```
imp
ort
ibm
_db
       from flask_session import Session
       from flask import Flask, render_template, redirect, request, session, jsonify
       from datetime import datetime
       # # Instantiate Flask object named app
       app = Flask(__name__)
       # # Configure sessions
       app.config["DB2_DATABASE"] = 'bludb'
       app.config["DB2_HOSTNAME"] = '9938aec0-8105-433e-8bf9-
       0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud'
       app.config["DB2_PORT"] = '32459'
       app.config["DB2_PROTOCOL"] = 'TCPIP'
       app.config["DB2_USER"] = 'mqs19694'
       app.config["DB2_PASSWORD"] = 'SsDjCqU5ECrgxjRF'
       Session(app)
       # Creates a connection to the database
       conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=9938aec0-8105-433e-8bf9-
       0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32459/bludb:userid=<mqs1969
       4>;password=<r1WNVdtZMECaCyhO>;sslConnection=true;",'','')
       db = conn.connection.cursor()
       @app.route("/")
       def index():
           shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice")
           shirtsLen = len(shirts)
           # Initialize variables
           shoppingCart = []
           shopLen = len(shoppingCart)
           totItems, total, display = 0, 0, 0
           if 'user' in session:
               shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
       price, id FROM cart GROUP BY samplename")
               shopLen = len(shoppingCart)
               for i in range(shopLen):
                   total += shoppingCart[i]["SUM(subTotal)"]
                   totItems += shoppingCart[i]["SUM(qty)"]
               shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice ASC")
```

```
        shirtsLen = len(shirts)
        return render_template ("index.html", shoppingCart=shoppingCart, shirts=shirts,
shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems, display=display,
session=session )
    return render_template ( "index.html", shirts=shirts, shoppingCart=shoppingCart,
shirtsLen=shirtsLen, shopLen=shopLen, total=total, totItems=totItems, display=display)
@app.route("/buy/")
def buy():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    qty = int(request.args.get('quantity'))
    if session:
        # Store id of the selected shirt
        id = int(request.args.get('id'))
        # Select info of selected shirt from database
        goods = db.execute("SELECT * FROM shirts WHERE id = :id", id=id)
        # Extract values from selected shirt record
        # Check if shirt is on sale to determine price
        if(goods[0]["onSale"] == 1):
            price = goods[0]["onSalePrice"]
        else:
            price = goods[0]["price"]
        samplename = goods[0]["samplename"]
        image = goods[0]["image"]
        subTotal = qty * price
        # Insert selected shirt into shopping cart
        db.execute("INSERT INTO cart (id, qty, samplename, image, price, subTotal)
VALUES (:id, :qty, :samplename, :image, :price, :subTotal)", id=id, qty=qty,
samplename=samplename, image=image, price=price, subTotal=subTotal)
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")
        shopLen = len(shoppingCart)
        # Rebuild shopping cart
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
        # Select all shirts for home page view
        shirts = db.execute("SELECT * FROM shirts ORDER BY samplename ASC")
        shirtsLen = len(shirts)
        # Go back to home page
        return render_template ("index.html", shoppingCart=shoppingCart, shirts=shirts,
shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems, display=display, 
session=session )
@app.route("/update/")
def update():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    qty = int(request.args.get('quantity'))
    if session:
        # Store id of the selected shirt
        id = int(request.args.get('id'))
        db.execute("DELETE FROM cart WHERE id = :id", id=id)
        # Select info of selected shirt from database
        goods = db.execute("SELECT * FROM shirts WHERE id = :id", id=id)
        # Extract values from selected shirt record
        # Check if shirt is on sale to determine price
        if(goods[0]["onSale"] == 1):
            price = goods[0]["onSalePrice"]
        else:
            price = goods[0]["price"]
        samplename = goods[0]["samplename"]
        image = goods[0]["image"]
        subTotal = qty * price
        # Insert selected shirt into shopping cart
        db.execute("INSERT INTO cart (id, qty, samplename, image, price, subTotal) 
VALUES (:id, :qty, :samplename, :image, :price, :subTotal)", id=id, qty=qty, 
samplename=samplename, image=image, price=price, subTotal=subTotal)
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal), 
price, id FROM cart GROUP BY samplename")
        shopLen = len(shoppingCart)
        # Rebuild shopping cart
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
        # Go back to cart page
        return render_template ("cart.html", shoppingCart=shoppingCart, 
shopLen=shopLen, total=total, totItems=totItems, display=display, session=session )
@app.route("/filter/")
def filter():
    if request.args.get('typeClothes'):
        query = request.args.get('typeClothes')
        shirts = db.execute("SELECT * FROM shirts WHERE typeClothes = :query ORDER BY
samplename ASC", query=query )
    if request.args.get('sale'):
        query = request.args.get('sale')
        shirts = db.execute("SELECT * FROM shirts WHERE onSale = :query ORDER BY
samplename ASC", query=query)
    if request.args.get('id'):
        query = int(request.args.get('id'))
        shirts = db.execute("SELECT * FROM shirts WHERE id = :query ORDER BY samplename
ASC", query=query)
    if request.args.get('kind'):
        query = request.args.get('kind')
        shirts = db.execute("SELECT * FROM shirts WHERE kind = :query ORDER BY
samplename ASC", query=query)
    if request.args.get('price'):
        query = request.args.get('price')
        shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice ASC")
    shirtsLen = len(shirts)
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    if 'user' in session:
        # Rebuild shopping cart
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")
        shopLen = len(shoppingCart)
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
        # Render filtered view
        return render_template ("index.html", shoppingCart=shoppingCart, shirts=shirts,
shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems, display=display,
session=session )
    # Render filtered view
    return render_template ( "index.html", shirts=shirts, shoppingCart=shoppingCart,
shirtsLen=shirtsLen, shopLen=shopLen, total=total, totItems=totItems, display=display)
@app.route("/checkout/")
def checkout():
    order = db.execute("SELECT * from cart")
    # Update purchase history of current customer
    for item in order:
        db.execute("INSERT INTO purchases (uid, id, samplename, image, quantity)
VALUES(:uid, :id, :samplename, :image, :quantity)", uid=session["uid"], id=item["id"],
samplename=item["samplename"], image=item["image"], quantity=item["qty"] )
    # Clear shopping cart
    db.execute("DELETE from cart")
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    # Redirect to home page
    return redirect('/')
@app.route("/remove/", methods=["GET"])
def remove():
    # Get the id of shirt selected to be removed
    out = int(request.args.get("id"))
    # Remove shirt from shopping cart
    db.execute("DELETE from cart WHERE id=:id", id=out)
    # Initialize shopping cart variables
    totItems, total, display = 0, 0, 0
    # Rebuild shopping cart
    shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")
    shopLen = len(shoppingCart)
    for i in range(shopLen):
        total += shoppingCart[i]["SUM(subTotal)"]
        totItems += shoppingCart[i]["SUM(qty)"]
    # Turn on "remove success" flag
    display = 1
    # Render shopping cart
    return render_template ("cart.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session )
@app.route("/login/", methods=["GET"])
def login():
    return render_template("login.html")
@app.route("/new/", methods=["GET"])
def new():
    # Render log in page
    return render_template("new.html")
@app.route("/logged/", methods=["POST"] )
def logged():
    # Get log in info from log in form
    user = request.form["username"].lower()
    pwd = request.form["password"]
    #pwd = str(sha1(request.form["password"].encode('utf-8')).hexdigest())
    # Make sure form input is not blank and re-render log in page if blank
    if user == "" or pwd == "":
        return render_template ( "login.html" )
    # Find out if info in form matches a record in user database
    query = "SELECT * FROM users WHERE username = :user AND password = :pwd"
    rows = db.execute ( query, user=user, pwd=pwd )
    # If username and password match a record in database, set session variables
    if len(rows) == 1:
        session['user'] = user
        session['time'] = datetime.now( )
        session['uid'] = rows[0]["id"]
    # Redirect to Home Page
    if 'user' in session:
        return redirect ( "/" )
    # If username is not in the database return the log in page
    return render_template ( "login.html", msg="Wrong username or password." )
@app.route("/history/")
def history():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    # Retrieve all shirts ever bought by current user
    myShirts = db.execute("SELECT * FROM purchases WHERE uid=:uid", uid=session["uid"])
    myShirtsLen = len(myShirts)
    # Render table with shopping history of current user
    return render_template("history.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session, myShirts=myShirts,
myShirtsLen=myShirtsLen)
@app.route("/logout/")
def logout():
    # clear shopping cart
    db.execute("DELETE from cart")
    # Forget any user_id
    session.clear()
    # Redirect user to login form
    return redirect("/")
@app.route("/register/", methods=["POST"] )
def registration():
    # Get info from form
    username = request.form["username"]
    password = request.form["password"]
    confirm = request.form["confirm"]
    fname = request.form["fname"]
    lname = request.form["lname"]
    email = request.form["email"]
    # See if username already in the database
    rows = db.execute( "SELECT * FROM users WHERE username = :username ", username =
username )
    # If username already exists, alert user
    if len( rows ) > 0:
        return render_template ( "new.html", msg="Username already exists!" )
    # If new user, upload his/her info into the users database
    new = db.execute ( "INSERT INTO users (username, password, fname, lname, email)
VALUES (:username, :password, :fname, :lname, :email)",
                    username=username, password=password, fname=fname, lname=lname,
email=email )
    # Render login template
    return render_template ( "login.html" )
@app.route("/cart/")
def cart():
    if 'user' in session:
        # Clear shopping cart variables
        totItems, total, display = 0, 0, 0
        # Grab info currently in database
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")
        # Get variable values
        shopLen = len(shoppingCart)
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
    # Render shopping cart
    return render_template("cart.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session)
```