Assignment -2

Python Programming

Assignment Date	25 September 2022
Student Name	Gishalin Rufina S G
Student Roll Number	960219104043
Maximum Marks	2 Marks

1. Create a Flask App

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

import sqlite3 as sql


app = Flask(__name__)
app.secret_key="123"




@app.route('/')
def homePage():
   return render_template('homePage.html')


@app.route('/aboutpage')
def AboutPage():
   return render_template('AboutPage.html')

@app.route('/signup')
def signup():
   return render_template('SignUp.html')

@app.route("/register",methods = ['POST','GET'])
def register():
   if request.method == 'POST':
      try:
         FirstName = request.form['firstname']
         MiddleName = request.form['middlename']
         LastName = request.form ['lastname']
         PhoneNumber = request.form ['phone']
         Address=request.form ['address']
         Email=request.form ['email']
         Password=request.form ['pass']

         with sql.connect("database.db") as con:
          cur = con.cursor()
```

```
            cur.execute("INSERT INTO User_Details
(FirstName,MiddleName,LastName,PhoneNumber,Address,Email,Password) VALUES
(?,?,?,?,?,?,?)",(firstname,MiddleName,LastName,PhoneNumber,Address,Email,Password) )
            con.commit()
            msg = "Record successfully added!"
        except:
         con.rollback()
         msg = "error in insert operation"


        finally:
          return render_template("list.html",msg = msg)
          con.close()

@app.route('/list')
def list():
  con = sql.connect("database.db")
  con.row_factory = sql.Row


  cur = con.cursor()
  cur.execute("select * from User_Details")

  User_Details = cur.fetchall();
  return render_template("list.html",User_Details = User_Details)

@app.route("/signIn")

def signIn():
   return render_template('Sign-in.html')


if __name__ == '__main__':
  app.run(debug = True)
```

2. Add the Home page, About Page and add bootstrap

homepage.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
```

```
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
<style>
* {
  box-sizing: border-box;
}

body {
  font-family: Arial;
  padding: 10px;
  background: Lavender;
}

/* Header/Blog Title */
.header {
  padding: 30px;
  text-align: center;
  background: Thistle;
}

.header h1 {
  font-size: 50px;
}

/* Style the top navigation bar */
.topnav {
  overflow: hidden;
  background-color:#FAEBD7 ;
}

/* Style the topnav links */
.topnav a {
  float: left;
  display: block;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
}

/* Change color on hover */
.topnav a:hover {
  background-color: #ddd;
  color: black;
}
```

```
/* Create two unequal columns that floats next to each other */
/* Left column */
.leftcolumn {
  float: left;
  width: 75%;
}

/* Right column */
.rightcolumn {
  float: left;
  width: 25%;
  background-color: #f1f1f1;
  padding-left: 20px;
}

/* Fake image */
.fakeimg {
  background-color: #aaa;
  width: 100%;
  padding: 20px;
}

/* Add a card effect for articles */
.card {
  background-color: Snow;
  padding: 20px;
  margin-top: 20px;
}

/* Clear floats after the columns */
.row:after {
  content: "";
  display: table;
  clear: both;
}

/* Footer */
.footer {
  padding: 20px;
  text-align: center;
  background: #ddd;
  margin-top: 20px;
}

/* Responsive layout - when the screen is less than 800px wide, make the two columns stack on top of each other instead of next to each other */
@media screen and (max-width: 800px) {
  .leftcolumn, .rightcolumn {
```

```
   width: 100%;
   padding: 0;
  }
}

/* Responsive layout - when the screen is less than 400px wide, make the navigation links stack on top of each other instead of next to each other */
@media screen and (max-width: 400px) {
  .topnav a {
   float: none;
   width: 100%;
  }
}
</style>
</head>
<body>
<center> <h1>Welcome to the project on the Inventory Management System for Retailers</h1></center>

<div class="header">
  <h1>Inventory Management System for Retailers</h1>
  <p>Taking the business to the next level!</p>
</div>


<div class="row">
  <div class="leftcolumn">
    <div class="card">
      <h2>Purpose of Inventory Management System</h2>
      <h5>The Inventory Management System is a real-time inventory database capable of connecting multiple stores.  This can be used to track the inventory of a single store or to manage the delivery of stock between several branches of a larger franchise. However, the system merely records sales and restocking data and provides warning of low stock at any location through email at a specified interval.</h5>
      <p>The goal is to reduce the stress of tracking rather than to holder all store maintenance. Further features may consist of the ability to create reports of sales, but again the explanation is left to the management. In addition, since theft does occasionally occur, the system provides solutions for confirming the store inventory and for correcting stock quantities.</p>

  <div class="rightcolumn">
    <div class="card">
      <h2>About Me</h2>

      <p>Gishalin Rufina<br>
Student at ACEW<br>
Kanyakumari district<br>
TamilNadu</p>
```

```
   </div>


</body>
</html>
```

AboutPage.html

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

<head>
<meta charset="utf-8">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
  <title>About the Project</title>
</head>

<body style="background:LightGoldenRodYellow;">
<h1> About the Project!</h1>

  <h5>
 Software Required:</h5>
<ul><li>Python</li>
<li> Flask </li>
<li> Docker</li>
</ul>
<h5>System Required:</h5>
<p>8GB RAM,Intel Core i3,OS-Windows/Linux/MAC ,Laptop or Desktop</p>


<h5>Get to know more</h5>

<p>Retail inventory management is the process of ensuring you carry merchandise that
shoppers want,
with neither too little nor too much on hand. By managing inventory, retailers meet
customer demand without running
out of stock or carrying excess supply.</p>
```

In practice, effective retail inventory management results in lower costs and a better understanding
of sales patterns. Retail inventory management tools and methods give retailers more information on which to
run their businesses. Applications have been developed to help retailers track and manage stocks related to their own products.
The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by
logging into the application.

Once retailers successfully log in to the application they can update their inventory details,
also users will be able to add new stock by submitting essential details related to the stock. They can view
details of the current inventory. The System will automatically send an email alert to the retailers if there is no
stock found in their accounts. So that they can order new stock.

</body>
</html>

3. Add the Sign in page and App the Signup Page + database connectivity

Sign-in.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta charset="utf-8">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
<title> Login Page </title>
<style>
Body {
  font-family: Calibri, Helvetica, sans-serif;
  background-color: pink;
}
button {
```

```
      background-color: #4CAF50;
      width: 100%;
       color: orange;
       padding: 15px;
       margin: 10px 0px;
       border: none;
       cursor: pointer;
        }
 form {
       border: 3px solid #f1f1f1;
   }
 input[type=text], input[type=password] {
       width: 100%;
       margin: 8px 0;
       padding: 12px 20px;
       display: inline-block;
       border: 2px solid green;
       box-sizing: border-box;
   }
 button:hover {
       opacity: 0.7;
   }
  .cancelbtn {
       width: auto;
       padding: 10px 18px;
       margin: 10px 5px;
   }


 .container {
       padding: 25px;
       background-color: lightblue;
   }
</style>
</head>
<body>
   <center> <h1> Sign in with us!</h1> </center>
   <form method= "POST" >
      <div class="container">
         <label>Username : </label>
         <input type="text" placeholder="Enter Username" name="username" required>
         <label>Password : </label>
         <input type="password" placeholder="Enter Password" name="password" required>
         <button type="submit">Login</button>

      </div>
   </form>
</body>
```

```
</html>
```

SignUp.html

```
<Html lang="en">
<head>
<meta charset="utf-8">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
<title>
SignUp Form
</title>
<style>
body {
  font-family: Arial;
  padding: 10px;
  background: Azure;
</style>
</head>
<body>
<br>
<br>
<form method ="POST" action = "{{ url_for('register') }}" >
<h1> SignUp with us!</h1>

<label> Firstname </label>
<input type="text" name="firstname" size="15"/> <br> <br>
<label> Middlename: </label>
<input type="text" name="middlename" size="15"/> <br> <br>
<label> Lastname: </label>
<input type="text" name="lastname" size="15"/> <br> <br>



<br>
<br>
<label>
Gender :
</label><br>
<input type="radio" name="male"/> Male <br>
<input type="radio" name="female"/> Female <br>
```

```
<input type="radio" name="other"/> Other
<br>
<br>

<label>
Phone :
</label>
<input type="text" name="country code"  value="+91" size="2"/>
<input type="text" name="phone" size="10"/> <br> <br>
Address
<br>
<textarea cols="80" rows="5" value="address">
</textarea>
<br> <br>
Email:
<input type="email" id="email" name="email"/> <br>
<br> <br>
Password:
<input type="Password" id="pass" name="pass"> <br>
<br> <br>
Re-type password:
<input type="Password" id="repass" name="repass"> <br> <br>
<input type="button" value="Submit"/>
</form>
</body>
</html>
```

Sqlite.py

```
import sqlite3

conn = sqlite3.connect('database.db')
print("Opened database successfully")

conn.execute('CREATE TABLE User_Details (FirstName TEXT,MiddleName TEXT,LastName
TEXT, PhoneNumber INTEGER,Address VARCHAR, Email VARCHAR, Password VARCHAR)')
print("Table created successfully");
conn.close()
```

app.py

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

import sqlite3 as sql


app = Flask(__name__)
```

```
app.secret_key="123"



@app.route('/')
def homePage():
   return render_template('homePage.html')


@app.route('/aboutpage')
def AboutPage():
   return render_template('AboutPage.html')

@app.route('/signup')
def signup():
   return render_template('SignUp.html')

@app.route("/register",methods = ['POST','GET'])
def register():
   if request.method == 'POST':
      try:
         FirstName = request.form['firstname']
         MiddleName = request.form['middlename']
         LastName = request.form ['lastname']
         PhoneNumber = request.form ['phone']
         Address=request.form ['address']
         Email=request.form ['email']
         Password=request.form ['pass']

         with sql.connect("database.db") as con:
          cur = con.cursor()
          cur.execute("INSERT INTO User_Details
(FirstName,MiddleName,LastName,PhoneNumber,Address,Email,Password) VALUES
(?,?,?,?,?,?,?)",(firstname,MiddleName,LastName,PhoneNumber,Address,Email,Password)
)
          con.commit()
          msg = "Record successfully added!"
      except:
       con.rollback()
       msg = "error in insert operation"


      finally:
       return render_template("list.html",msg = msg)
       con.close()

@app.route('/list')
```

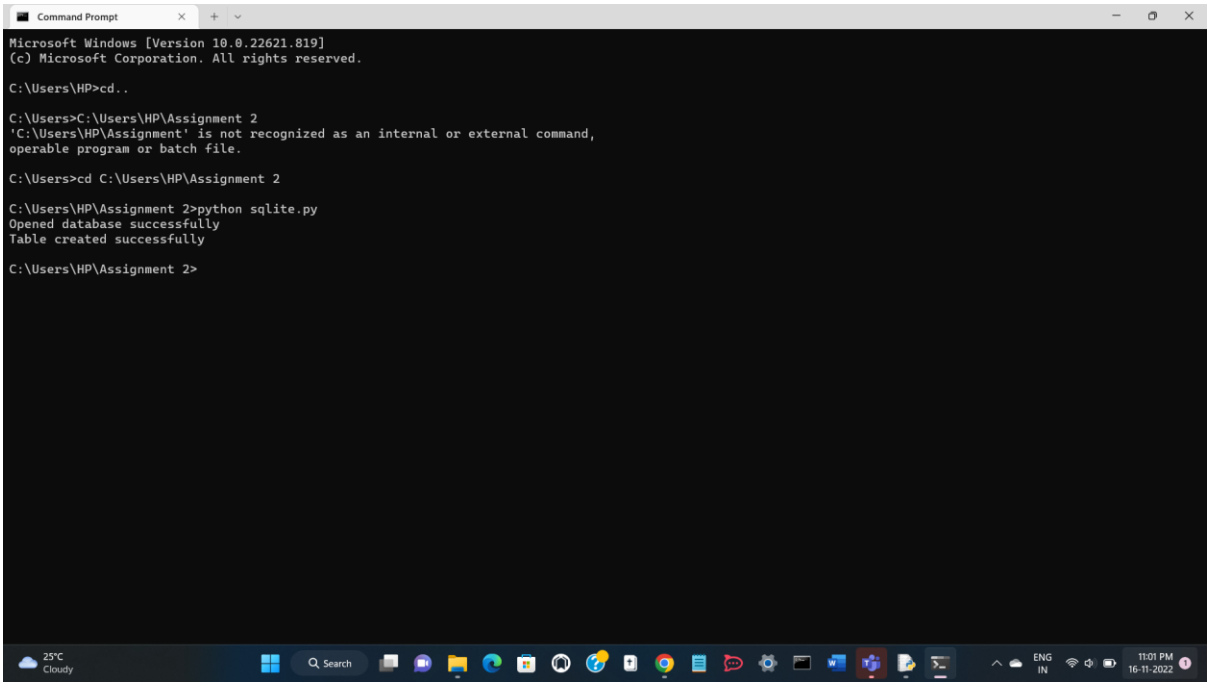
```
def list():
   con = sql.connect("database.db")
   con.row_factory = sql.Row


   cur = con.cursor()
   cur.execute("select * from User_Details")

   User_Details = cur.fetchall();
   return render_template("list.html",User_Details = User_Details)

@app.route("/signIn")

def signIn():
    return render_template('Sign-in.html')


if __name__ == '__main__':
   app.run(debug = True)
```

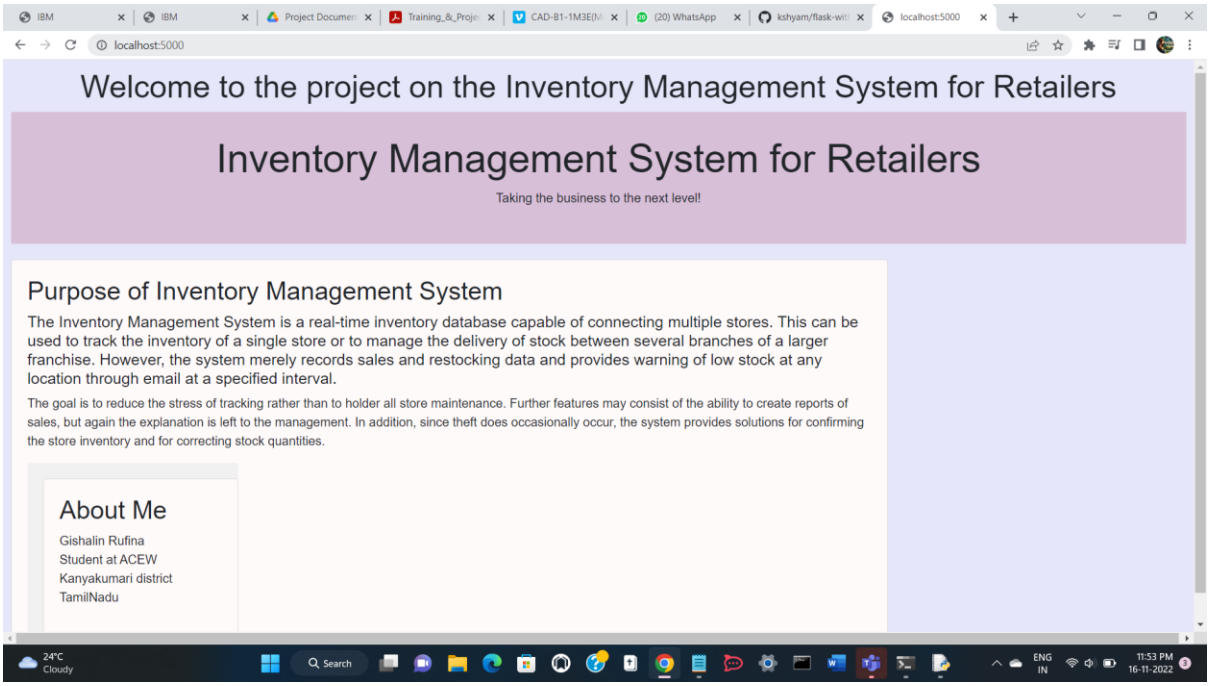
list.html

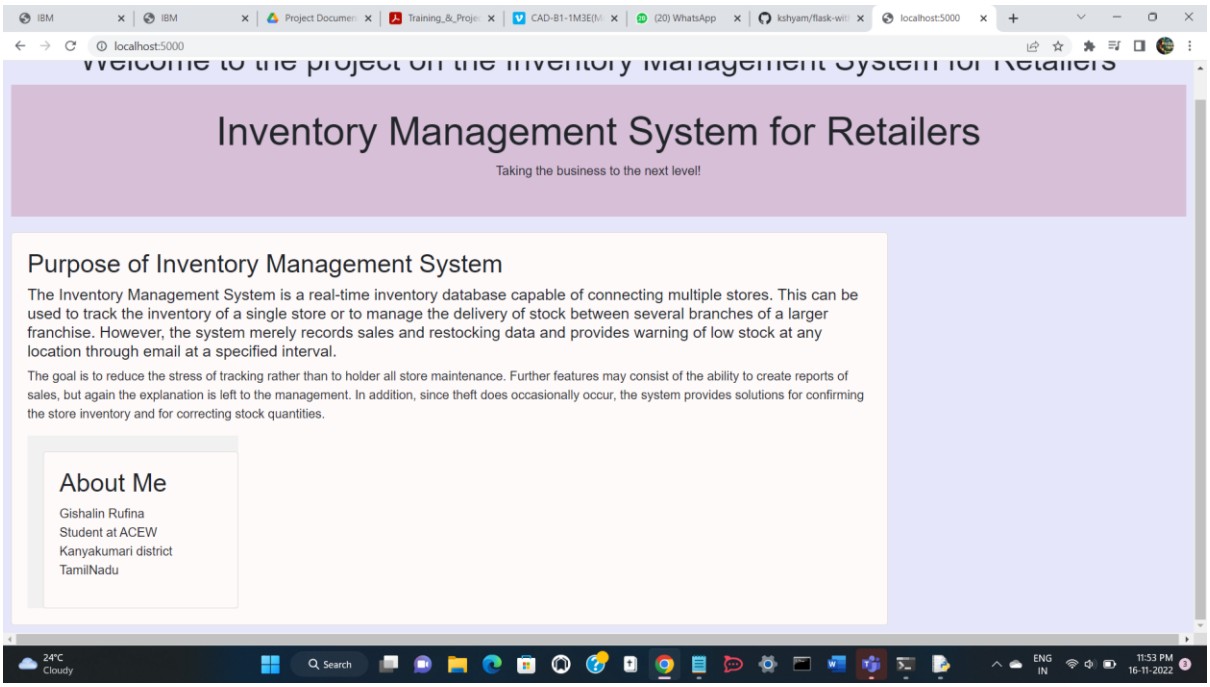
```
<!doctype html>
<html>
   <body>



    {{ msg }}

      <table border = 1>
         <thead>
            <td>FirstName</td>
            <td> MiddleName</td>
            <td>LastName</td>
            <td>PhoneNumber</td>
             <td>Address</td>
            <td>Email</td>
            <td>Password</td>
         </thead>

         {% for row in User_Details %}
            <tr>
               <td>{{row["FirstName"]}}</td>
               <td>{{row["MiddleName"]}}</td>
               <td> {{ row["LastName"]}}</td>
               <td>{{row["PhoneNumber"]}}</td>
```

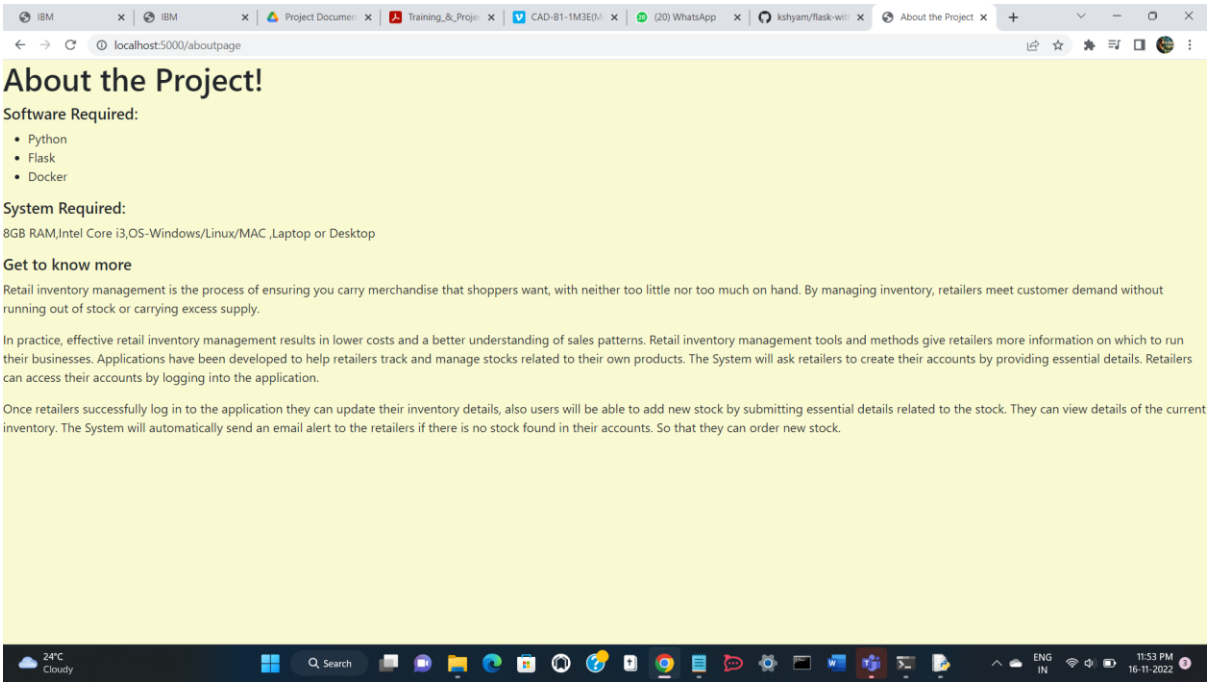
```
            <td>{{row["Address"]}}</td>
            <td>{{row["Email"]}}</td>
             <td>{{row["Password"]}}</td>
          </tr>
        {% endfor %}
      </table>


  </body>
</html>
```

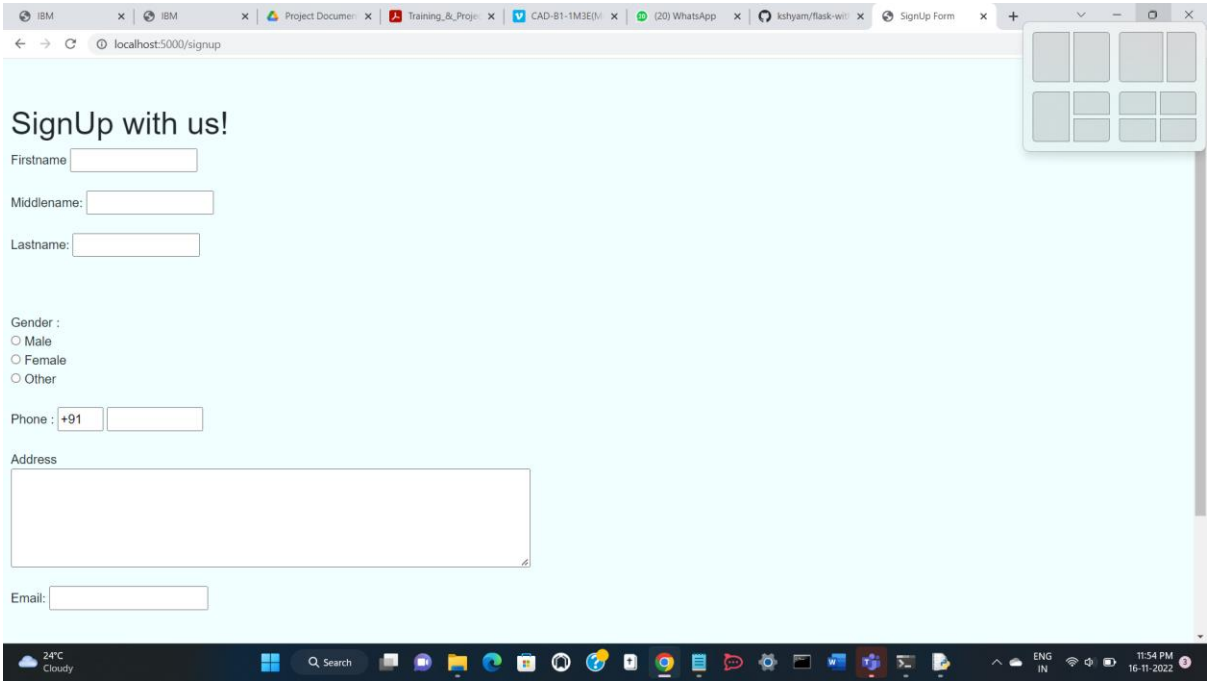
Output:

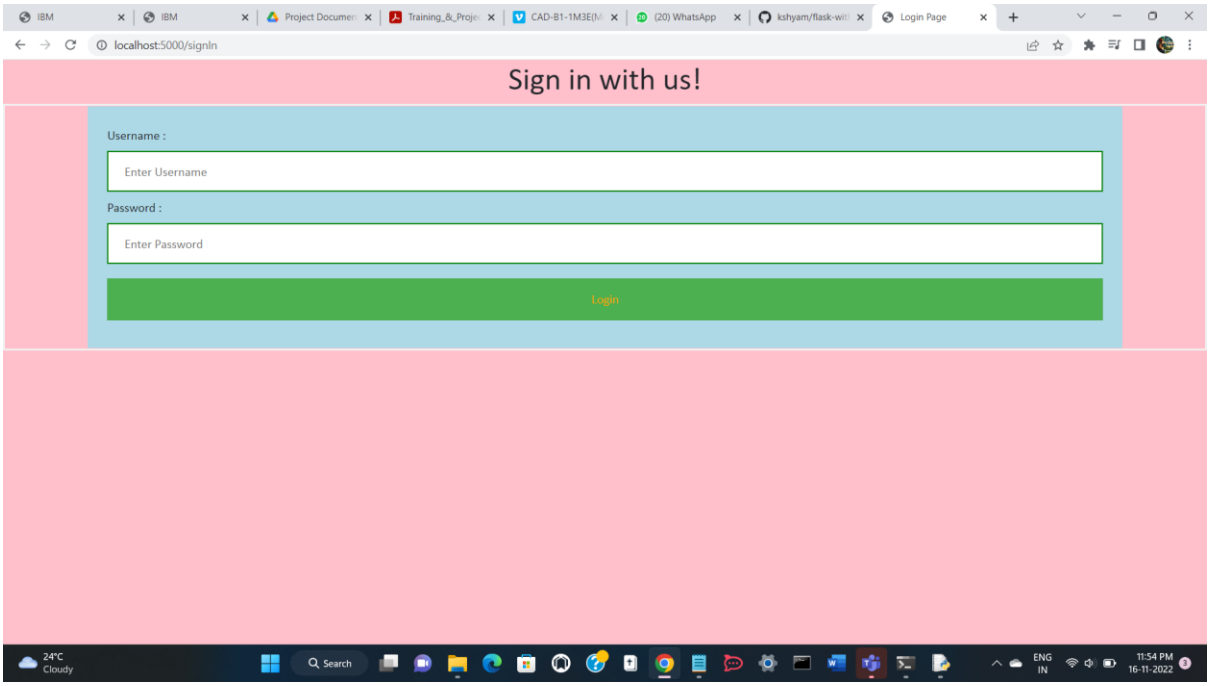


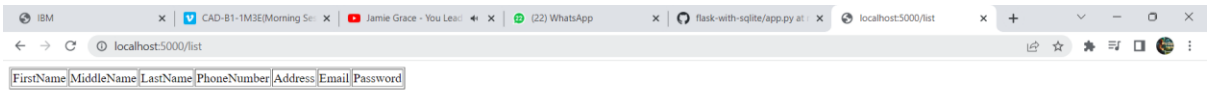

Welcome to the project on the Inventory Management System for Retailers
Inventory Management System for Retailers
Taking the business to the next level!
Purpose of Inventory Management System
The Inventory Management System is a real-time inventory database capable of connecting multiple stores. This can be used to track the inventory of a single store or to manage the delivery of stock between several branches of a larger franchise. However, the system merely records sales and restocking data and provides warning of low stock at any location through email at a specified interval.
The goal is to reduce the stress of tracking rather than to holder all store maintenance. Further features may consist of the ability to create reports of sales, but again the explanation is left to the management. In addition, since theft does occasionally occur, the system provides solutions for confirming the store inventory and for correcting stock quantities.
About Me
Gishalin Rufina
Student at ACEW
Kanyakumari district
TamilNadu


Inventory Management System for Retailers
Taking the business to the next level!
Purpose of Inventory Management System
The Inventory Management System is a real-time inventory database capable of connecting multiple stores. This can be used to track the inventory of a single store or to manage the delivery of stock between several branches of a larger franchise. However, the system merely records sales and restocking data and provides warning of low stock at any location through email at a specified interval.
The goal is to reduce the stress of tracking rather than to holder all store maintenance. Further features may consist of the ability to create reports of sales, but again the explanation is left to the management. In addition, since theft does occasionally occur, the system provides solutions for confirming the store inventory and for correcting stock quantities.
About Me
Gishalin Rufina
Student at ACEW
Kanyakumari district
TamilNadu


About the Project!
Software Required:
Python
Flask
Docker
System Required:
8GB RAM,Intel Core i3,OS-Windows/Linux/MAC ,Laptop or Desktop
Get to know more
Retail inventory management is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply.
In practice, effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information on which to run their businesses. Applications have been developed to help retailers track and manage stocks related to their own products. The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into the application.
Once retailers successfully log in to the application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock. They can view details of the current inventory. The System will automatically send an email alert to the retailers if there is no stock found in their accounts. So that they can order new stock.


SignUp with us!
Firstname
Middlename:
Lastname:
Gender :
Male
Female
Other
Phone : +91
Address
Email:


Login Page
localhost:5000/signIn
Sign in with us!
Username :
Enter Username
Password :
Enter Password
Login


localhost:5000/list
FirstName
MiddleName
LastName
PhoneNumber
Address
Email
Password