SE3020 – Distributed Systems

Assignment 02 – REST API

2022S1_REG_80

Student Name	Registration number
IT20207236	Balasooriya B.M.D.D
IT20190798	Rathnayaka R.M.S.Y
IT20491994	Dilshani M.G.S.U
IT20905422	Perera K.S.C

Contents

Introduction _

“Sani Agro” is online collaborative agri products store where customers can buy agri products through the system. Main advantage of this system is to provide much easy way for customers to buy agri products without having to physically attend and provide easy way to manage the system for the farmers.

This system has two user types which are **buyers** and **farmers**. A buyer can register to the system and login to the account. After that they can search the products and add to the cart as they wish. These products will be collected to the another page which calculate the total price and total number of products. Then buyers can click on the proceed to checkout button and navigate to the payment page. They can choose preferred payment method. There are two payment methods are available in this system. Those are **pay with credit cards** and **pay by mobile bill**.

After the payment, buyers can fill the delivery information forms and they can request the selected products shipped to their designated addresses. As well as system will be sent a payment confirmation via email to the buyer.

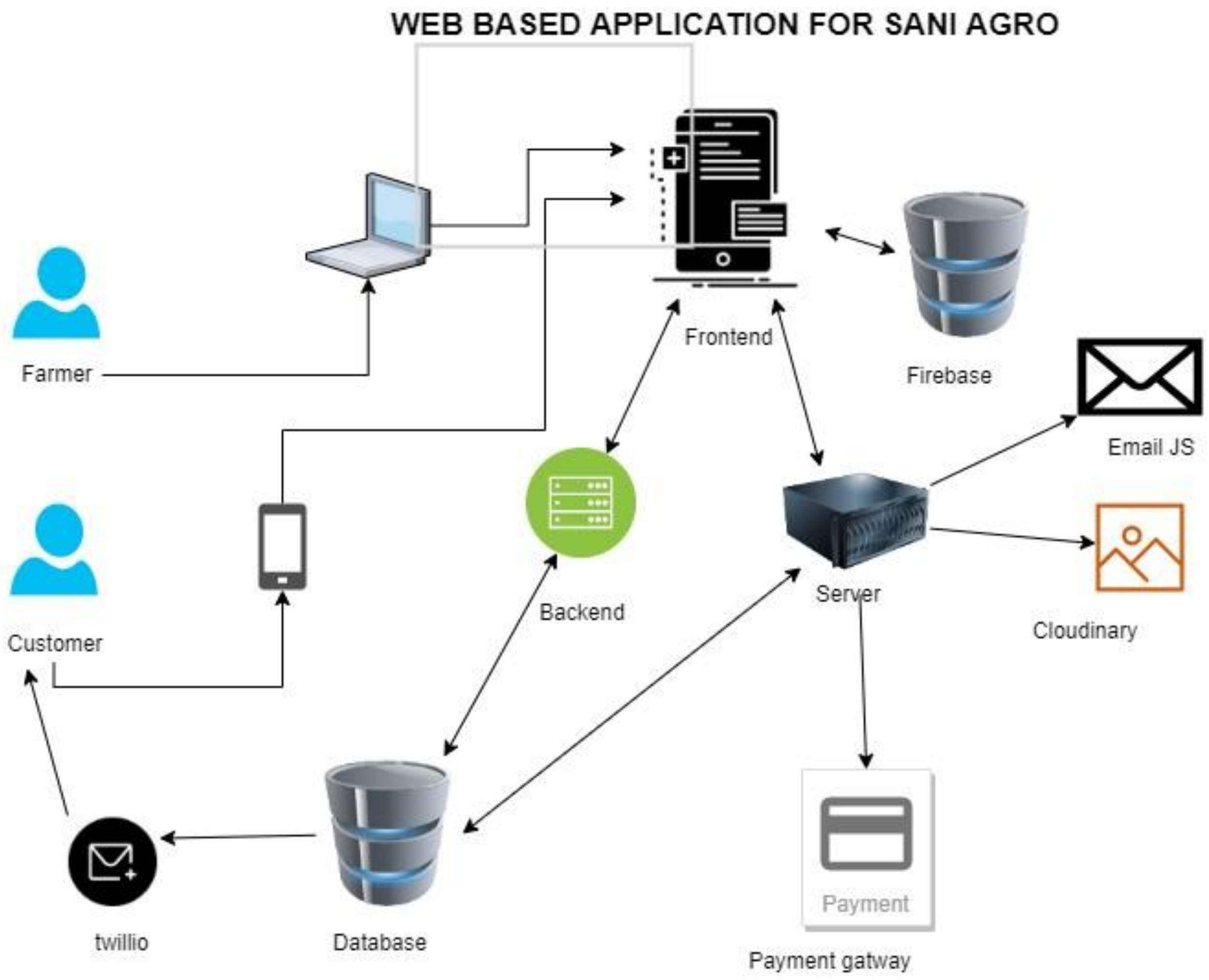
The farmers of the system can register to the system and login to their account. After that, they can view all the products and do the basic crud operations. These are adding new products, delete products, retrieve products from the database and update products.

Technologies

Following are the technologies that are used to implement this system.

- Backend – Node and Express js
- Frontend – React js
- Database – Mongodb

<u>High Level Architectural Diagram</u>



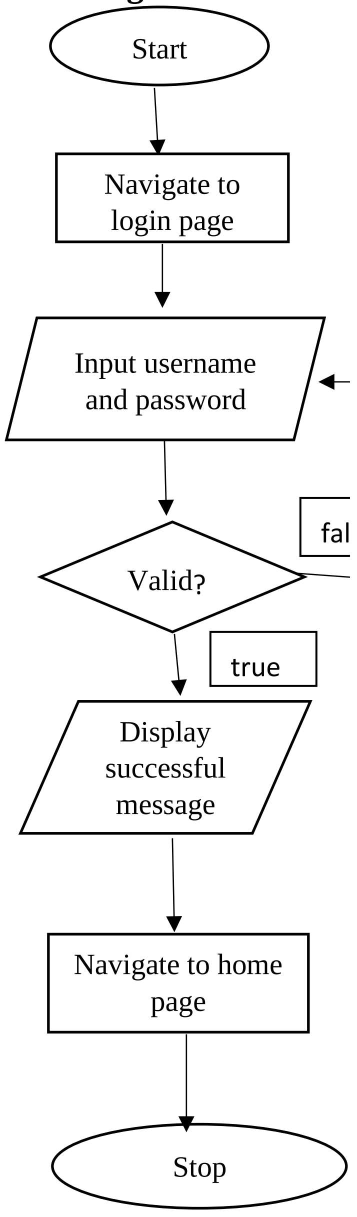
Interfaces and Interconnectivity

Service Interfaces

- Product service
- Cart service
- Delivery service
- Payment service

Main workflow of the system

1. User Login



When user entered invalid username and password system shows an error message. Then they must reenter the credentials.

If login is successful system displays login successfully message.

If they are valid, they can continue relevant workload related to them.

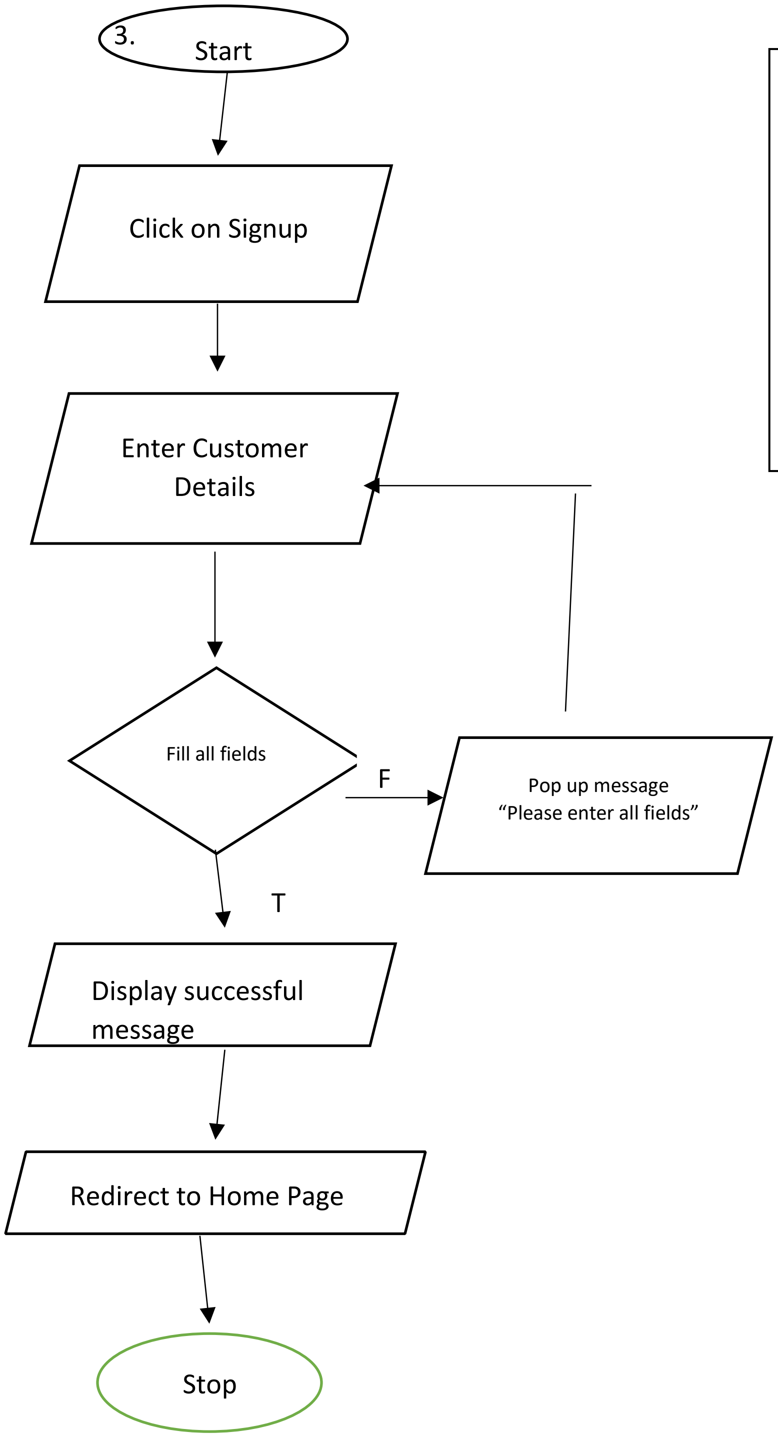
2. User Registration

This is User Registration.

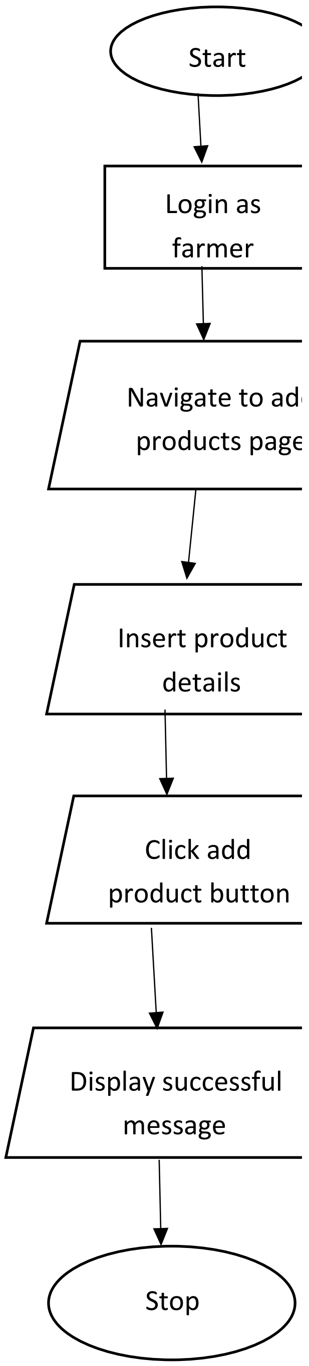
There are two main roles, • Farmers • Customers

When the registration user must give valid email and the password

According to the user role user can do a difference task in the system.



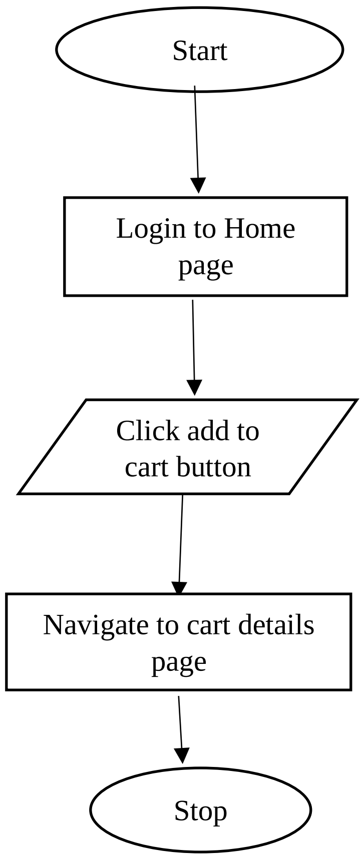
3. Upload the products



This function is only done by the farmers. They can add details and images of the product.

Then data will send to the database. Successful message will be displayed.

4. Add to cart



When customer login to the system, they can add multiple products to the cart. Through this function, customers can easily buy the products. They need to click on ‘add to cart’ button on the product card and then that product will added to the cart.

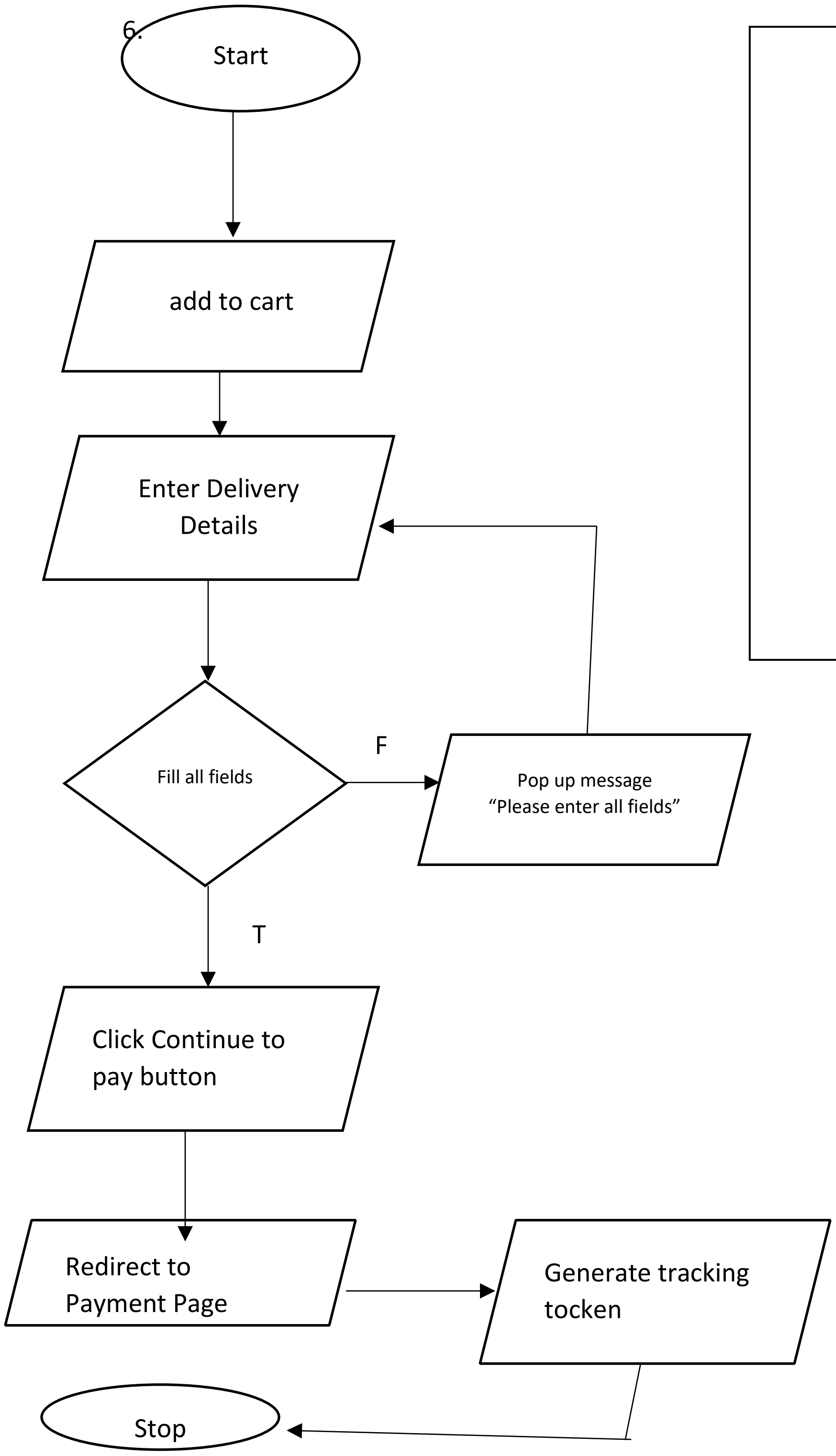
In the navigation bar there is a cart icon. After adding the product to the cart, customers can click on that cart icon and navigate to the cart details page.

If they want to delete the added products, they can click the delete button in the cart details page. And they can increase or decrease the quantity of the products. After total price of all the products are appeared in the side.

There are two types of making payment.

- Using credit card.
- Using mobile bill.

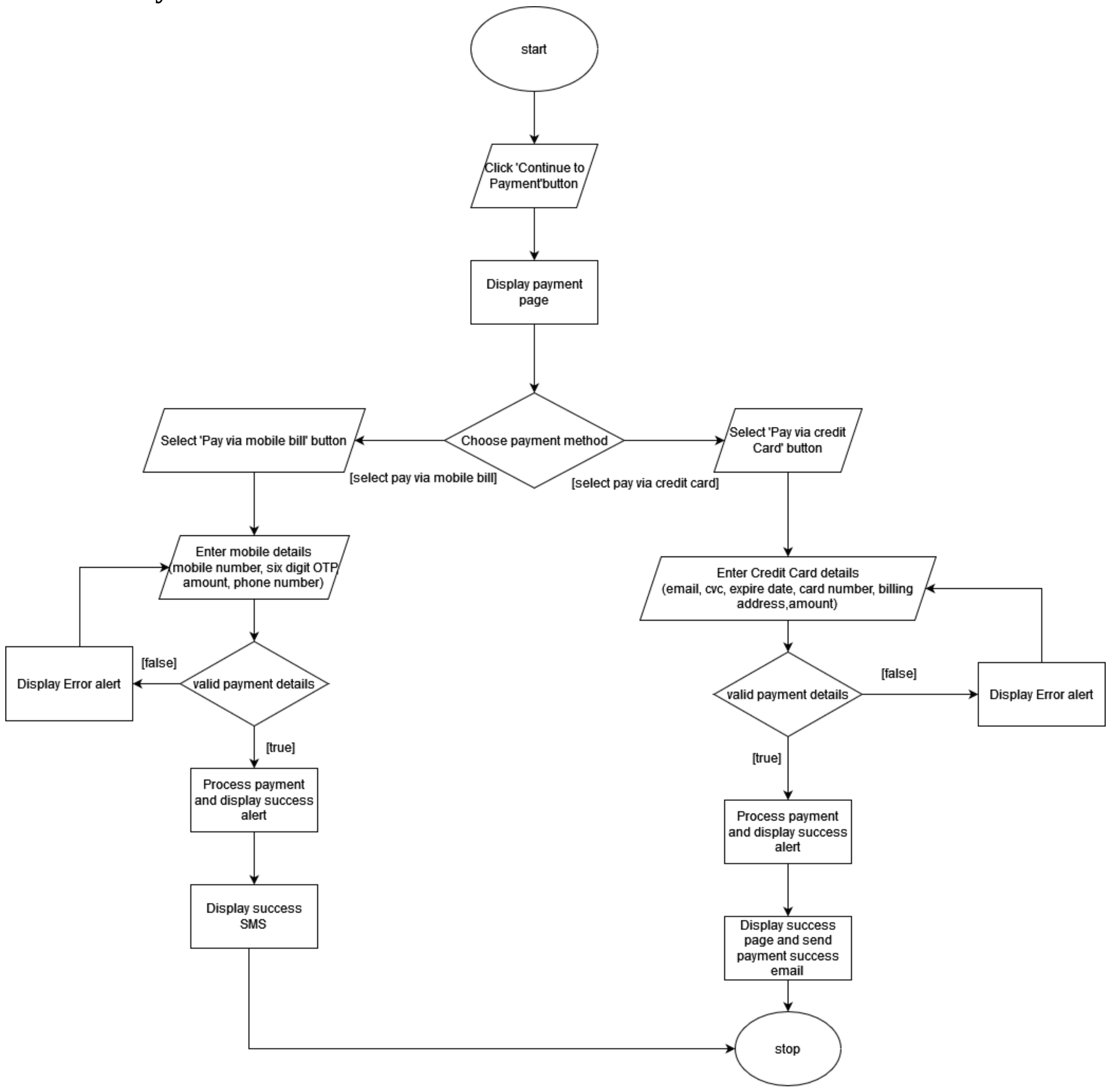
5. Delivery



At the end of the add to cart process, System will navigate to the delivery details page. User should enter the Full name, phone number, address. This form divided address few sections such as building number, street, city, and province. By submitting the form will redirect to the payment page.

These data save to the database and finally get the data from it and generate report of delivery details it will sent to the third-party delivery service and each user can get tracking token end of the delivery. It will also be generated as pdf which include user delivery details, issue date and tracking id

6. Payment


start
Click 'Continue to Payment'button
Display payment page
Choose payment method
[select pay via mobile bill]
Select 'Pay via mobile bill' button
[select pay via credit card]
Select 'Pay via credit Card' button
Enter mobile details (mobile number, six digit OTP amount, phone number)
valid payment details
[false]
Display Error alert
[true]
Process payment and display success alert
Display success SMS
Enter Credit Card details (email, cvc, expire date, card number, billing address,amount)
valid payment details
[false]
Display Error alert
[true]
Process payment and display success alert
Display success page and send payment success email
stop

Appendix

Backend Implementation

[Server.js] const express =

```
require("express"); const bodyParser =
require("body-parser"); const cors =
require("cors"); require("dotenv").config();
const stripe = require("stripe")(process.env.SECRET_KEY);
const mongoose = require("mongoose"); const app =
express(); app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true })); app.use(cors());
const URL = process.env.MONGODB_URL;
mongoose.connect(URL, {
useNewUrlParser: true,
useUnifiedTopology: true,
});
const connection = mongoose.connection;
connection.once("open", () => {
console.log("Mongodb connection success!");
});

// stripe router
const stripeRoute = require("./routes/stripe"); app.use("/payment",
stripeRoute);

//Customer router
const customerRouter = require("./routes/customers.js"); app.use("/customer",
customerRouter);

//Delivery address router
const deliveryRouter = require("./routes/Delivery.js"); //import  delivery routes
app.use("/delivery", deliveryRouter); //create delivery routes


//Products Router
```

```
const productsRouter = require("./routes/product.js"); app.use("/products",
productsRouter);


//Farmer Router
const farmerRouter = require("./routes/farmer.js");
app.use("/farmer", farmerRouter); const port =
process.env.PORT || 5000; app.listen(port, (error)
=> {
  console.log(`Server running on port ${port}`);
});
```

.env

```
MONGODB_URL =
mongodb+srv://admin:admin123@cluster0.qqijg.mongodb.net/agriProducts?retryWrites=true&w=majority
SECRET_KEY =
sk_test_51KuThEENtkzI6XdNuFDACHm53Jmto53PlRpP90Y4GkZmhwCnijb85w8UxYRyJclkd102cCuNBrUfh6
8yMAFM8qLp00BYIN4YMo
```

Models/Customer.js

```
const mongoose = require('mongoose'); const
Schema = mongoose.Schema;


const customerSchema = new Schema({
username : {         type : String,
require: true
  },


   email :{
type: String,
require: true    },
   phoneno : {
type: Number,
require: true
   },

   password: {
type: String,
require: true
   },
})
const Customer = mongoose.model("Customer", customerSchema); module.exports
= Customer;
```

Models/Delivery.js

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema; const
deliverySchema = new Schema({
   fullname : {
type : String,
require: true
  },

   phoneno :{
type: Number,
require: true
   },
   buildingNo : {
type: String,
require: true
   },     street : {
type: String,
require: true },
city: {       type:
String,
require: true
     },
   province: {
type: String,
require: true
     },


    Date: {
type:Date,
     }


})


const Delivery = mongoose.model("Delivery", deliverySchema); module.exports
= Delivery;
```

Models/Farmer.js

```
const mongoose = require('mongoose'); const
Schema = mongoose.Schema;


const farmerSchema = new Schema({


  username : {
type : String,
require: true

  },
   email :{
type: String,
require: true
   },

   phoneno : {
type: Number,
require: true
   },

   password: {
type: String,
require: true
   },

   state: {
type: String,
require: true
   },

})

const Farmer = mongoose.model("Farmer", farmerSchema); module.exports
= Farmer;
```

Models/Product.js

```
const mongoose = require('mongoose');

const Schema = mongoose.Schema;
const productSchema =  new Schema({
    productName : {
type : String,
required: true
    },

    category : {
type : String,
required: true
    },     price : {
type : String,

    },
    description:{
type:String,         required:
true
    },
    manufacDate :{
type: String,
required: true
    },
    image :{
type: String,
required: true
    }
})

const Product = mongoose.model("Product",productSchema); module.exports
= Product;
```

Models/Favourites.js

```
const mongoose = require('mongoose'); const
Schema = mongoose.Schema;
```

```
const favouriteSchema = new Schema({

    productName : {
type : String,
require: true
  },   price :{
type: String,
require: true
   },
   manufacDate : {
type: Number,
require: true
   },
description: {
type: String,
require: true
   },
   image :{
type: String,
required: true
},

})

const Favourites = mongoose.model("Favourite", favouriteSchema); module.exports
= Favourites;
```

Models/Feedback.js

```
const mongoose = require('mongoose'); //export const Schema
= mongoose.Schema; //store attributes in schema
```

```
const feedbackSchema = new Schema({ //create new object



  //properties     username :{
type: String, //data type


    },


    email :{          type: String,
//data type          required :true
//validation
    },


    type :{          type: String,
//data type          required :true
//validation


    },


    contactNumber :{
type: String, //data type


    },


    trackingno :{          type:
String, //data type
required :true //validation     },


    message :{
        type: String, //data type
required :true //validation
    },



})

const Feedback = mongoose.model("Feedback",feedbackSchema); //feedback schema data goes to feedback table

module.exports = Feedback;

routes/feedback.js
const router= require("express").Router(); const
{ disconnect } = require("mongoose"); let
Feedback = require("../models/Feedback");

//add feedback
router.route("/addf").post((req,res)=> {

    const username = req.body.username;
const email=req.body.email;     const
type=req.body.type;
    const contactNumber=req.body.contactNumber;
const trackingno=req.body.trackingno;     const
message=req.body.message

    const newFeedback= new Feedback({ //new feedback object

        username,
email,        type,
contactNumber,
trackingno,
message
   })


   newFeedback.save().then(()=>{         //to be
executed if successful
res.json("Feedback Added")
}).catch((err)=>{ //execute if not successful
console.log(err);
   })


})


 //view all the data from table by passenger
router.route("/readf").get((req,res)=>{
Feedback.find().then((Feedback)=>{
res.json(Feedback)       }) .catch((err)=>{
console.log(err)
     })


 })


 //view all the data from table by admin
router.route("/readfadmin").get((req,res)=>{
Feedback.find().then((Feedback)=>{
res.json(Feedback)     }) .catch((err)=>{

        console.log(err)
   })


})
```

```
 //view a specific feedback by id
 router.route("/getf/:id").get(async(req,res)=> {



    let userId= req.params.id;
    const user= await Feedback.findById(userId)
    .then((feedback) =>{
        res.status(200).send({ status : "user fetched",feedback})

    }).catch(()=> {
console.log(err.message);
        res.status(500).send({status: "Error with fetch user"});
    })
 })



//update a feedback
router.route("/updatef/:id").put(async(req,res)=> {
       let userId= req.params.id; //passing id through url as a parameter
       const { username,email,type,contactNumber,message } =req.body; //destructure frontend variables passing to
backend through a object

       const updateFeedback= {
username,        email,
type,

        contactNumber,
trackingno,        message,

       }
    const update = await Feedback.findByIdAndUpdate(userId, updateFeedback).then(() =>{
res.status(200).send({status: "User updated"})
    }).catch((err)=>{
console.log(err);
      res.status(500).send({status: "Error with update data", error:err.message});
    })

})

//delete feedback
router.route("/deletef/:id").delete(async(req,res) =>{
let userId=req.params.id;

    await Feedback.findByIdAndDelete(userId).then(() =>{
res.status(200).send({status: "Feedback deleted"})
  }).catch((err)=>{
console.log(err);
     res.status(500).send({status: "Error with delete Feedback", error:err.message});
  })

})

module.exports = router;
```

routes/ customers.js

```
const router = require("express").Router(); let
Customer = require("../models/Customer");
//Add Customer
```

```
router.route("/add").post((req,res) => {     const
username = req.body.username;     const email
= req.body.email;     const phoneno  =
Number(req.body.phoneno);     const password
= req.body.password;


    if(!username || !email || !phoneno || !password){         return
res.status(422).json({error:"please add all the feilds"})


    }
    Customer.findOne({email: email})
.then((savedCustomer)        =>        {
if(savedCustomer) {
            return res.status(422).json({error:"user already exists with that email"})
        }


    const newCustomer = new Customer({
username,         email,         phoneno,
password,
    })
    newCustomer.save().then(() => {
res.json("Customer Added")


    }).catch((err) => {
console.log(err);     })



}).catch((err) =>{
console.log(err);
}) })
router.route("/").get((req,res) => {
    Customer.find().then((customers) => {
res.json(customers)


    }).catch((err) => {
console.log(err)
    })
})
//update Customer using an ID
router.route("/update/:id").put(async (req, res) => {
let userId = req.params.id;
      const {username,email, phoneno,password} = req.body;


      const updateCustomer = {
username,           email,
phoneno,           password
      }
      const update = await Customer.findByIdAndUpdate(userId, updateCustomer).then(() => {
res.status(200).send({status: "Customer updated"})
      }).catch((err) => {
console.log(err);
          res.status(500).send({status: "Error with updating data", error: err.message});
      })
})
//Delete Customer Using an Id
router.route("/delete/:id").delete(async (req, res) => {
let userId = req.params.id;


      await Customer.findByIdAndDelete(userId).then(() => {
res.status(200).send({status: "Customer deleted"});
      }).catch ((err) => {
console.log(err.message);
```

```
          res.status(500).send({status: " Error with delete Customer", error: err.message});
        })
    })
router.route("/get/:id").get(async(req, res) => {


    let userId = req.params.id;
    const user = await Customer.findById(userId).then((customer) => {
res.status(200).send({status: " Customer fetched", customer})
    }).catch(() => {
console.log(err.message);
         res.status(500).send({status:"Error with get Customer" , error: err.message})
    }) })
router.route("/get/:email").get((req, res) => {


    Customer.findOne({email:email})
.then((customer) => {
        res.status(200).send({status: " Customer fetched", customer})
    }).catch(() => {
console.log(err.message);
         res.status(500).send({status:"Error with get user" , error: err.message})
    })
})
router.route("/signin").post((req,res) => {
{/*const email = req.body.email; const password
= req.body.password; */} const username =
req.body.username;     const email    =
req.body.email;     const phoneno  =
Number(req.body.phoneno);     const password
= req.body.password;
```

```
    const newCustomer = new Customer ({
username,         email,          phoneno,
password,


    })


 if(!email || !password){
     res.status(422).json({error:"Please add email or password"})
 }
 Customer.findOne({email:email})
.then(savedCustomer =>{
if(!savedCustomer){
         return  res.status(422).json({error:"Invalid Email or Password"})
      }


      Customer.findOne({password:password})
.then(savedCustomer =>{
if(savedCustomer){
             {/* res.json({message:"successfully signed in"}) */}


             res.json(Customer);
          }
          else{
              return res.status(400).json({error:" Email Password invalid"})
          }
      })
    .catch(err=>{
console.log(err)
    })
```

```
  }) }) module.exports =
router;
```
routes/

delivery.js

```
const router = require("express").Router(); let
Delivery = require("../models/Delivery");
//Add Delivery
router.route("/add").post((req,res) => {     const
fullname = req.body.fullname;     const phoneno
= Number(req.body.phoneno);     const
buildingNo   = req.body.buildingNo;     const
street   = req.body.street;     const city   =
req.body.city;     const province   =
req.body.province;


    const newDelivery = new Delivery({
fullname,         phoneno,
buildingNo,         street,         city,
province,


    })
    newDelivery.save().then(() => {
res.json("Delivery Added")


    }).catch((err) => {
console.log(err);
    })


})
//get all delivery addresses router.route("/").get((req,res)
=> {
    Delivery.find().then((delivery) => {
res.json(delivery)


    }).catch((err) => {
console.log(err)
    })
})
//update Delivery address using an ID
router.route("/update/:id").put(async (req, res) => {
let userId = req.params.id;
      const {fullname,phoneno,buildingNo,street,city,province} = req.body;


      const updateDelivery = {
fullname,           phoneno,
buildingNo,           street,
city,           province
      }
      const update = await Delivery.findByIdAndUpdate(userId, updateDelivery).then(() => {
res.status(200).send({status: "Delivery address updated"})
      }).catch((err) => {
console.log(err);
          res.status(500).send({status: "Error with updating data", error: err.message});
      })
})
//Delete Delivery Using an Id
router.route("/delete/:id").delete(async (req, res) => {
let userId = req.params.id;


      await Delivery.findByIdAndDelete(userId).then(() => {
res.status(200).send({status: "Delivery address deleted"});
      }).catch ((err) => {
console.log(err.message);
```

```
          res.status(500).send({status: " Error with delete Delivery", error: err.message});
      })
    })
router.route("/get/:id").get(async(req, res) => {


    let userId = req.params.id;
    const user = await Delivery.findById(userId).then((delivery) => {
res.status(200).send({status: " Delivery Address fetched", delivery})
    }).catch(() => {
console.log(err.message);
         res.status(500).send({status:"Error with get Delivery address" , error: err.message})
    }) })
module.exports = router;
```

routes/ farmer.js

```
const router = require("express").Router(); let
Farmer = require("../models/Farmer");
/*http://localhost:5000/farmer/add */
router.route("/create").post((req,res)=>{
const username = req.body.username;
const email = req.body.email;     const
phoneno = req.body.phoneno;     const
password = req.body.password;     const state
= req.body.state;

const newFarmer = new Farmer({

    username,
email,
phoneno,
password,     state,
}) newFarmer.save().then(()=>{
res.json("Farmer added")

}).catch((err)=>{
console.log(err);
})
})

/*http://localhost:5000/Farmer */ router.route("/").get((req,res)=>{

   Farmer.find().then((Farmers)=>{
res.json(Farmers)     }).catch((err)=>{
       console.log(err)
   })

})
/*http://localhost:5000/Farmer/update/ */
router.route("/update/:id").put(async (req, res) =>{
let FarmerID = req.params.id;
   const { username,email,phoneno,password,state} = req.body;

   const updateFarmer = {
username,       email,
phoneno,       password,
state
   }

   const update = await Farmer.findByIdAndUpdate(FarmerID,updateFarmer)
   .then(()=>{
       res.status(200).send({status: "Farmer Updated"})
```

```
    }).catch((err) =>{
console.log(err)
        res.status(500).send({status:"Updating data Err",error: err.message});
    })
})

/*http://localhost:5000/Farmer/delete/ */

router.route("/delete/:id").delete(async (req, res) =>{
let FarmerID = req.params.id;

    await Farmer.findByIdAndDelete(FarmerID)
    .then(()=>{
        res.status(200).send({status: "Farmer Deleted"});

    }).catch((err) =>{
console.log(err.message)
        res.status(500).send({status:"Deleting Err",error: err.message});
    }) })
router.route("/get/:id").get(async (req, res) =>{
let FarmerID = req.params.id;

    const Farmer = await Farmer.findById(FarmerID)
    .then((Farmer)=>{
        // res.status(200).send({status: "Farmer fetched", Farmer: Farmer});
res.json(Farmer);

    }).catch((err) =>{
console.log(err.message)
        res.status(500).send({status:"Fetching Err",error: err.message});
    }) })
module.exports = router;
```

routes/ customers.js const router =

```
require("express").Router(); let Product =
require("../models/Product");

/*http://localhost:5000/product/add */

router.route("/create").post((req,res)=>{

    const productName = req.body.productName;
    const category = req.body.category;     const
price = req.body.price;     const description =
req.body.description;     const manufacDate =
req.body.manufacDate;     const image =
req.body.image; const newProduct = new
Product({

    productName,
category,     price,
description,
manufacDate,
image,

})

newProduct.save().then(()=>{
res.json("Product inserted succeefuly.")
```

```
}).catch((err)=>{
console.log(err);
}) })
/*http://localhost:5000/product */

router.route("/").get((req,res)=>{

   Product.find().then((products)=>{
res.json(products)
}).catch((err)=>{
console.log(err)
    })

})
/*http://localhost:5000/product/update/ */

router.route("/update/:id").put(async (req, res) =>{
let productID = req.params.id;
   const { productName,category,price,description,manufacDate,image} = req.body;

   const updateProduct = {
productName,
category,      price,
description,
manufacDate,      image
   }

   const update = await Product.findByIdAndUpdate(productID,updateProduct)
   .then(()=>{
      res.status(200).send({status: "Product Updated"})
    }).catch((err) =>{
console.log(err)
        res.status(500).send({status:"Updating data Err",error: err.message});
    })
})
/*http://localhost:5000/product/delete/ */

router.route("/delete/:id").delete(async (req, res) =>{
let productID = req.params.id;

    await Product.findByIdAndDelete(productID)
    .then(()=>{
        res.status(200).send({status: "Product Deleted"});

    }).catch((err) =>{
console.log(err.message)
        res.status(500).send({status:"Deleting Err",error: err.message});
    })
})

router.route("/get/:id").get(async (req, res) =>{
let productID = req.params.id;

    const product = await Product.findById(productID)
    .then((product)=>{
        // res.status(200).send({status: "Product fetched", Product: Product});
res.json(product);

    }).catch((err) =>{
console.log(err.message)
        res.status(500).send({status:"Fetching Err",error: err.message});
    }) })
```

```
module.exports = router;
```

routes/ stripe.js

```
const router = require("express").Router(); const stripe =
require("stripe")(process.env.SECRET_KEY);

router.post("/add", async (req, res) => {
let status, error;   const { token, amount
} = req.body; try {     await
stripe.charges.create({       source:
token.id,       amount,       currency:
"usd",
    });     status = "sucess";
} catch (error) {
console.log(error);
status = "Failure";   }
res.json({ error, status });
}); module.exports =
router;
```

routes/ favourites.js

```
const router = require("express").Router(); let
Favourite = require("../models/Favourites");

/*http://localhost:5000/favourites/add */

router.route("/create").post((req,res)=>{

    const productName = req.body.productName;
const price = req.body.price;     const
```

```
manufacDate = req.body.manufacDate;     const
description = req.body.description;     const
image = req.body.image; const newFavourite =
new Favourite({

    productName,
    price,
manufacDate,
description,     image,

}) newFavourite.save().then(()=>{
res.json("Favourite added")

}).catch((err)=>{
console.log(err);
})
})

/*http://localhost:5000/favourite */ router.route("/").get((req,res)=>{

    Favourite.find().then((favourites)=>{
res.json(favourites)     }).catch((err)=>{
console.log(err)
    })

})
/*http://localhost:5000/favourite/delete/ */
router.route("/delete/:id").delete(async (req, res) =>{
let favouriteID = req.params.id;

    await Favourite.findByIdAndDelete(favouriteID)
```

```
   .then(()=>{
       res.status(200).send({status: " Deleted"});     }).catch((err) =>{
console.log(err.message)
       res.status(500).send({status:"Deleting Err",error: err.message});
   }) })
router.route("/get/:id").get(async (req, res) =>{
let favouriteID = req.params.id;

   const favourite = await Favourite.findById(favouriteID)
   .then((favourite)=>{
       // res.status(200).send({status: "Product fetched", Product: Product});
res.json(favourite);

   }).catch((err) =>{
console.log(err.message)
       res.status(500).send({status:"Fetching Err",error: err.message});
   }) })
module.exports = router;
```

Frontend Implementation

Payment.js

```
function Payment() {   const [{ address,
cart }] = useStateValue();   const navigate =
useNavigate();

  return (
    <div className="main">
      <div className="container-review">
```

```
<h2>Review Your order</h2>
<div className="container-address">
  <h5>Shipping Address</h5>
  <div className="address-details">
    <p>{address.fullname}</p>
    <p>{address.phoneno}</p>
    <p>{address.buildingNo}</p>
    <p>{address.street}</p>
    <p>{address.city}</p>
    <p>{address.province}</p>
  </div>
</div>

<div className="order">
  <h5>Your order</h5>
  <div>
    {cart?.map((product) => (
      <Product>
        <Image>
          <img src={product.image} alt="" />
        </Image>
        <Description>
          <h4>{product.title}</h4>
          <p>{product.price}</p>
        </Description>
      </Product>
    ))}
  </div>
</div>
```

```
      </div>
      <SubTotalContainer />
      <div className="payment-container">
       <h5>Payment Method</h5>

       <Checkout />
<Button
         className="me-4"
variant="outline-dark"          onClick={()
=> {            navigate("/mobilebillpay");
         }}
       >
         Pay Via Mobile Bill
       </Button>
      </div>
    </div>
  );
}
export default Payment;
```

Checkout.js

```
const MySwal = withReactContent(Swal);

const Checkout = (subTotal) => {
const navigate = useNavigate();
//emailjs
  //if payment sucessfull fire this
  const paymentSucess = () => {
MySwal.fire({      icon: "success",
title: "Payment was sucessful!",
time: 4000,
    });
    navigate("/checkout-success");
  };

  //if payment fail fire this   const
paymentFail = () => {     MySwal.fire({
icon: "error",       title: "Payment was
not sucessful!",       time: 4000,
    });
  };
  //stripe   const
publishableKey =

"pk_test_51KuThEENtkzI6XdNsSPiEkfVuyT4ruaOGV2NyjLVT2In4pD2EhYFbry7bN3ELd4Er5MSgS4YUCZhn
ELb91Us1ZNZ008WPxrNJK";
  const [{ cart }] = useStateValue();

  const tokenHandler = async (token) => {
    try {
      const response = await axios({        url:
"http://localhost:5000/payment/add",
method: "post",
        data: {
          amount: priceForStripe,
token,
       },
      });
      if (response.status === 200) {
paymentSucess();        console.log(`Your
Payment was sucessful`);
      }
    } catch (error) {
paymentFail();
console.log(error);
    }
  };

  const priceForStripe = getCartTotal(cart);

  return (     <StriprCheckout
stripeKey={publishableKey}      label="Pay
Now"      name="Pay with credit card"
shippingAddress      billingAddress
amount={priceForStripe}
description={`Total is $${priceForStripe}`}
token={tokenHandler}
    >
      <Button className="me-4" variant="outline-dark">        Pay Via Credit Card
      </Button>
    </StriprCheckout>
  );
};

export default Checkout;
```

```
MobilePay.js function
MobilePayModal() {
//country code   const
countryCode = "+94";
  //OTP   const [mobileNumber, setMobileNumber] =
useState(countryCode);   const [formExpand, setFormExpand] =
useState(false);   const [otp, setOtp] = useState("");

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

  const requestOtp = (e) => {
    e.preventdefault();     if
(mobileNumber.length >= 12) {
setFormExpand(true);
generateRecaptcha();       let appVerifier =
window.recaptchaVerifier;
```

```
      signInWithPhoneNumber(authentication, mobileNumber, appVerifier)
        .then((confirmationResult) => {
window.confirmationResult = confirmationResult;
        })
        .catch((error) => {
// Error; SMS not sent
console.log(error);
        });
    }
  };

  const verifyOTP = (e) => {
let otp = e.target.value;
setOtp(otp);
    //if user enter six digit only it should verify     if
(otp.length === 6) {       console.log(otp);       let
confirmationResult = window.confirmationResult;
confirmationResult        .confirm(otp)

        .then((result) => {
          // User signed in successfully.
          const user = result.user;
console.log(user);
        })
        .catch((error) => {
          // User couldn't sign in (bad verification code?)
        });
    }
```

```
 };

 return (
   <div className="form-container">
<Form onSubmit={requestOtp}>
       <h1>Pay Via Mobile Phone</h1>
       <div className="mb-3">
         <lable>Phone Number</lable>
         <input
type="tel"
           className="form-control telInput"
id="phoneNumberInput"            aria-
describedby="emailHelp"            value={mobileNumber}
onChange={(e) => setMobileNumber(e.target.value)}
         />
         <div className="form-text">Please enter your phone number</div>
       </div>
       {formExpand === true ? (
         <>
           <div className="mb-3">
             <label htmlFor="otpInput" className="form-label">
               OTP
             </label>
<input
               type="number"
className="form-control"
               id="otpInput"
value={otp}
onChange={verifyOTP}
```

```
            />
            <div id="otp-help" className="form-text">
              Please enter otp
            </div>
          </div>
        </>
      ) : null}
      {formExpand === false ? (
        <Button type="submit" className="btn btn-primary">
          Request OTP
        </Button>
      ) : null}
      <div id="recaptcha-container"></div>
    </Form>
  </div>
 );
}

export default MobilePayModal;
```

firebaseConfig.js

```
const firebaseConfig = {
  apiKey: "AIzaSyAFZ590ClneE7cYnC6XdTaM9HSmJdqb1l8",

  authDomain: "agri-product-app.firebaseapp.com",

  projectId: "agri-product-app",
  storageBucket: "agri-product-app.appspot.com",

  messagingSenderId: "582847577644",

  appId: "1:582847577644:web:0bd092b4fa84ceb76c9b89",
};

const app = initializeApp(firebaseConfig); export
const authentication = getAuth(app);

Mail.js import React from "react";
import emailjs from "emailjs-com";

function Mail() {
function sendEmail(e) {
    e.preventDefault();

    emailjs
      .sendForm(
        "service_7yavpge",
"template_motceg4",
        e.target,
        "fy8SgBtOKzkGQmN-T"
      )
      .then((res) => {
console.log(res);
      })
      .catch((err) => console.log(err));
```

```
 }
 return (
   <div>
     <form onSubmit={sendEmail}>
       <label>Name</label>
       <input type="text" name="name" />
       <label>Email</label>
       <input type="email" name="user_email" />
       <input type="Send Email" value="send" />
     </form>
   </div>
 );
}

export default Mail;
```

CheckoutSuccess.js

```
const CheckoutSucess = () => {
 return (
   <div>
     <h2>Checkout Sucess</h2>
     <Mail />
   </div>
 );
```

```
};

export default CheckoutSucess;



NotFound.js const
NotFound = () => {
return (
   <div className="contaier">
     <h3>Page Not found</h3>
     <h6>404</h6>
   </div>
  );
};

export default NotFound;

CartItems.js
function CartItems() {   const
navigate = useNavigate();

  let [qty, setqty] = useState(1);
//increase products quantity by 1   const
Increment = (e, id) => {     setqty(++qty);
    e.preventDefault();
  };
    //decrease products quantity by 1
const Decrement = (e, id) => {     setqty(--
qty);
```

```
    e.preventDefault();
  };

  const [{ cart }, dispatch] = useStateValue();

  //remove a product from the cart
const removeProduct = (e, id) => {
    e.preventDefault();

    dispatch({
      type: "DELETE_PRODUCT",
      id: id,
});
  };
  //if the cart has no any products
if (cart.length === 0)     return (
      <>
        <Header />
        <h3 style={{ textAlign: "center", fontSize: "5rem" }}>Empty Cart</h3>
        <br/>
        <center><Button onClick={() => navigate("/")}   style={{ backgroundColor: green[500] }}>Move to
Shop</Button></center>

      </>
    );

  console.log("cartitems >>>>", cart);
  return (
    <div>
```

```
   <Header />
   <Container>
     <Main>
       <ShoppingCart>
         {/* show cart items */}
         <h2>Shopping Cart</h2>

         {cart?.map((product) => (
           <Product>
             <Image>
               <img src={product.image} alt="" />
             </Image>
             <Description>
               <h4>{product.title}</h4>

               <Quantity>
                 <button onClick={Increment} min="1">
                   {" "}
                   +{" "}
                 </button>
                 <button> {qty} </button>
                 <button onClick={Decrement}> - </button>
               </Quantity>

               <p>Rs. {product.price * qty}</p>
               <button
onClick={(e) => {
                   if (
                     window.confirm(
                     "Are you sure you want to delete this record?"
                     )
                   )
                     removeProduct(e, product.id);
                 }}
                 style={{ backgroundColor: red[500] }}
               >
                 <DeleteForeverIcon />
               </button>

               {/* <CounterFunction/>  */}
             </Description>
           </Product>
         ))}
       </ShoppingCart>
       <Subtotal>
         <div>
           <p>
             {/* calculate total number of product types and total price*/}
             SubTotal ({cart.length} products ) :
             <strong> Rs. {getCartTotal(cart) * qty}.00</strong>
           </p>
           <small>
             <span>Click here to pay...</span>
           </small>
         </div>        <button
onClick={() => {
navigate("/delivery");
           }}
          >
            Proceed to checkout <CreditCardIcon />
          </button>
        </Subtotal>
      </Main>
    </Container>
  </div>
 );
}



const Quantity = styled.div``; export
default CartItems; reducer.js export
const initialState = {
 cart: [],
 favourite:[],
user: null,
address: [],
};

//calculate total cost in cart export const getCartTotal =
(cart) =>   cart.reduce((amount, item) => item.price +
amount, "");

const reducer = (state, action) => {
console.log("action >>>>", action);

 switch (action.type) {
//add a prodct to cart
```

```
   case "ADD_TO_CART":
     return {
...state,
       cart: [...state.cart, action.item],
     };
      //remove a prodct from cart

   case "DELETE_PRODUCT":
     const index = state.cart.findIndex(
       (cartItem) => cartItem.id === action.id
     );

     let newCart = [...state.cart];
if       (index       >=       0)       {
newCart.splice(index, 1);
     } else {
console.warn(`
           Can't remove the product of id ${index}
       `);
     }
     return {
...state,
       cart: newCart,
     };
        //add a prodct to wishlist

     case 'ADD_TO_WISHLIST':
       return {
...state,
          favourite :[...state.favourite, action.item],

};

     //remove a prodct from wishlist


case 'DELETE_FAVOURITE':

const index1 = state.favourite.findIndex(

   (favouriteItem)=>favouriteItem.id === action.id

);


let newFavourite = [...state.favourite];

if(index1 >=0 ){

newFavourite.splice(index1,1)



}else{

console.warn(`

       Can't remove the product of id ${index1}

   `)

}

return {

   ...state,

   favourite: newFavourite,

};

   case "SET_ADDRESS":

     return {

...state,

       address: { ...action.item },

     };
```

```
    case "SET_USER":
return state;     default:
      return state;
  }
};
export default reducer;



Stateprovide.js import
React,{createContext,useContext,useReducer} from "react";

export const StateContext = createContext()

export const StateProvider = ({reducer,initialState,children})=>(
    <StateContext.Provider value={useReducer(reducer,initialState)}>
        {children}
    </StateContext.Provider>
);

export const useStateValue=()=>useContext(StateContext);



Card.js

function Carditem({ id, image, title, price }) {

  const [{ cart },dispatch] = useStateValue();
console.log("cart >>>>", cart);   const
addToCart = (e) => {
    e.preventDefault();

    dispatch({
      type: "ADD_TO_CART",
      item: {
id,       title,
price,
image,

      },
    });
  };
  const [{ favourite },dispatch1] = useStateValue();

  console.log("favourite >>>>", favourite);
const addToFavourites = (e) => {
    e.preventDefault();

    dispatch1({
      type: "ADD_TO_WISHLIST",
      item: {
id,       title,
price,
image,

      },
    });
  };
  return (
    <Container>
      {/* implement products attributes in a card */}
      <Image>
        <img src={image} alt="" />
      </Image>
      <Description>
        <h5>{title}</h5>

        <p>Rs. {price}.00</p>        <FormControlLabel
control={<Checkbox icon={<FavoriteBorder />}
checkedIcon={<Favorite />}
         name="checkedH" onClick={addToFavourites}/>}

      />
        <button onClick={addToCart}>Add to Cart  <ShoppingCartIcon/></button>

      </Description>
    </Container>
  );
}

export default Carditem;

FavouriteItems.js function
FavouriteItems(){     const
navigate = useNavigate();
```

```
   const [{favourite}, dispatch] = useStateValue();
const removeFavouriite = (e, id) => {
      e.preventDefault();


      dispatch({
        type: "DELETE_FAVOURITE",
        id: id,
      });
     };
   if(favourite.length === 0)
return (
      <>
         <Header/>
               <h3 style={{textAlign:"center", fontSize:"5rem"}}>Empty Wishlist</h3>
               <br/>
               <center><Button onClick={() => navigate("/")}   style={{ backgroundColor: green[500] }}>Move
to Shop</Button></center>
      </>
   )



   console.log("favouriteitems >>>>", favourite);


   return(
      <div>
         <Header/>


         <Container>
         <Main>
```

```
              <Favourites>
                <h2>Your Wishlist</h2>
                {
                  favourite?.map((product)=>(
                   <Product>
                   <Image>
                     <img src={product.image}
                     alt=""/>
                   </Image>
                   <Description><h4>{product.title}</h4>



                    <p>{product.price}</p>
                    <button
       onClick={(e) => {
                  if (
                   window.confirm(
                     "Are you sure you want to delete this from favourites?"
                   )
                  )
                   removeFavouriite(e, product.id);
                 }}
                 style={{ backgroundColor: red[500] }}
                >
                 <DeleteForeverIcon />
                </button>


                   </Description>
                  </Product>
                     ))
                  }



                  </Favourites>
                  </Main>
         </Container>
            </div>
            )
}

export default FavouriteItems
```

FarmerSignIn.js

```
const FarmerSignin = (props) => {   const
[username, setUsername] = useState("");   const
[email, setEmail] = useState("");   const
[phoneno, setPhoneno] = useState("");   const
[password, setPassword] = useState("");   const
[state, setstate] = useState("");

  function sendData(e) {     if (!email ||
!password) {       alert("Please add email
or password");
     return;
```

```
    } else if (
      !/^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-
9]{1,3}])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/.test(
        email
      )
) {
      alert("Invalid email");
return;
    }

    e.preventDefault();

    const newFarmer = {
username,      email,
      phoneno,
password,
      state,
    };
    console.log(newFarmer);
    // alert("Success");

    axois
      .post("http://localhost:5000/farmer/signin", newFarmer)
      .then(() => {        alert("Sign
in successfully");

        //window.location='/profile/+{email}'
      })
      .catch((err) => {         alert("Invalid
email or password");
      });
  }
  return (
    <div>
      <Container>
        <Row>
          <Col>
            {" "}
            <Form className="container" onSubmit={sendData}>
              <div className="signin1">
                <div className="signin">
                  <Col xs={1} md={12}>
                    <Image                      className="im"
src="https://res.cloudinary.com/hidl3r/image/upload/v1631611510/itp/ulogin_b64etx.png"
                      roundedCircle
                    />
                  </Col>
                  <h1 className="login">Sign In</h1>
                  <br /> <br />
                  <Form.Group className="mb-3" controlId="formBasicEmail">
                    <Form.Control
type="email"
placeholder="Enter email"
value={email}
                      onChange={(e) => setEmail(e.target.value)}
                    />
                  </Form.Group>
                  <Form.Group className="mb-3" controlId="formBasicPassword">
                    <Form.Control                    type="password"
placeholder="Password"                    value={password}
onChange={(e) => setPassword(e.target.value)}
                    />
                  </Form.Group>
                  <Form.Group className="mb-3" controlId="formBasicCheckbox">
                    <Form.Check type="checkbox" label="Check me out" />
                  </Form.Group>
                  <br /> <br />
                  <Button variant="primary" size="lg" type="submit">
                    Sign In
                  </Button>
                  <br />
                  <br />
                  <br />
                  <h5>
                    <Link to="/farmersignup" id="link">
                      {" "}
                      Don't have an account?{" "}
                    </Link>
                  </h5>
                </div>
              </div>
            </Form>
          </Col>
          <Col>
            <br />
            <br />
            <br />
            <br />
            <br />
            <Image
              src="https://res.cloudinary.com/hidl3r/image/upload/v1633337384/itp/preview_nrqvph.gif"
              fluid
/>
          </Col>
        </Row>
      </Container>
    </div>
   );
};

export default FarmerSignin;
```

FarmerSignUp.js

```
const FarmerSignup = () => {

    const [username, setUsername] = useState("");
const [email,setEmail] = useState("");     const
[phoneno,setPhoneno] = useState("");     const
[password, setPassword] = useState("");     const
[state, setstate] = useState("");

    function sendData(e) {        if(!username || !email ||
!phoneno || !password){
```

```
            alert("Please fill  in all  fields");
return
        }



        else  if(!/^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-
9]{1,3}\.[0-9]{1,3}])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/.test(email)){
alert("Invalid email");
            return
         }

        else if(phoneno.trim().length !=10){
alert("Invalid phoneno");            return
        }

        e.preventDefault();

        const newFarmer ={
            username,
email,
            phoneno,
password,            state
        }
         console.log(newFarmer)
         //alert("Success");

         axois.post("http://localhost:5000/farmer/create", newFarmer).then(() => {
            alert("Registration Success");
            setUsername("");
setEmail("");            setPhoneno("");
setPassword("");
setstate("");


        }).catch((err) =>{
            alert(err)
        })
    }


    return(
        <div>
            <Container>
  <Row>
    <Col>  <br/><br/><br/><br/><br/><br/><br/><br/><br/>
     <Image
src="https://res.cloudinary.com/hidl3r/image/upload/v1652381357/AgriManagement/1b34759c948ebf
a256c9c05aee8fee11_ofe1yx.gif" fluid /></Col>
    <Col>  <Form className="container" onSubmit={sendData} >
          <div className = "signin1">
            <div className= "signin">
          <Col xs={1} md={12}  >
              <Image  className="im"
src="https://res.cloudinary.com/hidl3r/image/upload/v1631611510/itp/ulogin_b64etx.png"
roundedCircle />
              <br/> <br/>
            </Col>


              <h1 className="login">Sign Up</h1>
              <Form.Group className="mb-3" controlId="formBasicUsername">
```

```
                   <Form.Control type="text" placeholder="Enter Username"  value = {username}
onChange={(e) => setUsername (e.target.value)} />
              </Form.Group>
               <br/>

               <Form.Group className="mb-3" controlId="formBasicEmail">
                   <Form.Control type="email" placeholder="Enter email"  value = {email}
onChange={(e) => setEmail (e.target.value)} />

               </Form.Group>

               <br/>
              <Form.Group className="mb-3" controlId="formBasicPhoneno">
                   <Form.Control type="Number" placeholder="Enter Phone number"  value = {phoneno}
                     onChange={(e) => setPhoneno (e.target.value)} />

                   </Form.Group>
               <br/>
              <Form.Group className="mb-3" controlId="formBasicPassword">
              <Form.Control type="password" placeholder="Password" value = {password}
onChange={(e) => setPassword (e.target.value)}/>
              </Form.Group>

             <br/>
             <Form.Group className="mb-3" controlId="formBasicPassword">
<Form.Control type="text" placeholder="State" value = {state}
onChange={(e) => setstate (e.target.value)}/>
              </Form.Group><br/>
      <Button variant="primary" size="lg" type="submit">
```

```
          Sign Up
     </Button>
     <br/><br/>
      <h5>
      <Link to = "/farmersignin">Already have an account?  </Link>
     </h5>
     <br/>
              </div>
          </div>
        </Form></Col>
  </Row>

</Container>

     </div>
   )

}

export default FarmerSignup
```

ProductsAdd.js

```
function ProductsAdd(){

  const [productName, setproductName] = useState("");
const [category, setcategory] = useState("");   const
[price, serprice] = useState("");   const [description,
setdescription] = useState("");   const [manufacDate,
setmanufacDate] = useState("");   const [image,
setimage] = useState("");

 function                    sendData(e){
if(description.trim().length     <     2){
alert("Please      insert      a      good
Description");

   return

 }
  e.preventDefault();

  const newProduct ={

   productName,
category,
   price,
description,
   manufacDate,
   image

  }

  axios.post("http://localhost:5000/products/create",newProduct).then(()=>{

   alert("Product Added Successfully");
window.location = `/allproducts`;
```

```
   setproductName("");
setcategory("");    serprice("");
setdescription("");
setmanufacDate("");
setimage("");
}).catch((err=>{

     alert(err)
   }))


 }


   return(

<Form onSubmit={sendData}>
  <Form.Group className="container" controlId="vehicleNo">
   <Form.Label>Product Name</Form.Label>
   <Form.Control type="text" required placeholder="Enter Product Name"  maxlength="20"
onChange={(e)=>{

setproductName(e.target.value);


   }}/>
   <Form.Text className="text-muted">
ex : cucumber plant at 3 weeks
```

```
    </Form.Text>
  </Form.Group>

  <Form.Group className="container" controlId="vModel">     <Form.Label>Category</Form.Label>
    <Form.Control type="text" required placeholder="Enter Category" onChange={(e)=>{

setcategory(e.target.value);

}}/>
  </Form.Group>

  <Form.Group className="container" controlId="nicNo">
    <Form.Label>Price</Form.Label>
    <Form.Control type="text" required placeholder="ex :550LKR"  maxlength="10" onChange={(e)=>{

serprice(e.target.value);

}}/>
  </Form.Group>

  <Form.Group className="container" controlId="ownerName">
    <Form.Label>Description</Form.Label>
    <Form.Control type="text" required placeholder="Enter Description" onChange={(e)=>{

setdescription(e.target.value);

}}/>
  </Form.Group>
```

```
 <Form.Group className="container" controlId="manufacYear">
   <Form.Label>Manufactured Date</Form.Label>
   <Form.Control type="date" required placeholder="Enter Manufactured Date" maxlength="8"
onChange={(e)=>{

setmanufacDate(e.target.value);

}}/>
 </Form.Group>

 <Form.Group className="container" controlId="vType">
   <Form.Label>image</Form.Label>
   <Form.Control type="text" required placeholder="input image link" onChange={(e)=>{

setimage(e.target.value);

}}/>
 </Form.Group>



 <Form.Group className="container" controlId="formBasicCheckbox">
   <Form.Check type="checkbox" required label="Vehicle details checked" />


   <Button variant="primary" type="submit">
   Submit
 </Button>
 <Link to ="/"> <Button variant="info">home</Button></Link>


 </Form.Group>
```

```
</Form>


);
 }
export default ProductsAdd;
```

ProductsAll.js

```
function AllProducts(props){

   const [products, setProducts] = useState([]);
const [search, setSearch] = useState("");

   useEffect(() =>{     function getProducts(){
axios.get("http://localhost:5000/products/").then((res) =>{
setProducts(res.data);        }).catch((err) => {
alert(err.message);
       })
 }
    getProducts();
   },[])

   //"ProductDetails/+_id //check this
   //try with find one
```

```
  function Update(_id) {      console.log(_id)
props.history.push("/productdetails/"+_id);
 }

 return (

   //link to button's route id is not working inside table

   <div>
      <center> <h1>Product Details</h1> </center>







 <br/>

  <thead>
  <input type="text" placeholder = "Search Product " onChange ={(e) =>{
setSearch(e.target.value);
}} />



  </thead>
  <tbody>

  {products.filter(Product => {
```

```
                    if(search == ""){
return Product
                    }
                    else if(Product.productName.toLowerCase().includes(search.toLowerCase())){
                      return Product
                    }
                  }).


 map((Product) => {


   return(



<Main>
{
        products?.map((Product) => (


   <Carditem id={Product._id}
image={Product.image}
title={Product.productName}
price={Product.price}
// rating={Product.description}
// name={Product.manufacDate}

/>
  ))}
</Main>
```

```
  );

        })}
        </tbody>

 <br/>
 <br/>

   </div>

 )

}
```

DeliveryAddress.js

```
export default function DeliveryAddress(){  //adding function

   //creating states

  const [fullname,setFullname] = useState("");
  const [phoneno,setPhoneno] = useState("");
const [buildingNo,setBuildingNo] = useState("");
const [street,setStreet] = useState("");   const
[city,setCity] = useState("");   const
[province,setProvince] = useState("");

  const [value, setValue] =useState(  );



  function sendData(e){  //create event send data


    if(  !fullname || !phoneno || !street || !city || !province ){
alert("Kindly add username/contactNumber with delivery details!");
window.location = `/adddelivery`;       return
  }



    e.preventDefault(); //execute setData function, when click submit button


      const newDelivery ={


        fullname,
phoneno,
buildingNo,
street,          city,
          province

      }


     axios.post(`http://localhost:5000/delivery/add`, newDelivery).then(() =>{  //route from the backend
      alert("Delivery details Added.**To change delivery details please contact saniagro@gmail.com**")
//display if adding is successful        window.location =

`/payment`;   //redirect to adding page      }).catch((err) =>{

//display error if adding is not successful        alert(err)

    })

  }


   return(


     <div>

       <Header/>

       <br></br>

      <div sx={{ml:"auto"}}   value={value} onChange={(e,val)=>setValue(val)} >

          <div><a href="/addorder"><img style={{height: 90, width: 90, marginLeft:252}}
src="https://res.cloudinary.com/dorcq99nr/image/upload/v1652113795/AgriProducts/cart_hvasfk.jpg"
alt="Add to Cart" ></img></a>

           STEP 1   </div>

            </div>


            <div sx={{ml:"auto"}}   value={value} onChange={(e,val)=>setValue(val)} >

          <div><a href="/adddelivery"><img style={{height: 100, width: 100, marginLeft:352}}
src="https://res.cloudinary.com/dorcq99nr/image/upload/v1652115276/AgriProducts/delivery1_u8idbj
.jpg" alt="Add Delivery Adress"  ></img></a>

           NOW  STEP 2

          </div>

            </div>  <br/>


            <div sx={{ml:"auto"}}   value={value} onChange={(e,val)=>setValue(val)} >

          <div><a    href="/addpayment"><img    style={{height:    70,    width:    90,    marginLeft:492}}
src="https://cdn-icons-png.flaticon.com/512/4108/4108042.png" alt="Make your payment easily"
></img></a>
```

```
          NEXT STEP 3
         </div>
          </div>



    <Container>



    <Form className="container" onSubmit={sendData} >
     <div className = "delivery1">
     <div className = "delivery">
    <blockquote class="blockquote"><center>
       <h1 class="mb-0">Delivery Details</h1>
       </center></blockquote>
       <br></br>
    <div className="form-group">
    <label for="fullname" className="labels"> Full Name *: </label> <br/>
    <input type="text"  className="form-control" id="fullname" aria-describedby="em"
placeholder="Full name"  required     onChange={(e)=>{      setFullname(e.target.value);
     }}   />
    </div> <br/><br/>

    <div className="form-group">
    <label for="phoneno" className="labels"> Contact Number *: </label>
    <input type="text" className="form-control" id="phoneno" aria-describedby="em"
placeholder="Phone no" maxlength="10" size="50" required      onChange={(e)=>{
setPhoneno(e.target.value);
       }}   />
   </div> <br/><br/>
```

```
    <div className="form-group">
    <label for="buildingno" className="labels"> Building Number *:</label>
    <input type="text" className="form-control" id="buildingno" aria-describedby="em"
placeholder="9/8/A"          size="50"     required                                onChange={(e)=>{
setBuildingNo(e.target.value);
      }}   />
     </div> <br/><br/>

   <div class="form-group">
    <label for="street" className="labels"> Street *:</label>
  <input type="text"  class="form-control" id="street" aria-describedby="em" placeholder="flower road"
size="50" required    onChange={(e) =>{     setStreet(e.target.value);
   }}  />

</div> <br/><br/>

<div class="form-group">
<label for="city" className="labels"> City *:</label>
  <input type="text"  class="form-control" id="city" aria-describedby="em" placeholder="clombo"
size="50" required     onChange={(e) =>{     setCity(e.target.value);
   }}  />

</div> <br/><br/>



<div className="form-group ">
<i className="zmdi zmdi-email zmdi-hc-2x"></i>
```

```
     <label for=" province" className="labels">Province *:</label>
<select class="custom-select custom-select-lg mb-3" id="province"
onChange={(e)=>{        setProvince(e.target.value);
     }}  >
       <option>Choose</option>
      <option>Western Province</option>
      <option>Central Province</option>
      <option>Southern Province</option>
      <option>Uva Province</option>
      <option>Sabaragamuwa Province</option>
      <option>North Western Province</option>
      <option>North Central Province</option>
      <option>Nothern Province</option>
      <option>Eastern Province</option>
     </select>
   </div>

  <br/><br/>
  <center>
   <Button type="submit" class="btn btn-primary btn-lg">
   Continue to Payment
   </Button>



        </center>
        </div>
          </div>
      </Form>
```

```
</Container>



</div>
   );
}
```

GenerateTrackingToken.js

```
export default function DeliveryReport(){
const[Values, setValues] = useState([]);

   const[fullname, setFullname] = useState("");
const[phoneno, setPhoneno] = useState("");
const[buildingNo, setBuildingNo] = useState("");
const[street, setStreet] = useState("");     const[city,
setCity] = useState("");     const[province,
setProvince] = useState("");     const[date, setDate]
= useState("");     const [deliveries, setDeliveries] =
useState([]);     const [show, setShow] =
useState(false);

   const handleClose = () => setShow(false);
const handleShow = () => setShow(true);

   const [search, setSearch] = useState("");

   useEffect(()=>{       function getDeliveries(){
axios.get("http://localhost:5000/delivery/",).then((res)=>{
```

```
setDeliveries(res.data);          }).catch((err)=>{
alert(err.message)
        })
     }
     getDeliveries();
   },[])

   const deletedelivery = (id) =>{
axios.delete(`http://localhost:5000/delivery/delete/${id}`);
   }

   const UpdateDeliveryDeatails = (val) =>{
console.log('test====',val)
setValues(val);

       handleShow()
   }

   const createPDF = (_id, fullname,phoneno, buildingNo,street,city,province,date)=>{
console.log(_id);        console.log(fullname);        console.log(phoneno);
console.log(buildingNo);        console.log(street);        console.log(city);
console.log(province);        console.log(date);

       const unit = "pt";        const size = "A4";        const orientation =
"landscape";        const marginLeft = 40;        const doc = new
jsPDF(orientation, unit, size);        const title = `Sani Agro Delivery Tracking
Tocken     Tracking ID- ${_id}`;        const deliveryFullname = `Full Name:
${fullname}`;        const deliveryPhoneno = `Phoneno: ${phoneno}`;
const deliveryBuildingNo = `BuildingNo: ${buildingNo}`;        const
```

```
deliveryStreet = `Street: ${street}`;        const deliveryCity = `City: ${city}`;

const deliveryProvince = `Province: ${province}`;        const deliveryDate =

`Date: ${date}`;



       const image2 =
"https://res.cloudinary.com/dorcq99nr/image/upload/v1652970181/AgriProducts/process_delivery_ky
n9pc.png"

       const success = ` Dear Customer, ${fullname}. We will deliver your product within 2 working days.`

       const second = `If you have not received your item, please notify us using the Tracking Id to

Feedback and Complaints page.` ;        const third  = `We are committed to providing you with

quality services. Thank you`;         const issuedate =`Issue Date: ${date}`;

       const left = 20;

const top = 8;


       const lefts = 450;        const tops =

200;        const imgWidths = 350;

const imgHeights = 250;

doc.setFontSize(15);        doc.text(200,40

,title);        doc.text(60,200,

deliveryFullname );        doc.text(60,250,

deliveryPhoneno);        doc.text(60,

300,deliveryBuildingNo);        doc.text(60,

350,deliveryStreet);        doc.text(60,

400,deliveryCity);        doc.text(60,

450,deliveryProvince);



       doc.addImage(image2, 'PNG' , lefts, tops, imgWidths, imgHeights);
```

```
        doc.text(60,500,issuedate);
doc.text(80, 100, success);        doc.text(80,
120, second); doc.text(80, 140, third);
doc.save(`${fullname}'s
UserDeliveryToken.pdf`);

    }



    function sendData(e) {

      e.preventDefault();



      var fullname1=null;
var phoneno1=null;
var buildingNo1=null;
var street1=null;      var
city1= null;      var
province1= null;      var
date1 = null;

  //set values

      if (fullname ===""){
console.log('test null cond')
fullname1=Values.fullname;
        }else{
          fullname1=fullname
console.log('test not null cond')


      }
      if (phoneno ===""){
phoneno1=Values.phoneno
        }else{
          phoneno1=phoneno
      }


      if (buildingNo===""){
buildingNo1=Values.buildingNo


        }else{
         buildingNo1=buildingNo
      }


      if (street===""){
street1=Values.street
        }else{
street1=street
      }


      if (city===""){
city1=Values.city
}else{          city1=city
      }
```

```
    if (province===""){
province1=Values.province
     }else{
      province1=province
  }

    if (date===""){
date1 =Values.date
     }else{
date1 = date
  }

    const updateDelivery={
       id:Values._id,
fullname:fullname1,
phoneno:phoneno1,
buildingNo:buildingNo1,
street:street1,         city:city1,
province:province1,
date:date1
    }

    console.log('form submit =====',updateDelivery)

    axios.put(`http://localhost:5000/delivery/update/${updateDelivery.id}`, updateDelivery).then(()=>{
alert("Customer Delivery Details Updated");
       handleClose();
window.location = `/`;
```

```
      }).catch((err)=>{
console.log(err);          alert(err);

      })
  }



    return(
       <div>
           <h1>Delivery Details</h1>

           <InputGroup className="mb-3">
    <InputGroup.Text id="inputGroup-sizing-default">Search</InputGroup.Text>
    <FormControl      aria-label="Default"      aria-
describedby="inputGroup-sizing-default" onChange={(e) =>{
setSearch(e.target.value);
      }}
    />
  </InputGroup>

  {deliveries.filter(Delivery=>{
if(search === ""){          return
Delivery
      }
      else if(Delivery.fullname.toLowerCase().includes(search.toLowerCase())){
         return Delivery
      }
  })
  .map((val,key)=>{
        return(
         <div key={key} className="deliveries">
            <container length="100px">
                <row>
            <ListGroup>
            <ListGroup.Item>{val._id}</ListGroup.Item>
            <ListGroup.Item>{val.fullname}</ListGroup.Item>
            <ListGroup.Item>{val.phoneno}</ListGroup.Item>
            <ListGroup.Item>{val.buildingNo}</ListGroup.Item>
            <ListGroup.Item>{val.street}</ListGroup.Item>
            <ListGroup.Item>{val.city}</ListGroup.Item>
            <ListGroup.Item>{val.province}</ListGroup.Item>




             <Button className="generateDeliveryPdF" onClick={()=> createPDF(val._id, val.fullname,
val.phoneno, val.buildingNo, val.street,val.city, val.province, val.date )}>Generate report</Button>
            </ListGroup>

            </row>

            <Modal show={show} onHide={handleClose} className="getfunc">
              <Modal.Header closeButton>
              <Modal.Title>Update Details</Modal.Title>
              </Modal.Header>

              <Modal.Body>
```

```
                    <Form onSubmit={sendData}>


         <Form.Group controlId="container">
           <Form.Label for="fullname">Fullname</Form.Label>
           <Form.Control type="text"
defaultValue={Values.fullname}
onChange={(e)=>{
setFullname(e.target.value);
          }} />


         </Form.Group>
         <Form.Group controlId="container">
         <Form.Label for="phoneno">Phone Number</Form.Label>
         <Form.Control type="text"
defaultValue={Values.phoneno}
onChange={(e)=>{
setPhoneno(e.target.value);
         }} required />
        </Form.Group>



        <Form.Group controlId="container">
        <Form.Label for="buildingNo">BuildingNo</Form.Label>
        <Form.Control type="text"
defaultValue={Values.buildingNo}
onChange={(e)=>{        setBuildingNo(e.target.value);
        }} required/>
        </Form.Group>
```

```
       <Form.Group controlId="container">
       <Form.Label for="street">Phone number</Form.Label>
       <Form.Control type="Number"
defaultValue={Values.street}
onChange={(e)=>{        setStreet(e.target.value);
       }} required/>
       </Form.Group>

      <Form.Group>
       <Form.Label for="city">City</Form.Label>
<Form.Control type="text"
defaultValue={Values.city}
onChange={(e)=>{        setCity(e.target.value);
       }} />
      </Form.Group>

      <Form.Group>
       <Form.Label for="province">Province</Form.Label>
       <Form.Control type="text"
defaultValue={Values.province}
onChange={(e)=>{        setProvince(e.target.value);
       }} />
      </Form.Group>



      <Button  className="final" type="submit"> Edit details</Button>
      </Form>
```

```
        </Modal.Body>


        </Modal>
          <br />
      </container>
    </div>
          );
       }))}


          <Button variant="secondary" size="lg"><a href="/"> Back </a></Button>{' '}

</div>
   );
}
```

DeliveryFeedback.js

```
export default function DeliveryFeedback(){  //adding function

   //creating states   const [username,setUsername] =
useState("");   const [email,setEmail] = useState("");   const
[type,setType] = useState("");   const
[contactNumber,setContactNumber] = useState("");   const
[trackingno,setTrackingno] = useState("");   const
[message,setMessage] = useState("");
```

```
 function sendData(e){  //create event send data


   if(!(/^\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$/.test(email))){
alert("Incorrect Email type");
     return
 }



   e.preventDefault(); //execute setData function, when click submit button


     const newFeedback ={


       username,
       email,
type,
       contactNumber,
       trackingno,
message


     }


    axios.post(`http://localhost:5000/feedback/addf`, newFeedback).then(() =>{  //route from the
backend


       alert("Feedback Added.Mail us for any updates or inqueries- saniagro@gmail.com") //display if
adding is successful         window.location = `/`;  //redirect to adding page      }).catch((err) =>{
//display error if adding is not successful
       alert(err)
    })
 }
```

```
   return(


    <div>
      <br></br>
      <blockquote class="blockquote">
 <center><h1 class="mb-0">Complaints and Feedback of Your Delivery</h1></center>
</blockquote>
      <br></br>
       <form className="container"  onSubmit={sendData} >


<div className="form-group">
   <label for="Username">Enter Username</label>
   <input type="text" className="form-control" id="username" aria-describedby="em"
placeholder="Enter Username"      onChange={(e)=>{      setUsername(e.target.value);
    }}   />


  </div>
  <div class="form-group">
   <label for="email">Email Address</label>
   <input type="email"  class="form-control" id="email" aria-describedby="emailHelp"
placeholder="Email Address" require onChange={(e) =>{
     setEmail(e.target.value);
    }}  />
   <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone
else.</small>
  </div>


  Type
  <div class="form-check">
  <input class="form-check-input" type="radio" name="type" id="feedback" value="feedback"
onChange={(e) =>{

     setType(e.target.value);

    }}  />

  <label class="form-check-label" for="feedback">

    Feedback

  </label>

</div>

<div class="form-check">

  <input class="form-check-input"   type="radio" name="type" id="complaint" value="complint"  require
onChange={(e) =>{

     setType(e.target.value);

    }}  />

  <label class="form-check-label" for="complaint">

    Complaint

  </label>

</div>

<br></br>

<div class="form-group">

    <label for="contactNumber">Enter Contact Number</label>

    <input type="text" class="form-control" id="contactNumber" aria-describedby="em"
placeholder="Contact Number"  onChange={(e) =>{
setContactNumber(e.target.value);

    }}  />


  </div>


  <br></br>

<div class="form-group">

    <label for="trackingno">Tracking Number</label>
```

```
   <input type="text" class="form-control" id="trackingno" aria-describedby="em"
placeholder="Tracking Number" require onChange={(e) =>{
setTrackingno(e.target.value);
    }}  />

  </div>

  <div class="form-group">
   <label for="message">Description</label>
   <textarea class="form-control" id="message" rows="3" placeholder="Please include your Feedback or
Complaint" require onChange={(e) =>{       setMessage(e.target.value);
    }}></textarea>
  </div>

  <br/>
  <Button type="submit" >Submit</Button>




</form>



</div>
   );
}
```

home.js

```
function Home() {
return (
   <div>
      <Header />
      <br/>
<Container>
  <Row>
   <Col>



     <Carousel>
      <Carousel.Item  interval={1500}>
<img      width={100}      height={800}
className="d-block w-100"
          src="https://res.cloudinary.com/hidl3r/image/upload/v1652245751/AgriManagement/xavi-
mollu-Q14NlYLEI-unsplash_kevghd.jpg"
          alt="First slide"
        />
        <Carousel.Caption>
          <h3>Agriculture or farming is the practice</h3>
          <p>Agriculture is the art and science of cultivating the soil, growing crops and raising livestock. It
includes the preparation of plant and animal products for people to use and their distribution to
markets.</p>
        </Carousel.Caption>
      </Carousel.Item>
      <Carousel.Item  interval={1500}>
<img      width={100}      height={800}
className="d-block w-100"

src="https://res.cloudinary.com/hidl3r/image/upload/v1652246378/AgriManagement/hoovertung-
DjlNzHuv_Ck-unsplash_ikyp2n.jpg"
```

```
           alt="Second slide"
         />
         <Carousel.Caption>
           <h3>It substituted synthetic fertilizers</h3>
           <p>in the European Union more commonly known as ecological farming or biological farming, is
an agricultural system that uses fertilizers of organic origin such as compost manure, green manure, and
bone meal and places emphasis on techniques such as crop rotation and companion planting.</p>
</Carousel.Caption>
        </Carousel.Item>
        <Carousel.Item  interval={1500}>
<img      width={100}      height={800}
className="d-block w-100"

src="https://res.cloudinary.com/hidl3r/image/upload/v1652244740/AgriManagement/briennehong-
SbUlGuCu7Sg-unsplash_vbmogi.jpg"
           alt="Third slide"
         />
         <Carousel.Caption>
           <h3>Vegetables, grains, dairy products and meat.</h3>
           <p>
           These producers must follow the guidelines for organic food production. But they don't need to
go through the certification process. They can label their products as organic.
           </p>
         </Carousel.Caption>
        </Carousel.Item>
      </Carousel>


    </Col>


  </Row>
  <Row>
```

```
    <Col>1 of 3</Col>


  </Row>
    <AllProducts/>


</Container>


</div>




  );
}


export default Home;
```