STORE MANAGEMENT SYSTEM

BY

WIJETHUNGA P.M.L.R – 2019/E/174

NOVEMBER 2022

INDIVIDUAL DATABASE APPLICATION ASSESSMENT

EC5070 DATABASE SYSTEMS

SEMESTER 5

DEPARTMENT OF COMPUTER ENGINEERING

UNIVERSITY OF JAFFNA

Problem Statement for <JANAHITHA DB PROJECT>
<WIJETHUNGA P.M.L.R>
<09/09/2022>

HIGH LEVEL PROBLEM SUMMARY

<JANAHITHA enterprises is medium scale developed company that owns a name in the society but still runs on a local database and taking steps to change the systems into an online platform > < The primary success criteria is to implement an online web based database management system> <The main scope of this project is to give the company a database management system to keep and handle records effectively and update the system continuously on time for users to know the quantities of products on sale >

DETAILED PROBLEM STATEMENT

PART 1: FUNCTION

<Janahitha company is an Intermediate hardware distributing company which sells ironworks and all hardware related objects like Qt bars, cement , and pipes. They are mainly sold in large stocks to the customers who are owners of small scale hardware company. So the transactions happen so fast and in large scales. So the objects that were able to take an hour ago might not be available now. So the customers had to be sent off as they didn't have required amount of items. That is where the need of continuously updating and web based distributing database management system comes in. And as the company does considerable amount of money transactions being everything in a database management system is of utmost importance. company has one owner and 5 managing officers and around 10 employees. The company dals with lots of institutions and companies as in both buying side and selling side they have to work in collaborate with them. And managers have to work on the salaries of employees also. So The main purpose of this database management system is to provide the company a more reliable online database management system and web page with distributing data.
>

PART 2: TIME

<Upto today the company has a database on a local host and it is not updating daily. Using that we can only get the information about available items till the date it was updated. All the accounts are in written format. And my near goal is to have an online data management system for the Company and to build a view webpage for customers.
The long goal is to implement an online ordering system also .>

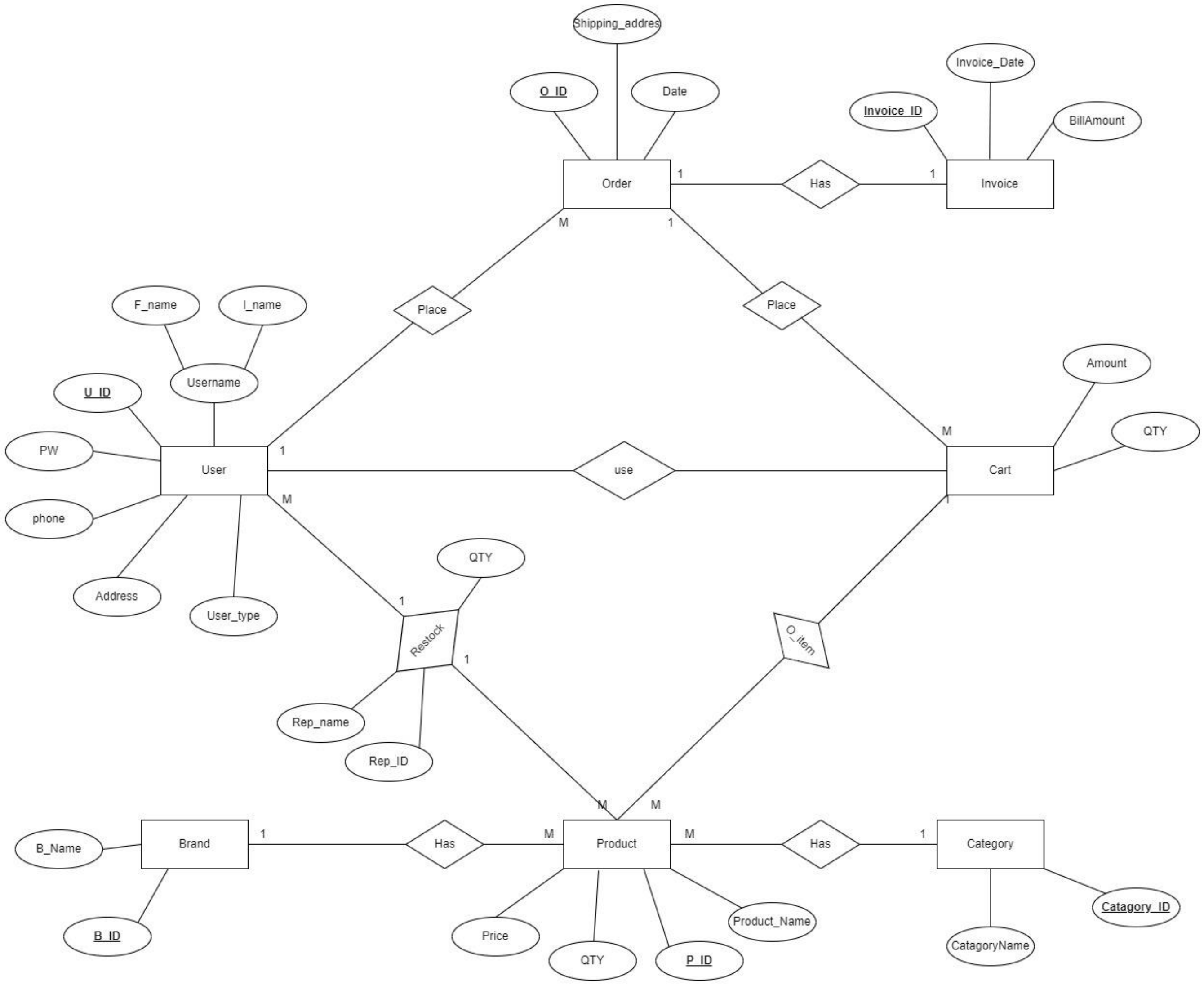
KEY STAKEHOLDERS

Sarath Rudrigo	*Owner/Managing director*
Faiz rees	*Manager*
Emplyeesof company/ customers	*Users*

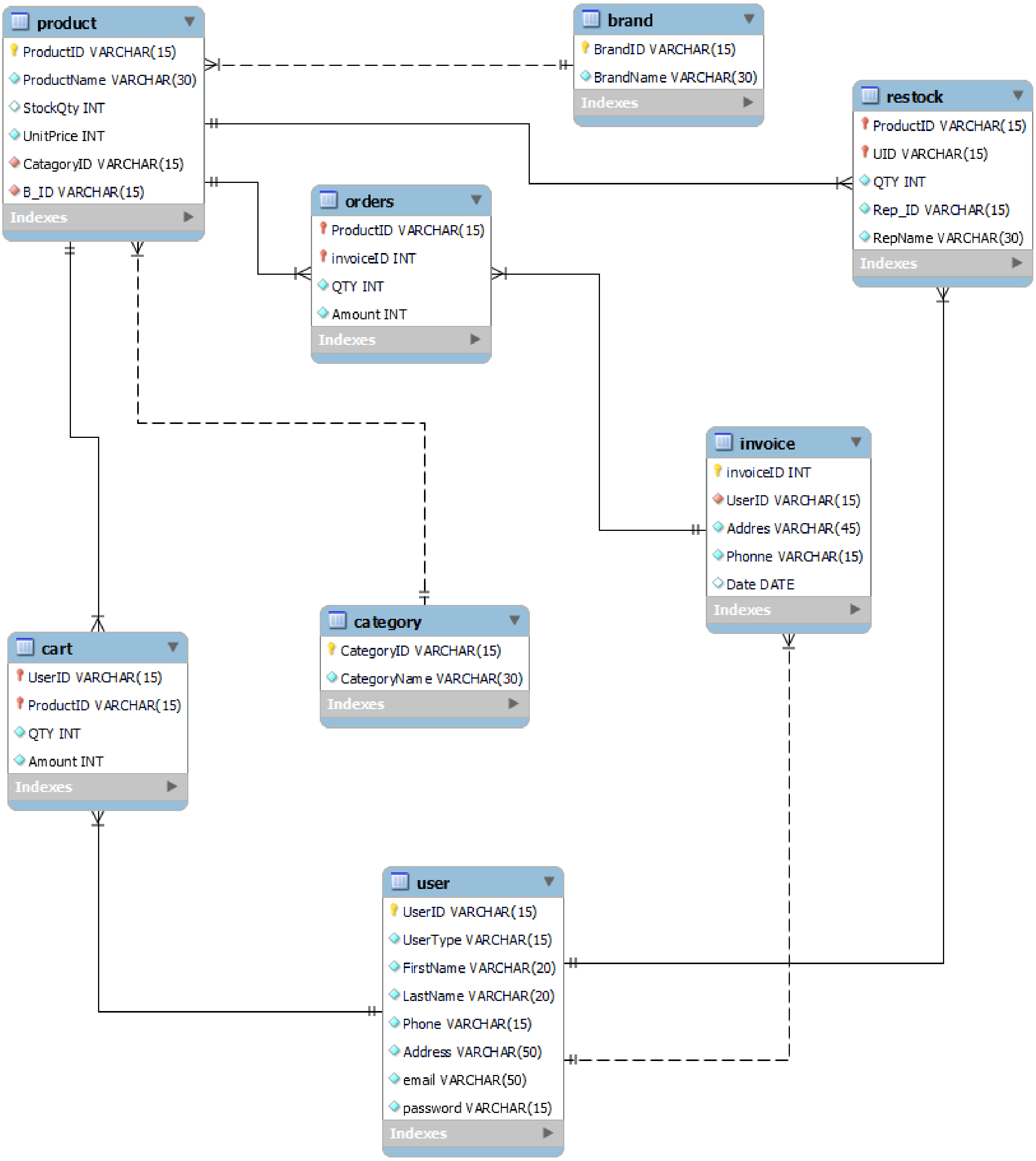
REVISION HISTORY

Date	**Version**	**Reason for change**	**Author**
10/11/2022	1.0	INITIAL RELEASE	WIJETHUNGA P.M.L.R

ER DIAGRAM

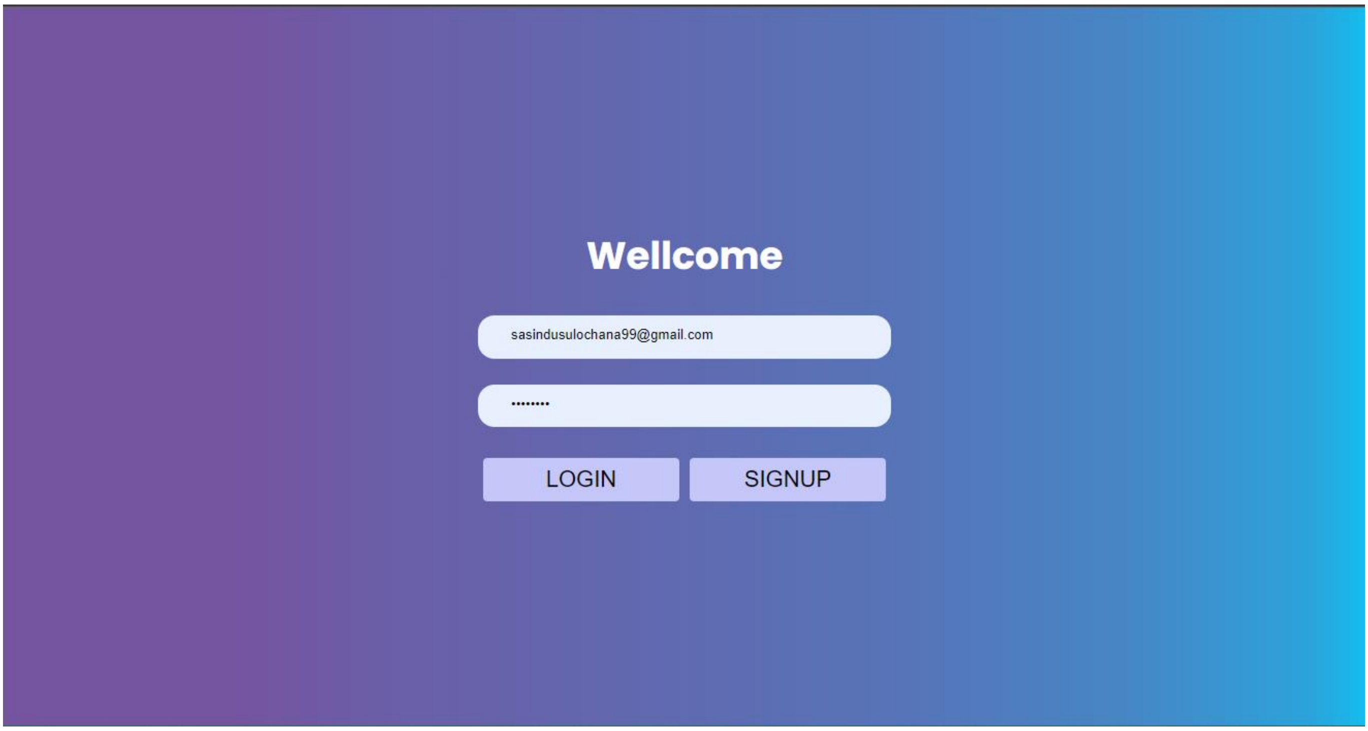


RELATION SCHEMA



GITHUB REPO :

LOGIN PAGE

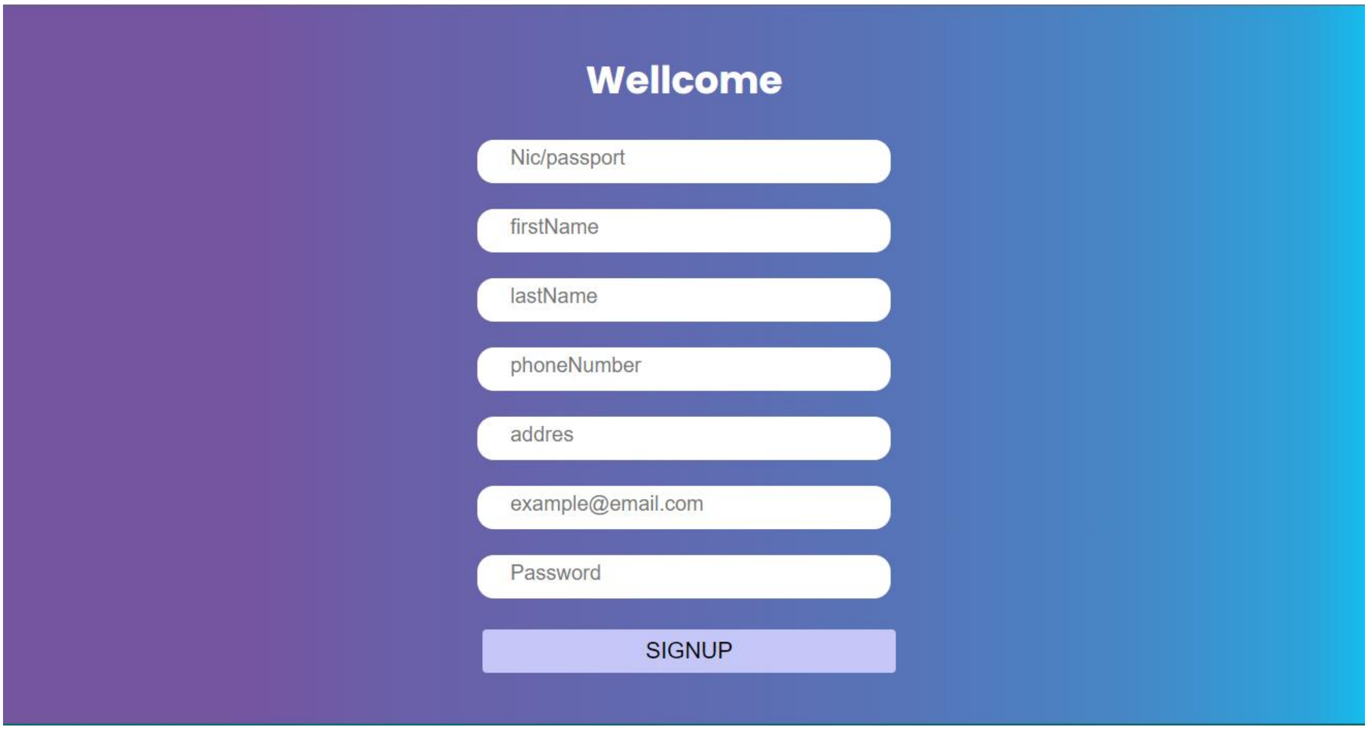


Wellcome

sasindusulochana99@gmail.com

••••••••

SIGNUP PAGE



Wellcome

Nic/passport

firstName

lastName

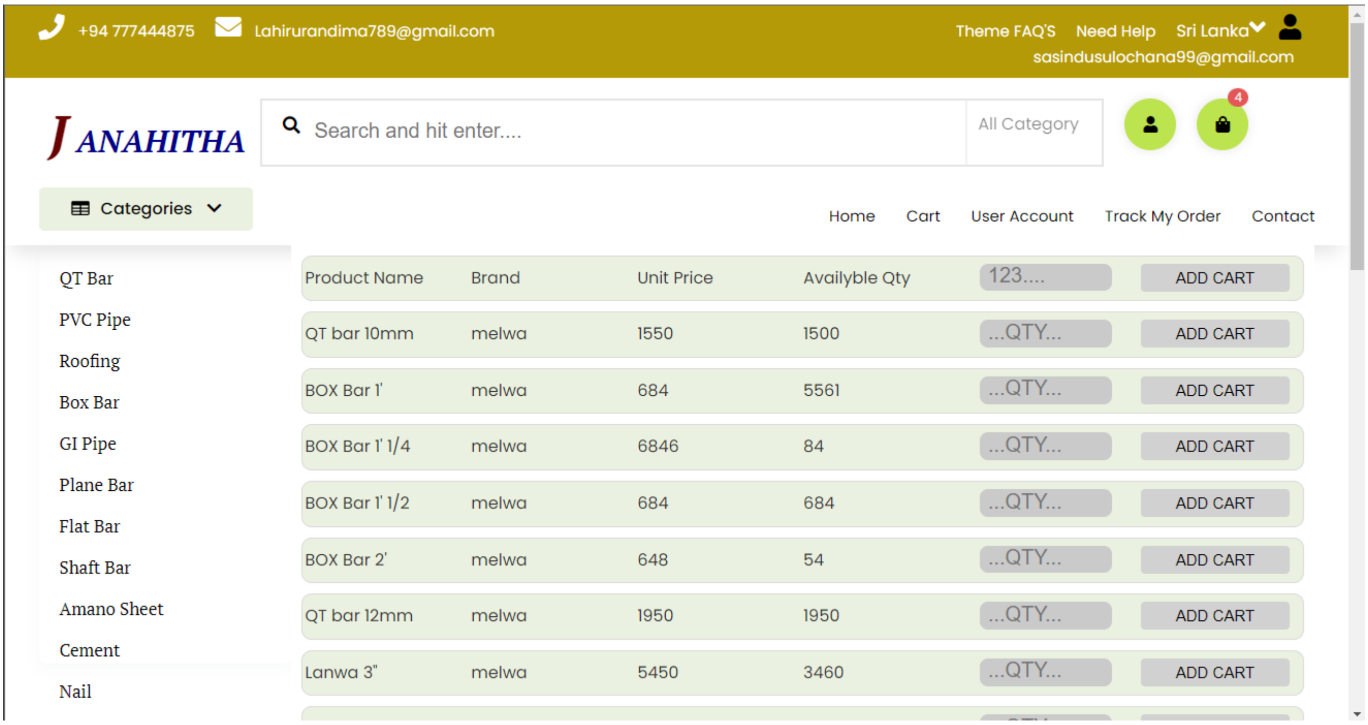
phoneNumber

addres

example@email.com

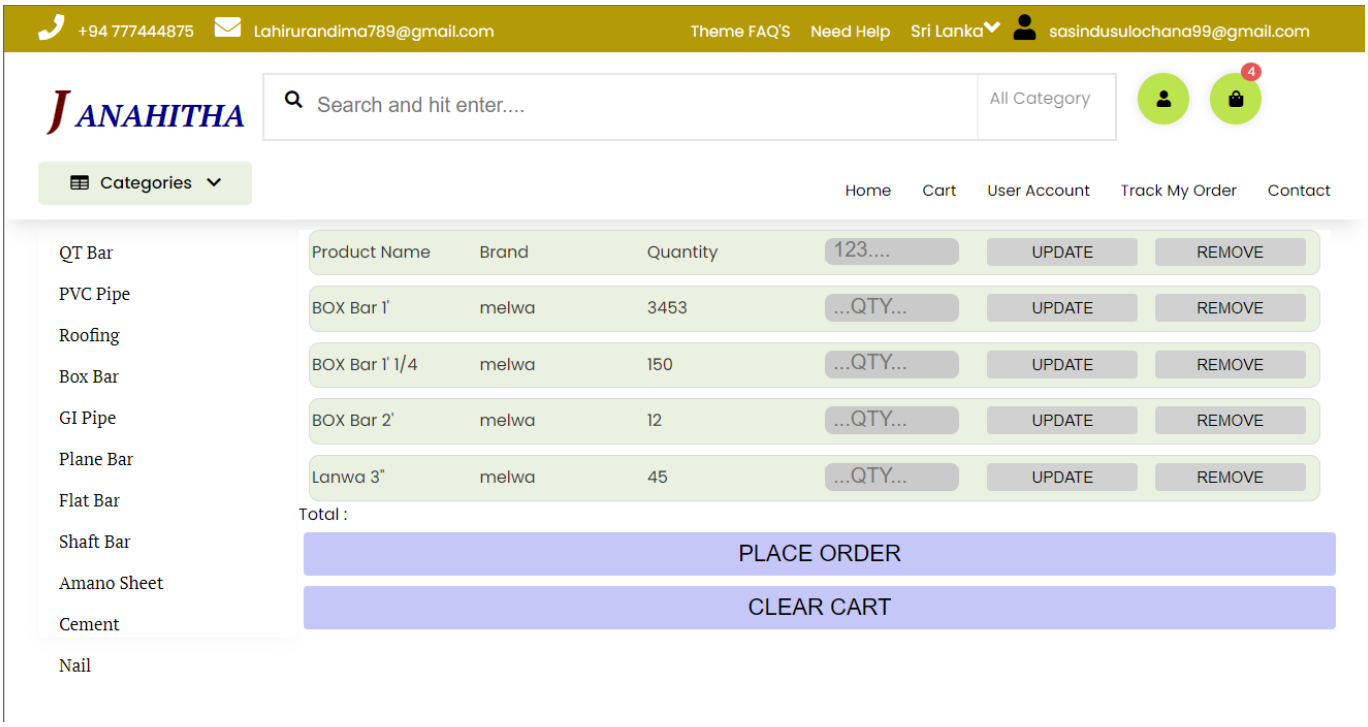
Password

MAIN PAGE



Product Name	Brand	Unit Price	Availyble Qty	123....	ADD CART
QT bar 10mm	melwa	1550	1500	...QTY...	ADD CART
BOX Bar 1'	melwa	684	5561	...QTY...	ADD CART
BOX Bar 1' 1/4	melwa	6846	84	...QTY...	ADD CART
BOX Bar 1' 1/2	melwa	684	684	...QTY...	ADD CART
BOX Bar 2'	melwa	648	54	...QTY...	ADD CART
QT bar 12mm	melwa	1950	1950	...QTY...	ADD CART
Lanwa 3"	melwa	5450	3460	...QTY...	ADD CART

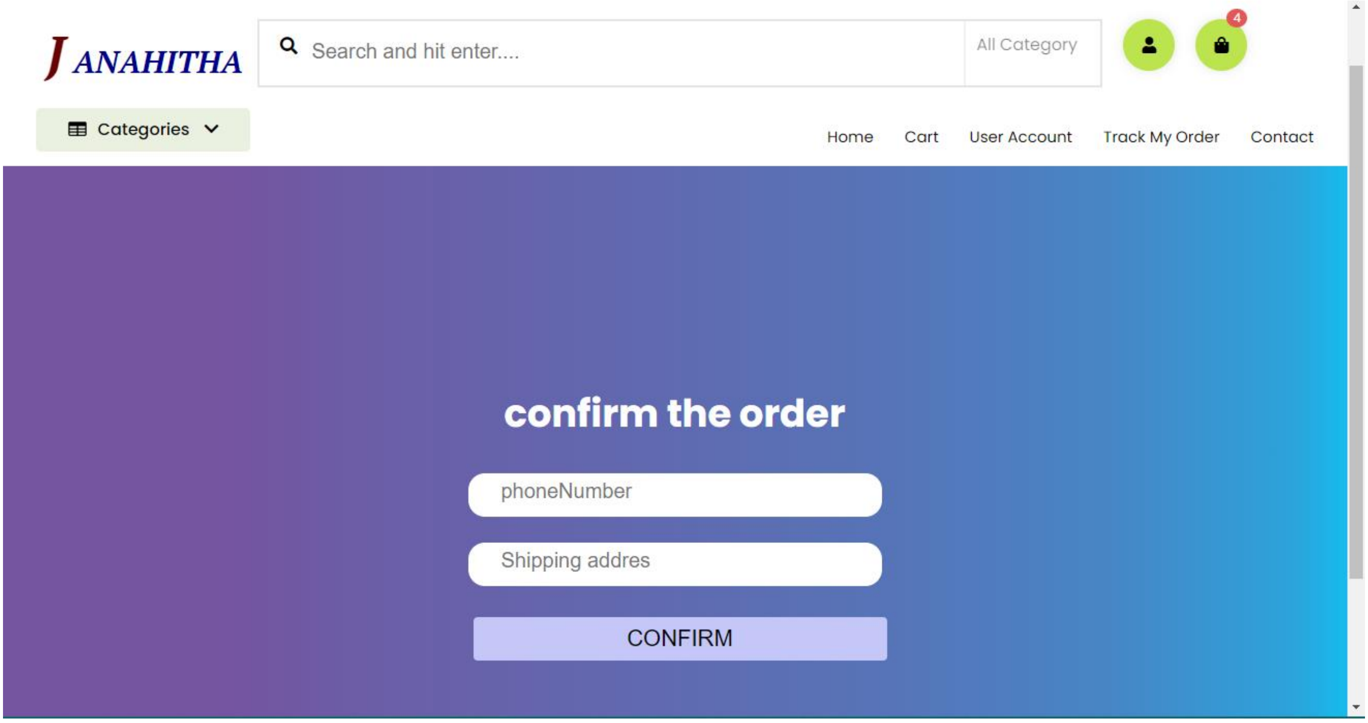
CART



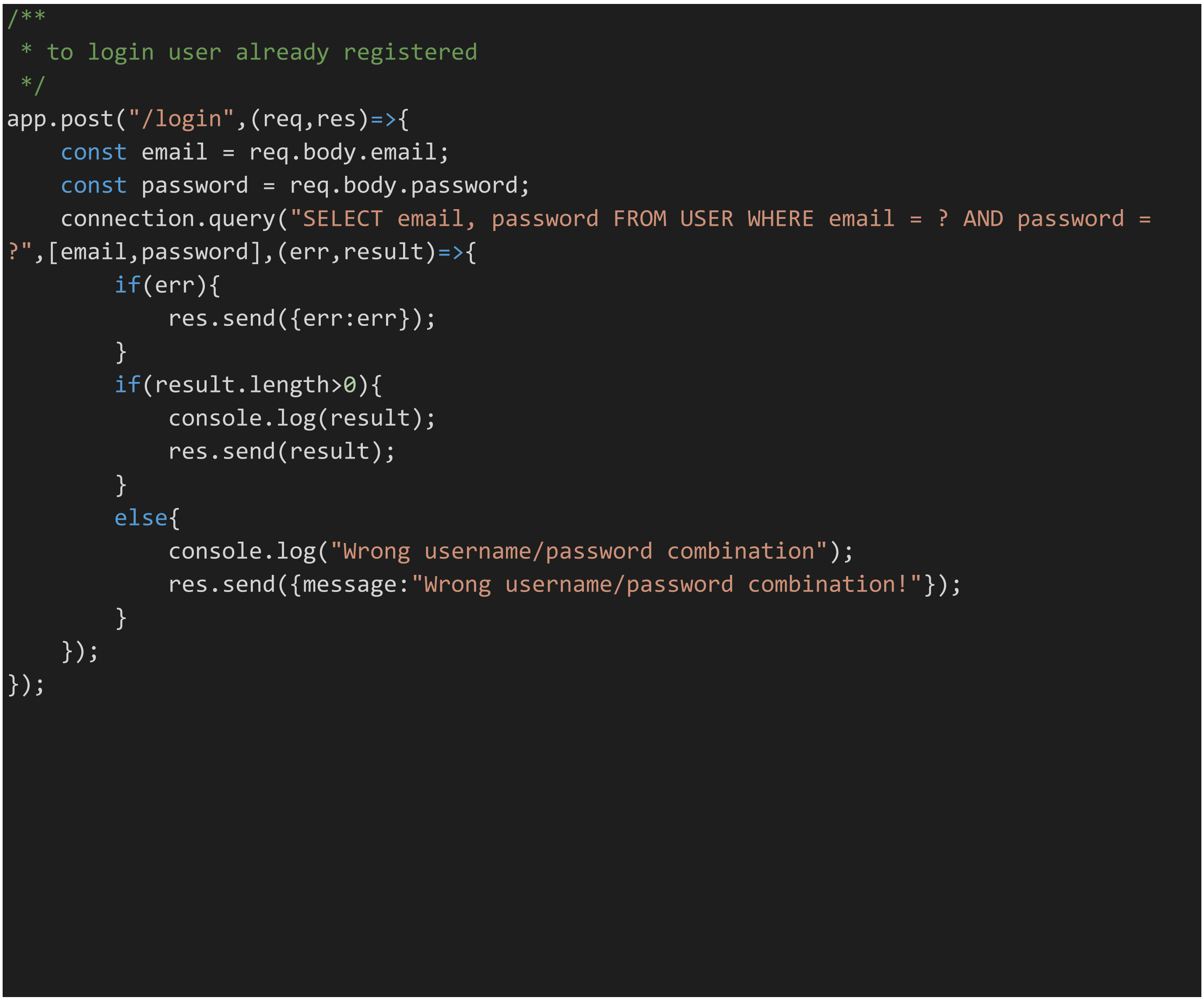
Product Name	Brand	Quantity	123....	UPDATE	REMOVE
BOX Bar 1'	melwa	3453	...QTY...	UPDATE	REMOVE
BOX Bar 1' 1/4	melwa	150	...QTY...	UPDATE	REMOVE
BOX Bar 2'	melwa	12	...QTY...	UPDATE	REMOVE
Lanwa 3"	melwa	45	...QTY...	UPDATE	REMOVE

Total :

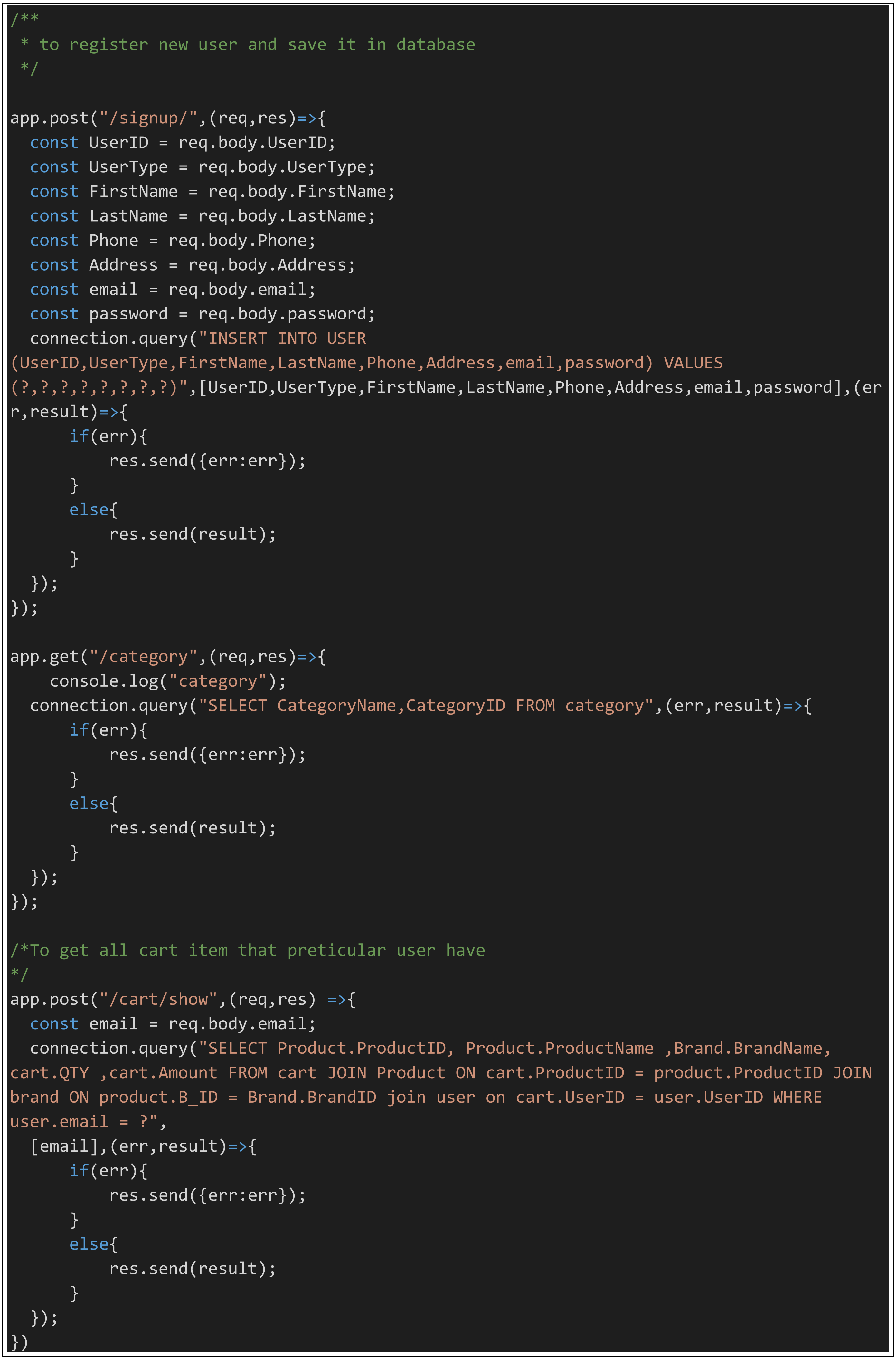
ORDER CONFIRM PAGE



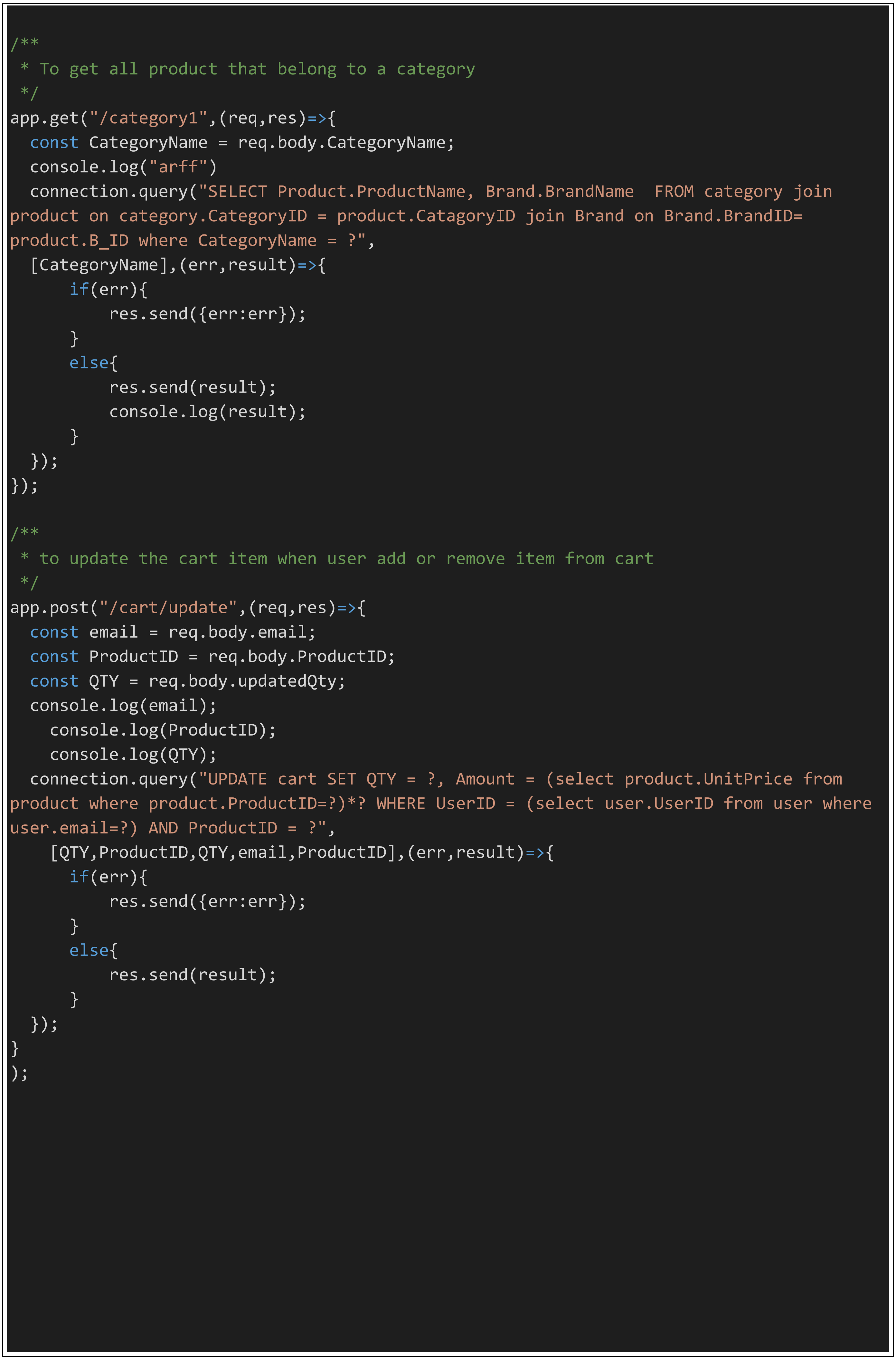
USED QUARYES



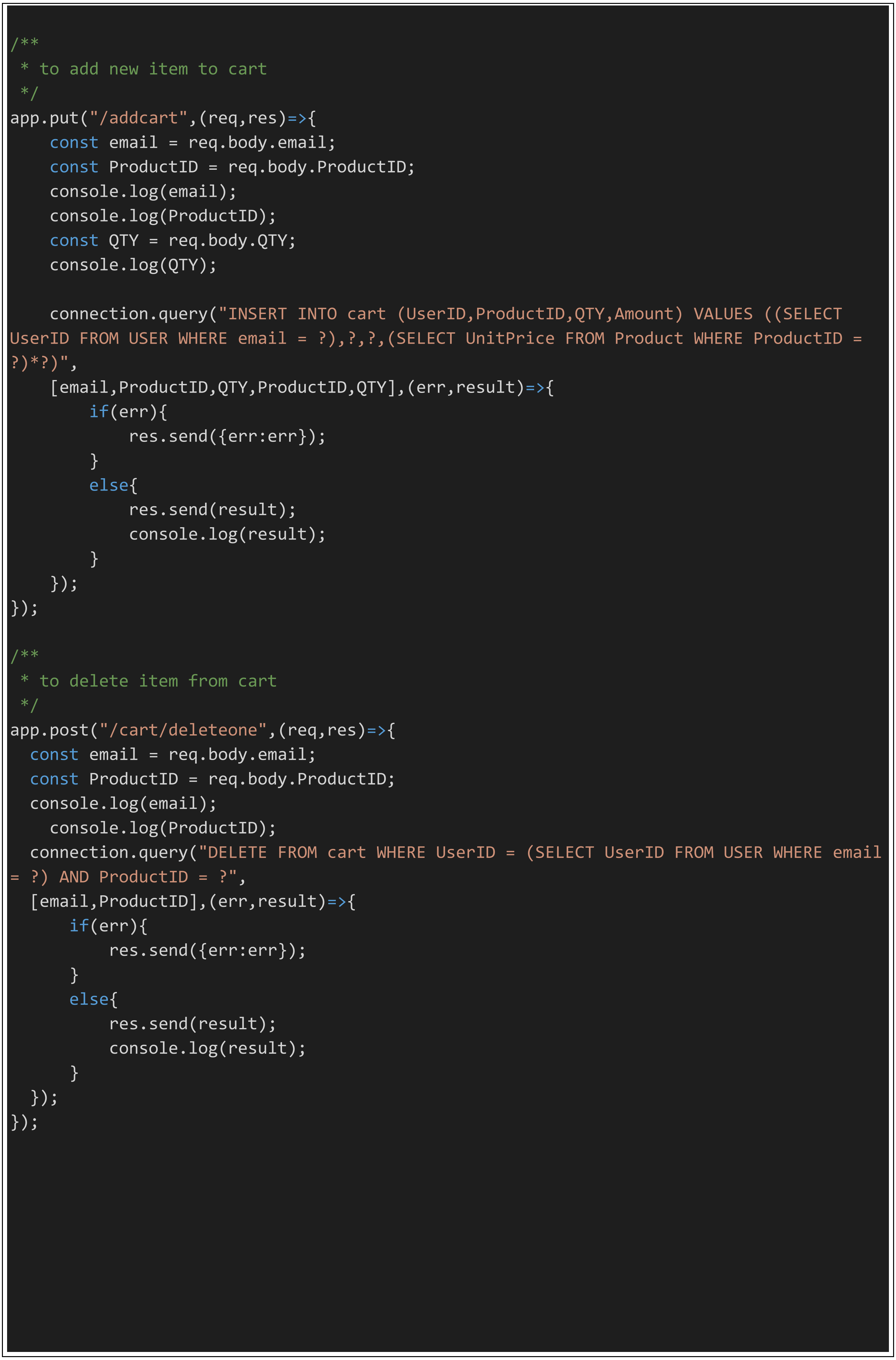
```
/**
 * to login user already registered
 */
app.post("/login",(req,res)=>{
    const email = req.body.email;
    const password = req.body.password;
    connection.query("SELECT email, password FROM USER WHERE email = ? AND password =
?",[email,password],(err,result)=>{
        if(err){
            res.send({err:err});
        }
        if(result.length>0){
            console.log(result);
            res.send(result);
        }
        else{
            console.log("Wrong username/password combination");
            res.send({message:"Wrong username/password combination!"});
        }
    });
});
```



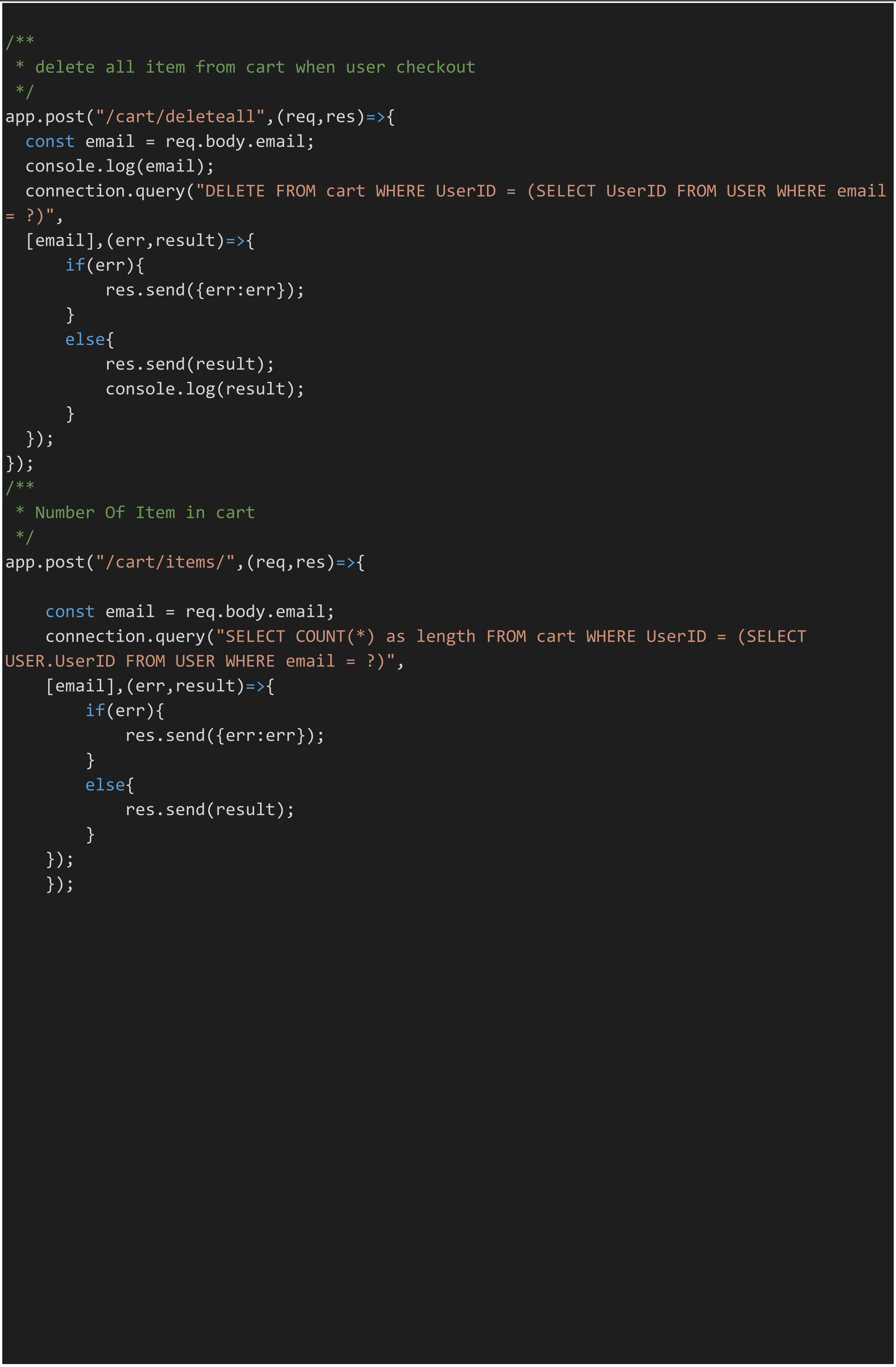
```
/**
 * to register new user and save it in database
 */

app.post("/signup/",(req,res)=>{
  const UserID = req.body.UserID;
  const UserType = req.body.UserType;
  const FirstName = req.body.FirstName;
  const LastName = req.body.LastName;
  const Phone = req.body.Phone;
  const Address = req.body.Address;
  const email = req.body.email;
  const password = req.body.password;
  connection.query("INSERT INTO USER
(UserID,UserType,FirstName,LastName,Phone,Address,email,password) VALUES
(?,?,?,?,?,?,?,?)",[UserID,UserType,FirstName,LastName,Phone,Address,email,password],(er
r,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
      }
  });
});

app.get("/category",(req,res)=>{
    console.log("category");
  connection.query("SELECT CategoryName,CategoryID FROM category",(err,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
      }
  });
});

/*To get all cart item that preticular user have
*/
app.post("/cart/show",(req,res) =>{
  const email = req.body.email;
  connection.query("SELECT Product.ProductID, Product.ProductName ,Brand.BrandName,
cart.QTY ,cart.Amount FROM cart JOIN Product ON cart.ProductID = product.ProductID JOIN
brand ON product.B_ID = Brand.BrandID join user on cart.UserID = user.UserID WHERE
user.email = ?",
  [email],(err,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
      }
  });
})
```



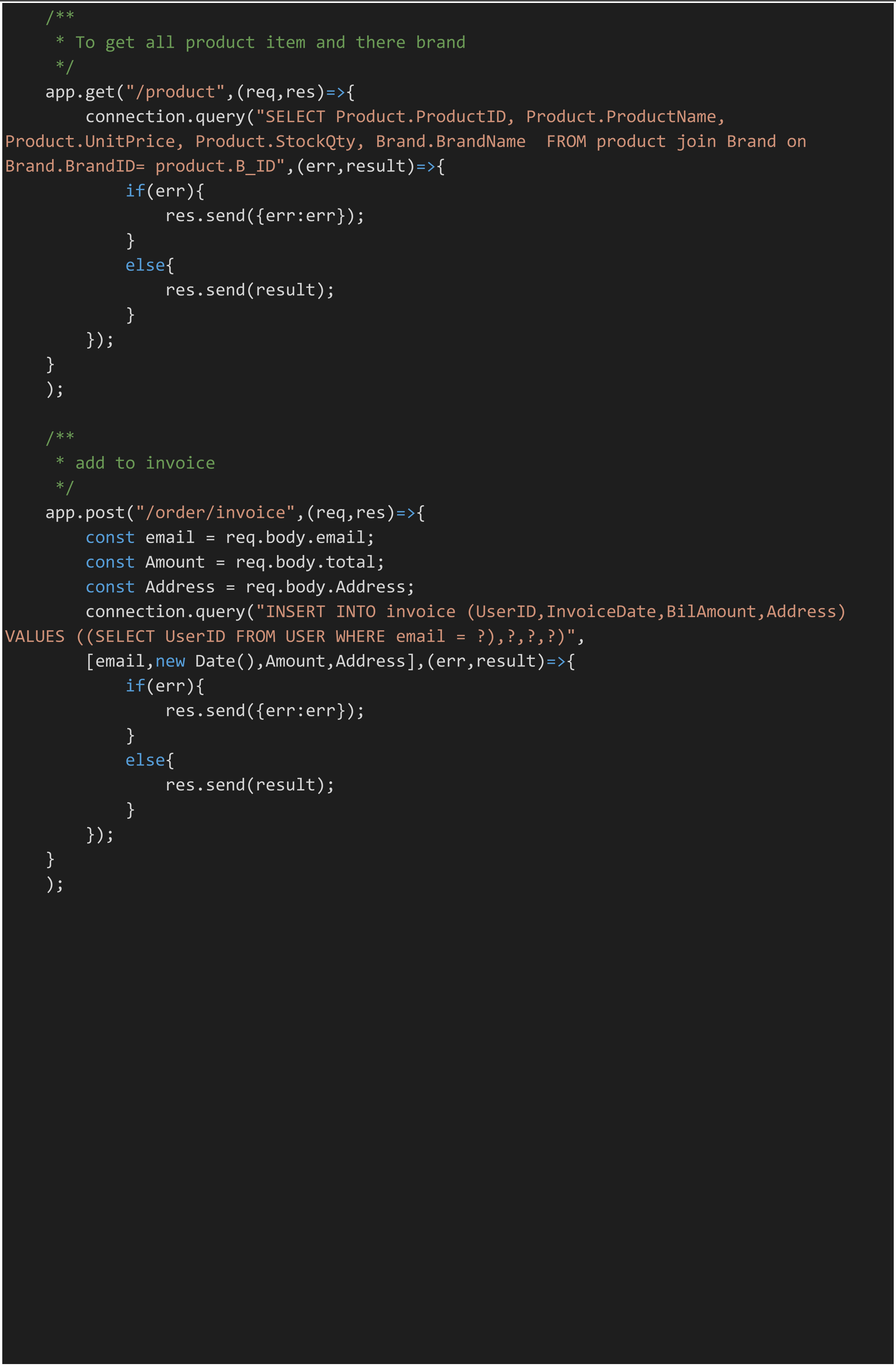
```
/**
 * To get all product that belong to a category
 */
app.get("/category1",(req,res)=>{
  const CategoryName = req.body.CategoryName;
  console.log("arff")
  connection.query("SELECT Product.ProductName, Brand.BrandName  FROM category join
product on category.CategoryID = product.CatagoryID join Brand on Brand.BrandID=
product.B_ID where CategoryName = ?",
  [CategoryName],(err,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
          console.log(result);
      }
  });
});

/**
 * to update the cart item when user add or remove item from cart
 */
app.post("/cart/update",(req,res)=>{
  const email = req.body.email;
  const ProductID = req.body.ProductID;
  const QTY = req.body.updatedQty;
  console.log(email);
    console.log(ProductID);
    console.log(QTY);
  connection.query("UPDATE cart SET QTY = ?, Amount = (select product.UnitPrice from
product where product.ProductID=?)*? WHERE UserID = (select user.UserID from user where
user.email=?) AND ProductID = ?",
    [QTY,ProductID,QTY,email,ProductID],(err,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
      }
  });
}
);
```



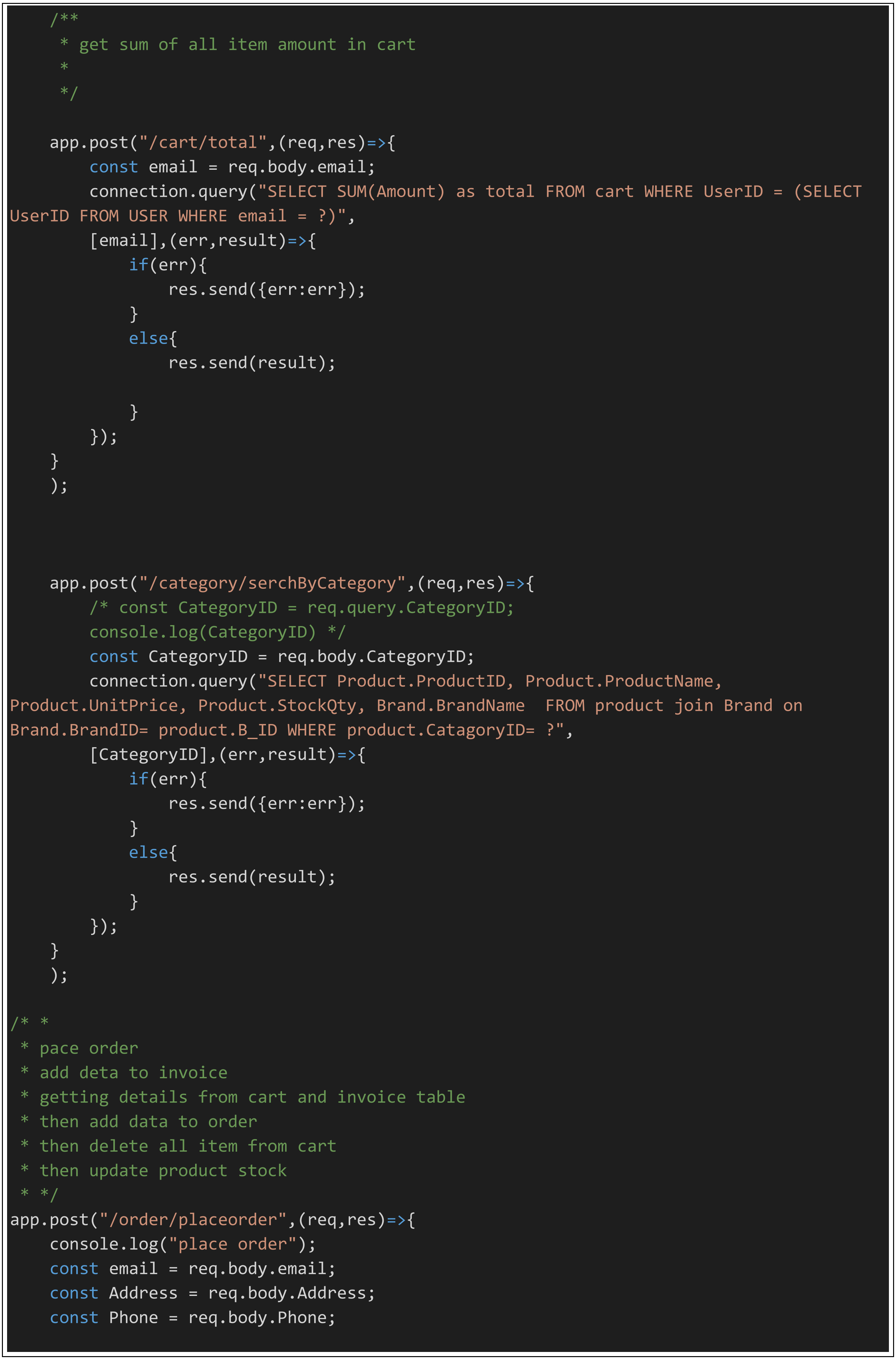
```
/**
 * to add new item to cart
 */
app.put("/addcart",(req,res)=>{
    const email = req.body.email;
    const ProductID = req.body.ProductID;
    console.log(email);
    console.log(ProductID);
    const QTY = req.body.QTY;
    console.log(QTY);

    connection.query("INSERT INTO cart (UserID,ProductID,QTY,Amount) VALUES ((SELECT UserID FROM USER WHERE email = ?),?,?,(SELECT UnitPrice FROM Product WHERE ProductID = ?)*?)",
    [email,ProductID,QTY,ProductID,QTY],(err,result)=>{
        if(err){
            res.send({err:err});
        }
        else{
            res.send(result);
            console.log(result);
        }
    });
});

/**
 * to delete item from cart
 */
app.post("/cart/deleteone",(req,res)=>{
  const email = req.body.email;
  const ProductID = req.body.ProductID;
  console.log(email);
    console.log(ProductID);
  connection.query("DELETE FROM cart WHERE UserID = (SELECT UserID FROM USER WHERE email = ?) AND ProductID = ?",
  [email,ProductID],(err,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
          console.log(result);
      }
  });
});
```



```
/**
 * delete all item from cart when user checkout
 */
app.post("/cart/deleteall",(req,res)=>{
  const email = req.body.email;
  console.log(email);
  connection.query("DELETE FROM cart WHERE UserID = (SELECT UserID FROM USER WHERE email
= ?)",
  [email],(err,result)=>{
      if(err){
          res.send({err:err});
      }
      else{
          res.send(result);
          console.log(result);
      }
  });
});
/**
 * Number Of Item in cart
 */
app.post("/cart/items/",(req,res)=>{

    const email = req.body.email;
    connection.query("SELECT COUNT(*) as length FROM cart WHERE UserID = (SELECT
USER.UserID FROM USER WHERE email = ?)",
    [email],(err,result)=>{
        if(err){
            res.send({err:err});
        }
        else{
            res.send(result);
        }
    });
    });
```



```
    /**
     * To get all product item and there brand
     */
    app.get("/product",(req,res)=>{
        connection.query("SELECT Product.ProductID, Product.ProductName, 
Product.UnitPrice, Product.StockQty, Brand.BrandName  FROM product join Brand on 
Brand.BrandID= product.B_ID",(err,result)=>{
            if(err){
                res.send({err:err});
            }
            else{
                res.send(result);
            }
        });
    }
    );

    /**
     * add to invoice
     */
    app.post("/order/invoice",(req,res)=>{
        const email = req.body.email;
        const Amount = req.body.total;
        const Address = req.body.Address;
        connection.query("INSERT INTO invoice (UserID,InvoiceDate,BilAmount,Address) 
VALUES ((SELECT UserID FROM USER WHERE email = ?),?,?,?)",
        [email,new Date(),Amount,Address],(err,result)=>{
            if(err){
                res.send({err:err});
            }
            else{
                res.send(result);
            }
        });
    }
    );
```



```
    /**
     * get sum of all item amount in cart
     *
     */

    app.post("/cart/total",(req,res)=>{
        const email = req.body.email;
        connection.query("SELECT SUM(Amount) as total FROM cart WHERE UserID = (SELECT
UserID FROM USER WHERE email = ?)",
        [email],(err,result)=>{
            if(err){
                res.send({err:err});
            }
            else{
                res.send(result);

            }
        });
    }
    );



    app.post("/category/serchByCategory",(req,res)=>{
        /* const CategoryID = req.query.CategoryID;
        console.log(CategoryID) */
        const CategoryID = req.body.CategoryID;
        connection.query("SELECT Product.ProductID, Product.ProductName,
Product.UnitPrice, Product.StockQty, Brand.BrandName  FROM product join Brand on
Brand.BrandID= product.B_ID WHERE product.CatagoryID= ?",
        [CategoryID],(err,result)=>{
            if(err){
                res.send({err:err});
            }
            else{
                res.send(result);
            }
        });
    }
    );

/* *
 * pace order
 * add deta to invoice
 * getting details from cart and invoice table
 * then add data to order
 * then delete all item from cart
 * then update product stock
 * */
app.post("/order/placeorder",(req,res)=>{
    console.log("place order");
    const email = req.body.email;
    const Address = req.body.Address;
    const Phone = req.body.Phone;
    connection.query("INSERT INTO invoice (UserID,addres,phone) VALUES ((SELECT UserID
FROM USER WHERE email = ?),?,?)",
    [email,Address,Phone]);
    connection.query("SELECT *,invoiceID FROM cart join invoice WHERE
cart.UserID=invoice.(SELECT UserID FROM USER WHERE email = ?)",
    [email],(err,result)=>{
        if(err){
            res.send({err:err});
        }
        else{
            console.log(result);
            result.forEach(element => {
                connection.query("INSERT INTO orders (ProductID,invoiceID,QTY) VALUES
(?,?,?)",
                [element.ProductID,element.invoiceID,element.QTY]);
                connection.query("DELETE FROM cart WHERE UserID = (SELECT UserID FROM
USER WHERE email = ?)",
                [email]);
                connection.query("UPDATE product SET StockQty = StockQty - ? WHERE
ProductID = ?",
                [element.QTY,element.ProductID]);
            });
        }
    });
}
);
```