SPRINT-4

TEAM ID	PNT2022TMID04866
Project Title	Inventory Management System for Retailers
DATE	11-11-2022
MARK	20 Points

Functional Requirement:

- Home page
- Register page
- Login page
- Dashboard
- Add Stock
- Update Stock
- Delete Stock
- View Stock

Code:

app.py:

```
from
flask
import
Flask,
render_te
mplate,
request,
redirect,
url_for,
session,
flash
            import ibm_db
            import sqlite3 as sql
            import re
            import alert
            from sendemail import sendmail,sendgridmail
            from dotenv import dotenv_values
            app = Flask(__name__)
```

```
envData=dotenv_values("./.env")
hostname=envData["HOSTNAME"]
port=envData["PORT"]
uid=envData["UID"]
pwd=envData["PWD"]
app.secret_key = 'a'
creds =
"DATABASE=bludb;HOSTNAME=%s;PORT=%s;SECURITY=SSL;SSLServerCertificate=./Digi
CertGlobalRootCA.crt;UID=%s;PWD=%s"%(hostname,port,uid,pwd)


conn = ibm_db.connect(creds,'','')
print(conn)
print("Connecting Successful!!!!!!!!")


@app.route('/')


def homer():
    return render_template('home.html')




@app.route('/login',methods =['GET', 'POST'])
def login():
    global userid
    msg = ''


    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
```

```
            msg = 'Logged in successfully !'
            return render_template('dashboard.html', msg = msg)
        else:
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)
```

```
@app.route('/register', methods =['GET', 'POST'])
def registet():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        sql = "SELECT * FROM users WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            insert_sql = "INSERT INTO  users VALUES (?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_stmt)
            msg = 'Please fill out the form !'
        if request.method == 'POST':
          msg = 'You have successfully registered! Please login !'
        #   sendgridmail(email,msg)
    return render_template('register.html', msg = msg)


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

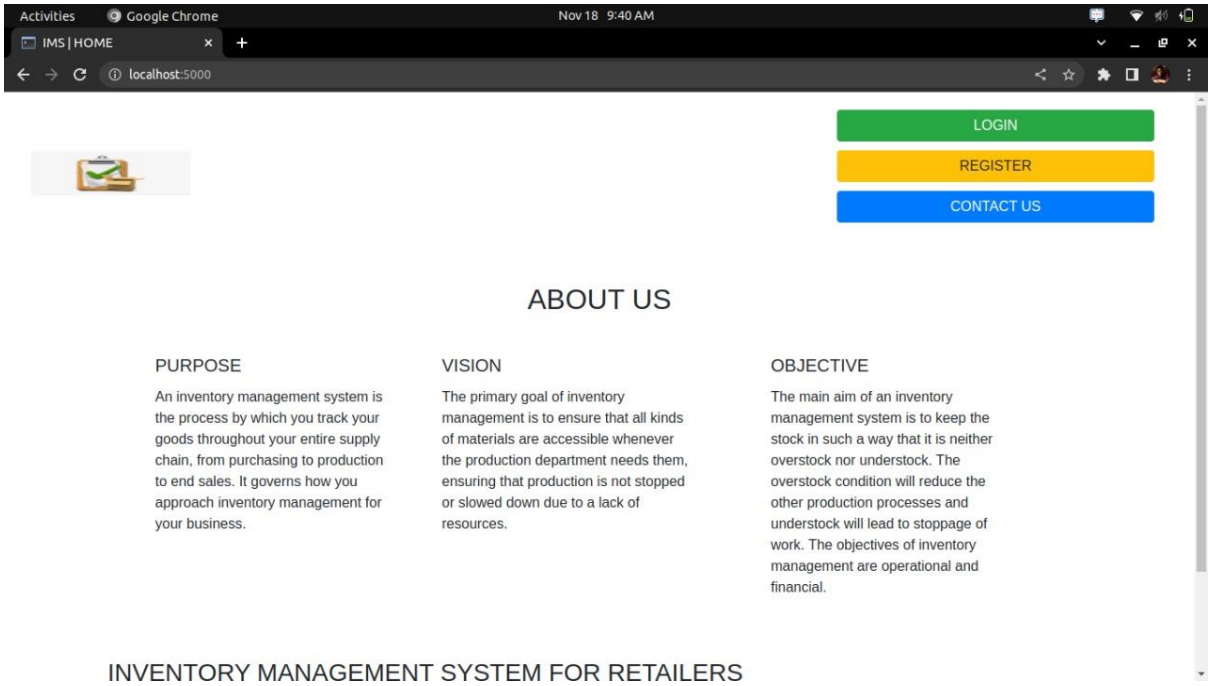
```
    msg=''
    if request.method == "POST":
        prodname=request.form['prodname']
        quantity=request.form['quantity']
        warehouse_location=request.form['warehouse_location']
        sql='SELECT * FROM product WHERE prodname =?'
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,prodname)
        ibm_db.execute(stmt)
        acnt=ibm_db.fetch_assoc(stmt)
        print(acnt)

        if acnt:
            msg='Product already exits!!'
            return render_template("add_stock.html",msg=msg)
        else:
            insert_sql='INSERT INTO product VALUES (?,?,?)'
            pstmt=ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(pstmt,1,prodname)
            ibm_db.bind_param(pstmt,2,quantity)
            ibm_db.bind_param(pstmt,3,warehouse_location)
            ibm_db.execute(pstmt)
            msg='You have successfully added the products!!'
            return render_template("dashboard.html")


    else:
        msg="fill out the form first!"
        return render_template('add_stock.html',meg=msg)


@app.route('/delete_stock',methods=['GET','POST'])
def delete_stock():
    if(request.method=="POST"):
        prodname=request.form['prodname']
        sql2="DELETE FROM product WHERE prodname=?"
        stmt2 = ibm_db.prepare(conn, sql2)
        ibm_db.bind_param(stmt2,1,prodname)
        ibm_db.execute(stmt2)


        flash("Product Deleted", "success")


        return render_template("dashboard.html")
```

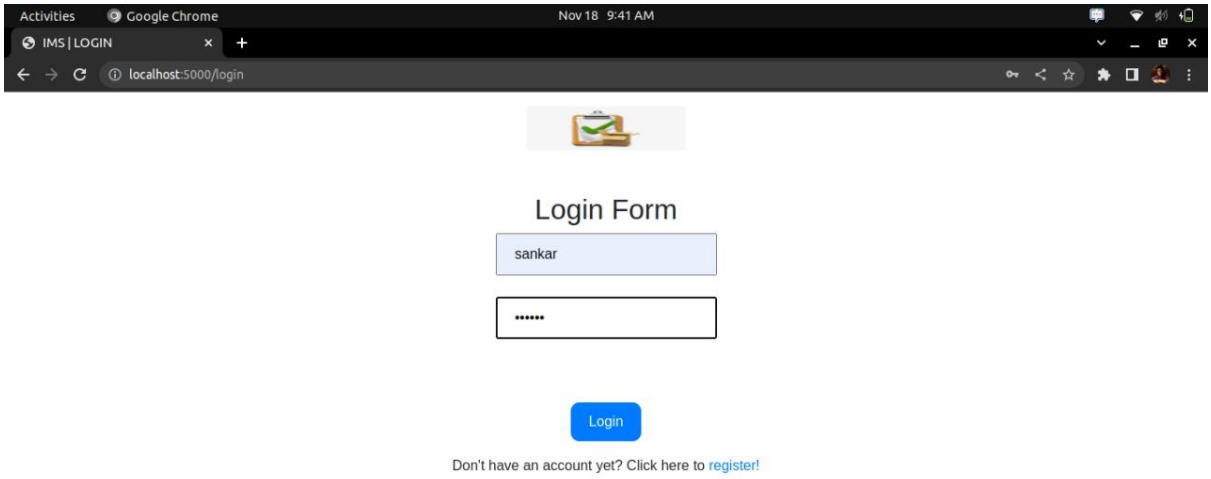
```
@app.route('/update_stock',methods=['GET','POST'])
def update_stock():
    mg=''
    if request.method == "POST":
        prodname=request.form['prodname']
        quantity=request.form['quantity']
        quantity=int(quantity)
        print(quantity)
        print(type(quantity))
        warehouse_location=request.form['warehouse_location']
        sql='SELECT * FROM product WHERE prodname =?'
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,prodname)
        ibm_db.execute(stmt)
        acnt=ibm_db.fetch_assoc(stmt)
        print(acnt)

        if acnt:
            insert_sql='UPDATE product SET  quantity=?,warehouse_location=?
WHERE prodname=? '
            pstmt=ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(pstmt,1,quantity)
            ibm_db.bind_param(pstmt,2,warehouse_location)
            ibm_db.bind_param(pstmt,3,prodname)
            ibm_db.execute(pstmt)
            mg='You have successfully updated the products!!'
            limit=5
            print(type(limit))
            if(quantity<=limit):
                   alert("Please update the quantity of the product {},
Atleast {} number of pieces must be added!".format(prodname,10))
            return render_template("dashboard.html",meg=mg)

        else:
             mg='Product not found!!'



    else:
         msg="fill out the form first!"
         return render_template('update_stock.html',meg=msg)


@app.route('/view_stock')
def view_stock():
```

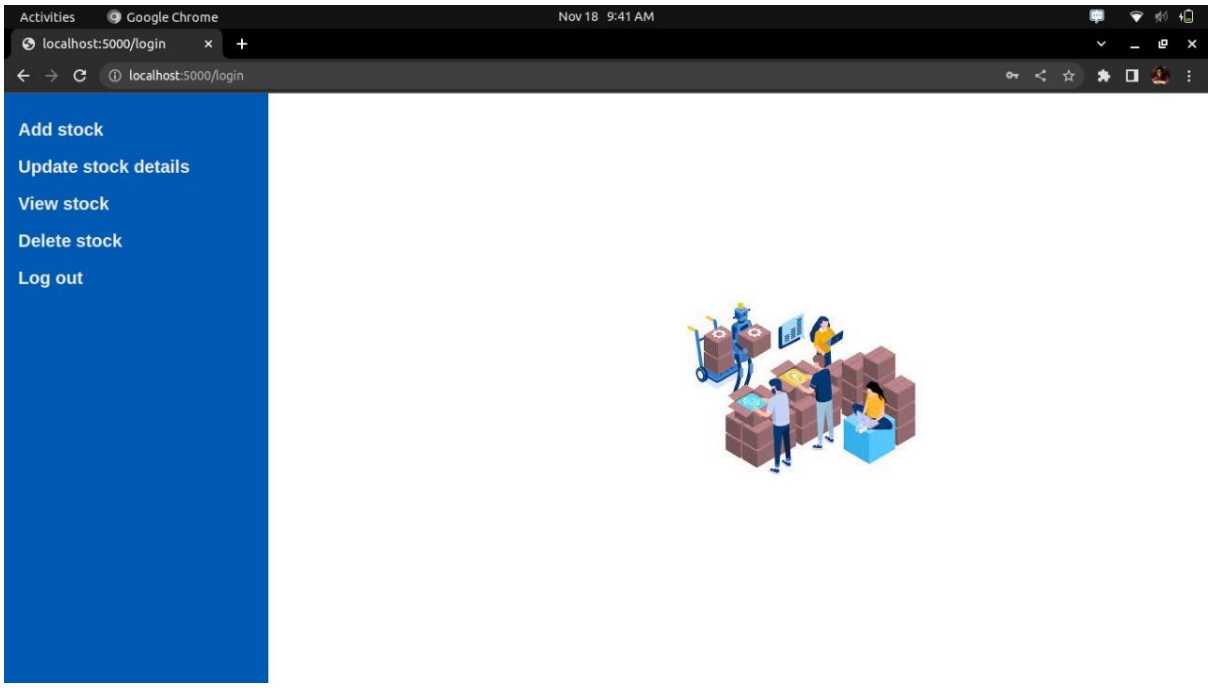
```
    sql = "SELECT * FROM product"
    stmt = ibm_db.prepare(conn, sql)
    result=ibm_db.execute(stmt)
    print(result)


    products=[]
    row = ibm_db.fetch_assoc(stmt)
    print(row)
    while(row):
        products.append(row)
        row = ibm_db.fetch_assoc(stmt)
        print(row)
    products=tuple(products)
    print(products)


    if result>0:
        return render_template('view.html', products = products)
    else:
        msg='No products found'
        return render_template('view.html', msg=msg)




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


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



@app.route('/logout')


def logout():
   session.pop('loggedin', None)
   session.pop('id', None)
   session.pop('username', None)
```

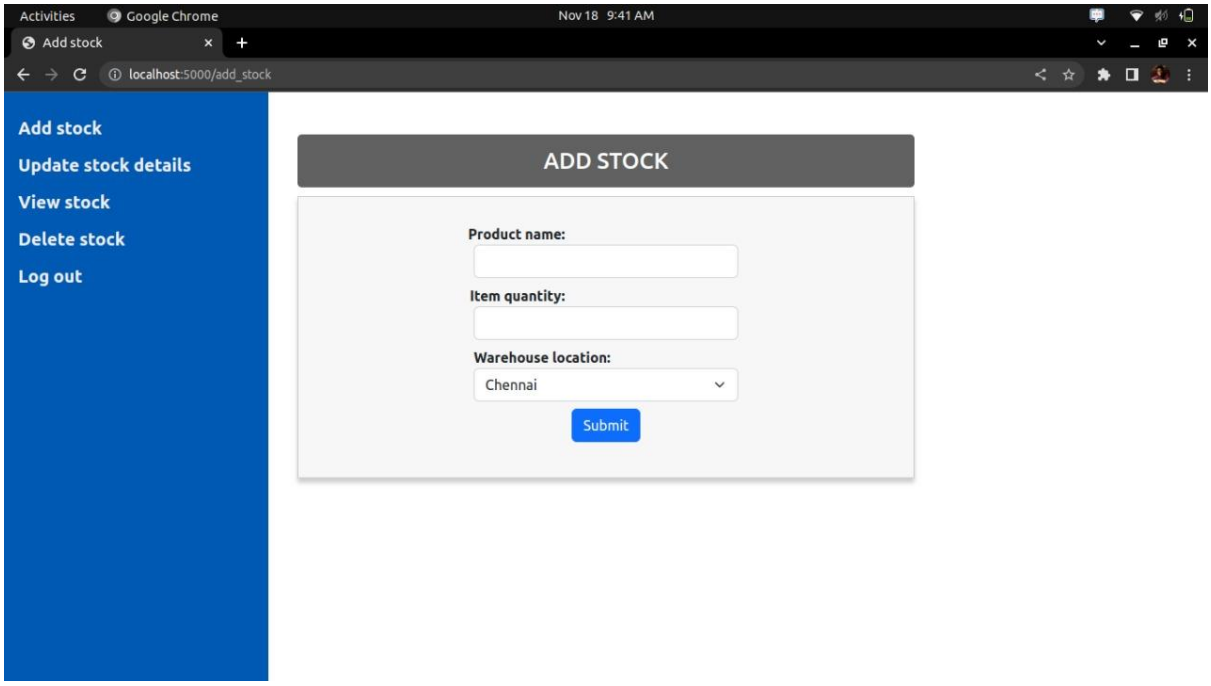
```
    return render_template('home.html')



if __name__ == '__main__':
    app.run(host='0.0.0.0')
```

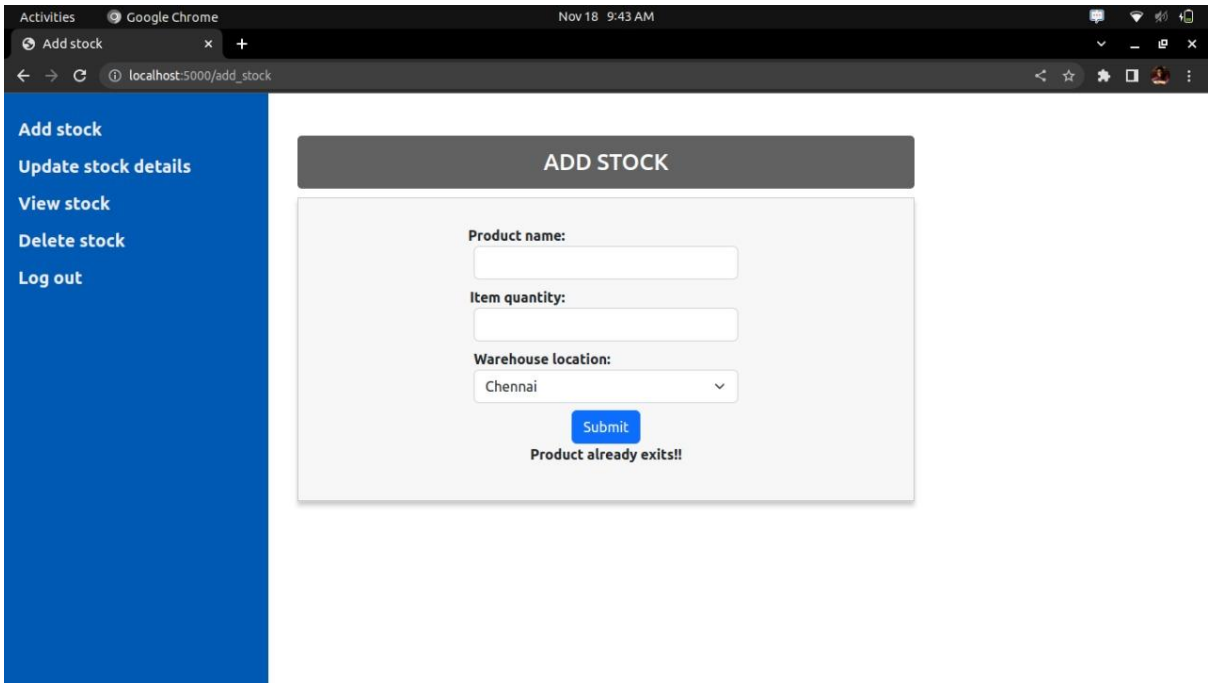
Output:

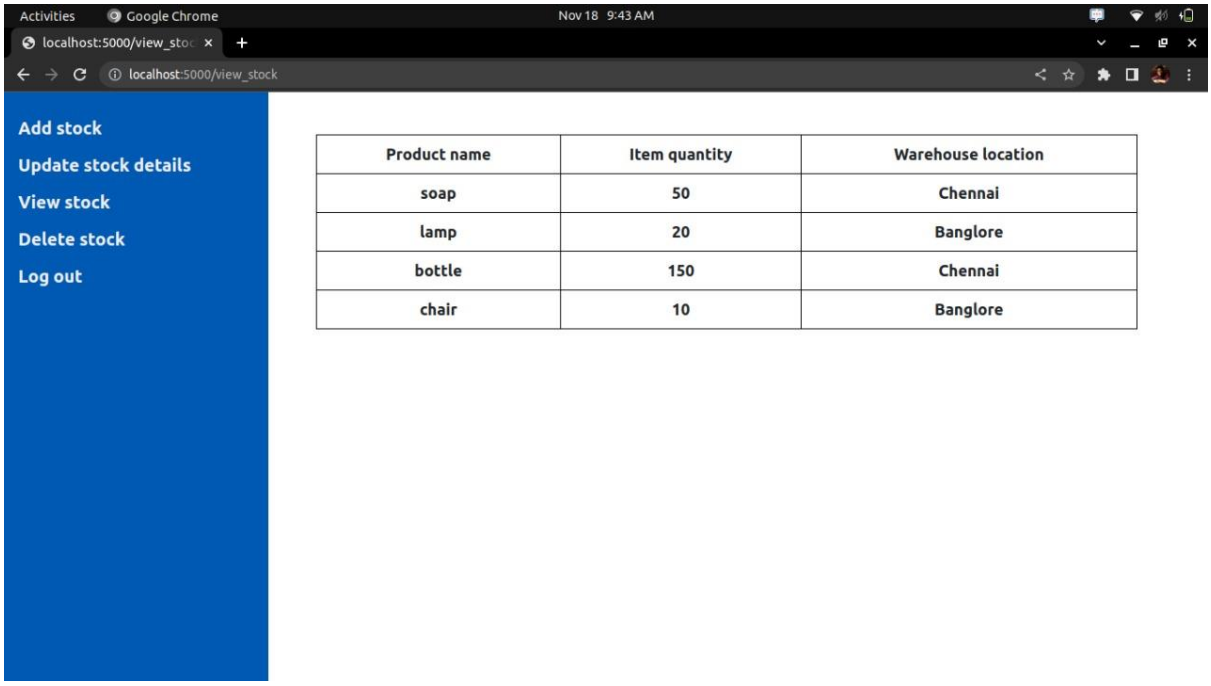


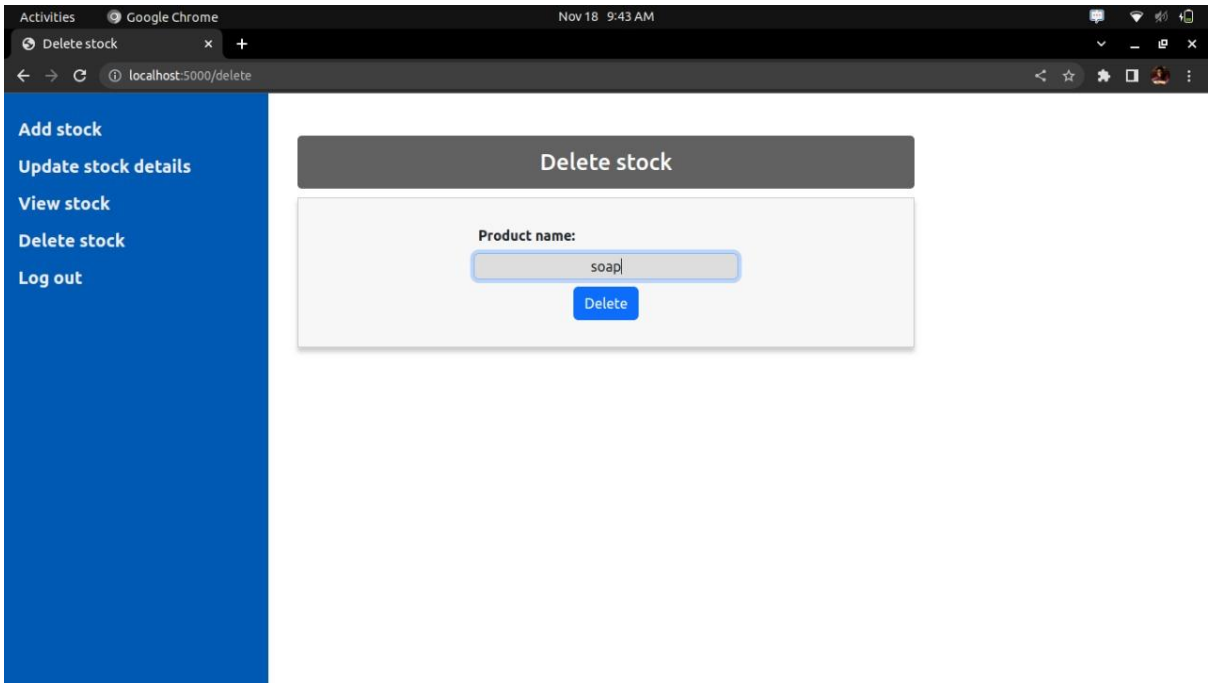

Activities
Google Chrome
Nov 18 9:41 AM
IMS | LOGIN
localhost:5000/login
Login Form
sankar
Login
Don't have an account yet? Click here to register!

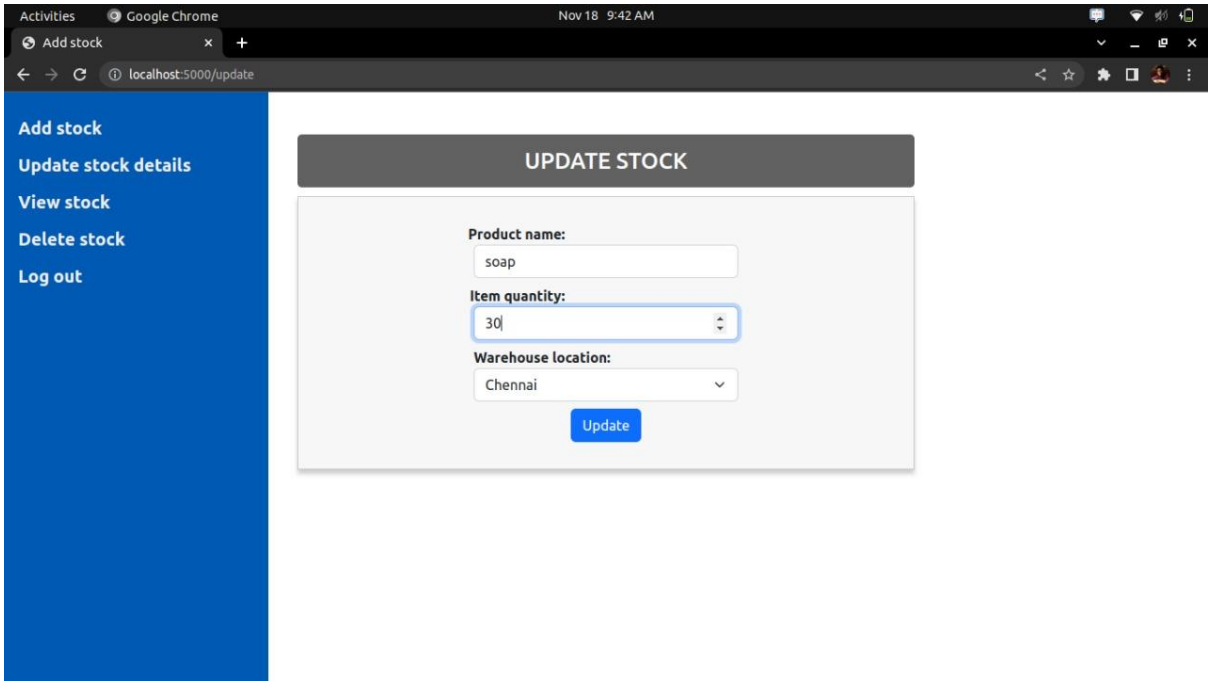

Activities
Google Chrome
Nov 18 9:41 AM
localhost:5000/login
localhost:5000/login
Add stock
Update stock details
View stock
Delete stock
Log out

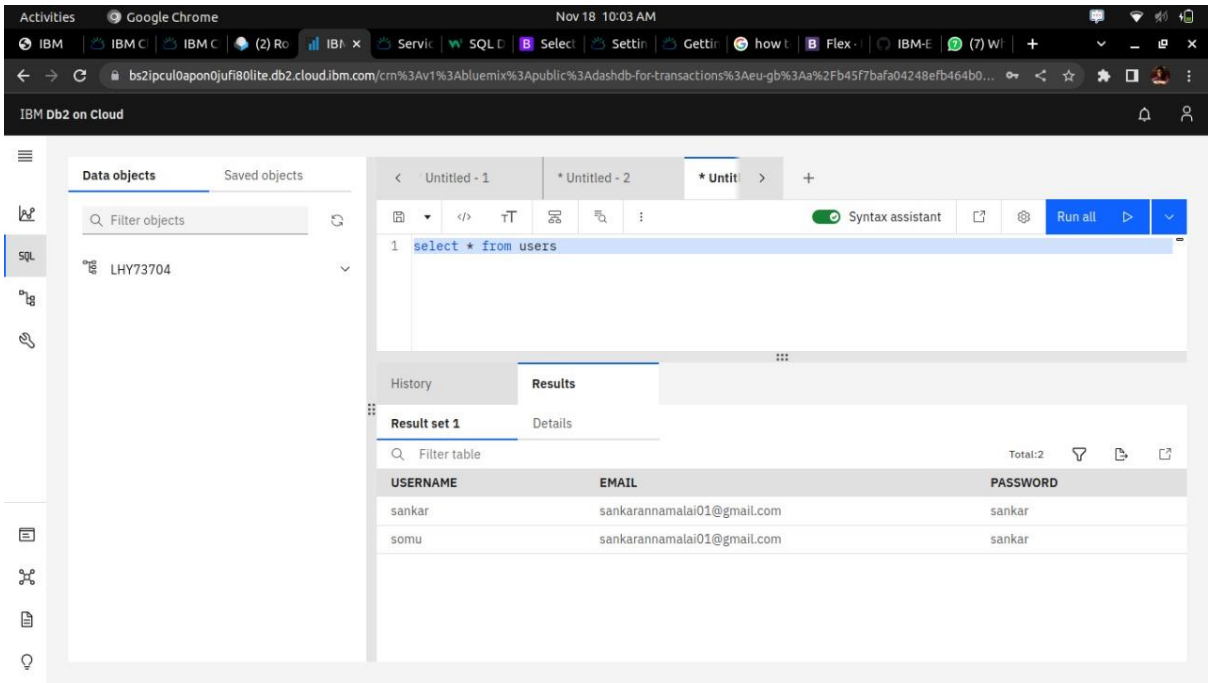

Activities
Google Chrome
Nov 18 9:41 AM
Add stock
localhost:5000/add_stock
Add stock
Update stock details
View stock
Delete stock
Log out
ADD STOCK
Product name:
Item quantity:
Warehouse location:
Chennai
Submit

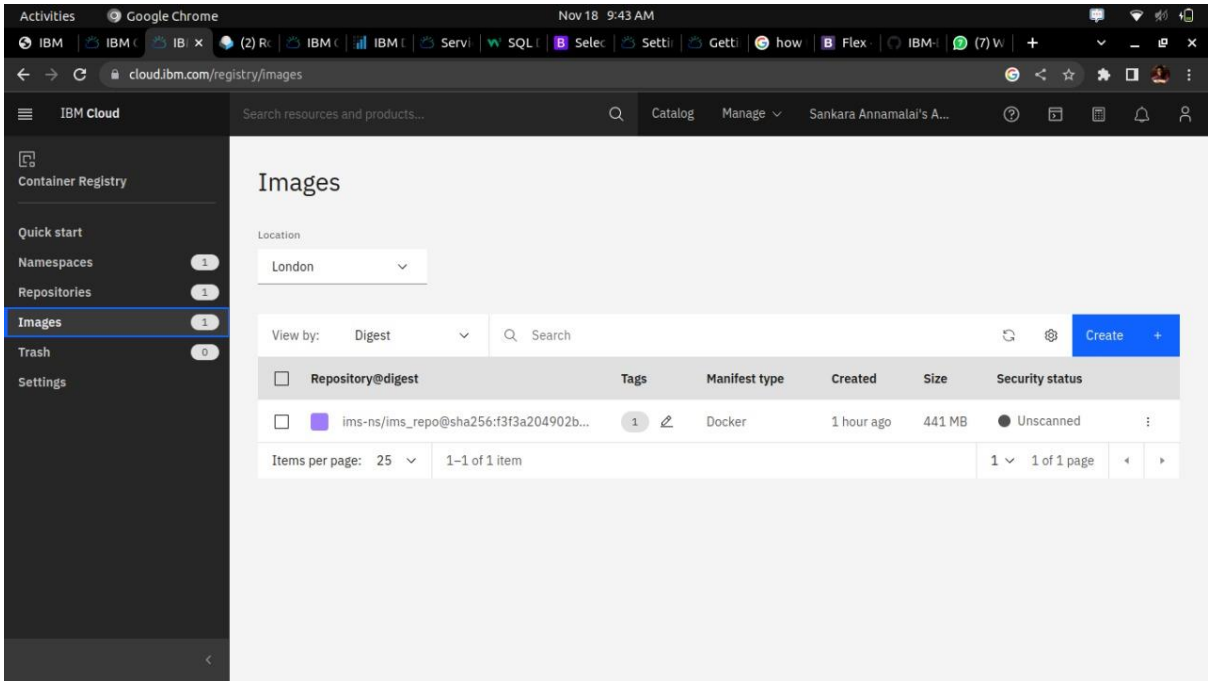

Activities
Google Chrome
Nov 18 9:43 AM
Add stock
localhost:5000/add_stock
Add stock
Update stock details
View stock
Delete stock
Log out
ADD STOCK
Product name:
Item quantity:
Warehouse location:
Chennai
Submit
Product already exits!!


Activities
Google Chrome
Nov 18 9:43 AM
localhost:5000/view_stoc
localhost:5000/view_stock
Add stock
Update stock details
View stock
Delete stock
Log out
Product name
Item quantity
Warehouse location
soap
50
Chennai
lamp
20
Banglore
bottle
150
Chennai
chair
10
Banglore


Activities
Google Chrome
Nov 18 9:43 AM
Delete stock
localhost:5000/delete
Add stock
Update stock details
View stock
Delete stock
Log out
Delete stock
Product name:
soap
Delete


Activities
Google Chrome
Nov 18 9:42 AM
Add stock
localhost:5000/update
Add stock
Update stock details
View stock
Delete stock
Log out
UPDATE STOCK
Product name:
soap
Item quantity:
30
Warehouse location:
Chennai
Update


Activities
Google Chrome
Nov 18 10:03 AM
IBM Db2 on Cloud
Data objects
Saved objects
Filter objects
LHY73704
Untitled - 1
* Untitled - 2
Syntax assistant
Run all
1 select * from users
History
Results
Result set 1
Details
Filter table
Total:2
USERNAME
EMAIL
PASSWORD
sankar
sankarannamalai01@gmail.com
sankar
somu
sankarannamalai01@gmail.com
sankar


Activities
Google Chrome
Nov 18 9:43 AM
cloud.ibm.com/registry/images
IBM Cloud
Search resources and products...
Catalog
Manage
Sankara Annamalai's A...
Container Registry
Quick start
Namespaces 1
Repositories 1
Images 1
Trash 0
Settings
Images
Location
London
View by: Digest
Search
Create
Repository@digest
Tags
Manifest type
Created
Size
Security status
ims-ns/ims_repo@sha256:f3f3a204902b...
1
Docker
1 hour ago
441 MB
Unscanned
Items per page: 25
1–1 of 1 item
1 of 1 page