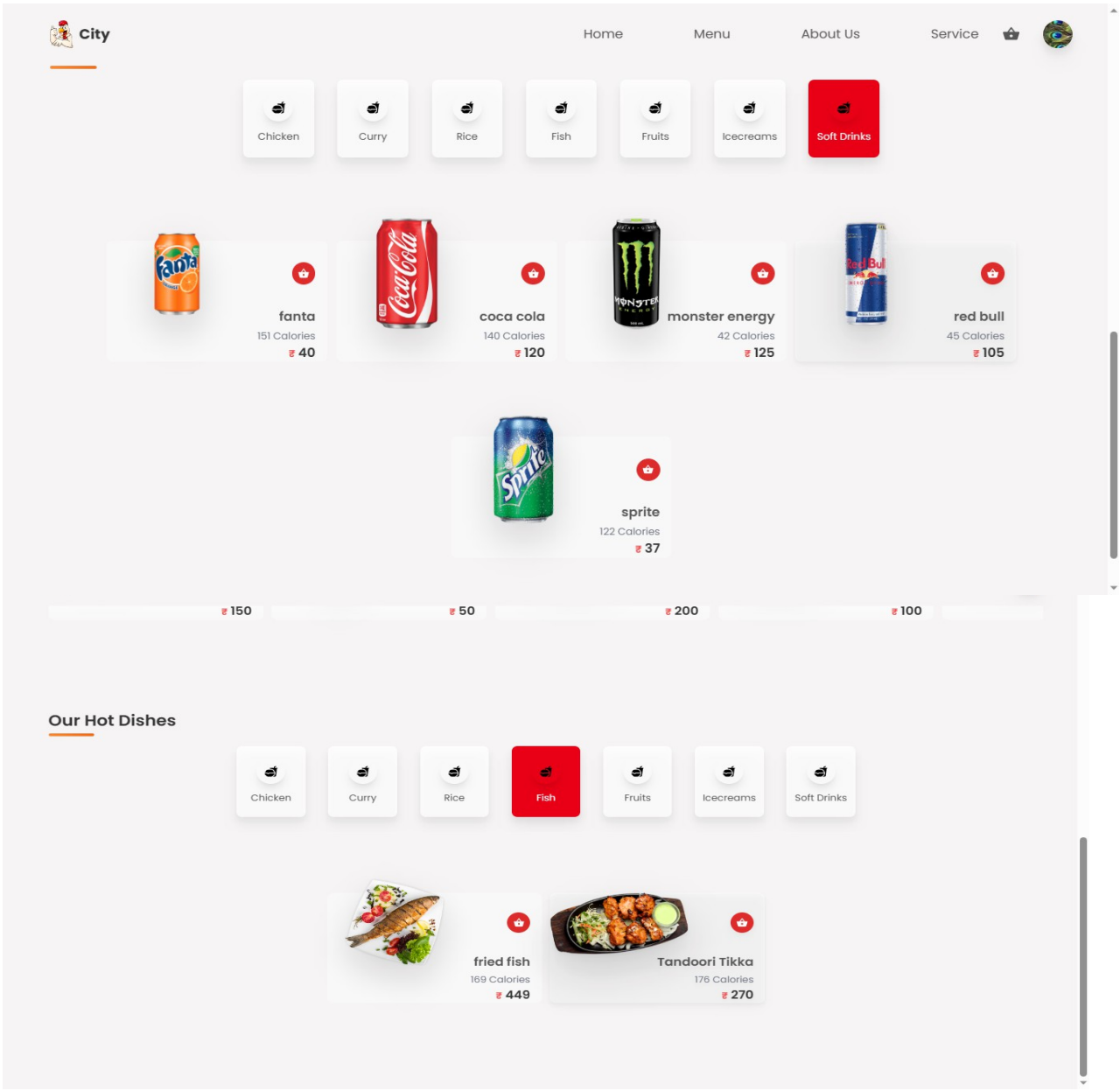


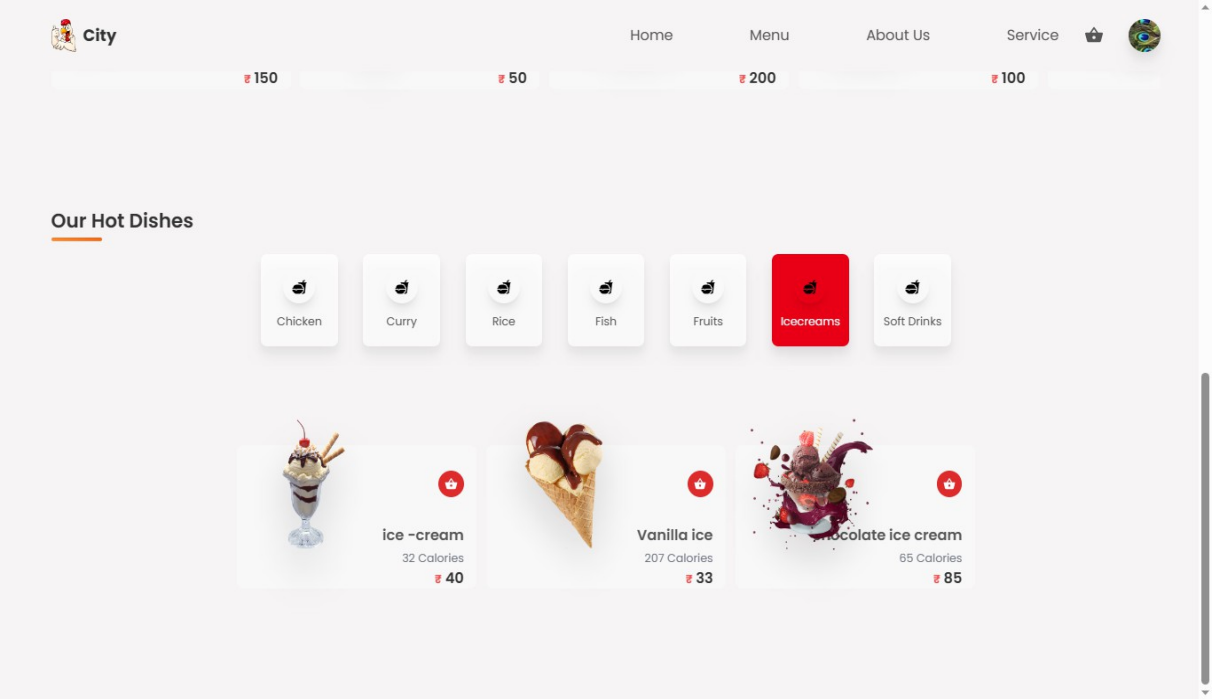



City
Home
Menu
About Us
Service
₹150
₹50
₹200
₹100
Our Hot Dishes
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Soft Drinks
chicken curry
100g Calories
₹120


City
Home
Menu
About Us
Service
₹150
₹50
₹200
₹100
Our Hot Dishes
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Soft Drinks
panner fried rice
488 Calories
₹149


City
Home
Menu
About Us
Service
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Soft Drinks
Blackberries
62 Calories
₹ 150
Blue grapes
78cal Calories
₹ 50
phomegranate
234cal Calories
₹ 200
pineapple
50 cal Calories
₹ 100
Bananaories
111cal Calories
₹ 50
sraw berry
500 Calories
₹ 100


City
Home
Menu
About Us
Service
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Soft Drinks
fanta
151 Calories
₹ 40
coca cola
140 Calories
₹ 120
monster energy
42 Calories
₹ 125
red bull
45 Calories
₹ 105
sprite
122 Calories
₹ 37
₹ 150
₹ 50
₹ 200
₹ 100
Our Hot Dishes
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Soft Drinks
fried fish
169 Calories
₹ 449
Tandoori Tikka
176 Calories
₹ 270



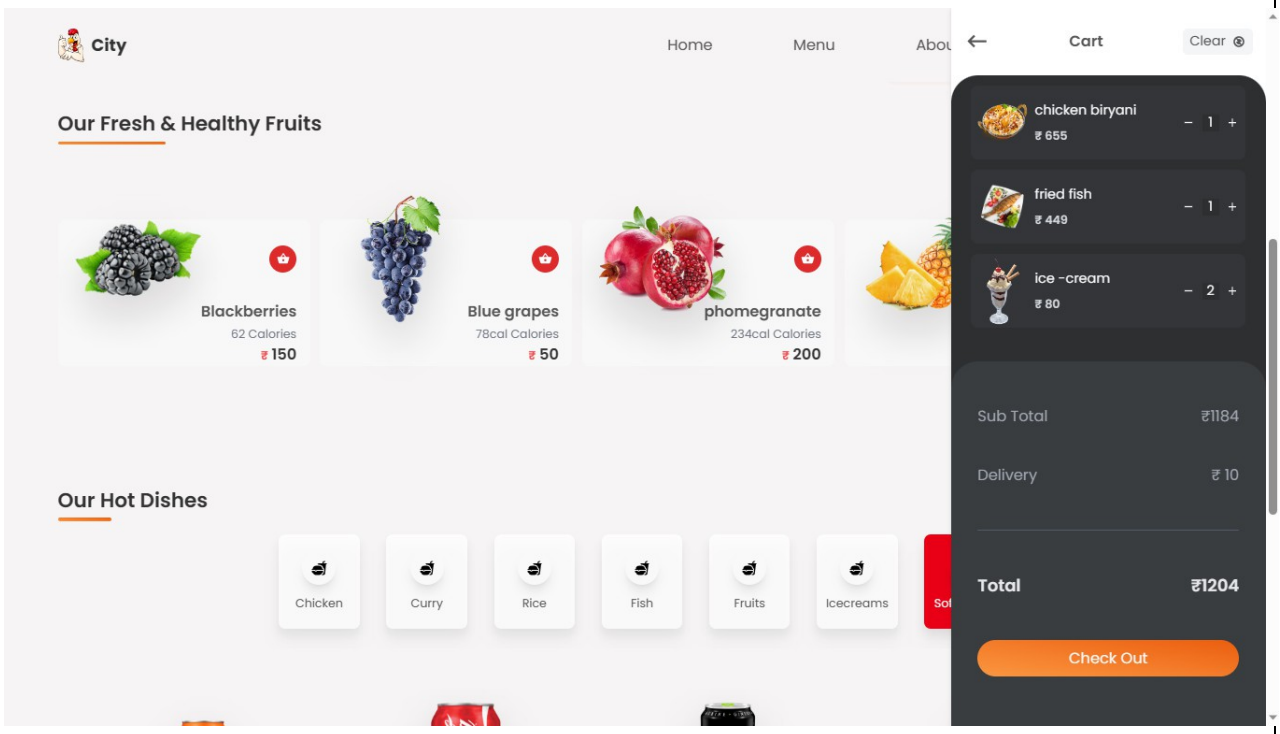
ORDER CONFIRMATION:

```
import React, { useEffect, useState } from "react";
import { BiMinus, BiPlus } from "react-icons/bi";
import { motion } from "framer-motion";
import { useStateValue } from "../context/StateProvider";
import { actionType } from "../context/reducer";
import { fetchCart } from "../utils/fetchLocalStorageData";
let items = [];

const CartItem = ({ item, setFlag, flag }) => {
  const [{ cartItems }, dispatch] = useStateValue();
  const [qty, setQty] = useState(item.qty);

  const cartDispatch = () => {
    localStorage.setItem("cartItems", JSON.stringify(items));
    dispatch({
      type: actionType.SET_CARTITEMS,
      cartItems: items,
    });
  };
```

```
 const updateQty = (action, id) => {
   if (action == "add") {
     setQty(qty + 1);
     cartItems.map((item) => {
       if (item.id === id) {
         item.qty += 1;
         setFlag(flag + 1);
       }
     });
     cartDispatch();
   } else {
     // initial state value is one so you need to check if 1 then remove it
     if (qty == 1) {
       items = cartItems.filter((item) => item.id !== id);
       setFlag(flag + 1);
       cartDispatch();
     } else {
       setQty(qty - 1);
       cartItems.map((item) => {
         if (item.id === id) {
           item.qty -= 1;
           setFlag(flag + 1);
         }
       });
       cartDispatch();
     }
   }
 };

 useEffect(() => {
   items = cartItems;
 }, [qty, items]);

 return (
    <div className="w-full p-1 px-2 rounded-lg bg-cartItem flex items-center
gap-2">
     <img
       src={item?.imageURL}
       className="w-20 h-20 max-w-[60px] rounded-full object-contain"
       alt=""
     />
```

```
      {/* name section */}
      <div className="flex flex-col gap-2">
        <p className="text-base text-gray-50">{item?.title}</p>
        <p className="text-sm block text-gray-300 font-semibold">
        ₹ {parseFloat(item?.price) * qty}
        </p>
      </div>

      {/* button section */}
      <div className="group flex items-center gap-2 ml-auto cursor-pointer">
        <motion.div
          whileTap={{ scale: 0.75 }}
          onClick={() => updateQty("remove", item?.id)}
        >
          <BiMinus className="text-gray-50 " />
        </motion.div>

          <p className="w-5 h-5 rounded-sm bg-cartBg text-gray-50 flex items-
center justify-center">
          {qty}
        </p>

        <motion.div
          whileTap={{ scale: 0.75 }}
          onClick={() => updateQty("add", item?.id)}
        >
          <BiPlus className="text-gray-50 " />
        </motion.div>
      </div>
    </div>
  );
};

export default CartItem;
```

output:


City
Home
Menu
Abou
Our Fresh & Healthy Fruits
Blackberries
62 Calories
₹ 150
Blue grapes
78cal Calories
₹ 50
phomegranate
234cal Calories
₹ 200
Our Hot Dishes
Chicken
Curry
Rice
Fish
Fruits
Icecreams
Cart
Clear
chicken biryani
₹ 655
1
fried fish
₹ 449
1
ice -cream
₹ 80
2
Sub Total
₹1184
Delivery
₹ 10
Total
₹1204
Check Out

Conclusion

Our project is only a humble venture to satisfy the needs to manage their project work. Several user friendly coding have also adopted. . A description of the background and context of the project and its relation to work already done in the area. Made statement of the aims and objectives of the project . The description of Purpose.Scope and applicability we define the problem on which we are working in the project.

We describe the requirement Specifications of the system and the actions that can be done on these things. We understand the problem domain and produce a model of the system, which describes operations that can be performed on the system. • We included features and operations in detail, including screen layouts we designed user interface and security issues related to system. Finally the system is implemented and tested according to test cases.

1. REFERENCE

2. Abhishek Singh, Adithya R, Vaishnav Kanade, Prof. Salma Pathan“ ONLINE FOOD ORDERING SYSTEM” International Research Journal of Engineering and Technology (IRJET) 2018.
3. Serhat Murat Alagoza, Haluk Hekimoglub,” A study on tam: analysis of customer attitudes in online food ordering system”, Elsevier Ltd. 2012.
4. Resham Shinde, Priyanka Thakare, Neha Dhomne, Sushmita Sarkar, ”Design and Implementation of Digital dining in Restaurants using Android”, International Journal of Advance Research in Computer Science and Management Studies 2014.
5. Noor Azah Samsudin, Shamsul Kamal Ahmad Khalid, Mohd Fikry Akmal Mohd Kohar, Zulkifli Senin, Mohd Nor Ihkasan,” A customizable wireless food ordering system with real time customer feedback”, IEEE Symposium on Wireless Technology and Applications(ISWTA) 2011. [7] Serhat Murat Alagoza, Haluk Hekimoglub,” A study on tam: analysis of customer attitudes in online food ordering system”, Elsevier Ltd. 2012.
6. Patel Krishna, Patel Palak, Raj Nirali, Patel Lalit,” Automated Food Ordering System”, International Journal of Engineering Research and Development (IJERD) 2015.
7. Google for problem solving

8. http://www.JSP.net/
9. http://www.tutorialspoint.com/mysql
10. https://www.tutorialspoint.com/java/
11. http://www.JSP.net

**********THANKYOU**********