# **CHAPTER 1: INTRODUCTION**

* 1. **Background**

The ease of access of internet to an individual sparks up the opportunities along with it. Rather than traditional system of ordering food through a phone call can be shifted to more digital approach. In today’s age many restaurants have selected idea of delivering food to the doorstep of their customers.

* 1. **Problem Statement**

In this fast paced world, an individual may not have time to make food by own or even go to a restaurant, our web application solves just that by allowing you to order your food on any device with few clicks.

* 1. **Objectives**

To provide interactive UI to users so they can order any food item available

* 1. **Applications**

Being business to customer, it has endless applications such as Ecommerce,

Trade, Internet banking, Online shopping, Online Auctions.

* 1. **Project Features**

Provides a beautiful UI through which user can navigate easily. User can choose any food item and add it to cart, then the user can checkout his/her order and is followed by a user details form and the order is placed. The order details and the user details is stored into the database. The admin that manages the webiste has the access to the order details and the user details.

* 1. **System Requirement**

Our WebApp can run on any device that supports a web browser like Chrome, Edge, Firefox, Opera, Safari, Torr.

* + 1. **Software Requirement**

Requires a web browser to run.

* + 1. **Hardware Requirement**

Requires sufficient hardware to be able to run a web browser

# **CHAPTER 2: LITERATURE REVIEW**

An ordering system is referred to as a set of detail of methods that is used in handling the order process. For better understanding on the topic of online food ordering system, we need to learn about the usual or traditional way of ordering food. The usual way of ordering depends on the menu cards of paper based. Waiters use paper to write the order of the customers. The records are arranged in paper. As with anything paper based, it is easy for things to get spoiled, and the data can be lost. Hence for systematic way of data storing, digitalization is required.

Self-service or self-ordering in restaurant industry refers to the restaurant taking the orders from customers through applying various types of technologies such as internet and many others. It has been successful for many companies and many online food ordering companies are thriving with it. Worldwide famous food delivery services have names such as Grubhub, DoorDash, Uber Eats, Zomato, Swiggy. In the context of our country Nepal there are several new startups and also some established companies such as Foodmandu, Bhojdeals, Pathao Food, Bhok lagyo,etc.

The first online delivery system was Waiter.com, also formerly known as World Wide Waiter, is an online restaurant delivery service that went online in early December 1995. It was founded by two Standford University Business School graduates Craig Cohen and Michael Adelberg. Waiter.com is considered the first online restaurant delivery service on the web; it pioneered the concept of online restaurant ordering in 1995 when it offered meal options from 60 Silicon Valley partner restaurants, expanding to over 1,300 restaurants in 2017. [1]

Foodmandu is an online food delivery service based in Kathmandu, Nepal. It is the first food delivery service in Nepal... It was founded by Mr. Manohar Adhikari on 11 November 2010. Initially, Foodmandu started delivering food from 10 restaurants and 15 employees. It now employs 500 people. Foodmandu offers service in Kathmandu Valley, Bhaktapur and Pokhara as of 2021. This service is available on Android as well as IOS devices.. As of 2020, it has been making more than 1,000 deliveries a day and has provided employment to more than 250 employees. [2]

# **CHAPTER 3: METHODOLOGY**

**3.1 Introduction**

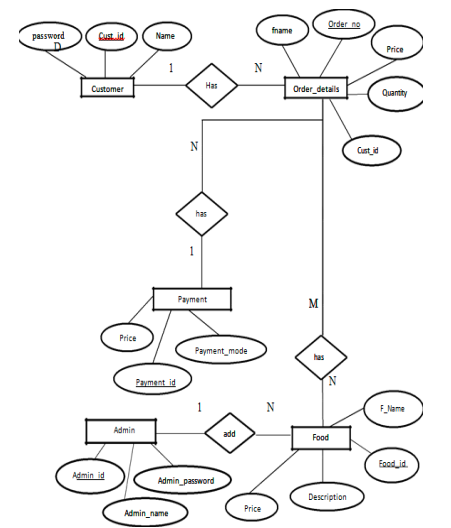
A food ordering system accepts orders from the customers, gets users data, and stores the data of order and user in database for futher delivery process. Our WebApp is able to do these processes step by step.

**3.2 Project Block Diagram**

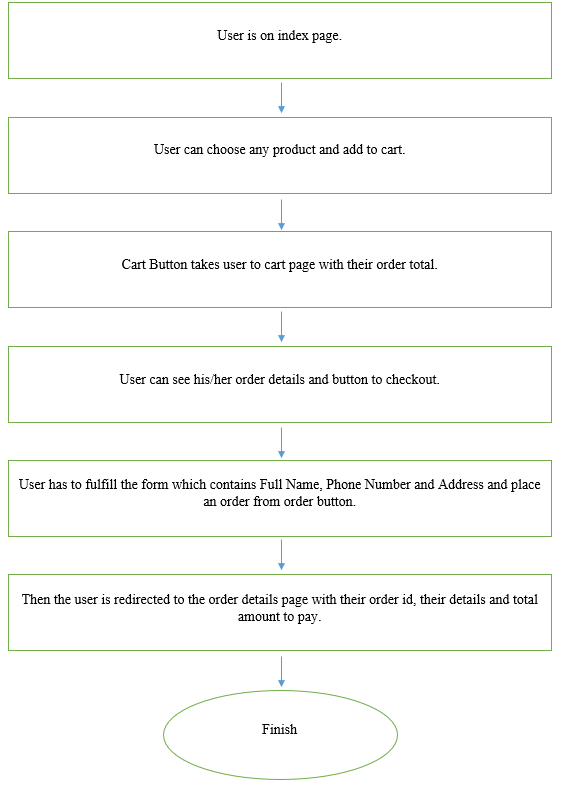
(Block or Flow Chart for CMP)

****

**3.3 System Analysis and Design**



**3.4 Working Principle**



**3.5 Implementation**

Our project is built on React, an open source JavaScript library used to build UIs. For the backend part, it uses Next.js libraries, and for the database, it uses MongoDB (leading global cloud database) and the Cloudinary website to store our product photos. The website has to be dynamic so that the actions performed by the user change the website’s parts, such as items in cart, total amount, and other details.

The functionality of fetching data from the server, i.e., server-side operations, is handled by Next.js libraries and components such as Redux Tool Kit. The Redux Toolkit is a set oftools that help simplify Redux development. It includes utilities for creating and managing Redux stores as well as for writing Redux actions and reducers. [3]

The database used in our web application is MongoDB. MongoDB is a document database used to build highly available and scalable internet applications. With its flexible schema approach, it's popular with development teams using agile methodologies. [4]. SQL databases are used to store structured data, while NoSQL databases like MongoDB are used to save unstructured data. MongoDB is used to save unstructured data in JSON format. MongoDB does not support advanced analytics and joins like SQL databases support. Using the Postman app, we checked the Get and Post methods to insert data into the database in JSON format. We connected our web application using the mongodb URL to our cluster, and using a library such as Mongoose, we connected our webapplication by placing our MONGO\_URL into the .env file.

For the photos of the products available, we opened an account in Cloudinary. The company provides cloud-based image and video management services. It enables users to upload, store, manage, manipulate, and deliver images and videos for websites and apps. All the images are stored there and the links to those images are inside our database, linking our webapp and the image. [5]

**3.6 Problem Faced**

The problems were faced during the development of the web application as React itself was new to us and, with little JavaScript knowledge, we understood some concepts. Next.js uses a different way of handling CSS styles, where we have to include the CSS class as className = {styles.button}. Next.js has a variety of libraries. The Redux toolkit was difficult to learn and also implement on our website, which was a hassle to say the least. The Redux Toolkit uses Redux Store, Redux Dispatch Action, async and other stuff, which was quite confusing and not beginner-friendly at all. Npm and Yarn two package managers were really confusing at first too, the two being totally different from one another.

 Connecting to our MongoDB database required a new package 'mongoose’ and the connection was made using the documentation of next.js on GitHub.

**CHAPTER 4: RESULT AND ANALYSIS**

In order to digitalized the ordering process of food, this project was oriented upon, with the help of present Web technologies it can be come true. User can order any item and then while entering his/her details, his details are stored into the database and the user can see that the order is being prepared, on the way, delivered. We also implemented an Admin Page which has access to control on the customer orders and the food items that is shown on the page. In order to access to the Admin page, one has to authenicate by entering username and password, only if it matches with the credentials that is in eht env file of our project, then only one is given access to the admin page, otherwise not. Admin can also add a new item and provide all the details about the item and then it is saved on the MongoDB and the image is rendered from a cloud image site knowns as Cloudinary.

Our web application is capable to do just that, and with the comments added in the day of mid-defense about the Admin page, Single Page Implementation, Sending Email to the customer are now added into the app for better User experience and better functionality. Overall we have a web application that is capable of getting orders from users, admin functionality for dynamic changes to the website, Single page implementation i.e. page doesn’t need to load for moving from one to another page.

**CHAPTER 5: CONCLUSION**

To summarize our project, web technologies has been advanced than the previous ASP and JSP pages, with the help of JavaScript libraries such as React, NodeJS and others, it has been a great way to make a website dynamic as well as User friendly in terms of UI/UX. Our project is Responsive Food Ordering Website which just does what it in the name, let users to order food from the website and is dynamic with Admin page, serverside rendering, database on MongoDB and is on Single Page with the help of React.

Being similar to ecommerce website, if this project were to be extended in the near future, it can be modified to any ecommerce website as it has those functionalites already made and can be scaled if required.

# **REFERENCES**

# [1]"Waiter.com - Wikipedia", *En.wikipedia.org*, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Waiter.com. [Accessed: 21- Sep- 2022].

[2]"Foodmandu", *EverybodyWiki Bios & Wiki*, 2022. [Online]. Available: https://en.everybodywiki.com/Foodmandu. [Accessed: 21- Sep- 2022].

[3]"Redux Toolkit | Redux Toolkit", *Redux-toolkit.js.org*, 2022. [Online]. Available: https://redux-toolkit.js.org/. [Accessed: 21- Sep- 2022].

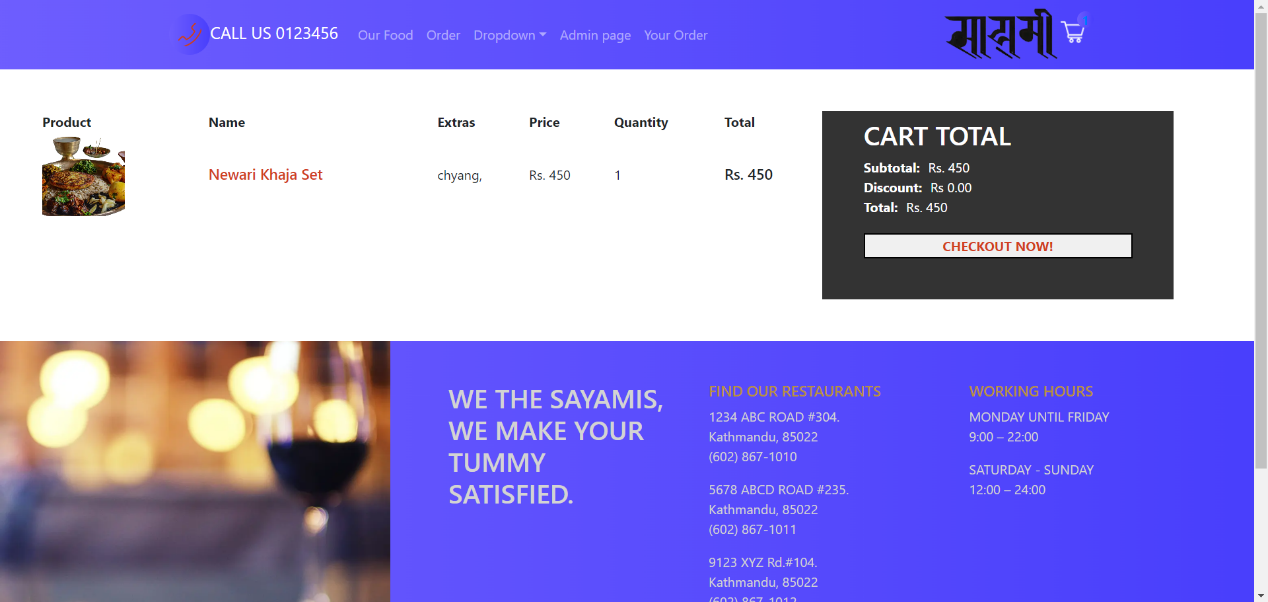
[4]D. (Preview)→ et al., "MongoDB: The Developer Data Platform | MongoDB", *MongoDB*, 2022. [Online]. Available: https://www.mongodb.com/. [Accessed: 21- Sep- 2022].

[5]"Cloudinary - Wikipedia", *En.wikipedia.org*, 2022. [Online]. Available: https://en.wikipedia.org/wiki/Cloudinary. [Accessed: 21- Sep- 2022].

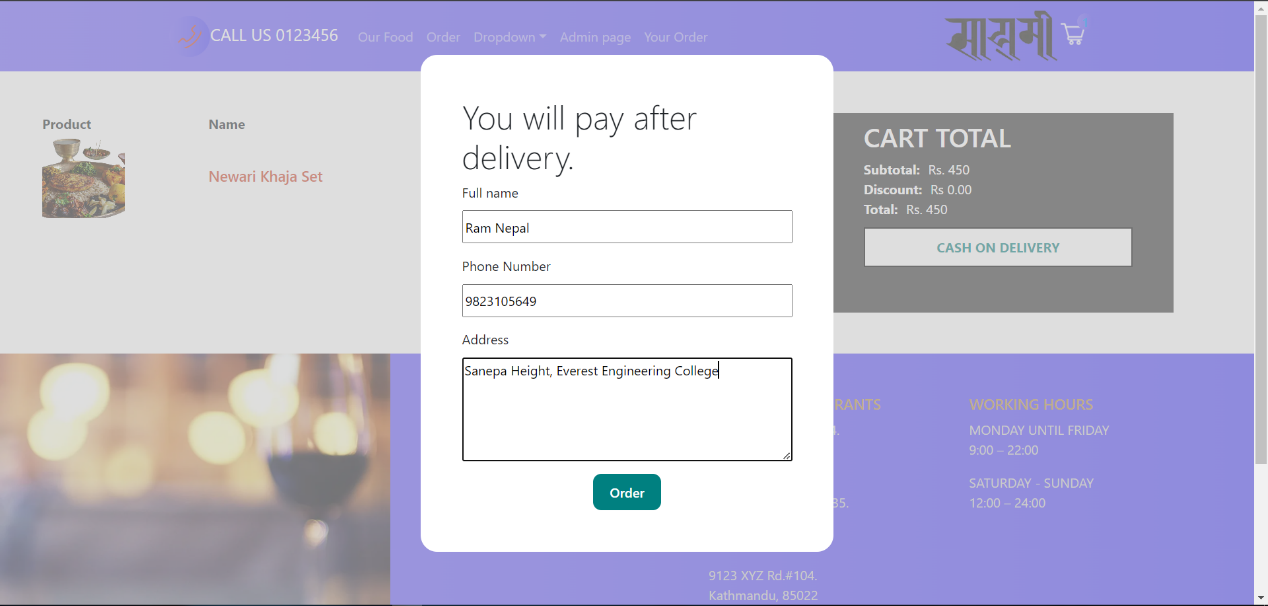
# **APPENDICES**

# Fig 1. Index Page

# Fig 2. Product Details Page



# Fig 3. Cart Page

 Fig. 4 User Details Form

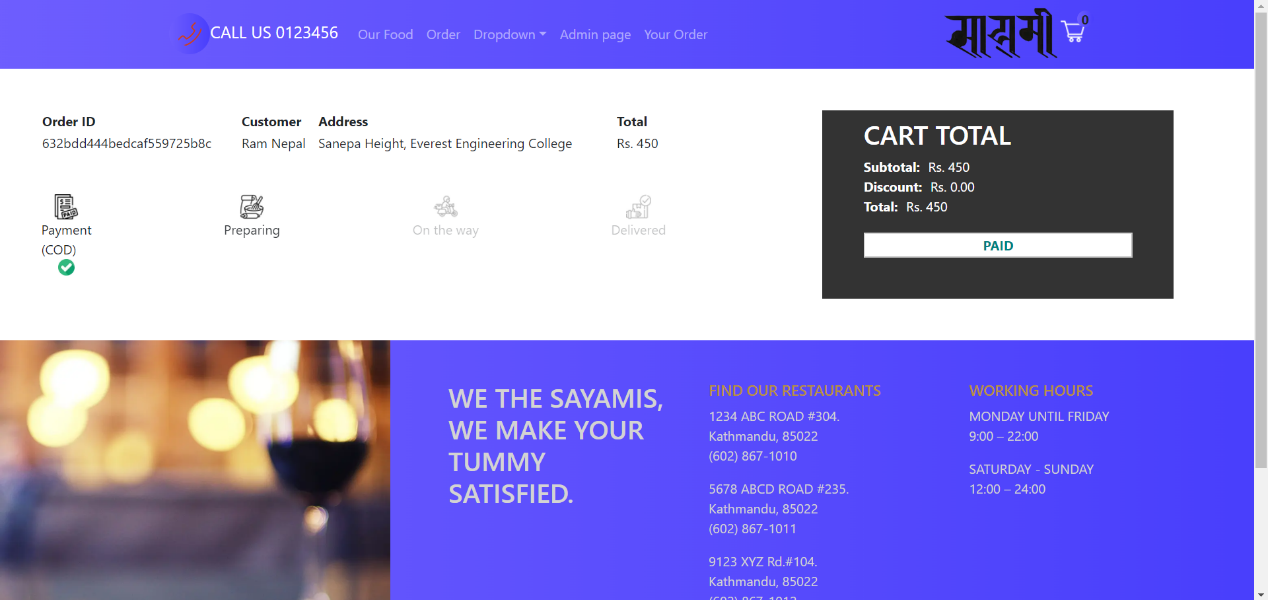


Fig.5 Order Details Page

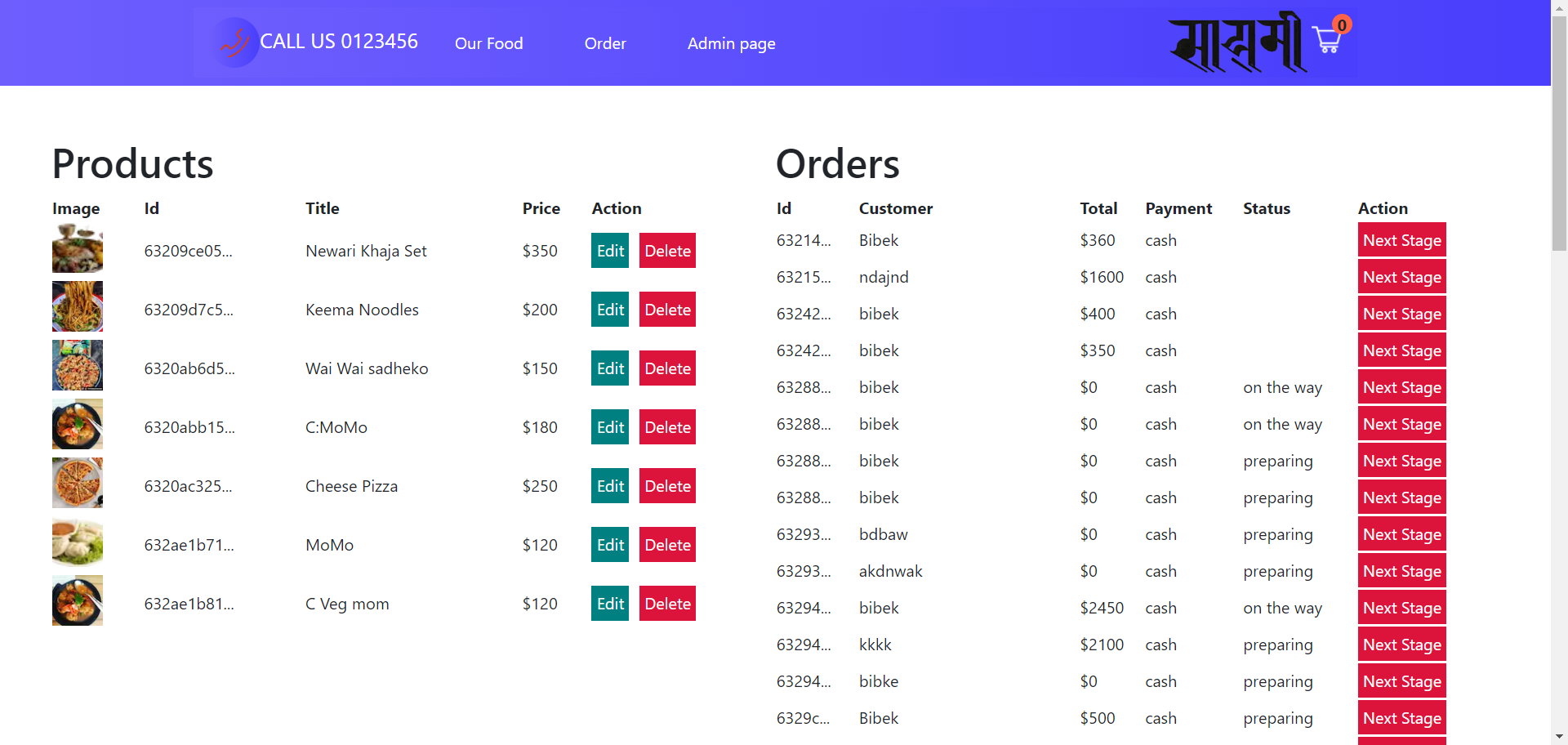


Fig.6 Admin Page