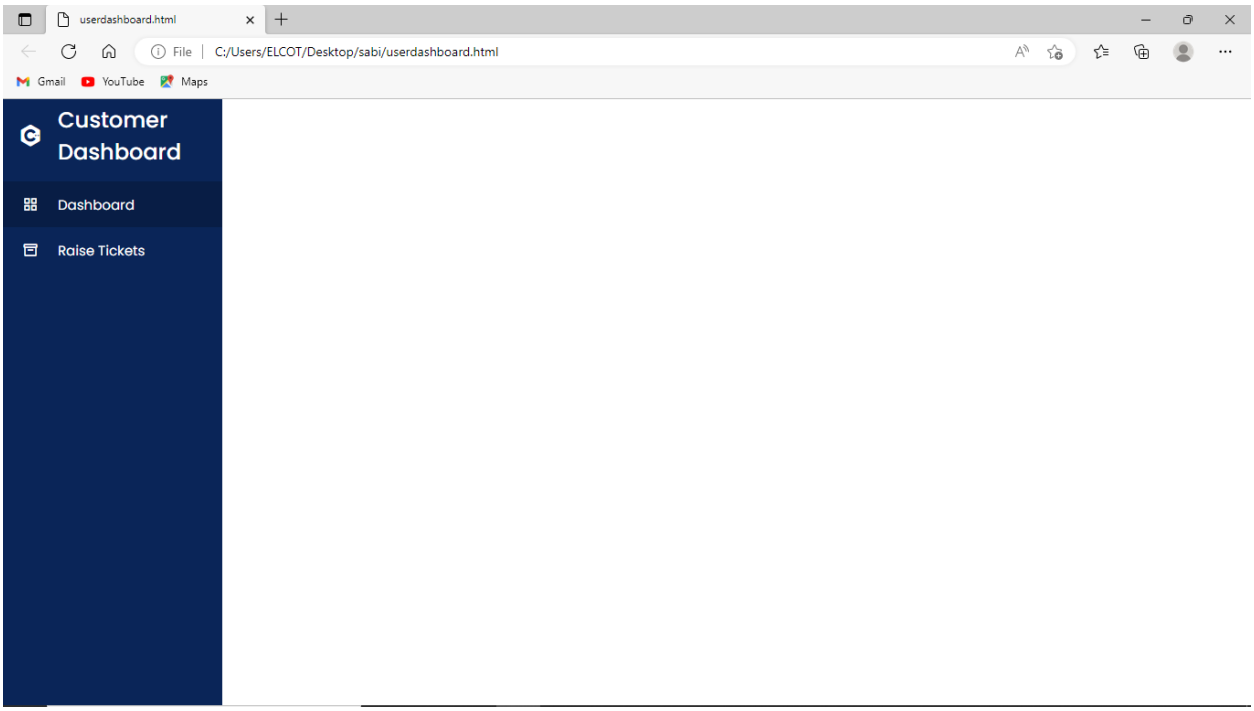
CUSTOMER CARE REGISTRY

TEAM ID:PNT2022TMID37726

Custdash.png:



APP.PY:

```
from __future__ import print_function

from audioop import add

import datetime

from unicodedata import name

from sib_api_v3_sdk.rest import ApiException

from pprint import pprint

from flask import Flask, render_template, request, redirect, url_for, session, flash

from markupsafe import escape

from flask import *

import ibm_db
```

```
import sib_api_v3_sdk

from init import randomnumber

from init import id

from init import hello


import datetime




conn =
ibm_db.connect("DATABASE=bludb;HOSTNAME="";PORT="";SECURITY=SSL;SSLServerCertificat
e="";UID="";PWD="""", '', '')

print(conn)

print("connection successful...")


app = Flask(__name__)

app.secret_key = 'your secret key'



@app.route('/')

def home():

   message = "TEAM ID : PNT2022TMID37544" +" "+ "BATCH ID : B1-1M3E "

   return render_template('index.html',mes=message)



@app.route('/signinpage', methods=['POST', 'GET'])

def signinpage():

   return render_template('signinpage.html')
```

```
@app.route('/agentsignin', methods=['POST', 'GET'])
def agentsignin():

    return render_template('signinpageagent.html')


@app.route('/signuppage', methods=['POST', 'GET'])
def signuppage():
    return render_template('signuppage.html')


@app.route('/agentRegister', methods=['POST', 'GET'])
def agentRegister():
    return render_template('agentregister.html')


@app.route('/forgotpass', methods=['POST', 'GET'])
def forgotpass():
    return render_template('forgot.html')


@app.route('/newissue/<name>', methods=['POST', 'GET'])
def newissue(name):
    name = name
```

```
    return render_template('complaint.html',msg=name)


@app.route('/forgot', methods=['POST', 'GET'])
def forgot():

    try:
        global randomnumber
        ida = request.form['custid']
        print(ida)
        global id
        id = ida
        sql = "SELECT EMAIL,NAME FROM Customer WHERE id=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, ida)
        ibm_db.execute(stmt)
        emailf = ibm_db.fetch_both(stmt)
        while emailf != False:
            e = emailf[0]
            n = emailf[1]
            break

        configuration = sib_api_v3_sdk.Configuration()
        configuration.api_key['api-key'] = ""

        api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
            sib_api_v3_sdk.ApiClient(configuration))
```

```
        subject = "Verification for Password"
        html_content = "<html><body><h1>Your verification Code is : <h2>" + \
           str(randomnumber)+"</h2> </h1> </body></html>"
        sender = {"name": "IBM CUSTOMER CARE REGISTRY",
                "email": "ibmdemo6@yahoo.com"}
        to = [{"email": e, "name": n}]
        reply_to = {"email": "ibmdemo6@yahoo.com", "name": "IBM"}
        headers = {"Some-Custom-Name": "unique-id-1234"}
        params = {"parameter": "My param value",
                "subject": "Email Verification"}
        send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
           to=to, reply_to=reply_to, headers=headers, html_content=html_content,
params=params, sender=sender, subject=subject)


        api_response = api_instance.send_transac_email(send_smtp_email)


        pprint(api_response)
        message = "Email send to:"+e+" for password"
        flash(message, "success")


    except ApiException as e:
        print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
        flash("Error in sending mail")
    except:
        flash("Your didn't Signin with this account")
    finally:
        return render_template('forgot.html')
```

```
@app.route('/verifyemail', methods=['POST', 'GET'])
def verifyemail():
    try:
        email = request.form['verifyemail']
        sql = "SELECT ID,NAME FROM Customer WHERE email=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.execute(stmt)
        emailf = ibm_db.fetch_both(stmt)
        while emailf != False:
            id = emailf[0]
            name = emailf[1]
            break
        configuration = sib_api_v3_sdk.Configuration()
        configuration.api_key['api-key'] = ""

        api_instance = sib_api_v3_sdk.TransactionalEmailsApi(
            sib_api_v3_sdk.ApiClient(configuration))
        subject = "Regarding of your Customer Id"
        html_content = "<html><body><h1>Your Customer Id  is : <h2>" + \
            str(id)+"</h2> </h1> </body></html>"
        sender = {"name": "IBM CUSTOMER CARE REGISTRY",
                  "email": "ibmdemo6@yahoo.com"}
        to = [{"email": email, "name": name}]
        reply_to = {"email": "ibmdemo6@yahoo.com", "name": "IBM"}
```

```
        headers = {"Some-Custom-Name": "unique-id-1234"}
        params = {"parameter": "My param value",
                "subject": "Email Verification"}
        send_smtp_email = sib_api_v3_sdk.SendSmtpEmail(
            to=to, reply_to=reply_to, headers=headers, html_content=html_content,
params=params, sender=sender, subject=subject)

        api_response = api_instance.send_transac_email(send_smtp_email)

        pprint(api_response)
        message = "Email send to:"+email+" for password"
        flash(message, "success")

    except ApiException as e:
        print("Exception when calling SMTPApi->send_transac_email: %s\n" % e)
        flash("Error in sending mail.")
    except:
        flash("Database not found in mail! Please Register Your account.", "danger")
    finally:
        return render_template('signinpage.html')


@app.route('/otp', methods=['POST', 'GET'])
def otp():
    try:
        otp = request.form['otp']
        cusid = id
```

```
        print(id)
        sql = "SELECT PASSWORD FROM Customer WHERE id=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, cusid)
        ibm_db.execute(stmt)
        otpf = ibm_db.fetch_both(stmt)
        while otpf != False:
            verify = otpf[0]
            break
        if otp == str(randomnumber):
            msg = "Your Password is "+verify+""
            flash(msg, "success")
            return render_template('forgot.html')
        else:
            flash("Wrong Otp", "danger")
    finally:
        return render_template('forgot.html')


@app.route('/admin', methods=['POST', 'GET'])
def admin():
    userdatabase = []
    sql = "SELECT * FROM customer"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        userdatabase.append(dictionary)
```

```
    dictionary = ibm_db.fetch_both(stmt)
if userdatabase:
    sql = "SELECT COUNT(*) FROM customer;"
    stmt = ibm_db.exec_immediate(conn, sql)
    user = ibm_db.fetch_both(stmt)

users = []
sql = "select * from ISSUE"
stmt = ibm_db.exec_immediate(conn, sql)
dict = ibm_db.fetch_both(stmt)
while dict != False:
    users.append(dict)
    dict = ibm_db.fetch_both(stmt)
if users:
    sql = "SELECT COUNT(*) FROM ISSUE;"
    stmt = ibm_db.exec_immediate(conn, sql)
    count = ibm_db.fetch_both(stmt)

agent = []
sql = "SELECT * FROM AGENT"
stmt = ibm_db.exec_immediate(conn, sql)
dictionary = ibm_db.fetch_both(stmt)
while dictionary != False:
    agent.append(dictionary)
    dictionary = ibm_db.fetch_both(stmt)

if agent:
```

```
    sql = "SELECT COUNT(*) FROM AGENT;"
    stmt = ibm_db.exec_immediate(conn, sql)
    cot = ibm_db.fetch_both(stmt)

  return
render_template("admin.html",complaint=users,users=userdatabase,agents=agent,message=u
ser[0],issue=count[0],msgagent = cot[0])



@app.route('/remove', methods=['POST', 'GET'])
def remove():

  otp = request.form['otpv']
  if otp == 'C':
    try:
      insert_sql = f"delete from customer"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.execute(prep_stmt)
      flash("delected successfully the Customer", "success")
    except:
      flash("No data found in Customer", "danger")
    finally:
      return redirect(url_for('signuppage'))
  if otp == 'A':
    try:
      insert_sql = f"delete from AGENT"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.execute(prep_stmt)
```

```
            flash("delected successfully the Agents", "success")
        except:
            flash("No data found in Agents", "danger")
        finally:
            return redirect(url_for('signuppage'))

    if otp == 'C':
        try:
            insert_sql = f"delete from AGENT"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.execute(prep_stmt)
            flash("delected successfully the Complaints", "success")
        except:
            flash("No data found in Complaints", "danger")
        finally:
            return redirect(url_for('signuppage'))

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        try:

            id = request.form['idn']
            global hello
            hello = id
            password = request.form['password']
            print(id, password)
```

```
        if id == '1111' and password == '1111':
            return redirect(url_for('admin'))

        sql = f"select * from customer where id='{escape(id)}' and
password='{escape(password)}'"
        stmt = ibm_db.exec_immediate(conn, sql)
        data = ibm_db.fetch_both(stmt)

        if data:
            session["name"] = escape(id)
            session["password"] = escape(password)
            return redirect(url_for("welcome"))

        else:
            flash("Mismatch in credetials", "danger")
    except:
        flash("Error in Insertion operation", "danger")

  return render_template('signinpage.html')

@app.route('/welcome', methods=['POST', 'GET'])
def welcome():
  try:
    id = hello
    sql = "SELECT ID,DATE,TOPIC,SERVICE_TYPE,SERVICE_AGENT,DESCRIPTION,STATUS FROM
ISSUE WHERE CUSTOMER_ID =?"
    agent = []
    stmt = ibm_db.prepare(conn, sql)
```

```
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.execute(stmt)
        otpf = ibm_db.fetch_both(stmt)
        while otpf != False:
            agent.append(otpf)
            otpf = ibm_db.fetch_both(stmt)
        sql = "SELECT COUNT(*) FROM ISSUE WHERE CUSTOMER_ID = ?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, id)
        ibm_db.execute(stmt)
        t = ibm_db.fetch_both(stmt)
        return render_template("welcome.html",agent=agent,message=t[0])
    except:

        return render_template("welcome.html")

@app.route('/loginagent', methods=['GET', 'POST'])
def loginagent():
    if request.method == 'POST':
        try:
            global loginagent
            id = request.form['idn']
            loginagent = id
            password = request.form['password']

            sql = f"select * from AGENT where id='{escape(id)}' and password='{escape(password)}'"
            stmt = ibm_db.exec_immediate(conn, sql)
```

```
            data = ibm_db.fetch_both(stmt)

            if data:
                session["name"] = escape(id)
                session["password"] = escape(password)
                return redirect(url_for("agentwelcome"))

            else:
                flash("Mismatch in credetials", "danger")
        except:
            flash("Error in Insertion operation", "danger")

    return render_template("signinpageagent.html")

@app.route('/delete/<ID>')
def delete(ID):
    sql = f"select * from customer where Id='{escape(ID)}'"
    print(sql)
    stmt = ibm_db.exec_immediate(conn, sql)
    student = ibm_db.fetch_row(stmt)
    if student:
        sql = f"delete from customer where id='{escape(ID)}'"
        stmt = ibm_db.exec_immediate(conn, sql)

        flash("Delected Successfully", "success")
        return redirect(url_for("admin"))
```

CUSTOMERDASH.HTML:

```
<!DOCTYPE html>
<!-- Designined by CodingLab | www.youtube.com/codinglabyt -->
<html lang="en" dir="ltr">
  <head>
    <meta charset="UTF-8">
    <!--<title> Responsiive Admin Dashboard | CodingLab </title>-->
    <link rel="stylesheet" href="style2.css">
    <!-- Boxicons CDN Link -->
    <link href='https://unpkg.com/boxicons@2.0.7/css/boxicons.min.css' rel='stylesheet'>
     <meta name="viewport" content="width=device-width, initial-scale=1.0">
   </head>
<body>
  <div class="sidebar">
    <div class="logo-details">
      <i class='bx bxl-c-plus-plus'></i>
      <span class="logo_name">Customer Dashboard</span>
    </div>
      <ul class="nav-links">
        <li>
          <a href="#" class="active">
            <i class='bx bx-grid-alt' ></i>
            <span class="links_name">Dashboard</span>
          </a>
        </li>
        <li>
          <a href="#">
```

```
          <i class='bx bx-box' ></i>
          <span class="links_name">Raise Tickets</span>
        </a>
      </li>

  </section>

  <script>
   let sidebar = document.querySelector(".sidebar");
let sidebarBtn = document.querySelector(".sidebarBtn");
sidebarBtn.onclick = function() {
  sidebar.classList.toggle("active");
  if(sidebar.classList.contains("active")){
  sidebarBtn.classList.replace("bx-menu" ,"bx-menu-alt-right");
}else
  sidebarBtn.classList.replace("bx-menu-alt-right", "bx-menu");
}
 </script>

</body>
</html>
```

STYLE2.CSS:

```
    /* Googlefont Poppins CDN Link */
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&displa
y=swap');
```

```
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Poppins', sans-serif;
}
.sidebar{
  position: fixed;
  height: 100%;
  width: 240px;
  background: #0A2558;
  transition: all 0.5s ease;
}
.sidebar.active{
  width: 60px;
}
.sidebar .logo-details{
  height: 80px;
  display: flex;
  align-items: center;
}
.sidebar .logo-details i{
  font-size: 28px;
  font-weight: 500;
  color: #fff;
  min-width: 60px;
  text-align: center
```

```
}
.sidebar .logo-details .logo_name{
  color: #fff;
  font-size: 24px;
  font-weight: 500;
}
.sidebar .nav-links{
  margin-top: 10px;
}
.sidebar .nav-links li{
  position: relative;
  list-style: none;
  height: 50px;
}
.sidebar .nav-links li a{
  height: 100%;
  width: 100%;
  display: flex;
  align-items: center;
  text-decoration: none;
  transition: all 0.4s ease;
}
.sidebar .nav-links li a.active{
  background: #081D45;
}
.sidebar .nav-links li a:hover{
  background: #081D45;
```

```
}
.sidebar .nav-links li i{
  min-width: 60px;
  text-align: center;
  font-size: 18px;
  color: #fff;
}
.sidebar .nav-links li a .links_name{
  color: #fff;
  font-size: 15px;
  font-weight: 400;
  white-space: nowrap;
}
.sidebar .nav-links .log_out{
  position: absolute;
  bottom: 0;
  width: 100%;
}
.home-section{
  position: relative;
  background: #f5f5f5;
  min-height: 100vh;
  width: calc(100% - 240px);
  left: 240px;
  transition: all 0.5s ease;
}
.sidebar.active ~ .home-section{
```

```
  width: calc(100% - 60px);
  left: 60px;
}
.home-section nav{
  display: flex;
  justify-content: space-between;
  height: 80px;
  background: #fff;
  display: flex;
  align-items: center;
  position: fixed;
  width: calc(100% - 240px);
  left: 240px;
  z-index: 100;
  padding: 0 20px;
  box-shadow: 0 1px 1px rgba(0, 0, 0, 0.1);
  transition: all 0.5s ease;
}
.sidebar.active ~ .home-section nav{
  left: 60px;
  width: calc(100% - 60px);
}
.home-section nav .sidebar-button{
  display: flex;
  align-items: center;
  font-size: 24px;
  font-weight: 500;
```

```
}
nav .sidebar-button i{
  font-size: 35px;
  margin-right: 10px;
}
.home-section nav .search-box{
  position: relative;
  height: 50px;
  max-width: 550px;
  width: 100%;
  margin: 0 20px;
}
nav .search-box input{
  height: 100%;
  width: 100%;
  outline: none;
  background: #F5F6FA;
  border: 2px solid #EFEEF1;
  border-radius: 6px;
  font-size: 18px;
  padding: 0 15px;
}
nav .search-box .bx-search{
  position: absolute;
  height: 40px;
  width: 40px;
  background: #2697FF;
```

```
  right: 5px;
  top: 50%;
  transform: translateY(-50%);
  border-radius: 4px;
  line-height: 40px;
  text-align: center;
  color: #fff;
  font-size: 22px;
  transition: all 0.4 ease;
}
.home-section nav .profile-details{
  display: flex;
  align-items: center;
  background: #F5F6FA;
  border: 2px solid #EFEEF1;
  border-radius: 6px;
  height: 50px;
  min-width: 190px;
  padding: 0 15px 0 2px;
}
nav .profile-details img{
  height: 40px;
  width: 40px;
  border-radius: 6px;
  object-fit: cover;
}
nav .profile-details .admin_name{
```

```
  font-size: 15px;
  font-weight: 500;
  color: #333;
  margin: 0 10px;
  white-space: nowrap;
}
nav .profile-details i{
  font-size: 25px;
  color: #333;
}
.home-section .home-content{
  position: relative;
  padding-top: 104px;
}
.home-content .overview-boxes{
  display: flex;
  align-items: center;
  justify-content: space-between;
  flex-wrap: wrap;
  padding: 0 20px;
  margin-bottom: 26px;
}
.overview-boxes .box{
  display: flex;
  align-items: center;
  justify-content: center;
  width: calc(100% / 4 - 15px);
```

```
  background: #fff;
  padding: 15px 14px;
  border-radius: 12px;
  box-shadow: 0 5px 10px rgba(0,0,0,0.1);
}
.overview-boxes .box-topic{
  font-size: 20px;
  font-weight: 500;
}
.home-content .box .number{
  display: inline-block;
  font-size: 35px;
  margin-top: -6px;
  font-weight: 500;
}
.home-content .box .indicator{
  display: flex;
  align-items: center;
}
.home-content .box .indicator i{
  height: 20px;
  width: 20px;
  background: #8FDACB;
  line-height: 20px;
  text-align: center;
  border-radius: 50%;
  color: #fff;
```

```
  font-size: 20px;
  margin-right: 5px;
}
.box .indicator i.down{
  background: #e87d88;
}
.home-content .box .indicator .text{
  font-size: 12px;
}
.home-content .box .cart{
  display: inline-block;
  font-size: 32px;
  height: 50px;
  width: 50px;
  background: #cce5ff;
  line-height: 50px;
  text-align: center;
  color: #66b0ff;
  border-radius: 12px;
  margin: -15px 0 0 6px;
}
.home-content .box .cart.two{
   color: #2BD47D;
   background: #C0F2D8;
 }
.home-content .box .cart.three{
   color: #ffc233;
```

```
   background: #ffe8b3;
 }
.home-content .box .cart.four{
   color: #e05260;
   background: #f7d4d7;
 }
.home-content .total-order{
  font-size: 20px;
  font-weight: 500;
}
.home-content .sales-boxes{
  display: flex;
  justify-content: space-between;
  /* padding: 0 20px; */
}

/* left box */
.home-content .sales-boxes .recent-sales{
  width: 65%;
  background: #fff;
  padding: 20px 30px;
  margin: 0 20px;
  border-radius: 12px;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.1);
}
.home-content .sales-boxes .sales-details{
  display: flex;
```

```
  align-items: center;
  justify-content: space-between;
}
.sales-boxes .box .title{
  font-size: 24px;
  font-weight: 500;
  /* margin-bottom: 10px; */
}
.sales-boxes .sales-details li.topic{
  font-size: 20px;
  font-weight: 500;
}
.sales-boxes .sales-details li{
  list-style: none;
  margin: 8px 0;
}
.sales-boxes .sales-details li a{
  font-size: 18px;
  color: #333;
  font-size: 400;
  text-decoration: none;
}
.sales-boxes .box .button{
  width: 100%;
  display: flex;
  justify-content: flex-end;
}
```

```
.sales-boxes .box .button a{
  color: #fff;
  background: #0A2558;
  padding: 4px 12px;
  font-size: 15px;
  font-weight: 400;
  border-radius: 4px;
  text-decoration: none;
  transition: all 0.3s ease;
}
.sales-boxes .box .button a:hover{
  background:  #0d3073;
}

/* Right box */
.home-content .sales-boxes .top-sales{
  width: 35%;
  background: #fff;
  padding: 20px 30px;
  margin: 0 20px 0 0;
  border-radius: 12px;
  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.1);
}
.sales-boxes .top-sales li{
  display: flex;
  align-items: center;
  justify-content: space-between;
```

```
  margin: 10px 0;
}
.sales-boxes .top-sales li a img{
  height: 40px;
  width: 40px;
  object-fit: cover;
  border-radius: 12px;
  margin-right: 10px;
  background: #333;
}
.sales-boxes .top-sales li a{
  display: flex;
  align-items: center;
  text-decoration: none;
}
.sales-boxes .top-sales li .product,
.price{
  font-size: 17px;
  font-weight: 400;
  color: #333;
}
/* Responsive Media Query */
@media (max-width: 1240px) {
  .sidebar{
    width: 60px;
  }
  .sidebar.active{
```

```
    width: 220px;
  }
  .home-section{
    width: calc(100% - 60px);
    left: 60px;
  }
  .sidebar.active ~ .home-section{
    /* width: calc(100% - 220px); */
    overflow: hidden;
    left: 220px;
  }
  .home-section nav{
    width: calc(100% - 60px);
    left: 60px;
  }
  .sidebar.active ~ .home-section nav{
    width: calc(100% - 220px);
    left: 220px;
  }
}
@media (max-width: 1150px) {
  .home-content .sales-boxes{
    flex-direction: column;
  }
  .home-content .sales-boxes .box{
    width: 100%;
    overflow-x: scroll;
```

```
    margin-bottom: 30px;
  }
  .home-content .sales-boxes .top-sales{
    margin: 0;
  }
}
@media (max-width: 1000px) {
  .overview-boxes .box{
    width: calc(100% / 2 - 15px);
    margin-bottom: 15px;
  }
}
@media (max-width: 700px) {
  nav .sidebar-button .dashboard,
  nav .profile-details .admin_name,
  nav .profile-details i{
    display: none;
  }
  .home-section nav .profile-details{
    height: 50px;
    min-width: 40px;
  }
  .home-content .sales-boxes .sales-details{
    width: 560px;
  }
}
@media (max-width: 550px) {
```

```
  .overview-boxes .box{
    width: 100%;
    margin-bottom: 15px;
  }
  .sidebar.active ~ .home-section nav .profile-details{
    display: none;
  }
}
  @media (max-width: 400px) {
  .sidebar{
    width: 0;
  }
  .sidebar.active{
    width: 60px;
  }
  .home-section{
    width: 100%;
    left: 0;
  }
  .sidebar.active ~ .home-section{
    left: 60px;
    width: calc(100% - 60px);
  }
  .home-section nav{
    width: 100%;
    left: 0;
  }
```

```
  .sidebar.active ~ .home-section nav{
    left: 60px;
    width: calc (100% - 60px);
  }
}
```