A Project Report

On

Online Mobile Store

Submitted for

The partial fulfillment for the award of degree of

Bachelor of computer application (B.C.A)

Session (2019-2022)

Supervised by:

Prof. Harprabh Singh

Submitted by:

Sachin /
Akashdeep

Roll no. 10721913645 / 10721913635

Department of computer science

BARING UNION CHRISTIAN COLLEGE BATALA

ACKNOWLEDGEMENT

The satisfaction that the successful completion of any task would be Incomplete without the mention of people whose careless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We are grateful to our project guide Prof. Harprabh singhfor the guidance, inspiration and constructive suggestions that helped me in the preparation of the project.

We are also thankful to our principal Dr. Edward Masih for the successful completion of the Course.

We are thankful to Dr. Narinder Singh ,Head of Dept of computer science, Baring christain College Batala for his endless support and co-operation.

We are also thankful to our parents who give us fully help for successful completion of the project.

Sachin

Akashdeep

DECLARATION

We, hereby, declare that this project work titled “Online Mobile Store” is our original work and no part of it has been submitted for any other degree purpose or published in any other from till date.

Sachin

Akashdeep

Abstract

In Today’s digitalised world every business like Hotels, Restaurants, and Digital markets are is getting Online, We have decided to build mobile store. This project aims at providing with the facility of Buying Mobile Phones Online. The user can see and buy mobile phones online. And at the same time Owner of The store can see orders, cancel orders, and add new Products.
This project uses programming languages like, HTML, CSS3, Javascript, nodejs and MySQL.
In this e-commerce based project, user can visit website and order any derised Product, using COD (Cash on Delivery) as a payment method.

INDEX

FRONT-END AND BACK-END

FRONT-END	HTML, CSS, Javascript
BACK-END	Node.js, MySQL

Javascript

JavaScript, often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behavior, often incorporating third-party libraries.
The programs in this language are called scripts. They can be written right in a web page's HTML and run automatically as the page loads.

Scripts are provided and executed as plain text. They don't need special preparation or compilation to run.

In this aspect, JavaScript is very different from another language called Java.
Today, JavaScript can execute not only in the browser, but also on the server, or actually on any device that has a special program called the JavaScript engine

Characteristics of Javascript

Using Javascript, you can make website dynamic and much do more.

Five important characteristics make Javascript's practical nature possible –

- Validating User's Input. JavaScript is very useful while using forms
- Simple client side calculations
- Security
- Flexibility
- Familiarity

Summary

- Lightweight, and interpreted scripting programming language
- Javascript is used to create dynamic and interactive web content like applications.

- Javascript files are saved with the ".js" file extension.

SQL

SQL is a structured query language, which is computer language for storing manipulating and retrieving stored in relational database.

SQL is a standard language for accessing and manipulating databases.

What is SQL?

- SQL stands for Structured Query Language.
- SQL lets you access and manipulate databases.
- SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.

What Can SQL do?

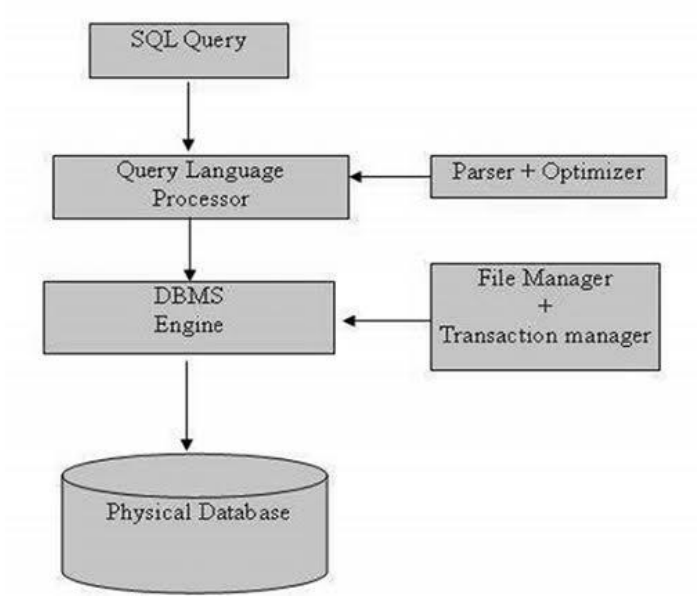
- SQL can execute queries against a database.
- SQL can retrieve data from a database.
- SQL can insert records in a database.
- SQL can update records in a database.
- SOL can delete records from a database.
- SQL can create new databases.
- SQL can create new tables in a database.
- SQL can create stored procedures in a database.
- SQL can create views in a database.
- SQL can set permissions on tables, procedures, and views.

History of SQL

- 1970-Dr.Edger F. “Ted” cod of IBM is known as the father of relation database . He described a relational model for database.
- 1974 – Structured Query Language appeared.
- 1978 – IBM Worked to developed cod’s ideas and released product named system/R.
- 1986-IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by relational Software and its later becoming oracle.

SQL Process

When you are executing and SQL commands for any RDBMS, the system determines the best way to carry out your request and SQL engine figures out how to interpret the task.



SDLC

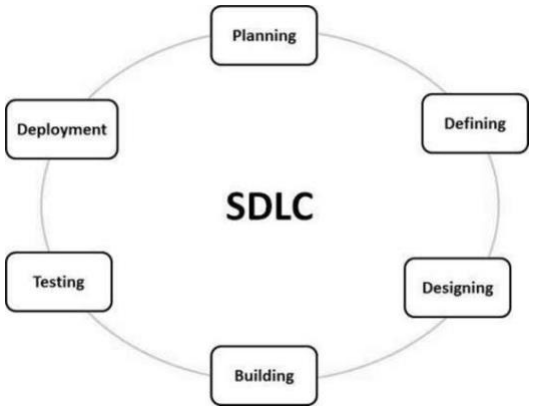
Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality softwares. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

- SDLC is the acronym of Software Development Life Cycle.
- It is also called as Software Development Process.
- SDLC is a framework defining tasks performed at each step in the software development process.
- ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

What is SDLC?

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



A typical Software Development Life Cycle consists of the following stages –

Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an SRS (Software Requirement Specification) document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS – Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third

party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. Are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

SDLC Models

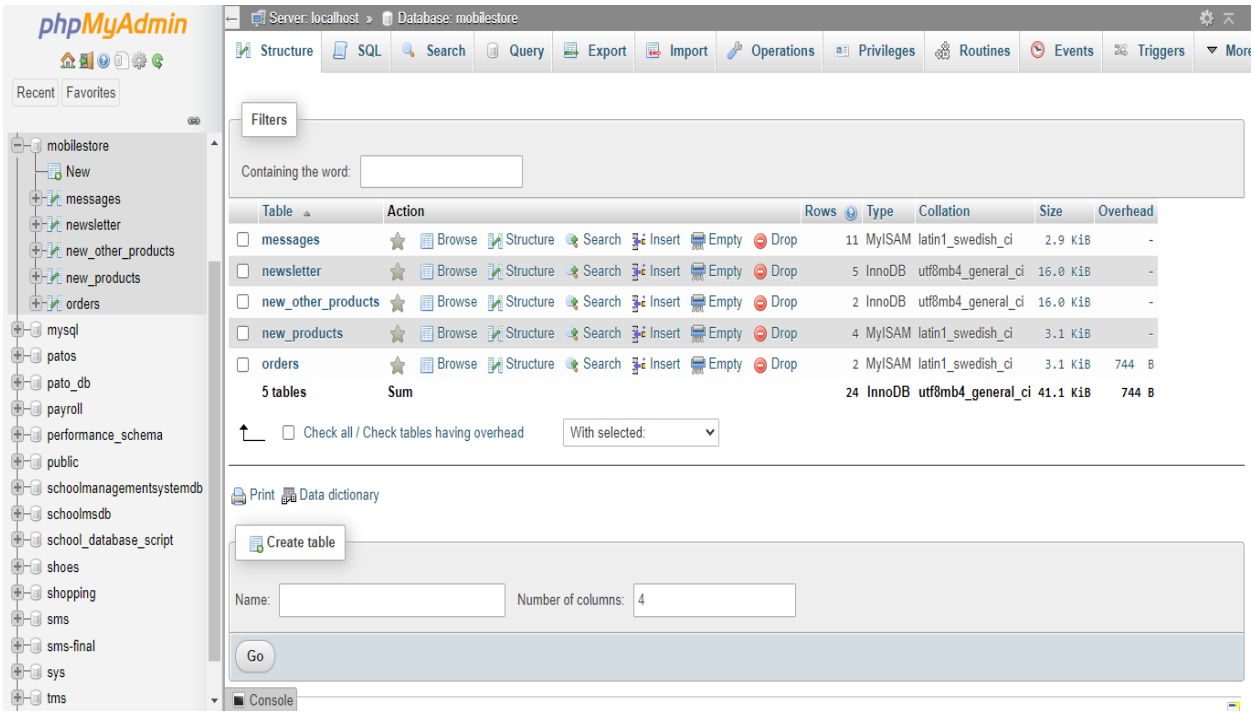
There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development.

Following are the most important and popular SDLC models followed in the industry –

- Waterfall Model
- Iterative Model
- Spiral Model
- V-Model
- Big Bang Model

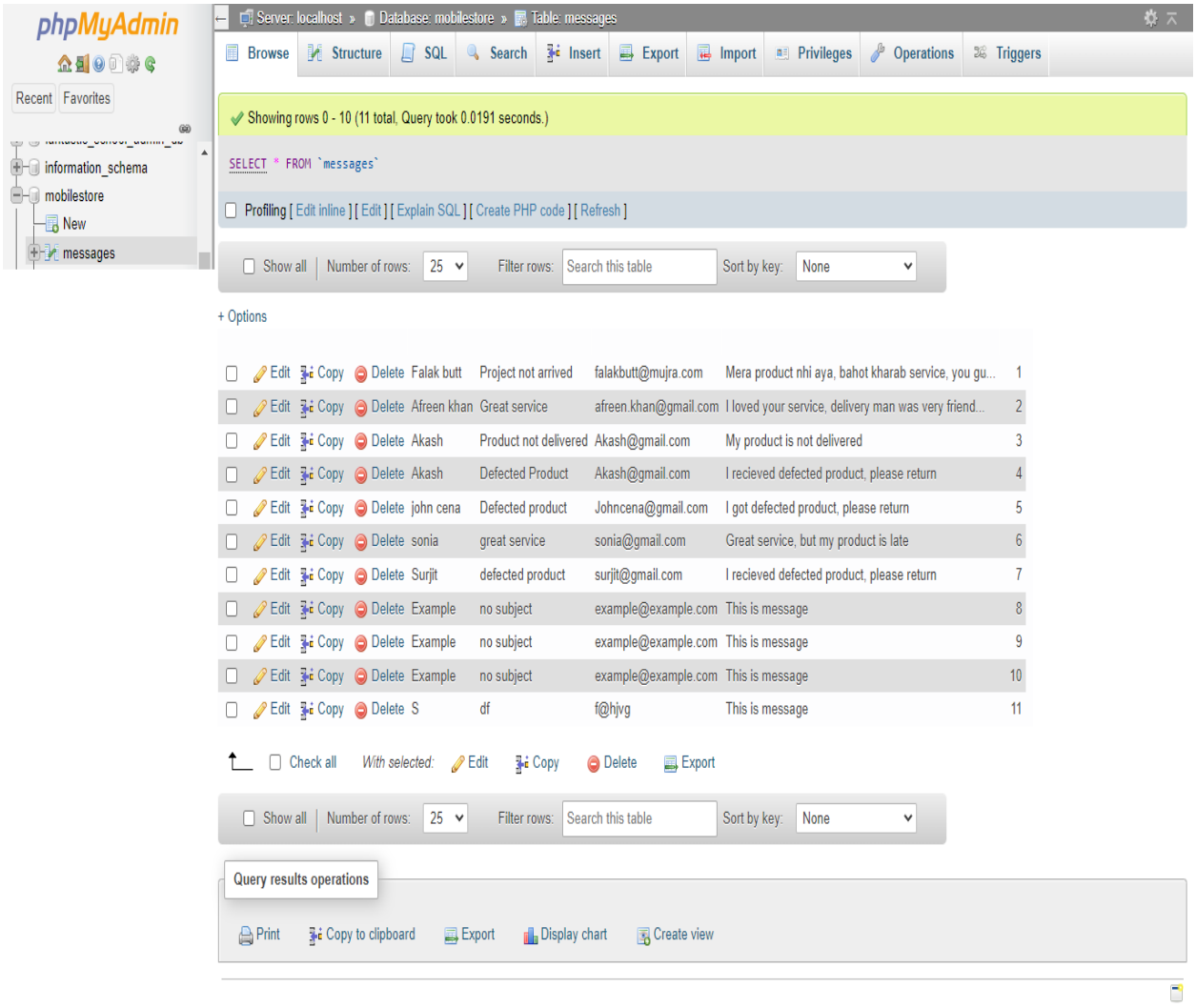
Database

Database Name: mobile store



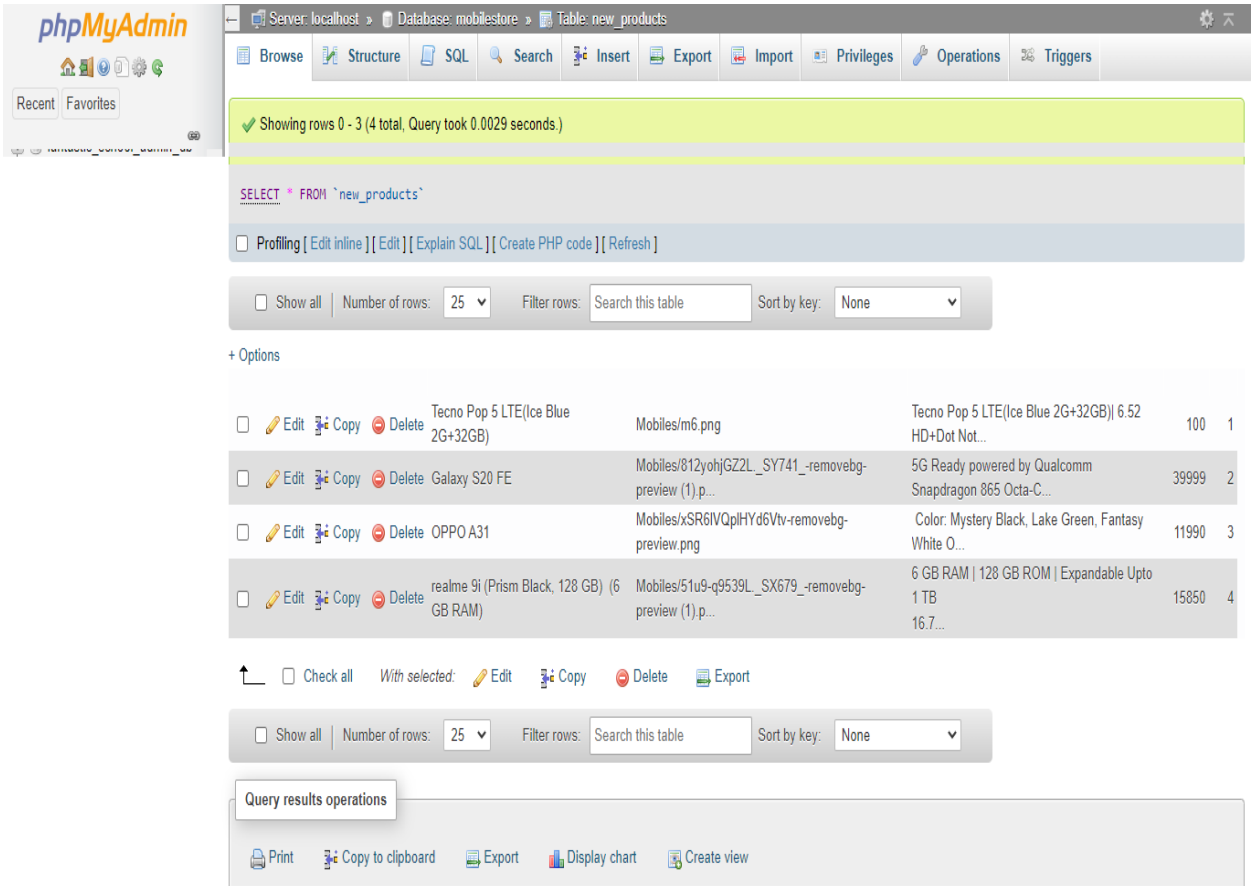
This Database store the information about users and products.

Table Name: messages



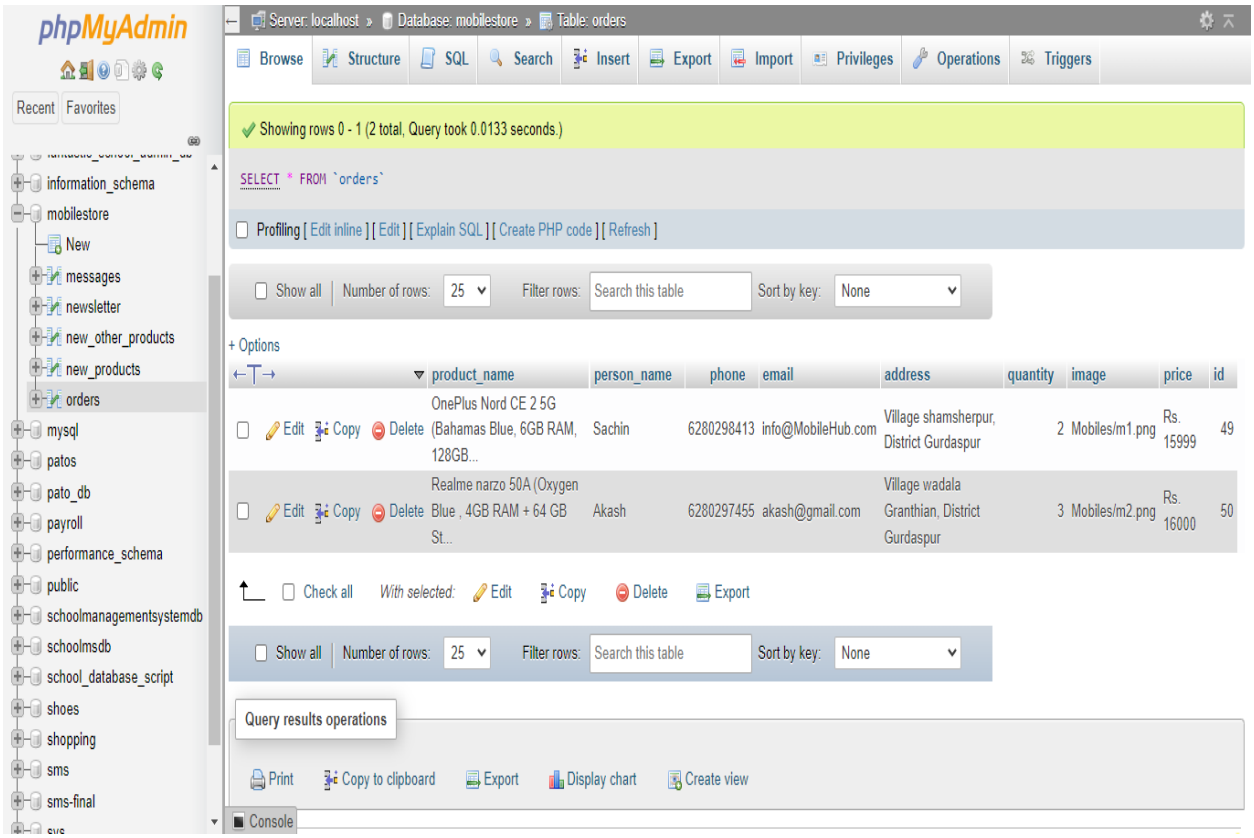
This table store the messages sent by users.

Table Name: new_products



This table the information of newly added products.

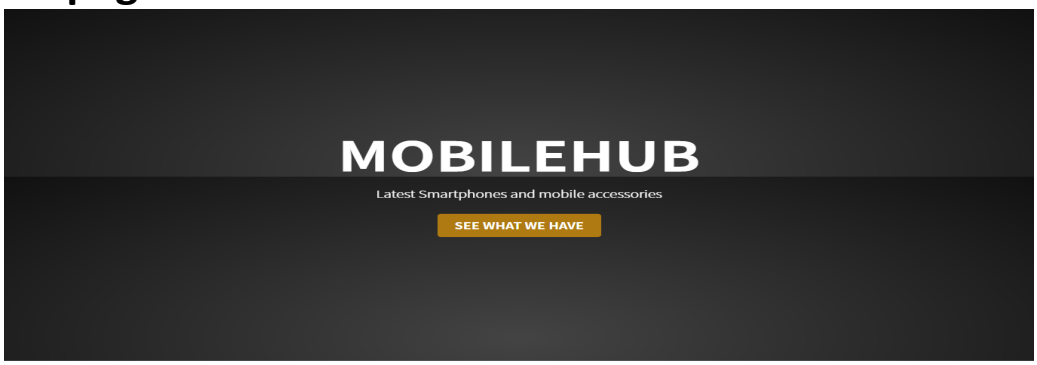
Table Name: orders

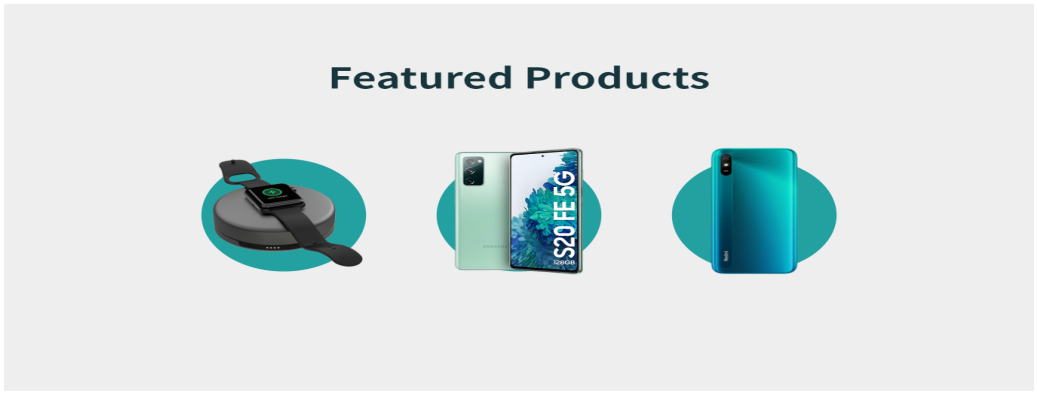


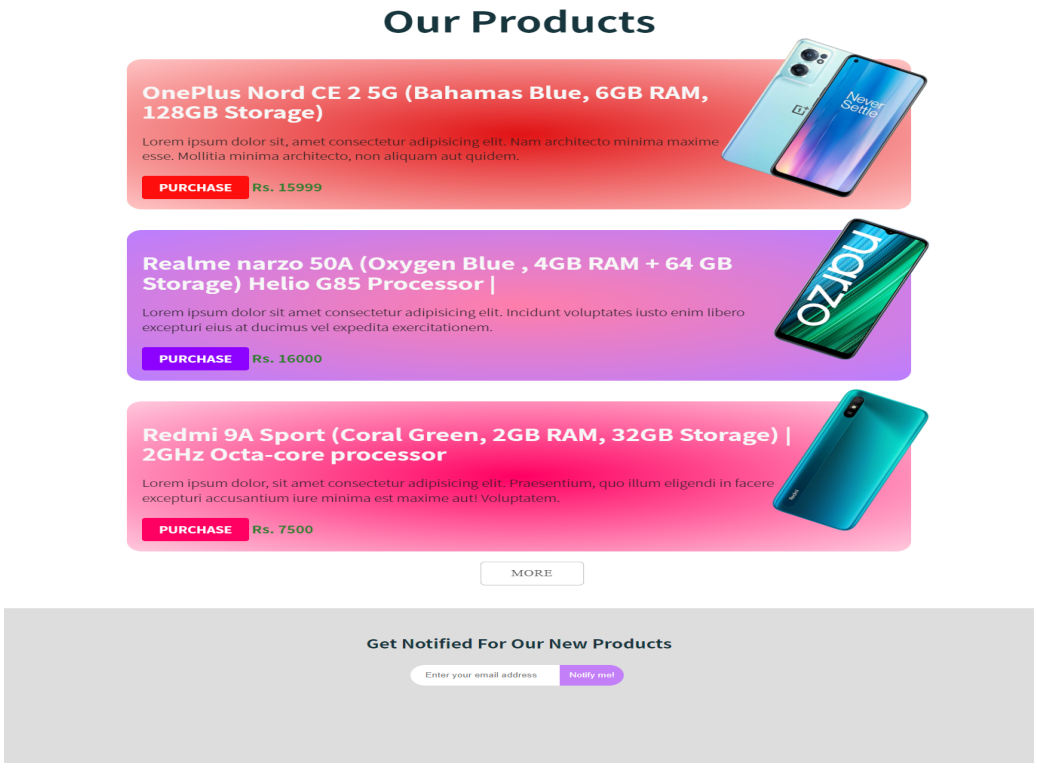
This table store the information of booked orders.

Screenshots

Home page

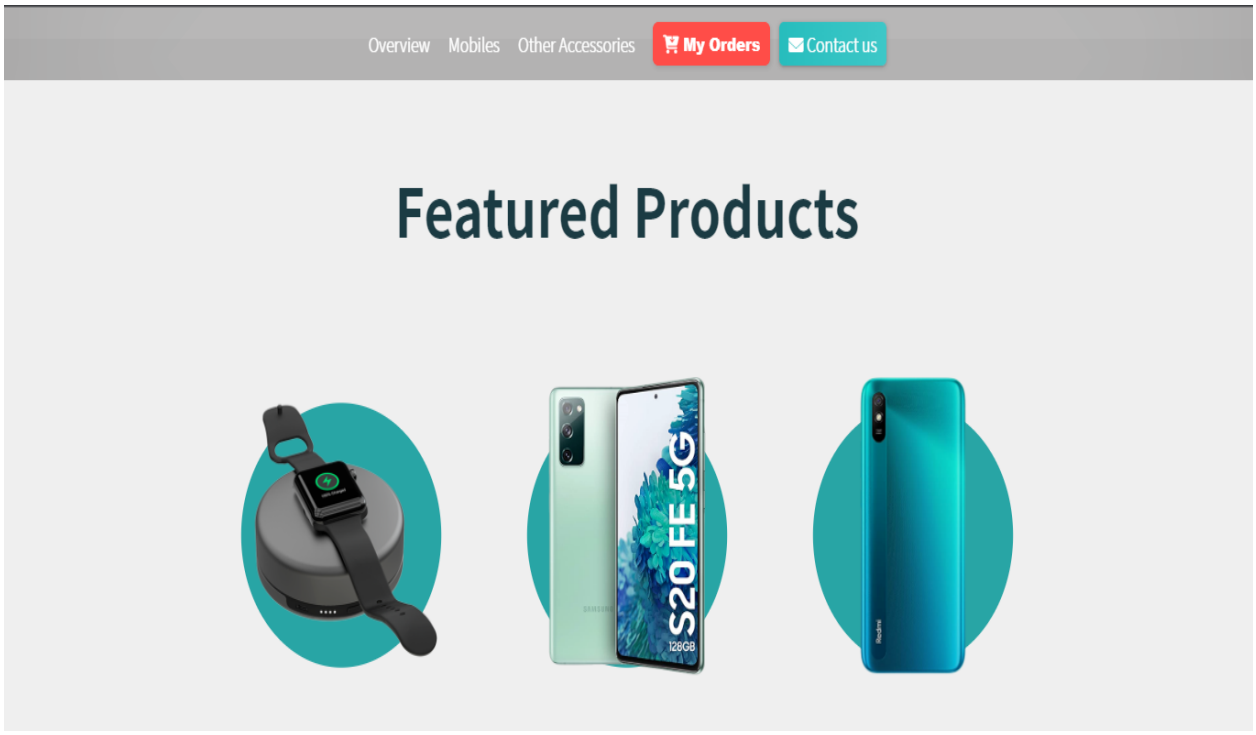






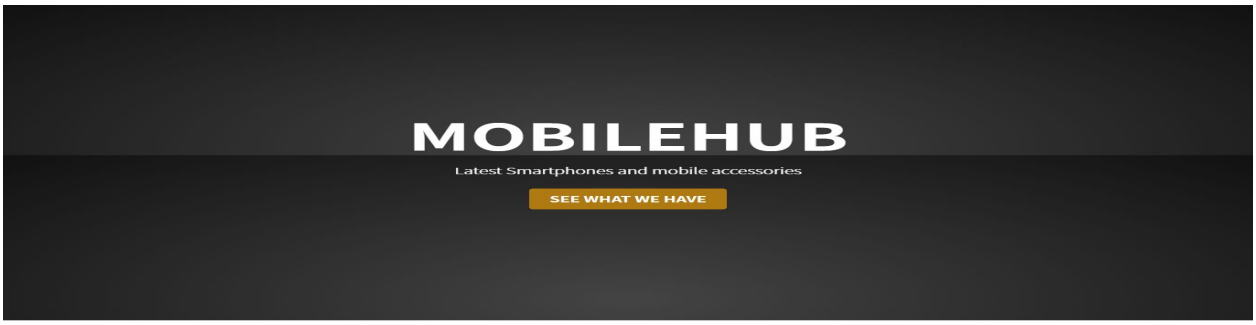
This is home page from here we can go to other sections of website.

Navbar



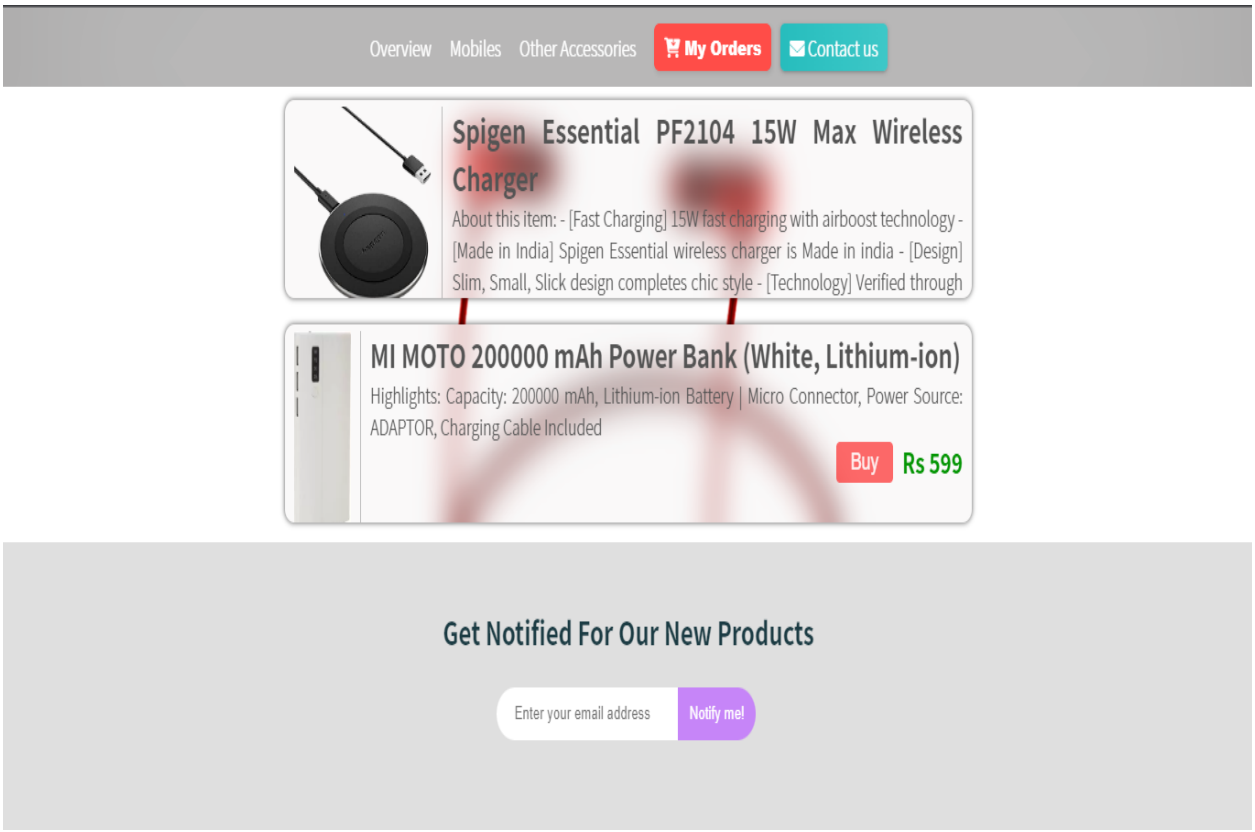
From Navbar we can click different button at open different sections
Of store

Mobile Section



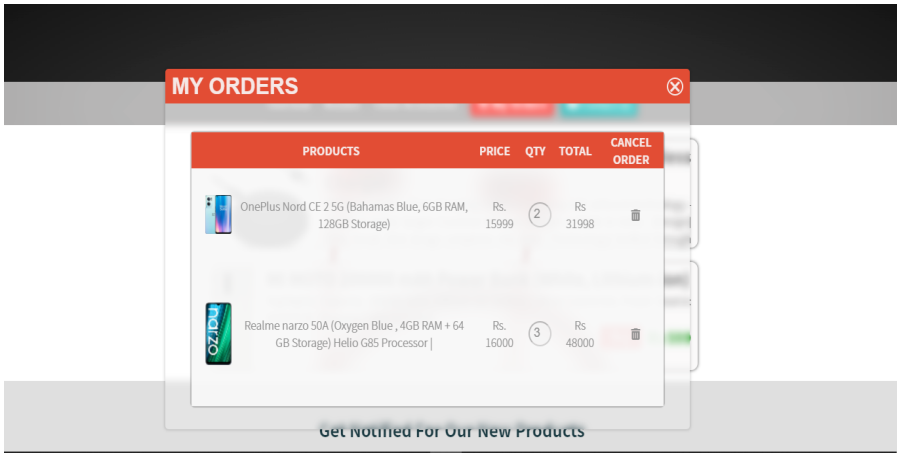
By Clicking on Mobiles button it leads to Mobiles section

Other Accessories



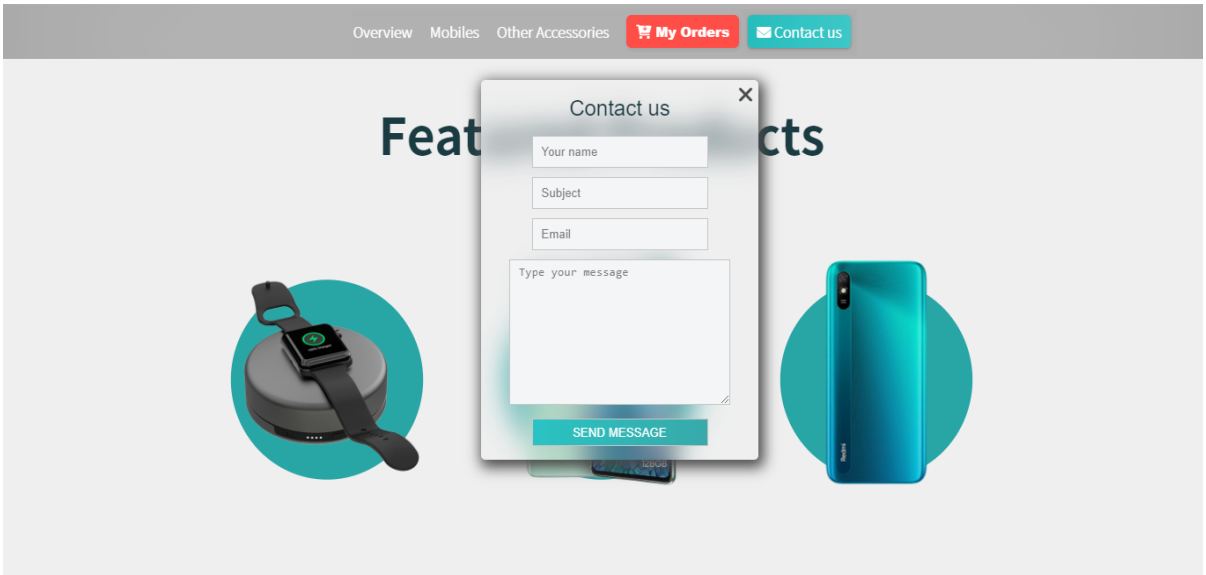
By clicking on Other Accessories button we can go to another section, where We can buy other mobile related products

My Orders



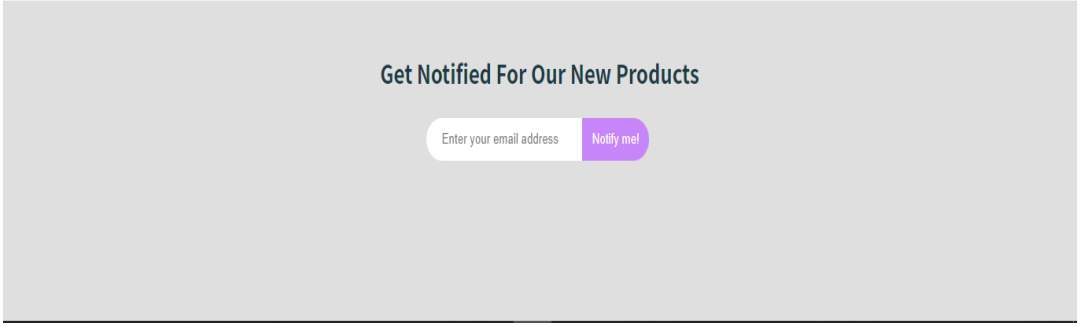
Clicking on MY ORDERS button leads to Orders section, where user can Review or delete orders.

Contact us Box



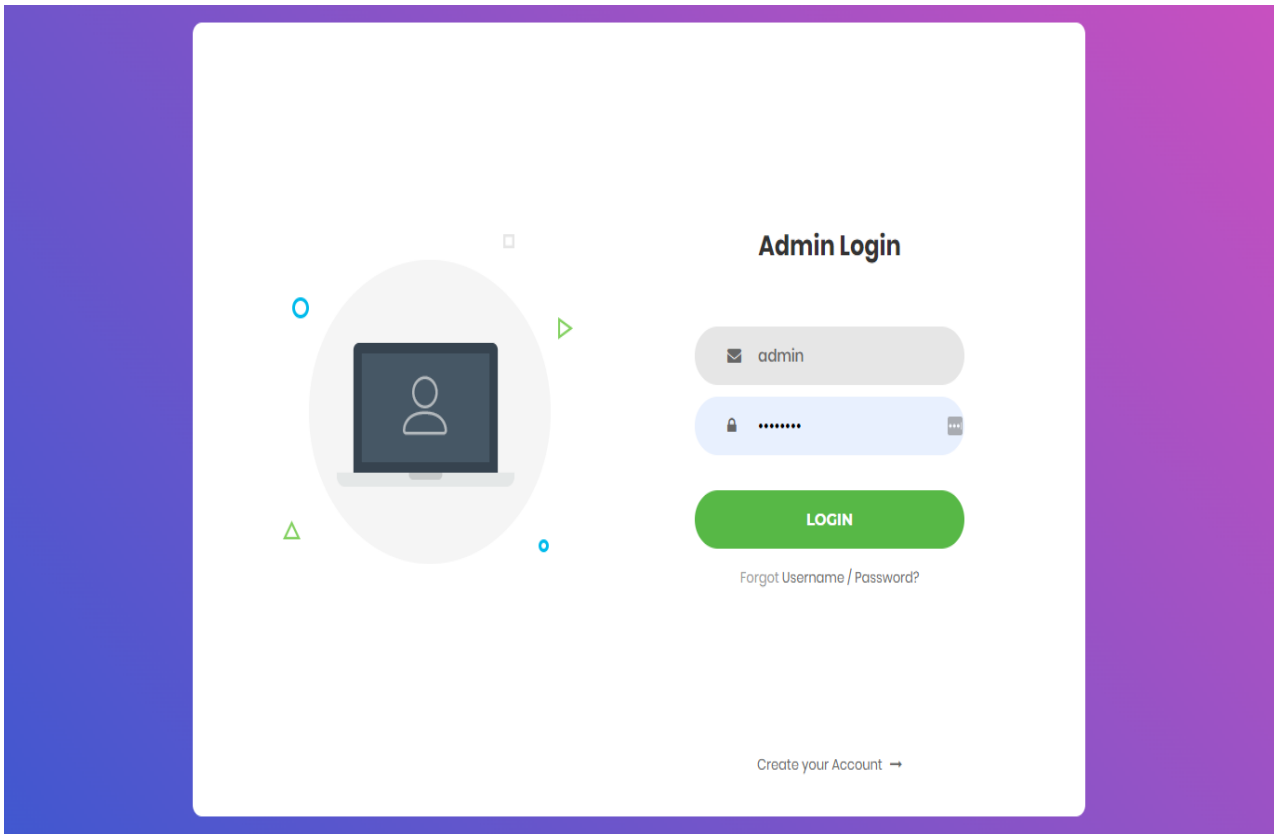
In this section we can send our feedback

Newsletter

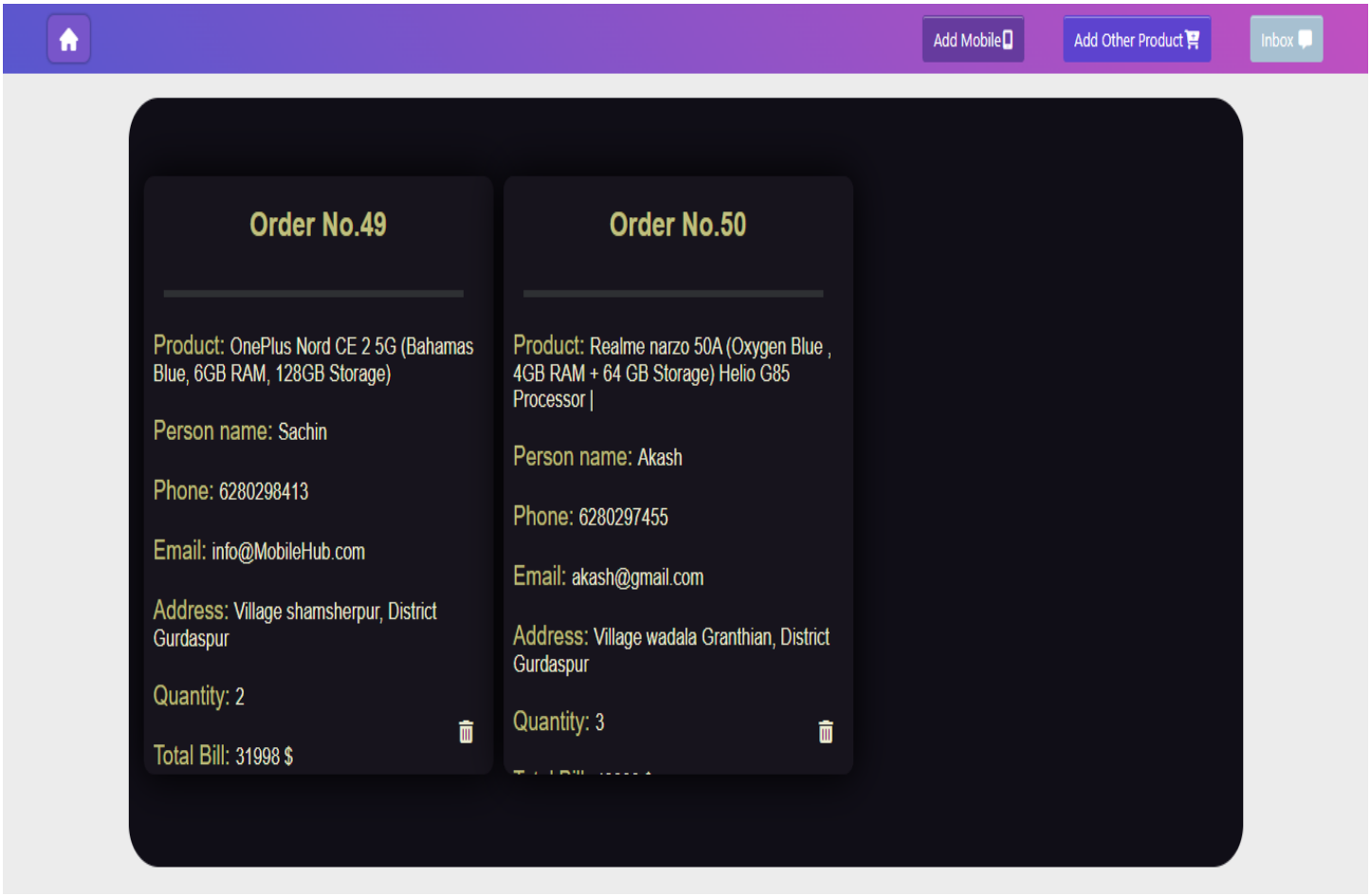


At the bottom of page, we can enter our email at subscribe to Newsletter about new products

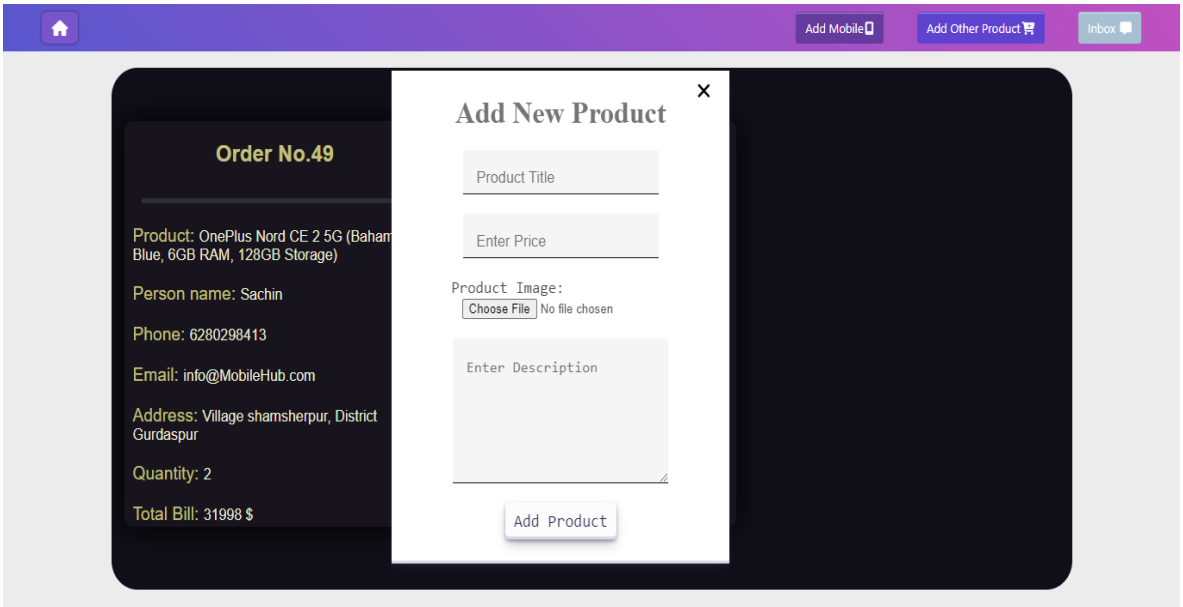
Admin login page



Admin Panel : Orders

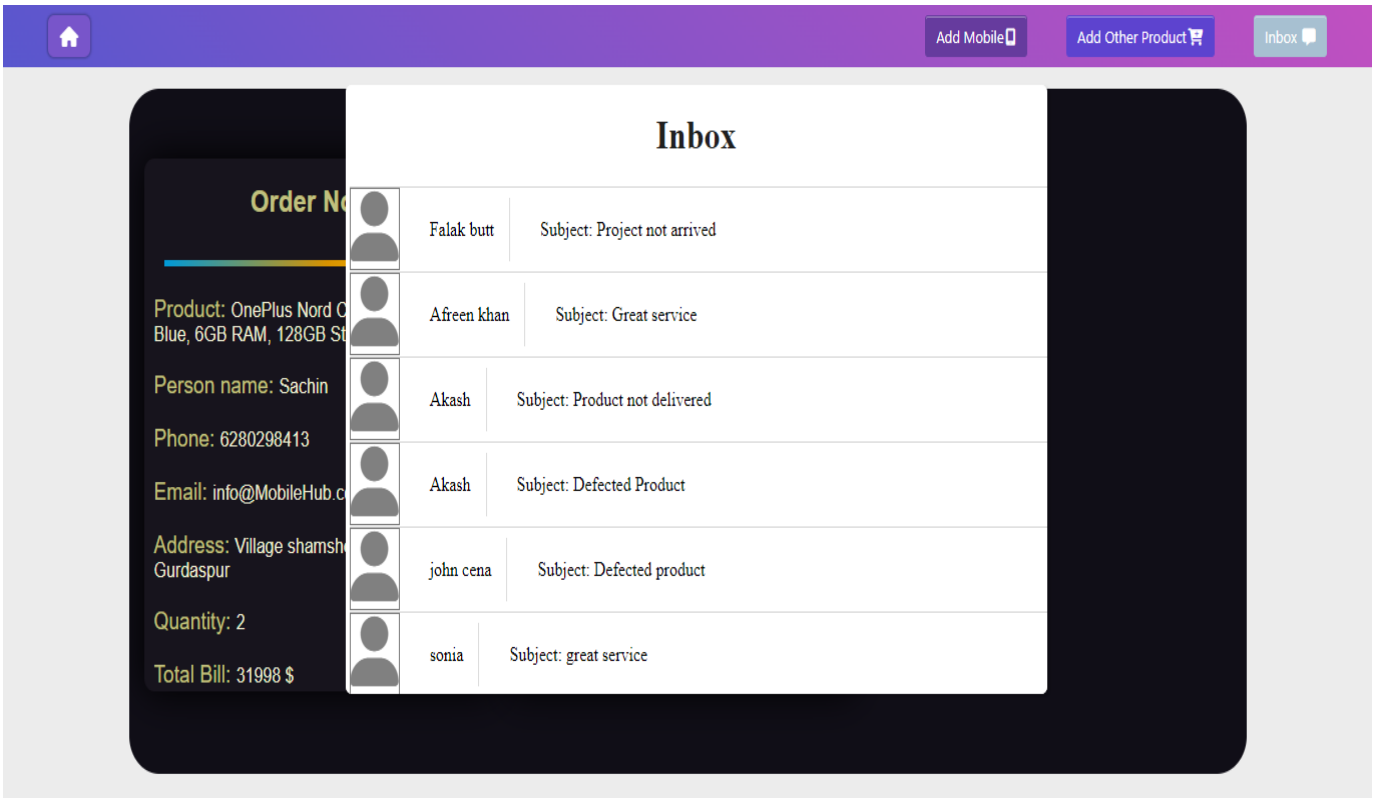


Add new product



Form to add new product

Inbox



In inbox we can see feedback sent by users

Homepage coding in HTML

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <link href="https://fonts.googleapis.com/css2?
family=Noto+Sans+JP:wght@300;700&display=swap"
rel="stylesheet">
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/
all.min.css" integrity="sha512-
KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvq
cyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g=="
crossorigin="anonymous" referrerpolicy="no-referrer" />
  <!-- font-family: 'Noto Sans JP', sans-serif; -->
  <link rel="stylesheet" href="styles.css">
  <link href="https://unpkg.com/aos@2.3.1/dist/aos.css"
rel="stylesheet">
  <script src="https://unpkg.com/aos@2.3.1/dist/aos.js"></script>
  <script defer
src="https://use.fontawesome.com/releases/v5.15.4/js/all.js"
integrity="sha384-
rOA1PnstxnOBLzCLMcre8ybwbTmemjzdNlILg8O7z1lUkLXozs4
DHonlDtnE7fpc" crossorigin="anonymous"></script>
  <link rel="stylesheet" href="path/to/font-awesome/css/font-
awesome.min.css">
  <link rel="shortcut icon" href="./logos/Mobile Store-logos.jpeg"
type="image/x-icon">

  <title>MobileHub | Mobile Store</title>
</head>
```

```
<body>
  <header id="header" class="hero">

    <img class="" src="" alt="" id="header-img">

    <div class="container spacing">
      <h1 class="primary-title animate__animated animate__flipInY">MobileHub</h1>
      <p class="animate__animated animate__flipInX">Latest Smartphones and mobile
accessories</p>
      <a href="#featured" class="btn animate__animated animate__rotateInUpLeft animate__delay-
1s">See what we have</a>
    </div>

  </header>

  <main>

    <nav id="nav-bar">
      <a href="#featured" id="overview" class="nav-link">Overview</a>
      <a class="nav-link show-mobiles" id="mobileLink" style="cursor: pointer;">Mobiles</a>
      <a class="nav-link" id="other-accessories" style="cursor: pointer;">Other Accessories</a>
      <a type="button" class="my-orders-btn" ><i class="fa-solid fa-cart-arrow-down"></i> My
Orders</a>
      <a type="button" id="reg-btn" ><i class="fa-solid fa-envelope"></i> Contact us</a>
    </nav>
<!-- My orders -->
  <div class="my-orders-container">
    <div class="myorders-top">
      <h1 class="myorders-h1">MY ORDERS</h1>
      <i class="fa-solid fa-xmark" id="close-myorders"></i>
    </div>

    <div class="myorders-table">
      <table>
        <thead class="myorders-headings">
          <th>PRODUCTS</th>
          <th>PRICE</th>
          <th>QTY</th>
          <th>TOTAL</th>
          <th>CANCEL ORDER</th>
        </thead>
        <tbody class="myorders-body">
          <!-- <tr class="table-row">
            <td class="myorder-title"><img src="https://via.placeholder.com/50" style="display: inline;"
alt=""><p style="display: inline;">Lorem, ipsum dolor sit amet consectetur adipisicing elit.
fdgfdhgdhgddgfdhdg</p></td>
            <td class="myorder-price">£59.99</td>
            <td class="qty"><input type="number" name="" id="" disabled value="3"></td>
            <td class="myorder-total">£182.95</td>
            <td><i class="fa-solid fa-trash-can"></i></td>
          </tr> -->
        </tbody>
      </table>
    </div…
```

Note: Rest of the code can be seen inside index.html

Javascript code

```
const purchaseBtn = document.querySelectorAll('.btn-purchase');
const buyContainer = document.querySelector('.buy-container');
const closeBuy = document.querySelector('.close-buy');
const successMsg = document.querySelector('.buy-success');
const inputs = document.querySelectorAll('.buy-input');
const buyBtn = document.querySelector('.buy-btn');
const featuredItem = document.querySelectorAll('.featured_item');
const notifyMe = document.getElementById('notify-me');
const notifyMsg = document.querySelector('.notify-msg');
const inputArr = [];

function hideNotify(){
   notifyMsg.style.display = 'none';
}
notifyMe.addEventListener('click', ()=>{
   notifyMsg.style.display = 'flex';
   setTimeout(hideNotify, 3000)
})

function hideSuccessMsg(){
   successMsg.style.display = "none";
}
function showSuccess(){
   inputs.forEach((inp)=>{
      if(inp.value.length>0){
         inputArr.push(inp.value);
      }
   });
   if(inputArr.length>=5){
      successMsg.style.display = "flex";
      buyContainer.style.display = 'none';
      setTimeout(hideSuccessMsg, 3000);
   }
   else{
      alert("Please fill all your details");
   }
}
```

```
buyBtn.addEventListener('click', showSuccess);

/*contact us*/

const contactBtn = document.getElementById('reg-btn');
const contactUs = document.querySelector('.contact-us');
const contactClose = document.querySelector('.close-contact');

contactBtn.addEventListener('click',()=>{
   contactUs.classList.toggle('active');
   contactUs.style.transition = 'all 200ms linear';
   contactUs.style.transform = 'scale(0.1)';
   setTimeout(()=>{
      contactUs.style.transform = 'scale(1)'
   }, 1);
});

contactClose.addEventListener('click',()=>{
   contactUs.style.transform = 'scale(0.1)';
   setTimeout(()=>{
      contactUs.classList.remove('active')
   }, 200);
})

//MakingTabs:
//Show Mobile Section
const mobiles = document.querySelector('.mobiles');
const showMobiles = document.getElementById("mobileLink");
const mainSection = document.getElementById('main-sec');

const products = document.getElementById('products');
const productsDiv = document.createElement('div');
productsDiv.appendChild(products);

const moreBtn = document.getElementById('more-pd-btn');
moreBtn.style.display = 'none';
mobiles.innerHTML = productsDiv.innerHTML;

//Adding content to mobile Section
const moreMobiles = document.createElement('div');
const pContainer = document.createElement('div');
pContainer.setAttribute('class','container');
pContainer.appendChild(moreMobiles);
```

```
function showmore(){
    mobiles.style.display = 'block';
    mobiles.appendChild(pContainer);
    const div = document.createElement('section');
    div.appendChild(mobiles);
    mainSection.innerHTML = div.innerHTML;
    moreBtn.style.display = 'none';

}
showMobiles.addEventListener('click', ()=>{
    showmore();
});

//Show Homepage
const homepage = document.getElementById('home-
sec');
const homebtn = document.getElementById('overview');
//By default on pageload
    homepage.style.display = 'block';
        let homeDiv = document.createElement('div');
        homeDiv.appendChild(homepage);
        mainSection.innerHTML = homeDiv.innerHTML;
        mainSection.append(productsDiv)
        moreBtn.style.display = 'block';
```

```
homebtn.addEventListener('click', ()=>{
    homepage.style.display = 'block';
    let homeDiv = document.createElement('div');
    homeDiv.appendChild(homepage);
    mainSection.innerHTML = homeDiv.innerHTML;
    mainSection.append(productsDiv)
    moreBtn.style.display = 'block';

})


//My Orders
const closeMyOrders = document.getElementById('close-myorders');
const openMyOrders = document.querySelector('.my-orders-btn');
const myOrdersBox = document.querySelector('.my-orders-
container');

openMyOrders.addEventListener('click', ()=>{
    myOrdersBox.style.display = 'block';
    myOrdersBox.style.transform = 'scale(0.1)';
    myOrdersBox.style.transition = 'all 100ms linear';
    setTimeout(()=>{
        myOrdersBox.style.transform = 'scale(1)'
    }, 1);
});
closeMyOrders.addEventListener('click', ()=>{
    myOrdersBox.style.transform = 'scale(0.1)'
    setTimeout(()=>{
        myOrdersBox.style.display = 'none';
    }, 100);
});
```

```
//Getting Orders Data from MySQL and displaying in MyOrders

async function MyOrdersData(){

    let response = await fetch('/getorders');
    response.json().then(data=> {
        for(let i = 0; i<data.length; i++){
            console.log(Number(data[i].price))
            const tbody = document.querySelector('.myorders-body');
            const tr = document.createElement('tr');
            tr.setAttribute('class', 'table-row');
            tr.setAttribute('id',`${data[i].id}`)
            tr.innerHTML = `<td>
            <span class="myorder-title"><img src="${data[i].image}" style="display:
inline;" alt=""><p style="display: inline; text-align:center; margin:auto;">$
{data[i].product_name}</p></span>
            </td>
            <td class="myorder-price" "><span></span>${data[i].price}</td>
            <td class="qty"><input type="number" name="" id="" disabled value="$
{data[i].quantity}"></td>
            <td class="myorder-total" >Rs ${parseFloat(data[i].price.match(/(\
d+)/))*parseInt(data[i].quantity)}</td>
            <td class="cancel-order" title='Cancel Order'><i
onClick="deleteOrder(event)" class="fa-solid fa-trash-can"></i></td>`;
            tbody.appendChild(tr);
        }})
}
MyOrdersData()
//Deleting Order from My Orders
async function deleteOrder(e){
    let cancelNotif = document.querySelector('.order-canceled');
    cancelNotif.style.transition = 'all 700ms ease-in-out';
    cancelNotif.style.transform = "translateX(0)";

    setTimeout(()=>{
        cancelNotif.style.transform = "translateX(350px)";
    }, 2000);

    let id = e.currentTarget.parentElement.parentElement.id;
    e.currentTarget.parentElement.parentElement.remove();
    const formData = {id: id}
```

```
    const fetchh = await fetch('/deleteorder', {
        method: 'Post', // Method itself
        mode: 'cors',
        headers: {
         'Content-type': 'application/json; charset=UTF-8' // Indicates the content
        },
        body: JSON.stringify(formData)
       });

}
//Getting Products Data from MySQL and displaying in Products
async function getProducts(){
    let response = await fetch('/getproduct');
    response.json().then((data)=>{
        for(let i = 0; i<data.length;i++){
            let innerColor = `rgb(${Math.random()*255}, ${Math.random()*255},${
{Math.random()*255})`;
            let div = document.createElement('article');
            div.setAttribute('class', 'product rproduct4 spacing');
            div.setAttribute('style', `background: radial-gradient(${innerColor},rgb(${
{Math.random()*255}, ${Math.random()*255},${Math.random()*255}))`);

            div.innerHTML = `<img src="${data[i].product_image}" alt=""
class="product_image">
            <h3 class="product_title">${data[i].product_name}</h3>
            <p class="product_description">${data[i].description}</p>
            <a class="btn btn-purchase" onclick="showContainer(event)" style='background-color:
${innerColor};'>Purchase</a>
            <span class="product-price">Rs ${data[i].price}</span>`;
            moreMobiles.appendChild(div);
        }
    })
}
getProducts()
async function getOtherProducts(){
    let response = await fetch('/getothers');
    let div = document.createElement('div');
    let div2 = document.createElement('div');

    div.setAttribute('class', 'accessories');
    div2.appendChild(div);


    response.json().then((data)=>{
        for(let i = 0; i<data.length;i++){
            const otherAccessories = document.createElement('div');
            otherAccessories.innerHTML = `<div class="accessory">
                                <img src="${data[i].product_image}">
                                <div class="acc_details">
                                    <h2>${data[i].product_name}</h2>
                                    <p>${data[i].description}</p>
                                    <div class="buttons">
                                        <button onClick='showAccessoryContainer(event)'>Buy</button>
                                        <span>Rs ${data[i].price}</span>
```

Code on Server

```
import express, { urlencoded } from 'express';
import path, {dirname} from 'path';
import { fileURLToPath } from 'url';
import mysql from 'mysql';

import multer from 'multer';
const storage = multer.diskStorage({
    destination: (req, file, cb)=>{
        cb(null, './public/Mobiles')
    },
    filename: (req, file, cb)=>{
        cb(null, file.originalname)
    }
})
const upload = multer({ storage: storage });

//create mysql connection
const connection = mysql.createConnection({
    host     : 'localhost',
    user     : 'root',
    database : 'mobilestore',
    password : 'mysql',
    multipleStatements: true
});
//connect
connection.connect((err)=>{
    if(err){
        throw err;
    }
    else{
        console.log('MySql is connected...')
    }

});

const __dirname = path.dirname(fileURLToPath(import.meta.url));
const app = express();
app.use(express.json());
app.use(urlencoded({ extended : true}));
app.use(express.static('./public'));
```

```
app.post('/buy', (req, res)=>{

    let insertOrder = `INSERT INTO orders
(product_name, person_name, phone, email,
address, quantity, image, price) VALUES ('$
{req.body.title}','${req.body.fullname}','$
{req.body.phone}','${req.body.email}','$
{req.body.address}','${req.body.quantity}','$
{req.body.image}','${req.body.price}');`;
    connection.query(insertOrder,
(err,result)=>{
        if (err) {
            throw err;
        }
        else{
            console.log('new order inserted to
database!')
        }
    })
    console.log(req.body.price)

    setTimeout(()=>{

res.sendFile(path.join(__dirname,'public/index.
html'));
    },2400)
});
```

```
app.post('/contact',(req,res)=>{
    let insertMessage = `INSERT INTO messages (name, subject, email,
message) VALUES ('${req.body.name}','${req.body.subject}','$
{req.body.email}','${req.body.message}');`
    connection.query(insertMessage,(err,result)=>{
        if(err){
            throw err;
        }
        else{
            console.log('Mail inserted to database!');
        }
    })

    res.sendFile(path.join(__dirname,'./messageSent.html'))
});
app.get('/getcontact', (req,res)=>{
    const gettMessages = 'SELECT * FROM messages;';
    let selectedMessages = '';
    connection.query(gettMessages, (err,result)=>{
        if(err){
            throw err;
        }
        else{
            selectedMessages = result;

        }
        res.send(selectedMessages)
    })
})
```

Conclusion

This fully fledged e-commerce project was made in a hope that it will be appreciated by Everyone since a lot of hard-work has been put into this by us. We hope that you like our project. Thanks you and regards.