INDEX

|  |  |  |  |
| --- | --- | --- | --- |
| S.R | Title | No | Sign |
|  |  |  |  |
| 1 | Introduction | 2 |  |
| 2 | Technology used | 3 |  |
| 3 | Objectives | 4 |  |
| 4 | System Flow Chart | 5 |  |
| 5 | DFD / UML Diagram | 9 |  |
| 6 | ERD | 19 |  |
| 7 | Screen Shots | 21 |  |
| 8 | References | 29 |  |

Introduction

This project is aimed at developing a Web application that depicts Food order and purchasing the Food Online.

Using this software, Hotels can improve the efficiency of their services to providing their food to the customer. This web application involves all the features of the latest Food order websites .

This website divided into two modules

1.Client side

2.Admin side

Technology used

* Frontend
* React JS (Frontend library)
* Css (Style the webpages)
* Backend
* Express (Backend Framework)
  + - Node JS
    - Mongo DB (Database)
* Other Tools

* Mongoose
* Connect with mongodb and also run the CRUD opration with database.
* Nodemon
* When we save any js file automatically reflact on the server without run any command like node index.js.
* Morgan
* Get status code of every get and post request.
* Concurrently
* To run frontend and backend server simultaneously.
* Redux
* For State management.

Objective

Objective is to provide the website to the hotel which unable to easy to order food from online for the users or customers.

This website is reduce the manual operation and maintain all records of order information on the database.

Main concept of this website is to maintain the orders reports and to maintain customer records and also generates the various reports for analysis for the admin.

System flow Chart

Flow Chart of Admin side

Update food

Insert food

View food

Admin

No

Login

Yes

View User

View food

Delete food

View all User Order

Logout

Flow Chart of Client side

Home page

Add Cart

Update cart

Delete cart

Register

No

Login

Yes

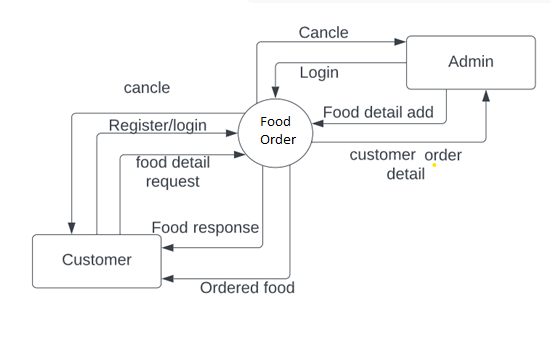
Order Food

View Ordered Food

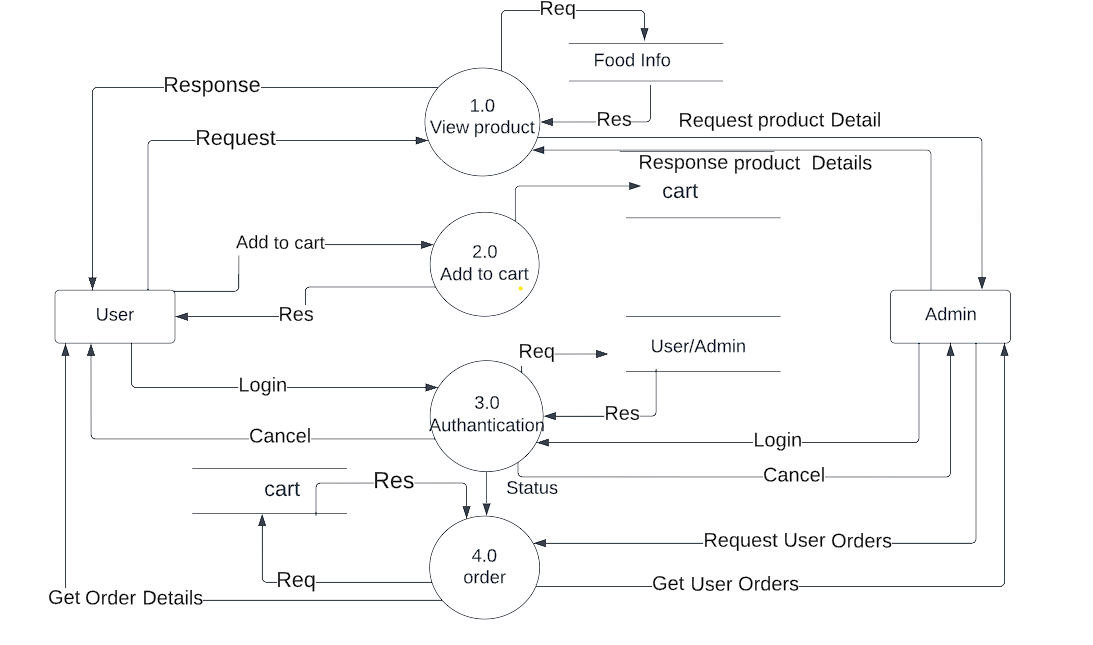
Log out

DFD

0 Level for food order

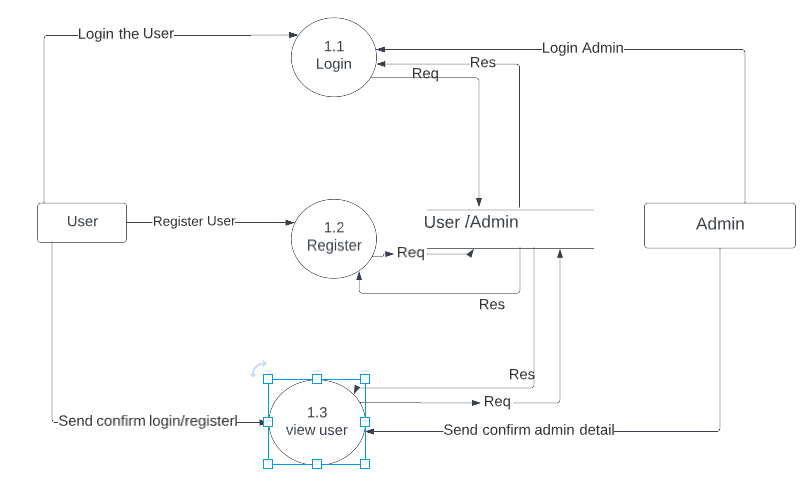


1st level dfd for admin & client

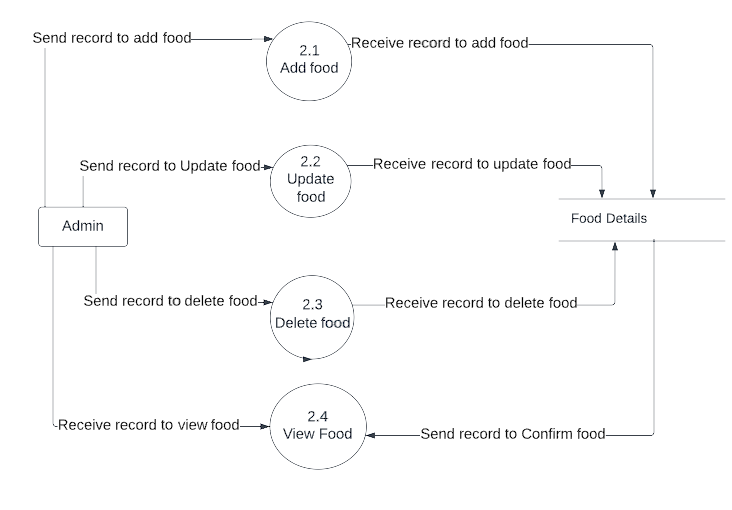


2nd level dfd User & Client

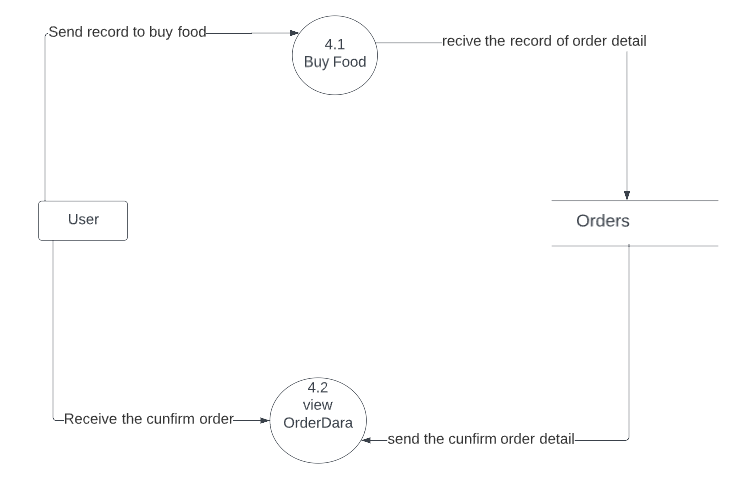
1.0 registration & login for admin & User



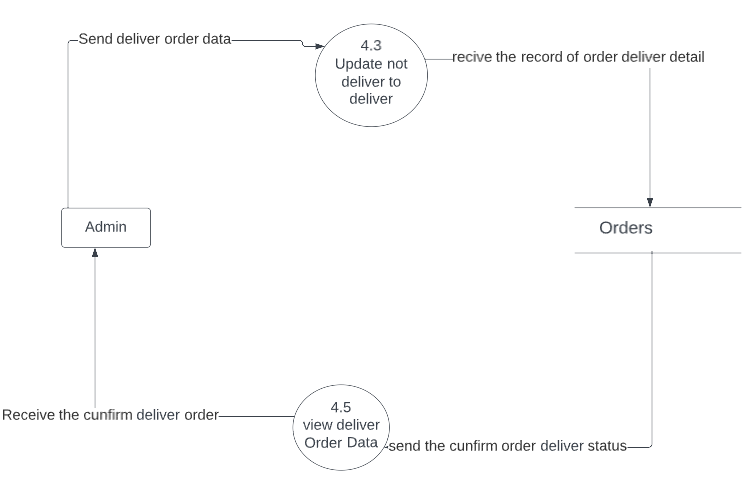
2.0 View food from admin Side



4.0 Order process from client side

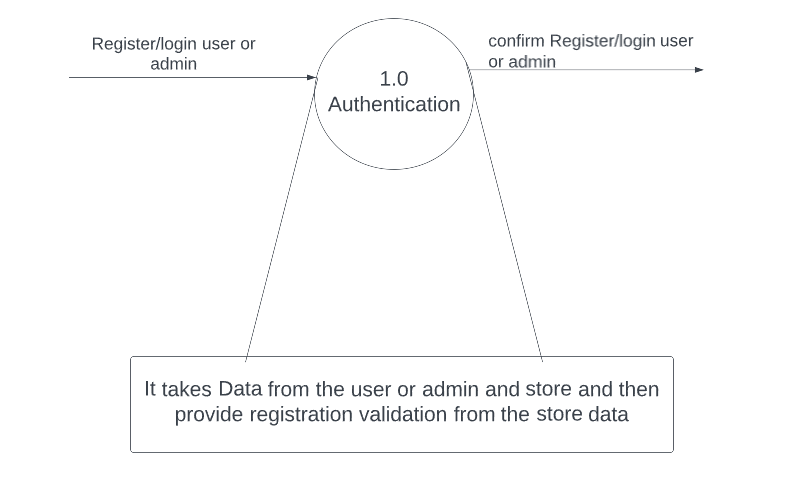


4.0 Order process from Admin side

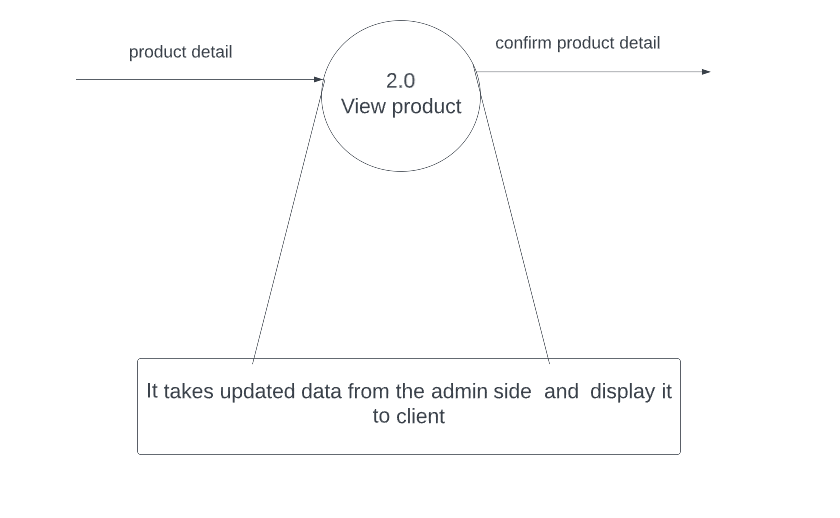


Process specification

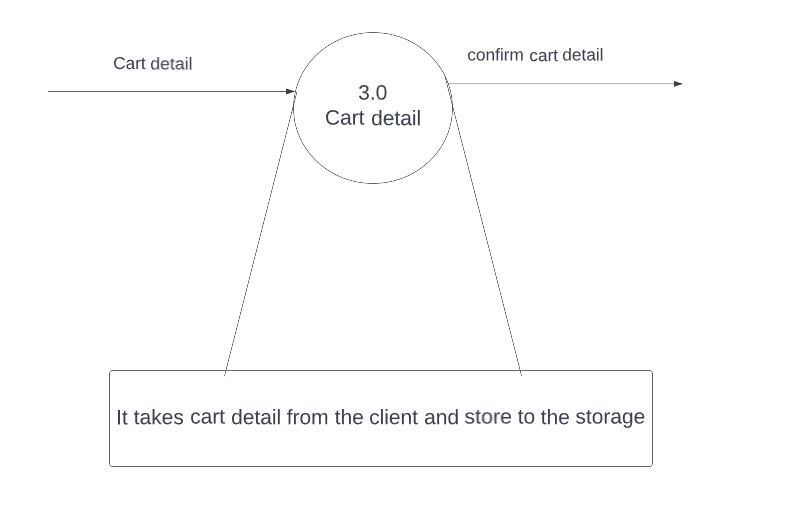
1. 0 Authentication process



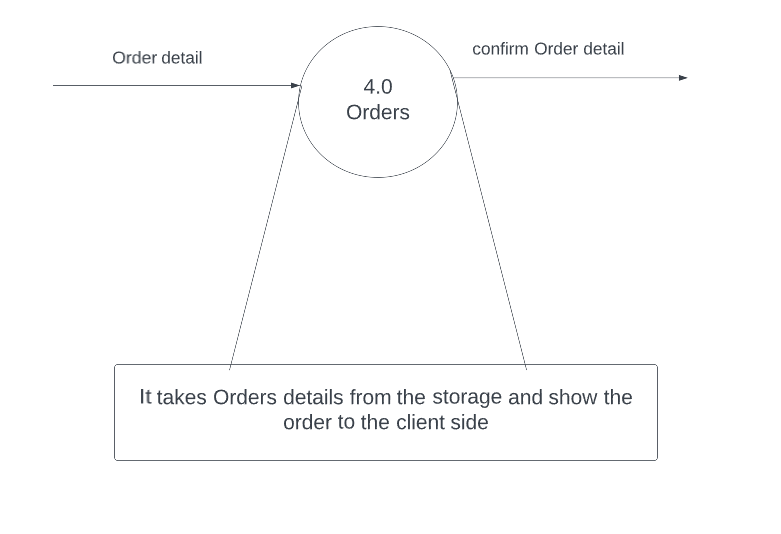
2.0 View product process



3.0 Add to cart process



4.0 Orders process



Database

Structure of the Database

1.User

{

\_id : int32,

name : string,

email : string,

pass : string

isAdmin : boolean

}

2.Foods

{

\_id : int32,

name : string,

varientes : array,

prices : array,

category : string,

image : string

}

3.Orders

{

\_id : int32,

name : string,

email : string,

userid : string,

orderItem : array,

shippingAddress : {

address : string,

pin : string,

city : string

},

orderAmount : string,

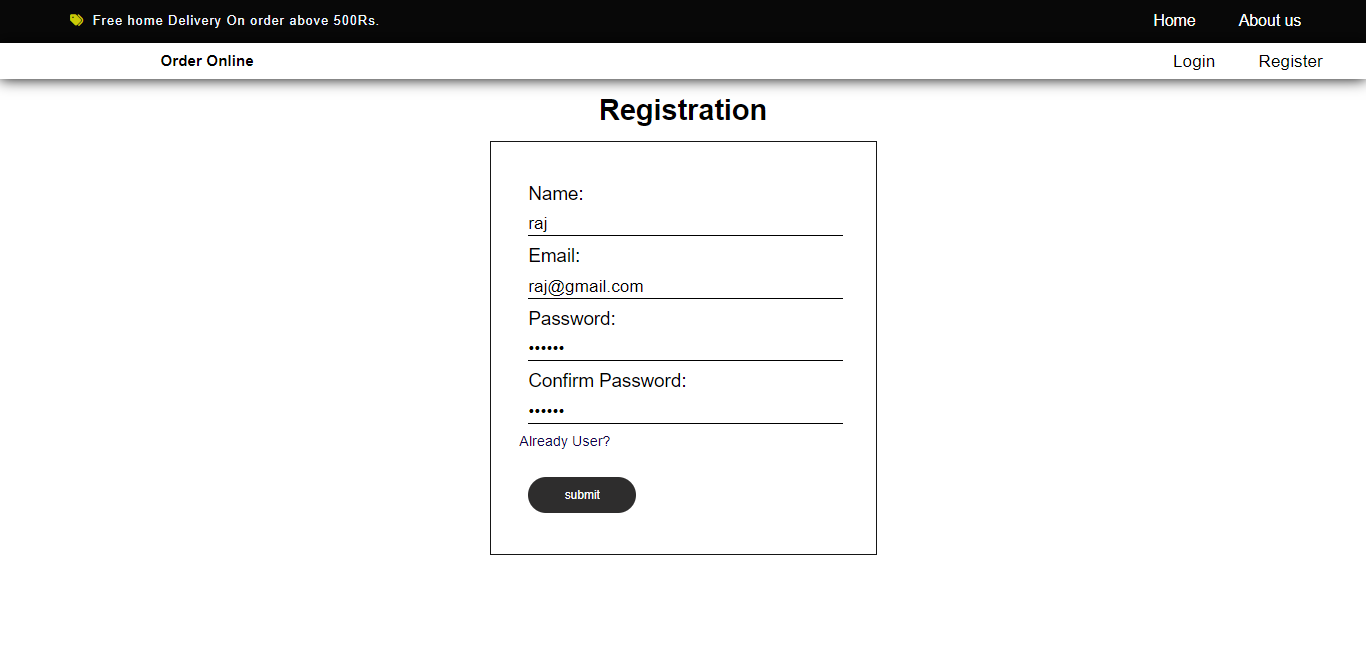
isDelivered : boolean

}

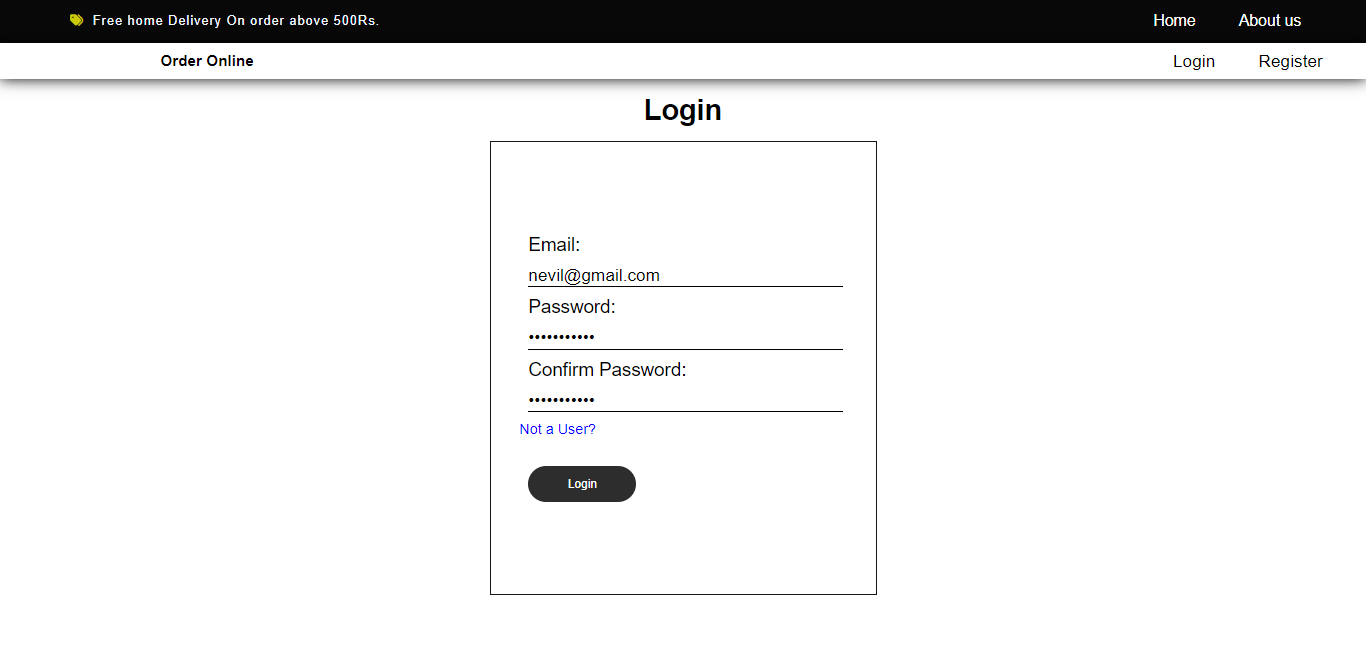
Note : shippingAddress is Object

Screenshots

1.Registration page

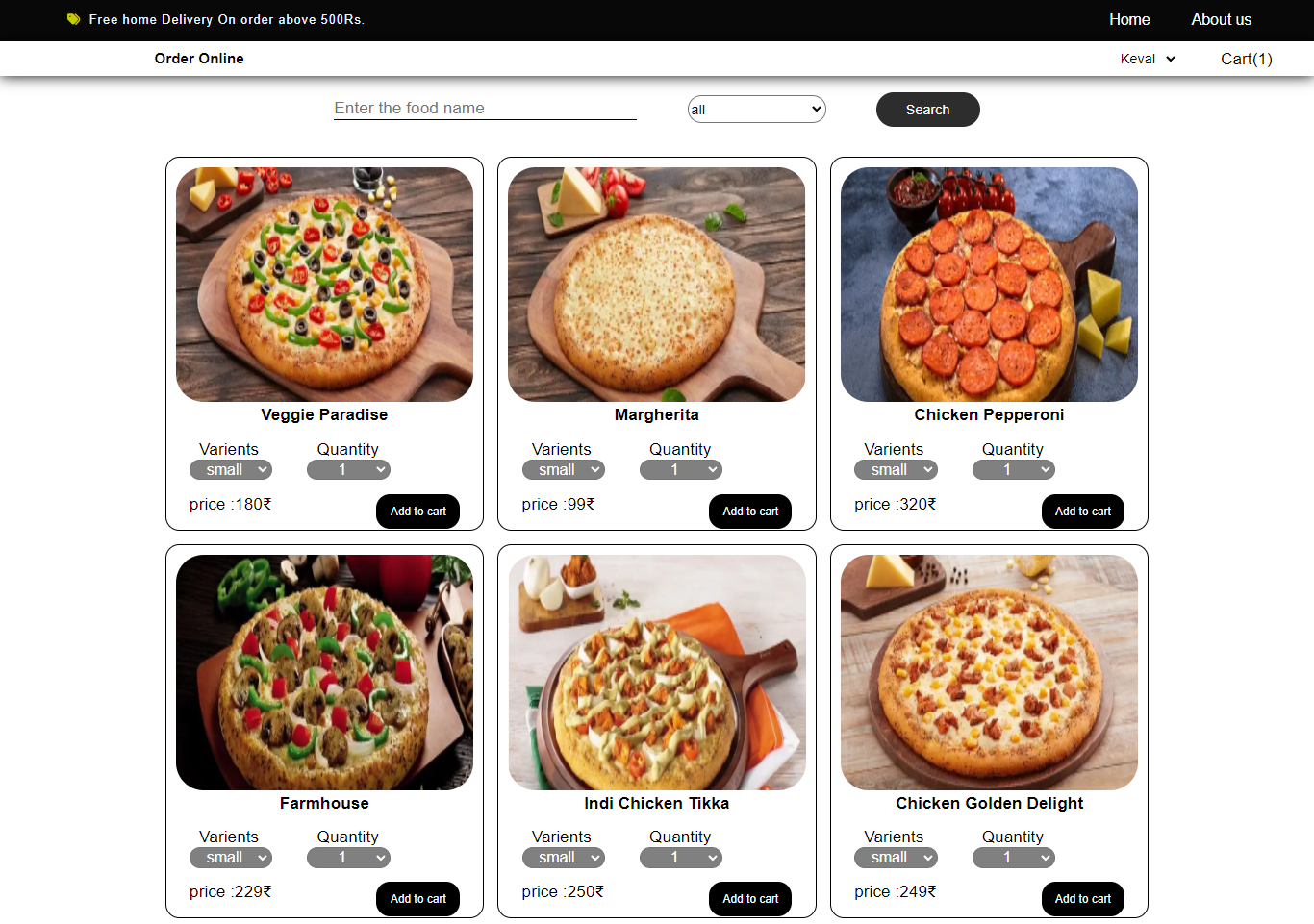


1. Login page

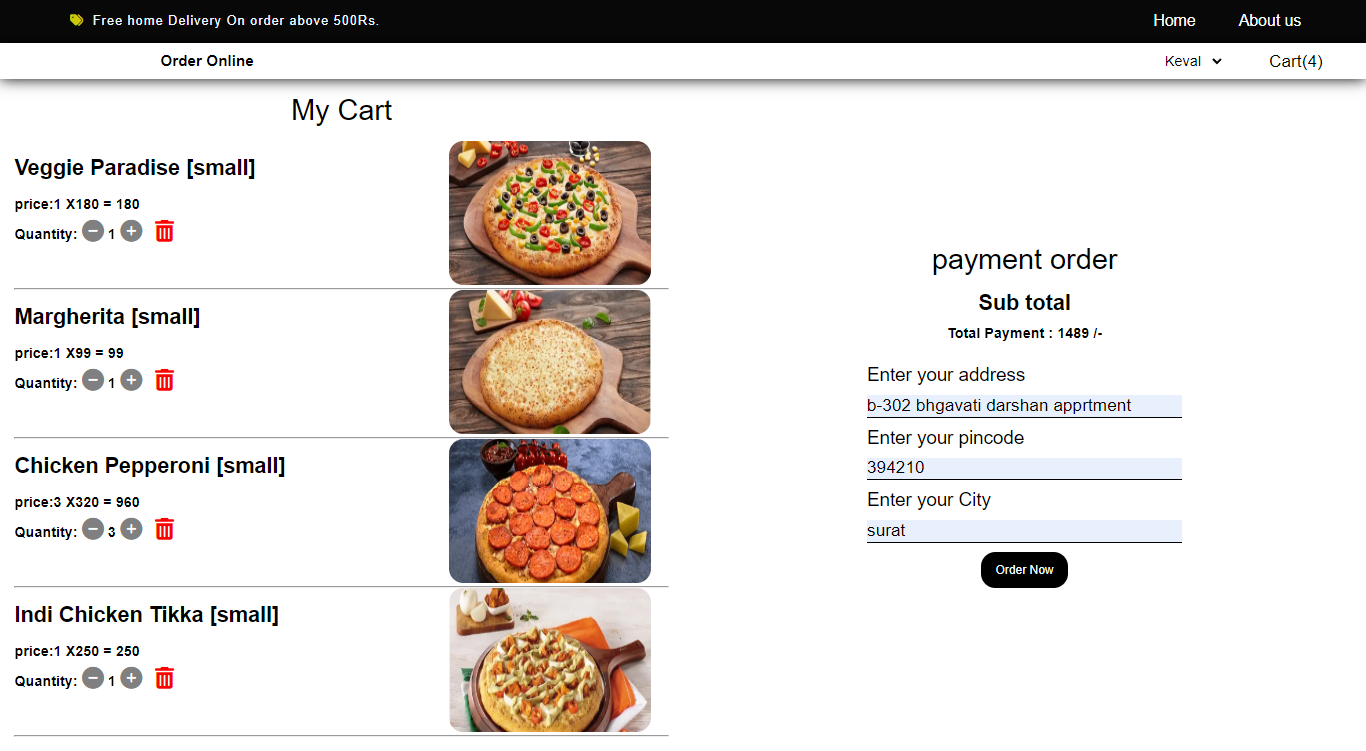


Client/User

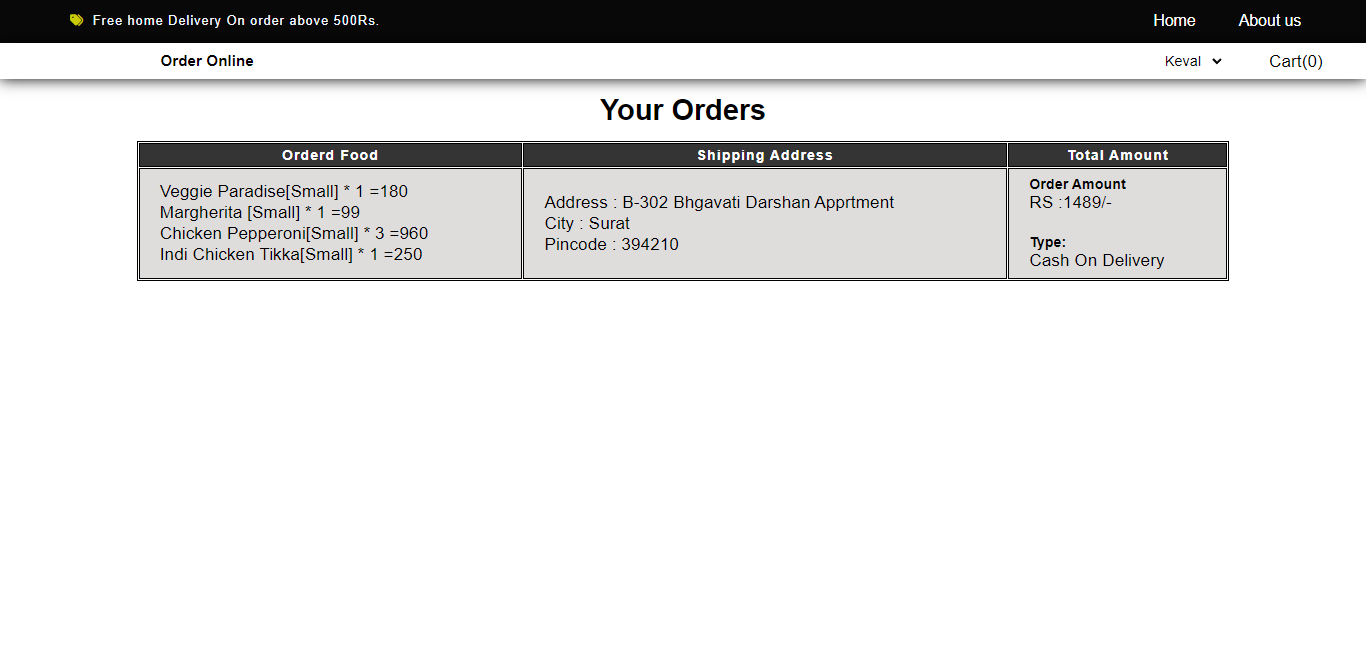
3.Home page



4. Add to cart page



5.Order Details

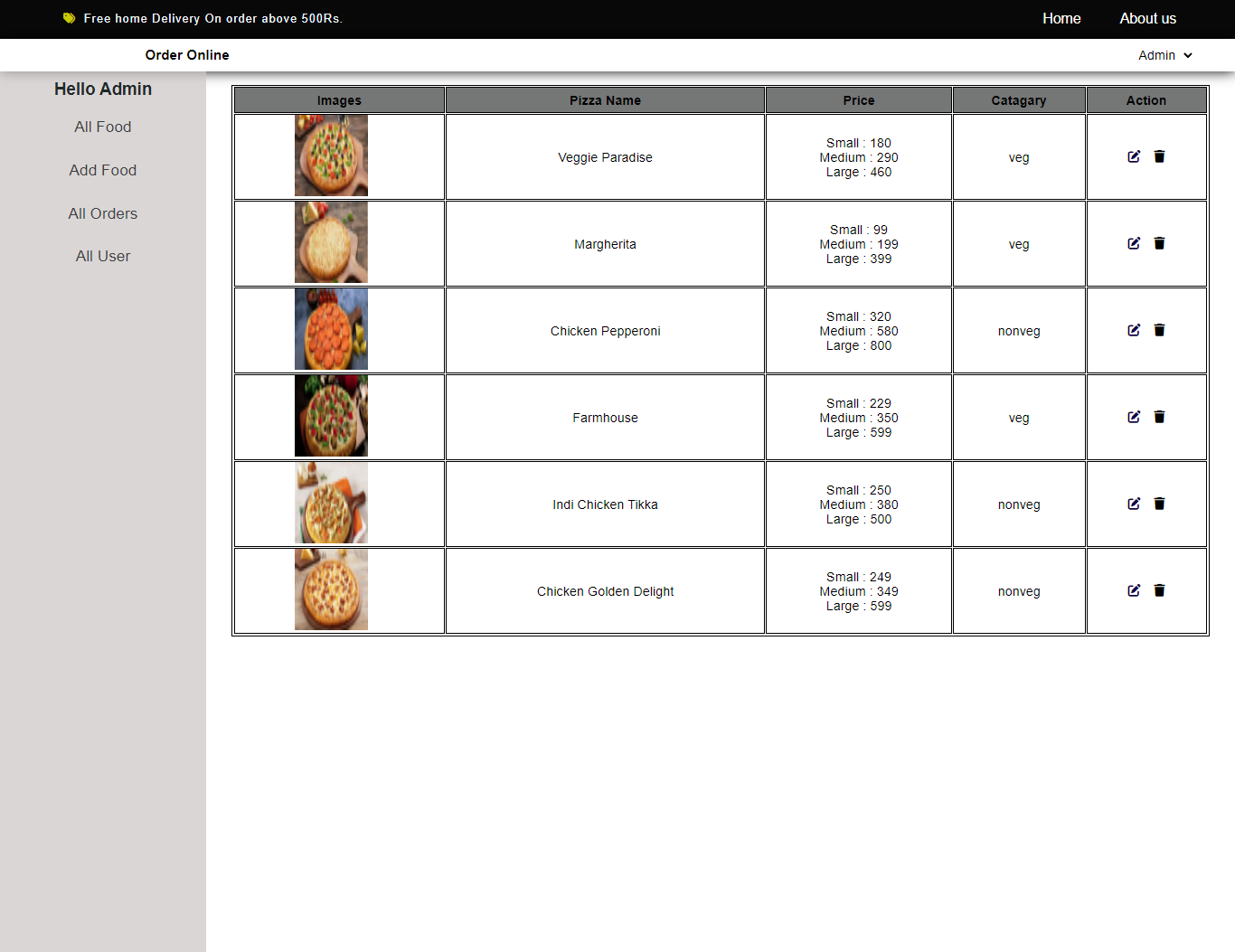


6. About us page

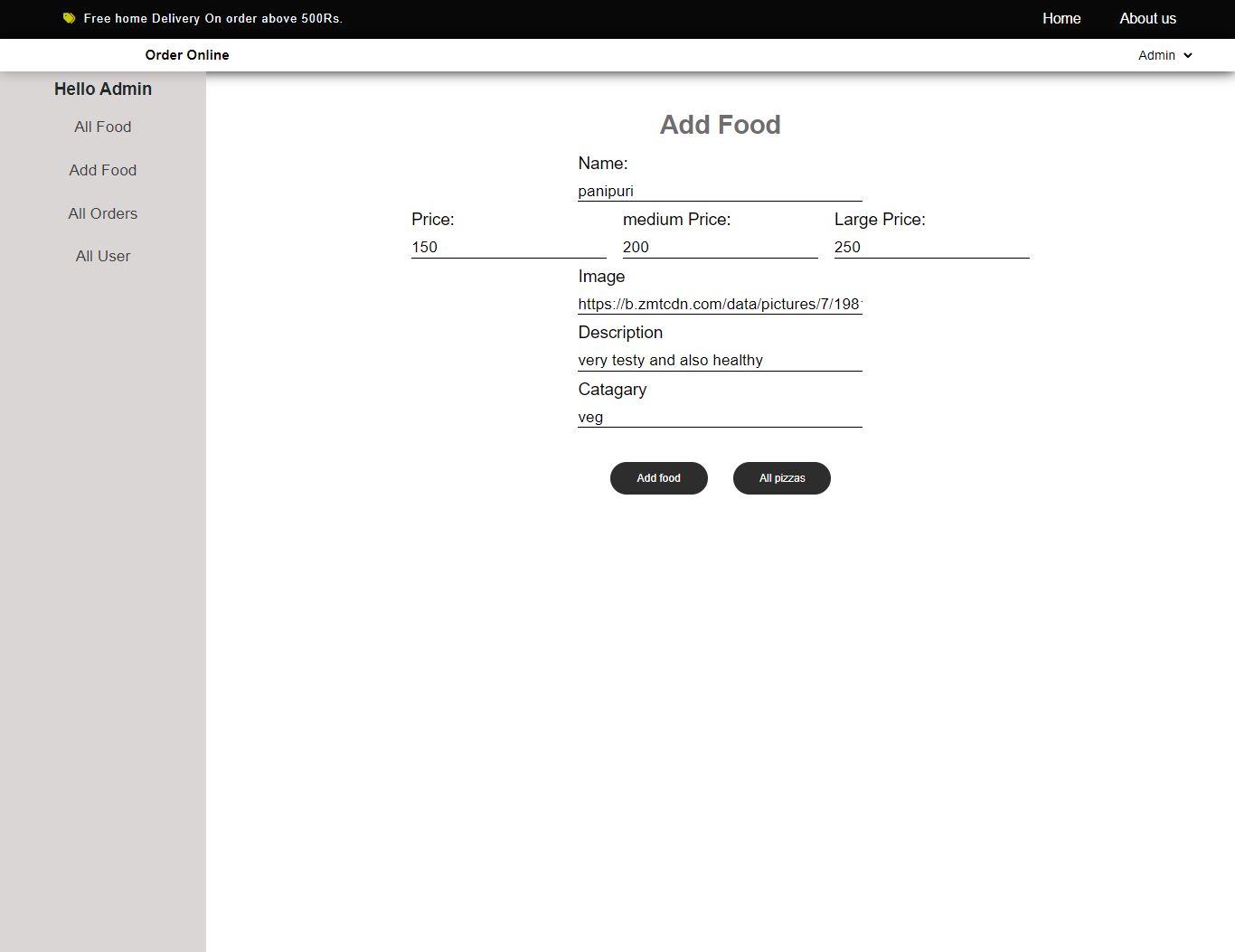


Admin

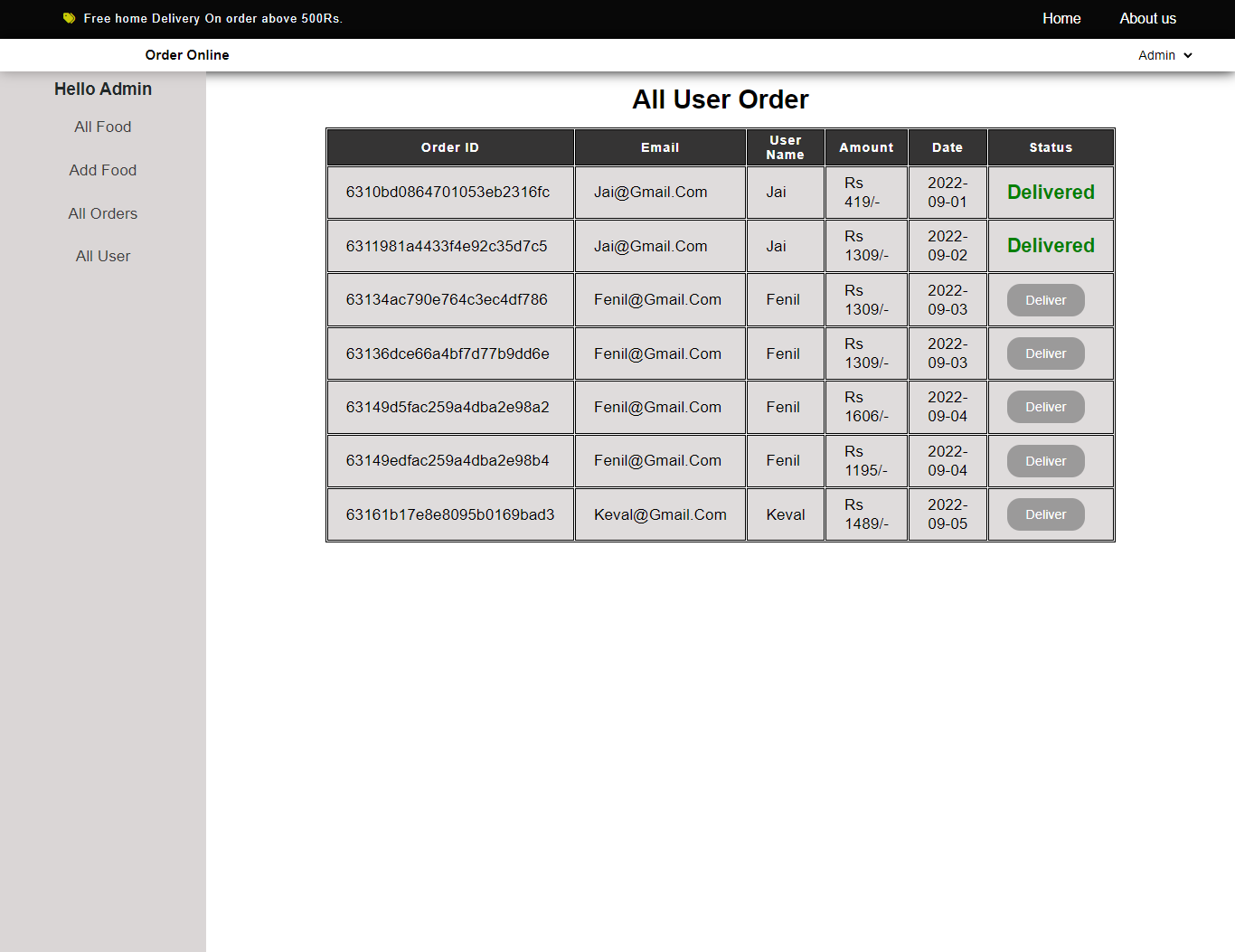
7. All foods



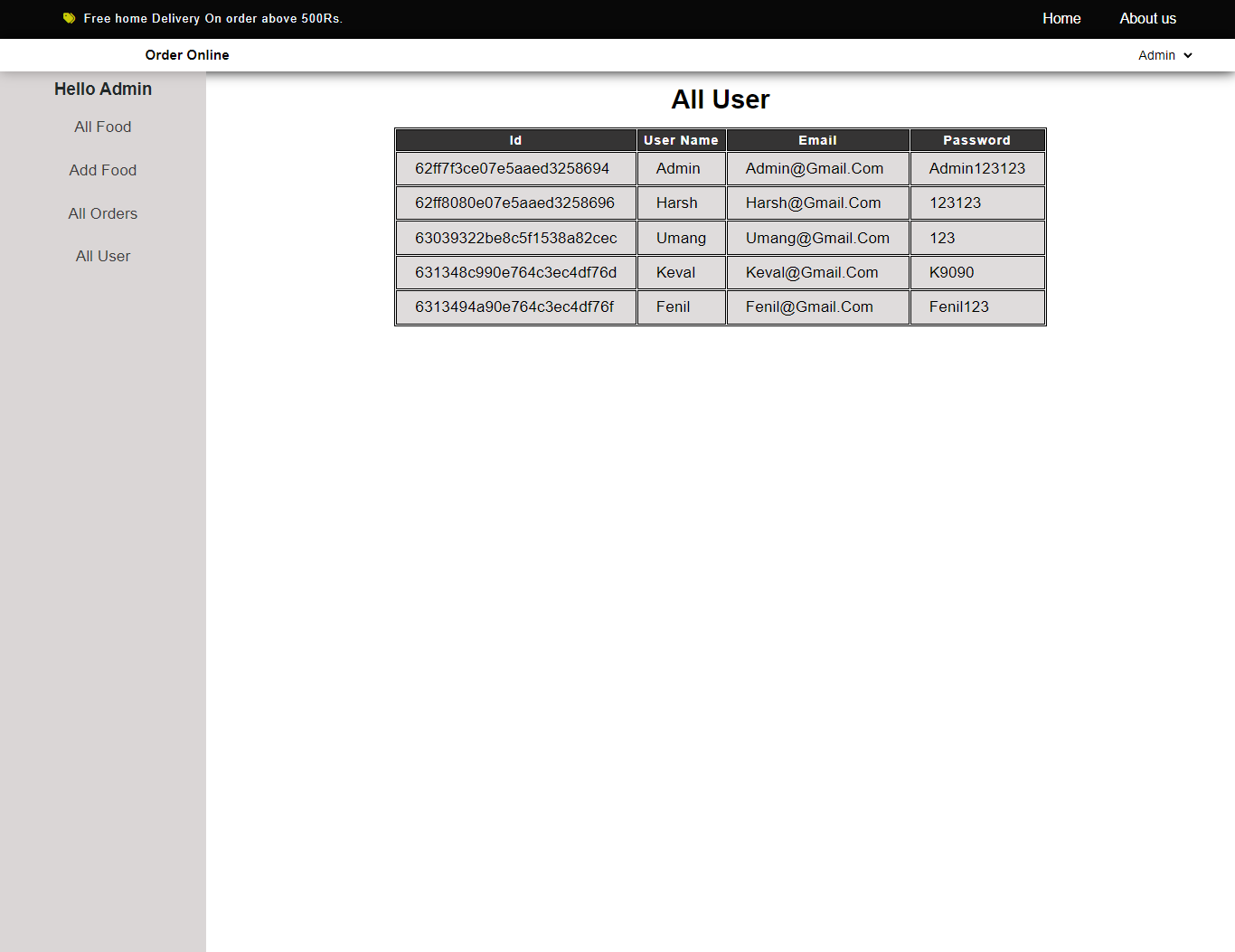
8. Add food



9 . All User Order Status



10. All User



References

1. React.org (for react)

2. <https://nodejs.org/en/docs/> (for node js)

3. <https://mongoosejs.com/docs/models.html> (For creating schema and connection with database and perform the opration with database)

4. <https://expressjs.com/en/4x/api.html#express.router> (For api request in express)

5. techinfoyt youtube channel (For Database connection)

6. <https://react-redux.js.org/tutorials/quick-start> (for redux state mangement tools)