**TRANG WEB BÁN HÀNG**

**Ngô Văn Khánh**

**Tin10A5**

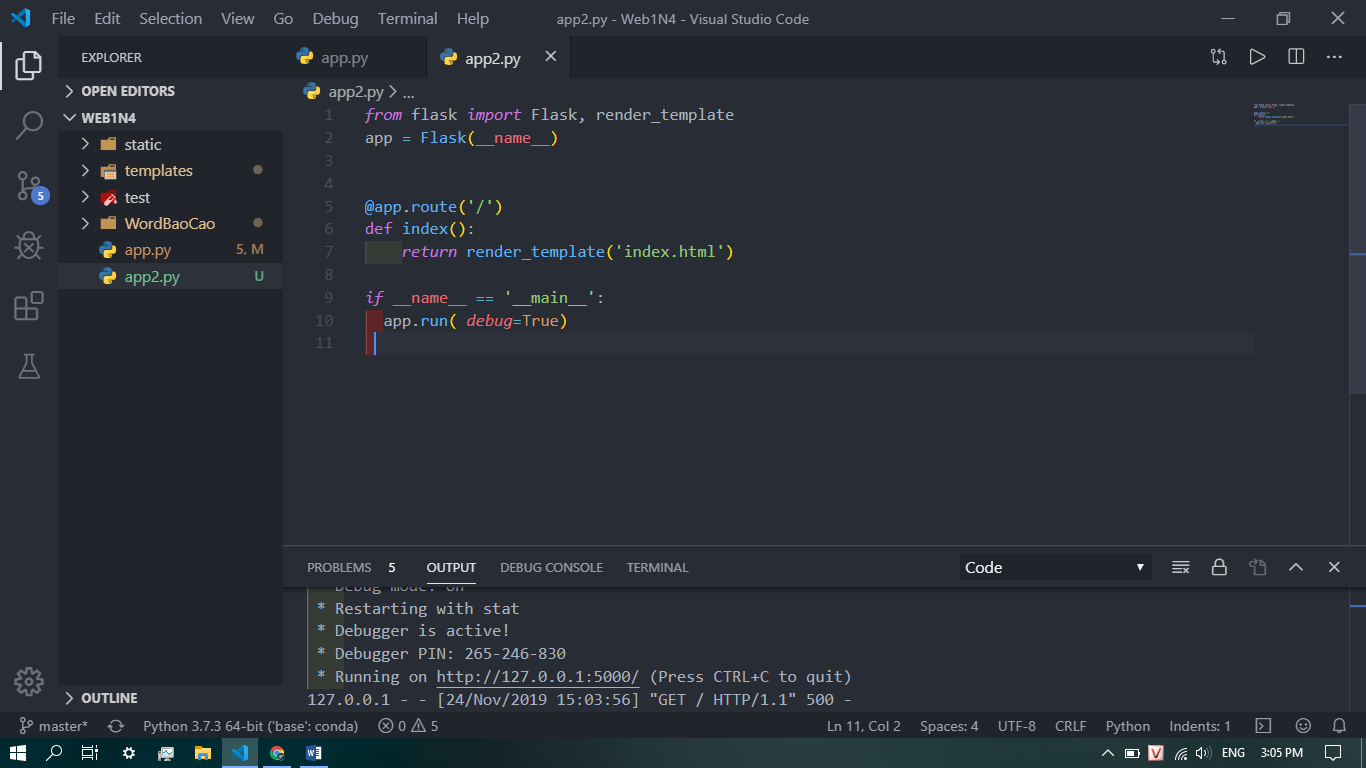
**16103100478**

1. Giới thiệu
2. Mô tả
   1. Vai trò Admin( Đang phát triển)

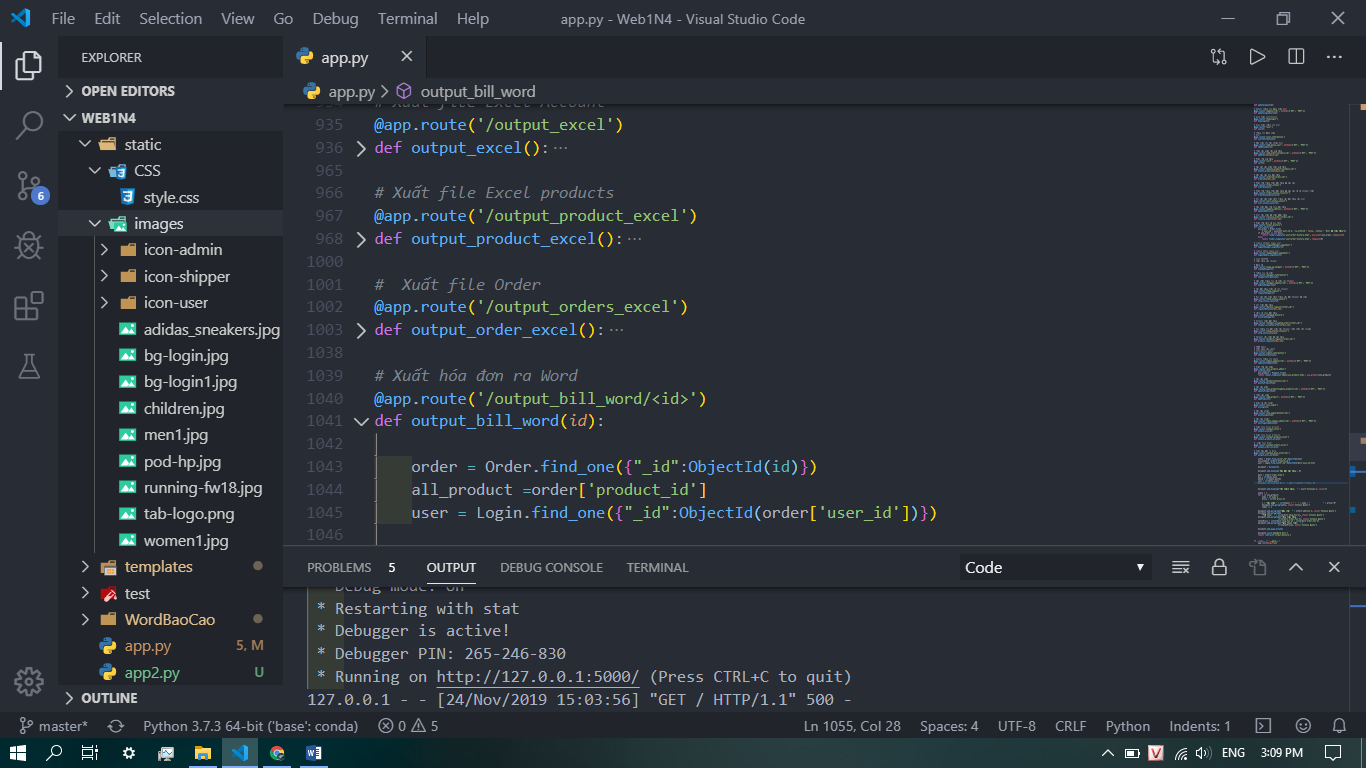
Quản lý thông tin các user, shipper, sản phẩm, đơn hàng. Có quyền truy cập, thêm, sửa, xóa database.

* 1. Vai trò User
     + - Thêm sản phẩm vào giỏ hàng
       - Tiến hành gửi đơn hàng
       - Các bước xác nhận đơn hàng thông báo qua email
       - Nạp tiền để tiến hành thực hiện chức năng mua hàng
       - Quản lý thông tin cá nhân
  2. Vai trò Shipper
     + - Nhận đơn hàng mà user đã gửi đi
       - Xác nhận các bước đơn hàng
       - Hoàn thành đơn hàng để nhận được phí ship
       - Quản lý thông tin cá nhân

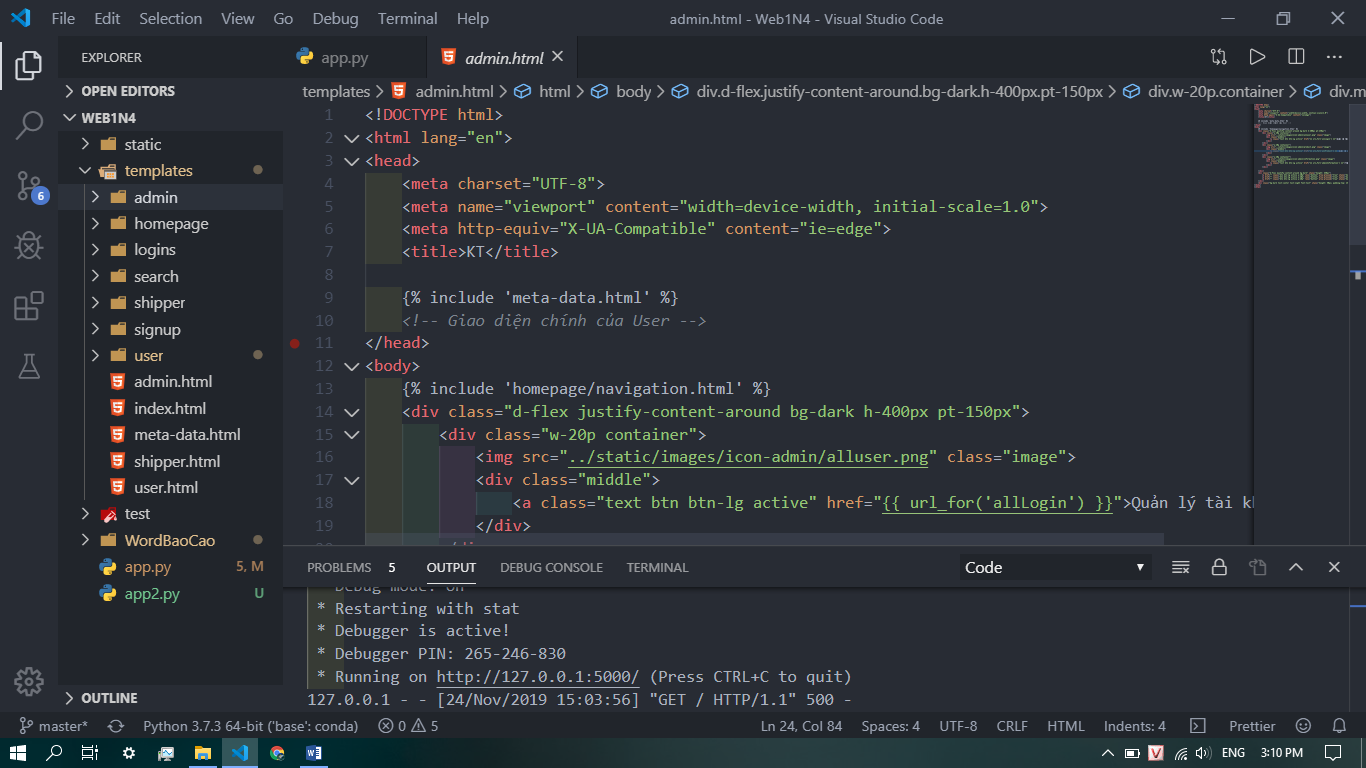
1. Cách xây dựng trang Web Python bằng Flask
   * + - Hình mẫu



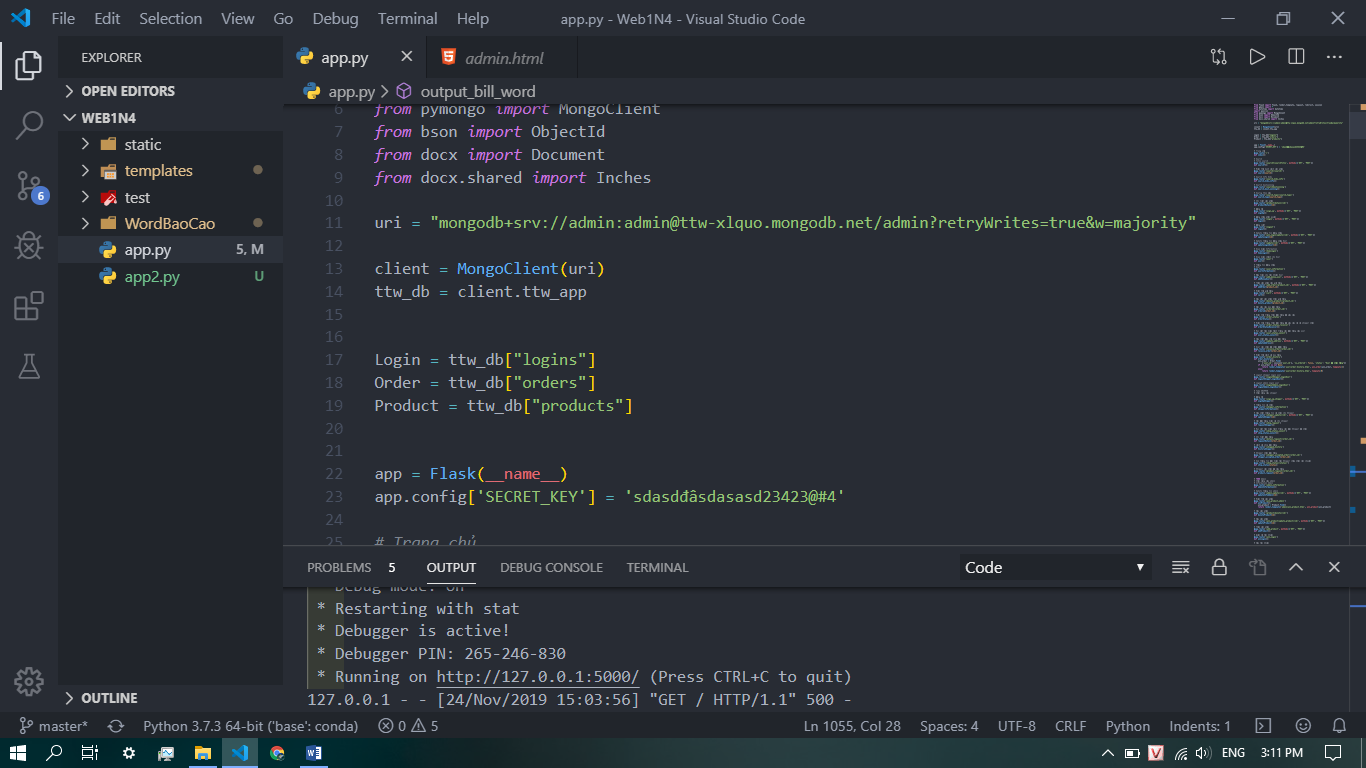
* + - * Tạo các folder con theo cấu trúc: static– templates – app.py(file)
      * Trong folder static có 2 folder con là CSS( chứa file style.css) và images( chứa các file ảnh của web)



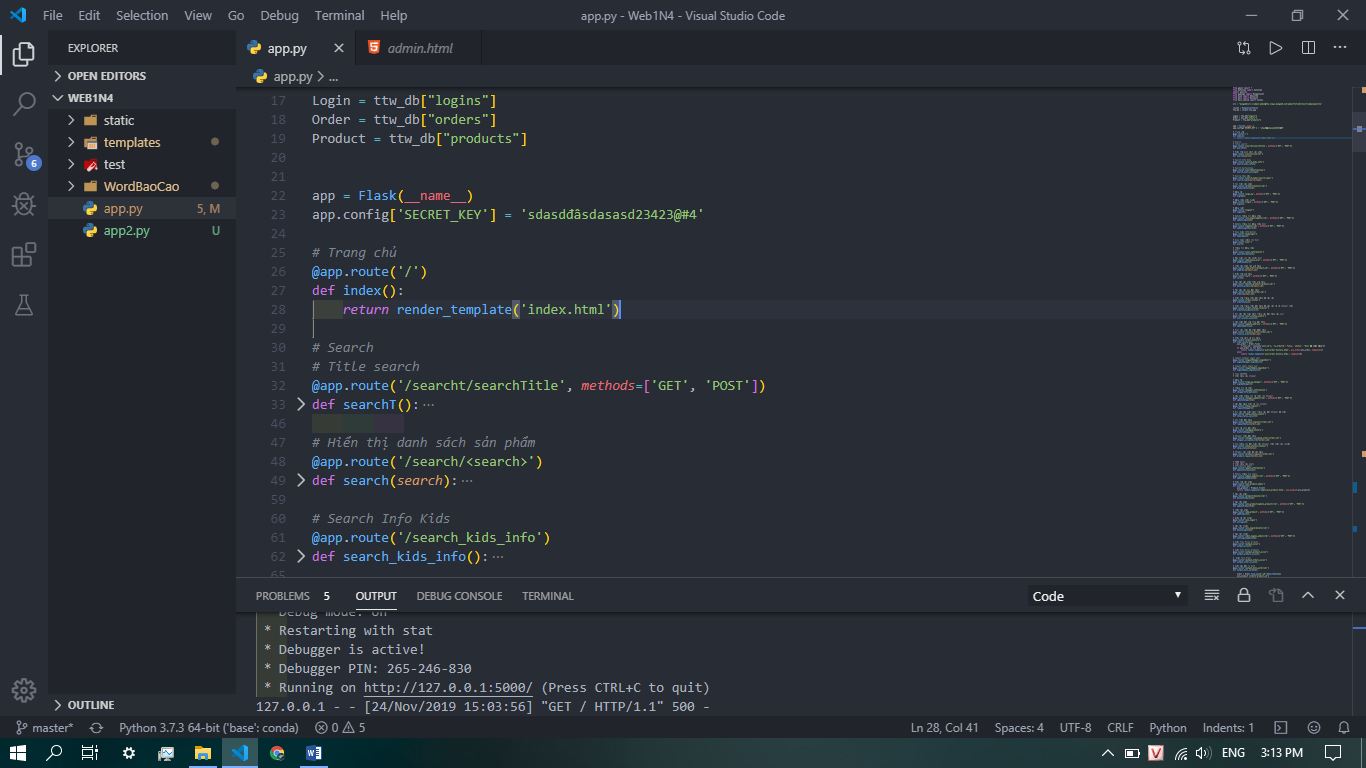
* + - * Folder templates chứa tất cả các trang html



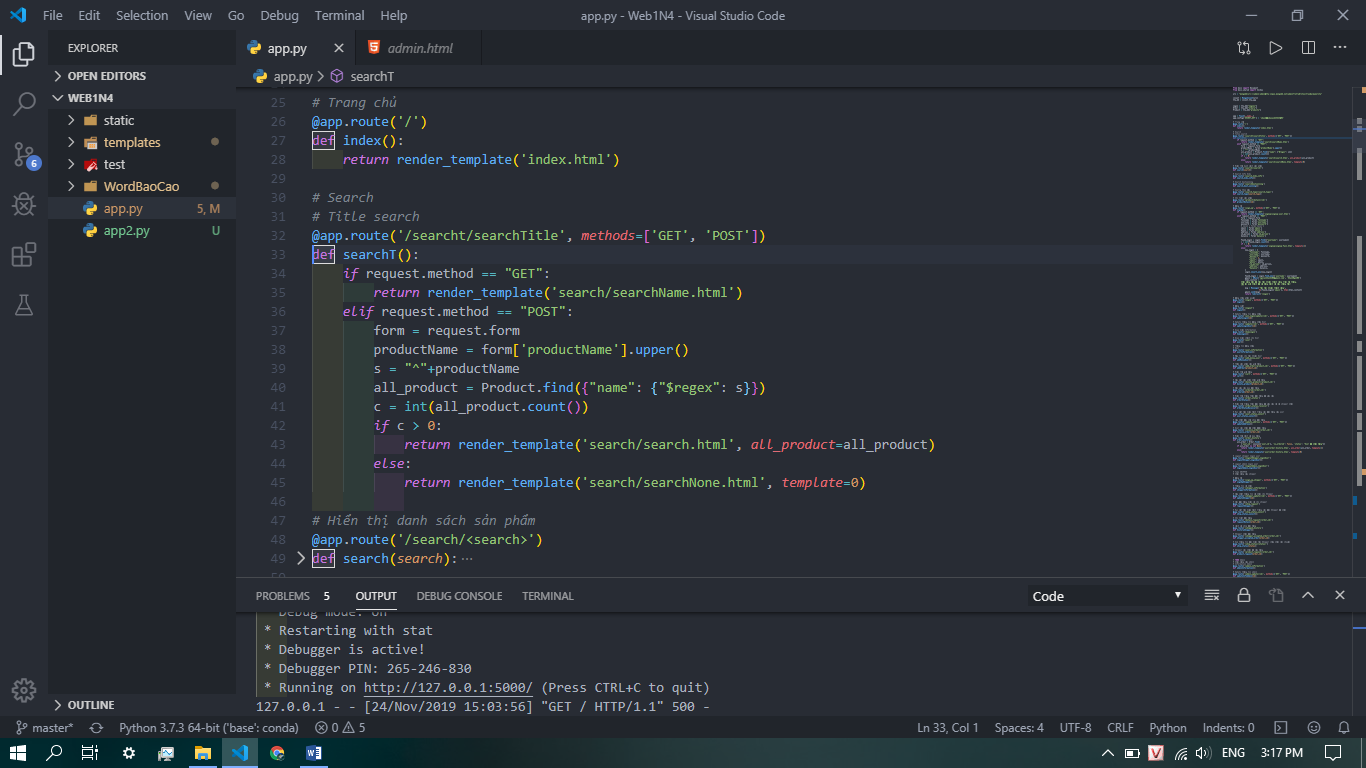
* + - * File app.py chứa code của chương trình



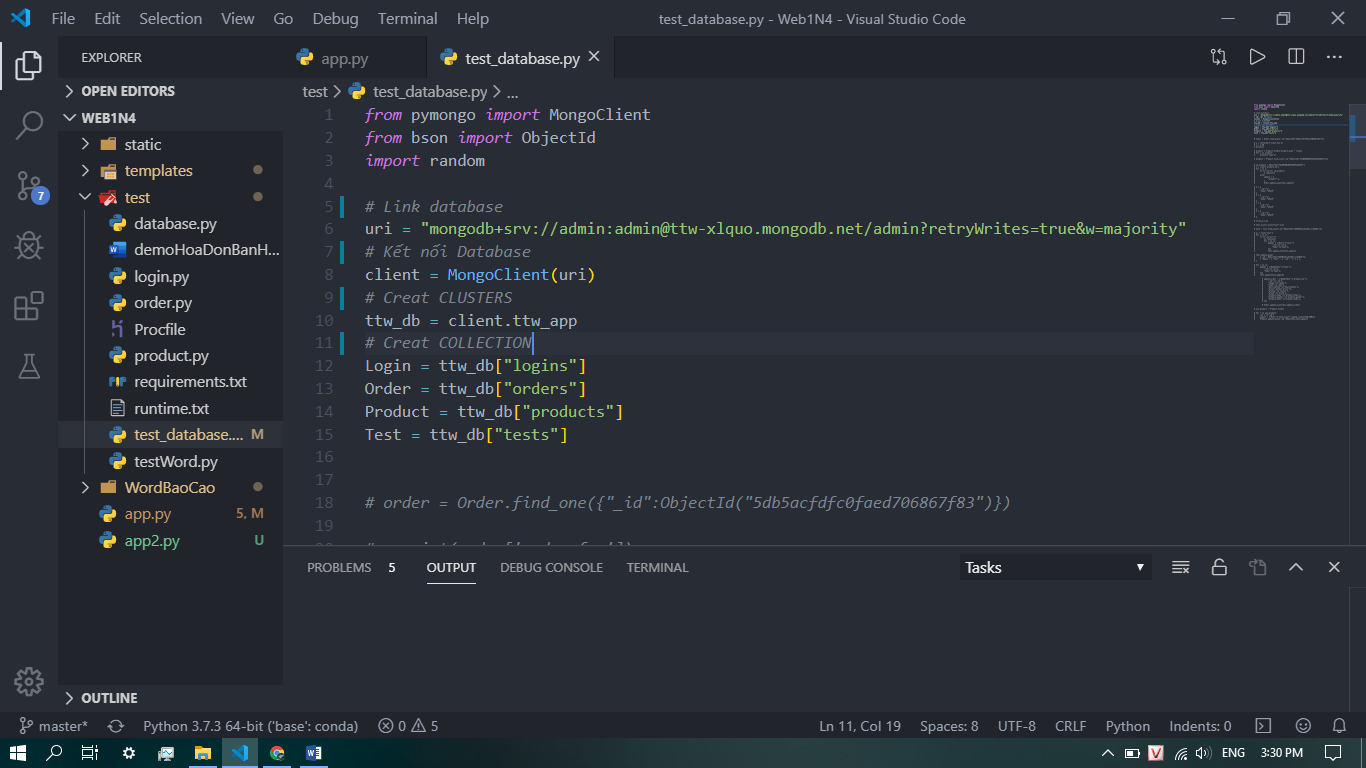
* + - * #Trang chủ: khi thực hiện hàm index sẽ trả về trang ‘index.html’ ở trong folder templates



* + - * #Search có 2 phương thức methods=[‘GET’,’POST’]. Khi trang được yêu cầu, method == GET sẽ thực hiện trả về trang ‘searchName.html’. Khi trang gửi đi yêu cầu(form method = ‘POST’), method == POST sẽ thực hiện lấy dữ liệu Client gửi đi và thực hiện yêu cầu. Tìm được kết quả thì trả về và gửi sang bên ‘search.html’ .



* + - * Thực hiện kết nối đến Database. Trong 1 Collections sẽ có nhiều Document, mỗi Document là một bản ghi( chứa các trường dữ liệu)



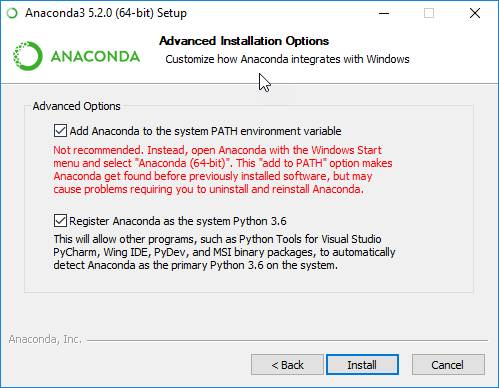
* + - * Code chương trình file app.py

|  |  |
| --- | --- |
|  | from flask import Flask, render\_template, request, redirect, session |
|  | from gmail import \* |
|  | from datetime import datetime |
|  | import pyexcel |
|  | from pymongo import MongoClient |
|  | from bson import ObjectId |
|  | from docx import Document |
|  | from docx.shared import Inches |
|  |  |
|  | # uri để kết nối đến database |
|  | uri = "mongodb+srv://admin:admin@ttw-xlquo.mongodb.net/admin?retryWrites=true&w=majority" |
|  |  |
|  | # thực hiện kết nối |
|  | client = MongoClient(uri) |
|  | # get data |
|  | ttw\_db = client.ttw\_app |
|  |  |
|  | # get collection |
|  | Login = ttw\_db["logins"] |
|  | Order = ttw\_db["orders"] |
|  | Product = ttw\_db["products"] |
|  |  |
|  |  |
|  | app = Flask(\_\_name\_\_) |
|  | app.config['SECRET\_KEY'] = 'sdasdđâsdasasd23423@#4' |
|  |  |
|  | # Trang chủ |
|  | @app.route('/') |
|  | def index(): |
|  | return render\_template('index.html') # trả về trang index.html |
|  |  |
|  | # Search |
|  | # Title search |
|  | @app.route('/searcht/searchTitle', methods=['GET', 'POST']) |
|  | def searchT(): |
|  | if request.method == "GET": |
|  | return render\_template('search/searchName.html') |
|  | elif request.method == "POST": |
|  | form = request.form |
|  | productName = form['productName'].upper() |
|  | s = "^"+productName |
|  | all\_product = Product.find({"name": {"$regex": s}}) |
|  | c = int(all\_product.count()) |
|  | if c > 0: |
|  | return render\_template('search/search.html', all\_product=all\_product) |
|  | else: |
|  | return render\_template('search/searchNone.html', template=0) |
|  |  |
|  | # Hiển thị danh sách sản phẩm |
|  | @app.route('/search/<search>') |
|  | def search(search): |
|  | session['search'] = search |
|  | all\_product\_gender = Product.find({'product\_gender': search}) |
|  | if all\_product\_gender is not None: |
|  | return render\_template('search/search.html', all\_product=all\_product\_gender) |
|  | # elif session['search'] == "Kids": |
|  | elif search == "Kids": |
|  | all\_product\_kids = Product.find({'product\_kids': True}) |
|  | if all\_product\_kids is not None: |
|  | return render\_template('search/search.html', all\_product=all\_product\_kids) |
|  |  |
|  | # Search Info Kids |
|  | @app.route('/search\_kids\_info') |
|  | def search\_kids\_info(): |
|  | product = Product.find({"product\_kids": True}) |
|  | return render\_template('search/search.html', all\_product=product) |
|  |  |
|  | # Search bestselling |
|  | @app.route('/searchbestselling') |
|  | def search\_best\_selling(): |
|  | all\_product = Product.find({"sold\_count": {"$gt": 150}}) |
|  | if all\_product is not None: |
|  | return render\_template('search/bestselling.html', all\_product=all\_product) |
|  |  |
|  | # Searcg theo mẫu |
|  | @app.route('/search\_type/<search\_type>') |
|  | def search\_type(search\_type): |
|  | session['search\_type'] = search\_type |
|  | all\_product\_type = Product.find({"product\_type": search\_type}) |
|  | if all\_product\_type is not None: |
|  | return render\_template('search/search.html', all\_product=all\_product\_type) |
|  |  |
|  | # Chi tiết sản phẩm |
|  | @app.route('/product/detail/<id>') |
|  | def productDetail(id): |
|  | product = Product.find\_one({"\_id": ObjectId(id)}) |
|  | view\_ex = int(product['view']) |
|  | view\_new = view\_ex + 1 |
|  | update = {"$set": {"view": view\_new}} |
|  | Product.update\_one({"\_id": ObjectId(id)}, update) |
|  | return render\_template('search/detail.html', detail\_product=product) |
|  |  |
|  | # Đăng ký |
|  | @app.route('/sign\_up', methods=['GET', 'POST']) |
|  | def signUp(): |
|  | if request.method == 'GET': |
|  | return render\_template('signup/signup-user.html') |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  | fullname = form['fullname'] |
|  | username = form['username'] |
|  | password = form['password'] |
|  | # level = form['level'] |
|  | email = form['email'] |
|  | phone = form['phone'] |
|  | address = form['address'] |
|  | id\_person = form['id\_person'] |
|  | balance = form['balance'] |
|  |  |
|  | found\_login = Login.find({"username": username}) |
|  | c = int(found\_login.count()) |
|  | if c > 0: |
|  | return render\_template('signup/signup-fail.html', template=1) |
|  | else: |
|  | new\_login = { |
|  | "fullname": fullname, |
|  | "username": username, |
|  | "password": password, |
|  | "level": 2, |
|  | "email": email, |
|  | "phone": phone, |
|  | "id\_person": id\_person, |
|  | "address": address, |
|  | "balance": balance, |
|  | } |
|  | Login.insert\_one(new\_login) |
|  |  |
|  | found\_login = Login.find\_one({'username': username}) |
|  | gmail = GMail('ngovankhanh108@gmail.com', 'KhanhNgo108') |
|  | html\_content = ''' |
|  | Chúc mừng bạn đã tạo tài khoản thành công trên hệ thống. |
|  | Cảm ơn quý khách đã sử dụng dịch vụ của chúng tôi! |
|  | ''' |
|  | msg = Message('Tạo tài khoản thành công', |
|  | to=found\_login['email'], html=html\_content) |
|  | gmail.send(msg) |
|  | return redirect('/login') |
|  |  |
|  | # Đăng nhập phân quyền |
|  | @app.route('/login', methods=['GET', 'POST']) |
|  | def login(): |
|  | if request.method == 'GET': |
|  | return render\_template('logins/login-user.html') # trả về html khi người dùng yêu cầu |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  | login\_username = form['username'] |
|  | login\_password = form['password'] |
|  | login = Login.find\_one({'username': login\_username}) # tìm kiếm username có trong database |
|  | if login is None: # nếu không tìm thấy username |
|  | return render\_template('logins/login-fail.html') # thông báo đăng nhập không thành công |
|  | else: |
|  | if login\_password == login['password']: # so sánh password người dùng nhập với password trong database nếu trùng thì đăng nhập |
|  | session['loggedin'] = True # lưu phiên đăng nhập |
|  | session['level\_loggedin'] = str((login['level'])) # lưu phiên thông tin level của người đăng nhập |
|  | # phân quyền đăng nhập |
|  | if session['level\_loggedin'] == "2": # user level = 2, lưu vào phiên đăng nhập |
|  | session["user\_id"] = str(login["\_id"]) # lưu session id = id của user đăng nhập |
|  | return render\_template('user.html', login=login) # trả về html và truyền login sang html |
|  | elif session['level\_loggedin'] == "1":# level = 1 sẽ là admin |
|  | session["admin\_id"] = str(login["\_id"]) |
|  | return render\_template('admin.html', login=login) |
|  | elif session['level\_loggedin'] == "3": #level sẽ là shipper |
|  | session["shipper\_id"] = str(login["\_id"]) |
|  | return render\_template('shipper.html', login=login) |
|  | else: |
|  | return render\_template('logins/login-fail.html') # nếu password sai, trả về Login Fail |
|  |  |
|  | # Đăng xuất |
|  | @app.route('/logout') |
|  | def logout(): |
|  | if "loggedin" in session: # nếu tồn tại phiên trong session |
|  | if session['loggedin'] == False: # đã đăng xuất thì trả về trang chủ |
|  | return redirect('/') |
|  | else: |
|  | session['loggedin'] = False # phiên bằng false và kiểm tra level đang đăng nhập và thực hiện xóa phiên |
|  | if session['level\_loggedin'] == "1": # thực hiện kiểm tra phiên đăng nhập |
|  | del session['admin\_id'] # thực hiện xóa session để dăng xuất tài khoản |
|  | del session['level\_loggedin'] |
|  | elif session['level\_loggedin'] == "2": |
|  | del session['user\_id'] |
|  | del session['level\_loggedin'] |
|  | elif session['level\_loggedin'] == "3": |
|  | del session['shipper\_id'] |
|  | del session['level\_loggedin'] |
|  | return redirect('/') |
|  | else: |
|  | return redirect('/') |
|  |  |
|  | # Update thông tin đăng nhập |
|  | @app.route('/all\_login/update/<id>', methods=['GET', 'POST']) # <id> sẽ được truyền vào bằng id của document(bản ghi) |
|  | def updateLogin(id): # thực hiện hàm update với 1 tham số id được truyền vào |
|  | login = Login.find\_one({"\_id": ObjectId(id)}) # thực hiện tìm kiếm trường "\_id" trong collection với tham số id được truyền vào |
|  | if request.method == 'GET': |
|  | return render\_template('update\_login.html', login=login) # trả về trang update truyền document vừa tìm được sang html |
|  | elif request.method == 'POST': # nếu người dùng thực hiện gửi dữ liệu, thực hiện lấy dữ liệu |
|  | form = request.form # lấy dữ liệu từ form |
|  | fullname = form['fullname'] # gán dữ liệu ở form vào biến |
|  | username = form['username'] |
|  | password = form['password'] |
|  | level = form['level'] |
|  |  |
|  | # update database |
|  | update\_login = {"$set": { |
|  | "fullname": fullname, |
|  | "username": username, |
|  | "password": password, |
|  | "level": level, |
|  | }} |
|  | Login.update\_one(login, update\_login) |
|  | return redirect('/all\_login') |
|  |  |
|  | # Update thông tin đăng nhập User |
|  | @app.route('/update/<id>', methods=['GET', 'POST']) |
|  | def updateLoginUser(id): # hàm update thông tin của user |
|  | login = Login.find\_one({"\_id": ObjectId(id)}) # tìm user đang đăng nhập |
|  | if request.method == 'GET': |
|  | return render\_template('user/user-update.html', update\_user=login) |
|  | elif request.method == 'POST': |
|  | # lấy dữ liệu từ form |
|  | form = request.form |
|  | fullname = form['fullname'] |
|  | username = form['username'] |
|  | password = form['password'] |
|  | phone = form['phone'] |
|  | email = form['email'] |
|  | address = form['address'] |
|  | # level = form['level'] |
|  |  |
|  | # thực hiện update bằng pymongo |
|  | update\_login = {"$set": { |
|  | "fullname": fullname, |
|  | "username": username, |
|  | "password": password, |
|  | "level": 2, |
|  | "phone": phone, |
|  | "email": email, |
|  | "address": address, |
|  | }} |
|  | Login.update\_one(login, update\_login) |
|  | return redirect('/homeLogin') |
|  |  |
|  | # Giao diện Information |
|  | @app.route('/homeLogin') |
|  | def homeLogin(): # hàm giao diện của tài khoản khi đăng nhập |
|  | if "loggedin" in session: |
|  | if session['loggedin'] == True: # khi có phiên đăng nhập tồn tại, thực hiện kiểm tra |
|  | if session['level\_loggedin'] == "2": |
|  | return render\_template('user.html') # trả về trang user nếu session == 2 |
|  | elif session['level\_loggedin'] == "1": |
|  | return render\_template('admin.html')# trả về trang admin nếu session == 1 |
|  | elif session['level\_loggedin'] == "3": |
|  | return render\_template('shipper.html')# trả về trang shipper nếu session == 3 |
|  | else: |
|  | return redirect('/login') |
|  | return redirect('/login') |
|  |  |
|  | # Giao diện chính cho User |
|  | @app.route('/user') |
|  | def user(): |
|  | if "user-loggedin" in session: |
|  | if session['user-loggedin'] == True: |
|  | return render\_template('user.html') |
|  | else: |
|  | return redirect('/login') |
|  | else: |
|  | return redirect('/login') |
|  |  |
|  | # Thông tin đăng nhập |
|  | # User |
|  | @app.route('/user\_information') |
|  | def userInformation(): |
|  | id = session["user\_id"] |
|  | login = Login.find\_one({"\_id": ObjectId(id)}) |
|  | if login is not None: |
|  | return render\_template('user/information.html', found\_user=login) |
|  | else: |
|  | return 'Login is not found' |
|  |  |
|  | # Nạp tiền cho tài khoản User |
|  | @app.route('/add\_money\_user', methods=['GET', 'POST']) |
|  | def addMoneyUser(): # hàm nạp tiền cho tài khoản user |
|  | user\_login = Login.find\_one({"\_id": ObjectId(session["user\_id"])}) # thực hiện tìm kiếm tài khoản đang đăng nhập với id lưu session |
|  | if request.method == 'GET': |
|  | return render\_template('user/add-money.html', found\_user=user\_login) |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  | money = int(form['money']) # lấy số tiền user nhập muốn nạp |
|  | ex\_money = int(user\_login['balance']) # lấy số tiền hiện có trong tài khoản của user |
|  | balance = money + ex\_money # tính total bằng số tiền cũ + số tiền user nhập |
|  |  |
|  | # thực hiện update số tiền trong tài khoản |
|  | add\_money = {"$set": { |
|  | "balance": balance, |
|  | }} |
|  |  |
|  | Login.update\_one(user\_login, add\_money) |
|  | return redirect('/add\_money\_user') |
|  |  |
|  | # Thêm sản phẩm vào giỏ hàng |
|  | @app.route('/add\_order/<product\_id>', methods=['GET', 'POST']) |
|  | def addOrder(product\_id): # thực hiện hàm tạo đơn order, thêm sản phẩm vào giỏ hàng |
|  | if "loggedin" in session: # nếu tồn tại phiên đăng nhập thì thực hiện |
|  | if session['loggedin'] == True: |
|  | if session['level\_loggedin'] == "2": # nếu là người dùng vơi vai trò user mới thực hiện |
|  | # tìm kiếm thông tin user |
|  | found\_user = Login.find\_one({"\_id": ObjectId(session["user\_id"])}) |
|  | # tìm kiếm thông tin sản phẩm |
|  | found\_product = Product.find\_one({"\_id": ObjectId(product\_id)}) |
|  | # tìm kiếm thông tin đơn order |
|  | found\_order = Order.find\_one({"user\_id": session['user\_id'], "is\_ordered": False, "status": "Shipper chưa nhận đơn"}) |
|  | # nếu tìm thấy sản phẩm thì tiếp tục thực hiện |
|  | if found\_product is not None: |
|  | # nếu đơn order đã tồn tại thực hiện thêm sản phẩm |
|  | if found\_order is not None: |
|  | # lấy tổng tiền hiện tại của đơn order |
|  | ex\_fee = found\_order['order\_fee'] |
|  | # cập nhật tổng tiền = giá bán sản phẩm + tổng tiền hiện tại |
|  | order\_fee = int(found\_product['price'])+int(ex\_fee) |
|  | # phí ship sẽ bằng 0,1 x tổng tiền |
|  | ship\_fee = 0.1 \* \ |
|  | int(int(found\_product['price'])+int(ex\_fee)) |
|  | # Tăng số lượng đã bán của sản phẩm |
|  | sold\_count\_ex = int(found\_product['sold\_count']) |
|  | sold\_count\_new = sold\_count\_ex + 1 |
|  | update\_product2 = { |
|  | "$set": {"sold\_count": sold\_count\_new}} |
|  | Product.update\_one(found\_product, update\_product2) |
|  | # Cập nhật giá tiền mới |
|  | found\_order\_update\_2 = {"$set": { |
|  | "order\_time": datetime.now(), |
|  | "order\_fee": order\_fee, |
|  | "ship\_fee": ship\_fee, |
|  | # Thêm 1 sản phẩm vào giỏ hàng |
|  | }, "$addToSet": {"product\_id": { # thông tin sản phẩm |
|  | "\_id": found\_product['\_id'], |
|  | "image": found\_product['image'], |
|  | "name": found\_product['name'], |
|  | "description": found\_product['description'], |
|  | "price": int(found\_product['price']), |
|  | "status": found\_product['status'], |
|  | "brand": found\_product['brand'], |
|  | "product\_type": found\_product['product\_type'], |
|  | "product\_gender": found\_product['product\_gender'], |
|  | "product\_kids": found\_product['product\_kids'], |
|  | }}} |
|  | Order.update\_one(found\_order, found\_order\_update\_2) |
|  | b = session['search'] |
|  | a = "/search/" + b |
|  | return redirect(a) |
|  | else: # trường hợp còn lại là chưa tồn tại đơn order, và sẽ tạo mới |
|  | # Tăng số lượng đã bán của sản phẩm |
|  | sold\_count\_ex = int(found\_product['sold\_count']) |
|  | sold\_count\_new = sold\_count\_ex + 1 |
|  | update\_product2 = { |
|  | "$set": {"sold\_count": sold\_count\_new}} |
|  | Product.update\_one(found\_product, update\_product2) |
|  | # Tạo 1 đơn mới |
|  | add\_order = { |
|  | "user\_id": session['user\_id'], # lưu id người mua |
|  | "product\_id": [{ # lưu thông tin sản phẩm |
|  | "\_id": found\_product['\_id'], |
|  | "image": found\_product['image'], |
|  | "name": found\_product['name'], |
|  | "description": found\_product['description'], |
|  | "price": int(found\_product['price']), |
|  | "status": found\_product['status'], |
|  | "brand": found\_product['brand'], |
|  | "product\_type": found\_product['product\_type'], |
|  | "product\_gender": found\_product['product\_gender'], |
|  | "product\_kids": found\_product['product\_kids'], |
|  | }], |
|  | "address": found\_user['address'], # địa chỉ người mua |
|  | "order\_time": datetime.now(), # thời gian gửi yêu cầu |
|  | "order\_fee": int(found\_product['price']), # tiền sản phẩm |
|  | "ship\_fee": 0.1\*int(found\_product['price']), #phí ship |
|  | "is\_ordered": False, # trạng thái order |
|  | "status": "Shipper chưa nhận đơn", # trạng thái của đơn hàng khi vận chuyển |
|  | "shipper\_id": "null", # id của shipper nhận đơn |
|  | } |
|  | Order.insert\_one(add\_order) |
|  |  |
|  | b = session['search'] |
|  | a = "/search/" + b |
|  | return redirect(a) |
|  | else: |
|  | return 'Not found' |
|  | else: |
|  | return redirect('/login') |
|  | else: |
|  | return redirect('/login') |
|  | else: |
|  | return redirect('/login') |
|  |  |
|  | # Hiển thị giỏ hàng |
|  | @app.route('/cart', methods=['GET', 'POST']) |
|  | def cart(): # hàm hiển thị giỏ hàng |
|  | if "loggedin" in session: # nếu tồn tại phiên đăng nhập |
|  | if session['loggedin'] == True: |
|  | if session['level\_loggedin'] == "2": # kiểm tra nếu level phiên đăng nhập == 2 là user thì thực hiện |
|  | # tìm đơn order hiện tại của user |
|  | found\_order = Order.find\_one({"user\_id": session['user\_id'], 'is\_ordered': False, 'status': "Shipper chưa nhận đơn"}) |
|  | # nếu người dùng yêu cầu trang giỏ hàng thì sẽ trả về trang "cart.html" |
|  | if request.method == 'GET': |
|  | if found\_order is not None: # nếu order tồn tại thì thực hiện hiển thị |
|  | cart = found\_order |
|  | order\_product = cart['product\_id'] |
|  | return render\_template('user/cart.html', cart=cart, order\_product=order\_product, template=1) |
|  | else: # nếu không tìm thấy đơn order nào sẽ trả về đơn hàng trống |
|  | return render\_template('user/cart.html', template=0) |
|  | # chức năng cập nhật thời gian yêu cầu nhận hàng, mặc định sẽ là giao hàng bất cứ lúc nào(chỉnh sửa thời gian muốn nhận hàng theo ý muốn) |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  | request\_time = form['request\_time'] |
|  | # thực hiện lưu mới vào database |
|  | update = {"$set": { |
|  | 'request\_time': request\_time, |
|  | }} |
|  | Order.update\_one(found\_order, update) |
|  | return redirect('/cart') |
|  | # nếu level loggedin == 3 là shipper thì không có quyền được vào trang giỏ hàng |
|  | elif session['level\_loggedin'] == "3": |
|  | return redirect('/') |
|  | # nếu level loggedin == 1 là 1 thì không có quyền được vào trang giỏ hàng |
|  | elif session['level\_loggedin'] == "1": |
|  | return redirect('/') |
|  | else: |
|  | return redirect('/login') |
|  | else: |
|  | return redirect('/login') |
|  |  |
|  | # Xóa một sản phẩm khỏi giỏ hàng |
|  | @app.route('/delete\_product/<product\_id>') |
|  | def delete\_product(product\_id): # hàm xóa 1 sản phẩm có trong giỏ hàng |
|  | # tìm đơn order hiện có của user đang đăng nhập |
|  | order = Order.find\_one({'user\_id': session['user\_id'], 'is\_ordered': False, 'status': "Shipper chưa nhận đơn"}) |
|  |  |
|  | c = order['product\_id'] |
|  | # nếu tồn tại đơn order |
|  | if len(c) > 1: |
|  | # thực hiện tìm thông tin sản phẩm muốn xóa = id sản phẩm |
|  | product = Product.find\_one({"\_id": ObjectId(product\_id)}) |
|  | # Giảm số lượng đã bán của sản phẩm |
|  | sold\_count\_ex = int(product['sold\_count']) |
|  | sold\_count\_new = sold\_count\_ex - 1 |
|  | update\_product2 = {"$set": {"sold\_count": sold\_count\_new}} |
|  | Product.update\_one(product, update\_product2) |
|  | # Xóa 1 sản phẩm |
|  | Order.update\_one( |
|  | # update đơn order hiện tại user muốn xóa sản phẩm |
|  | {'\_id': ObjectId(order['\_id'])}, |
|  | # xóa 1 object khỏi trưởng product\_id thuộc kiểu array , $pull là lấy ra |
|  | {'$pull': {"product\_id": {'\_id': ObjectId(product\_id)}}}, |
|  | ) |
|  | # cập nhật lại giá tiền khi xóa 1 sản phẩm |
|  | order\_fee\_ex = int(order['order\_fee']) |
|  | product\_price = int(product['price']) |
|  | new\_order\_fee = int(order\_fee\_ex) - int(product\_price) |
|  | new\_ship\_fee = (0.1 \* new\_order\_fee) |
|  | # thực hiện cập nhật trên database |
|  | Order.update\_one( |
|  | {'\_id': ObjectId(order['\_id'])}, |
|  | {'$set': {"order\_fee": new\_order\_fee, "ship\_fee": new\_ship\_fee}}, |
|  | ) |
|  | return redirect('/cart') |
|  | # trường hợp đơn order tồn tại duy nhất 1 sản phẩm |
|  | elif len(c) == 1: |
|  | #thực hiện tìm kiếm thông tin sản phẩm bằng id |
|  | product = Product.find\_one({"\_id": ObjectId(product\_id)}) |
|  | # Giảm số lượng đã bán của sản phẩm |
|  | sold\_count\_ex = int(product['sold\_count']) |
|  | sold\_count\_new = sold\_count\_ex - 1 |
|  | update\_product2 = {"$set": {"sold\_count": sold\_count\_new}} |
|  | Product.update\_one(product, update\_product2) |
|  | # thực hiện xóa document hiện tại khi chỉ còn 1 object(sản phẩm) trong trường "product\_id" |
|  | Order.delete\_one(order) |
|  | return redirect('/cart') |
|  |  |
|  | # Gửi yêu cầu mua đơn hàng |
|  | @app.route('/ordered/<order\_id>') |
|  | def ordered(order\_id): # hàm gửi yêu cầu nhận id của đơn order |
|  | # tìm thông tin đơn order |
|  | found\_order = Order.find\_one({"\_id": ObjectId(order\_id)}) |
|  | # tìm thông tin người dùng gửi yêu cầu |
|  | found\_user = Login.find\_one({"\_id": ObjectId(found\_order['user\_id'])}) |
|  | # nếu tồn tại đơn order |
|  | if found\_order is not None: |
|  | # tổng tiền sẽ bằng phí sản phẩm + phí ship |
|  | total = int(found\_order['order\_fee']) + int(found\_order['ship\_fee']) |
|  | # lấy số tiền hiện có của user |
|  | ex\_balance = int(found\_user['balance']) |
|  | # nếu số tiền hiện có của user > tổng tiền đơn hàng thì thực hiện |
|  | if ex\_balance >= total: |
|  | # Trừ tiền |
|  | balance = ex\_balance - total |
|  | update\_user = {"$set": { |
|  | "balance": balance, |
|  | }} |
|  | Login.update\_one(found\_user, update\_user) |
|  | # cập nhật trạng thái đơn hàng chuyển từ false sang True (đã gửi yêu cầu mua hàng) |
|  | update\_order = {"$set": { |
|  | "is\_ordered": True, |
|  | }} |
|  | Order.update\_one(found\_order, update\_order) |
|  | # Gửi Mail |
|  | gmail = GMail('ngovankhanh108@gmail.com', 'KhanhNgo108') |
|  | html\_content = '''Đơn hàng của bạn đã được gửi đi, |
|  | Tài khoản của bạn sẽ bị trừ đi {{total}}$ tổng chi phí sản phẩm và vận chuyển của đơn hàng này. |
|  | Cảm ơn bạn đã sử dụng dịch vụ của chúng tôi''' |
|  | html\_content1 = html\_content.replace("{{total}}", str(total)) |
|  | msg = Message( |
|  | 'GỬI YÊU CẦU', |
|  | to=found\_user['email'], |
|  | html=html\_content1 |
|  | ) |
|  | gmail.send(msg) |
|  | return render\_template('user/accepted.html', template=1) |
|  | else: |
|  | return render\_template('user/accepted.html', template=0) |
|  | else: |
|  | return 'Order is not found' |
|  |  |
|  | # Hiển thị trạng thái đơn hàng đã yêu cầu |
|  | @app.route('/order\_status') |
|  | def orderStatus(): # hàm hiển thị tất cả các đơn hàng đã gửi đi của user |
|  | # tìm tất cả các đơn hàng của user đăng nhập với id bằng phiên id đăng nhập hiện tại, tình trạng order = true(đã gửi yêu cầu),shipper\_id = null(đã gửi yêu cầu và chưa shipper nhận đơn) |
|  | all\_order = Order.find({'user\_id': session['user\_id'], 'is\_ordered': True, 'shipper\_id': "null"}) |
|  | # đếm số lượng đơn order tìm thấy trong database |
|  | c = int(all\_order.count()) |
|  | # nếu > 0(tồn tại 1 đơn trở lên) sẽ thực hiện hiển thị |
|  | if c > 0: |
|  | return render\_template('user/order-status.html', all\_order=all\_order, template=1) |
|  | else: # trả về template thông báo hiện không có đơn nào tồn tại |
|  | return render\_template('user/order-status.html', template=0) |
|  |  |
|  | # Hiển thị trạng thái đơn hàng đã yêu cầu và có shipper nhận |
|  | @app.route('/order\_status\_receiver') |
|  | def orderStatusReceiver(): # hàm hiển thị đơn hàng đã có shipper nhận |
|  | # thực hiện tìm kiếm những đơn có shipper chấp nhận vận chuyển, thông tin vận chuyển lưu vào status |
|  | all\_order = Order.find({'user\_id': session['user\_id'], 'is\_ordered': True,"status": "Shipper đã nhận hàng, bắt đầu tiến hành vận chuyển"}) |
|  | c = int(all\_order.count()) |
|  | if c > 0: |
|  | return render\_template('user/order-status.html', all\_order=all\_order, template=1) |
|  | else: |
|  | return render\_template('user/order-status.html', template=0) |
|  |  |
|  | # Hai nút tìm kiếm tình trạng các đơn hàng của user |
|  | @app.route('/user\_status\_select') |
|  | def user\_status\_select(): |
|  | # hàm phụ để hiện thị 2 nút bấm cho người dùng khi người dùng vào trang các đơn hiện tại |
|  | return render\_template('user/user-status-select.html') |
|  |  |
|  | # Cập nhật địa chỉ ship đơn hàng |
|  | @app.route('/update\_address', methods=['GET', 'POST']) |
|  | def updateAddress(): |
|  | # hàm cập nhật địa chỉ ship hàng trước khi gửi yêu cầu mua hàng, mặc định sẽ là địa chỉ mặc định khi đăng ký tài khoản |
|  | found\_order = Order.find\_one({'user\_id': session['user\_id'], 'is\_ordered': False, 'status': "Shipper chưa nhận đơn"}) |
|  | if request.method == 'GET': |
|  | return render\_template('user/update-address.html', found\_order=found\_order) |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  | address = form['address'] |
|  | # thực hiện update địa chỉ ship hàng |
|  | update = {"$set": { |
|  | "address": address, |
|  | }} |
|  | Order.update\_one(found\_order, update) |
|  | return redirect('/cart') |
|  |  |
|  | # User xác nhận đã nhận được hàng |
|  | @app.route('/receive\_order/<order\_id>') |
|  | def receive\_order(order\_id): |
|  | # khi nhận được hàng, user thực hiện bấm nút xác nhận "Đã nhận được hàng." |
|  | # tìm đơn order hiện tại |
|  | found\_order = Order.find\_one({"\_id": ObjectId(order\_id)}) |
|  | # nếu tìm thấy đơn order |
|  | if found\_order is not None: |
|  | # Chuyển trạng thái đơn sang false và cập nhật trạng thái status = "User đã nhận được hàng" |
|  | update1 = {"$set": { |
|  | "status": "User đã nhận hàng", |
|  | "is\_ordered": False, |
|  | }} |
|  | Order.update\_one(found\_order, update1) |
|  | # Add phí vận chuyển cho Shipper |
|  | # tìm thông tin shipper qua id |
|  | id\_shipper = found\_order['shipper\_id'] |
|  | found\_shipper = Login.find\_one({"\_id": ObjectId(id\_shipper)}) |
|  | # thực hiện cộng tiền ship vào tài khoản shipper |
|  | ship\_fee = found\_order['ship\_fee'] |
|  | if found\_shipper['balance'] == 0: |
|  | update2 = {"$set": {"balance": ship\_fee, }} |
|  | Login.update\_one(found\_shipper, update2) |
|  | else: |
|  | balance = int(found\_shipper['balance'])+int(ship\_fee) |
|  | update3 = {"$set": {"balance": balance, }} |
|  | Login.update\_one(found\_shipper, update3) |
|  | # Gửi mail cho Shipper |
|  | gmail = GMail('ngovankhanh108@gmail.com', 'KhanhNgo108') |
|  | html\_content = '''Bạn đã vận chuyển thành công đơn hàng đến tay người dùng |
|  | Bạn sẽ nhận được {{ship\_fee}}$ phí ship cho đơn hàng này. |
|  | Cảm ơn bạn đã tham gia vào hệ thống của chúng tôi''' |
|  | html\_content1 = html\_content.replace("{{ship\_fee}}", str(ship\_fee)) |
|  | msg = Message( |
|  | 'NHẬN ĐƠN', |
|  | to=found\_shipper['email'], |
|  | html=html\_content1 |
|  | ) |
|  | gmail.send(msg) |
|  | return redirect('/order\_status') |
|  | else: |
|  | return 'Order is not found' |
|  |  |
|  | # Hiển thị lịch sử mua hàng |
|  | @app.route('/order\_history') |
|  | def order\_history(): |
|  | # thực hiện tìm kiếm những đơn hàng mà user đã xác nhận giao hàng thành công |
|  | all\_order = Order.find( |
|  | {'user\_id': session['user\_id'], 'is\_ordered': False, 'status': "User đã nhận hàng"}) |
|  | if all\_order is not None: |
|  | # nếu tồn tại thì hiển thị các đơn hàng |
|  | return render\_template('user/order-history.html', all\_order=all\_order, template=1) |
|  | else: |
|  | # thông báo cho người dùng biết không có đơn hàng nào tồn tại |
|  | return render\_template('user/order-history.html', template=0) |
|  |  |
|  | # Logout shipper login user |
|  | @app.route('/logoutShipper\_loginUser') |
|  | def logoutShipper\_loginUser(): |
|  | # hàm thực hiện đăng xuất phiên đăng nhập |
|  | if session['loggedin'] == False: |
|  | return redirect('/login') |
|  | else: |
|  | session['loggedin'] = False |
|  | if session['level\_loggedin'] == "1": # thực hiện kiểm tra lần lượt các session level |
|  | del session['admin\_id'] # thực hiện xóa session để đăng xuất |
|  | del session['level\_loggedin']# thực hiện xóa session để đăng xuất |
|  | elif session['level\_loggedin'] == "2": |
|  | del session['user\_id'] |
|  | del session['level\_loggedin'] |
|  | elif session['level\_loggedin'] == "3": |
|  | del session['shipper\_id'] |
|  | del session['level\_loggedin'] |
|  | return redirect('/login') |
|  |  |
|  | # Logout admin login user |
|  | @app.route('/logoutAdmin\_loginUser') |
|  | def logoutAdmin\_loginUser(): |
|  | if session['loggedin'] == False: |
|  | return redirect('/login') |
|  | else: |
|  | session['loggedin'] = False |
|  | if session['level\_loggedin'] == "1": |
|  | del session['admin\_id'] |
|  | del session['level\_loggedin'] |
|  | elif session['level\_loggedin'] == "2": |
|  | del session['user\_id'] |
|  | del session['level\_loggedin'] |
|  | elif session['level\_loggedin'] == "3": |
|  | del session['shipper\_id'] |
|  | del session['level\_loggedin'] |
|  | return redirect('/login') |
|  |  |
|  | # Todo SHIPPER |
|  | # Chức năng của shipper |
|  |  |
|  | # Đăng ký |
|  | @app.route('/sign\_up\_shipper', methods=['GET', 'POST']) |
|  | def signUpShipper(): |
|  | if request.method == 'GET': |
|  | return render\_template('signup/signup-shipper.html') |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  | fullname = form['fullname'] |
|  | username = form['username'] |
|  | password = form['password'] |
|  | # level = form['level'] |
|  | email = form['email'] |
|  | phone = form['phone'] |
|  | address = form['address'] |
|  | id\_person = form['id\_person'] |
|  | # balance = form['balance'] |
|  |  |
|  | login = Login.find({"username": username}) |
|  | c = int(login.count()) |
|  | if c > 0: |
|  | return render\_template('signup/signup-fail.html', template=0) |
|  | else: |
|  | new\_login = { |
|  | "fullname": fullname, |
|  | "username": username, |
|  | "password": password, |
|  | "level": 3, |
|  | "email": email, |
|  | "phone": phone, |
|  | "id\_person": id\_person, |
|  | "address": address, |
|  | "balance": 0, |
|  | } |
|  | Login.insert\_one(new\_login) |
|  |  |
|  | found\_login = Login.find\_one({'username': username}) |
|  | gmail = GMail('ngovankhanh108@gmail.com', 'KhanhNgo108') |
|  | html\_content = ''' |
|  | Chúc mừng bạn đã tạo tài khoản thành công trên hệ thống. |
|  | Cảm ơn quý khách đã sử dụng dịch vụ của chúng tôi! |
|  | ''' |
|  | msg = Message('Tạo tài khoản thành công', |
|  | to=found\_login['email'], html=html\_content) |
|  | gmail.send(msg) |
|  | return redirect('/login') |
|  |  |
|  | # Thông tin cá nhân |
|  | @app.route('/shipper\_information') |
|  | def shipperInformation(): |
|  | login = Login.find\_one({"\_id": ObjectId(session['shipper\_id'])}) |
|  | if login is not None: |
|  | return render\_template('shipper/information.html', found\_shipper=login) |
|  | else: |
|  | return 'Login is not found' |
|  |  |
|  | # Cập nhật thông tin cá nhân cho Shipper |
|  | @app.route('/shipper\_update/<id>', methods=['GET', 'POST']) |
|  | def updateShipper(id): |
|  | shipper = Login.find\_one({"\_id": ObjectId(session['shipper\_id'])}) |
|  | if shipper is not None: |
|  | if request.method == 'GET': |
|  | return render\_template('shipper/shipper-update.html', update\_shipper=shipper) |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  |  |
|  | password = form['password'] |
|  |  |
|  | update = {"$set": { |
|  | "password": password, |
|  | }} |
|  | Login.update\_one(shipper, update) |
|  | else: |
|  | return 'Shipper is not found' |
|  | return redirect('/shipper\_information') |
|  |  |
|  | # Các đơn hàng hiện có cho shipper |
|  | @app.route('/ship\_request') |
|  | def requestShipper(): |
|  | # thực hiện tìm kiếm các đơn order user đã gửi và chưa có người nhận |
|  | all\_order = Order.find( |
|  | {'is\_ordered': True, 'status': "Shipper chưa nhận đơn"}) |
|  | c = int(all\_order.count()) |
|  | if c > 0: |
|  | # trả về trang html các yêu cầu đơn order cho shipper |
|  | return render\_template('shipper/ship-request.html', template=1, all\_order=all\_order) |
|  | else: |
|  | return render\_template('shipper/ship-request.html', template=0) |
|  |  |
|  | # Hai nút tìm kiếm tình trạng các đơn Shipper đã nhận |
|  | @app.route('/ship\_status\_select') |
|  | def ship\_status\_select(): |
|  | return render\_template('shipper/ship-status-select.html') |
|  |  |
|  | # Chi tiết đơn hàng |
|  | @app.route('/detail\_request/<order\_id>') |
|  | def requestDetail(order\_id): # hàm nhận id của đơn order để thực hiện |
|  | # thực hiện tìm kiếm đơn order |
|  | found\_order = Order.find\_one({"\_id": ObjectId(order\_id)}) |
|  | if found\_order is not None: |
|  | # tìm kiếm thông tin user từ id\_user lưu trong đơn order |
|  | user = Login.find\_one({"\_id": ObjectId(found\_order['user\_id'])}) |
|  | # thực hiện lấy mảng chứa thông tin các sản phẩm |
|  | order\_product = found\_order['product\_id'] |
|  | return render\_template('shipper/detail-request.html', found\_order=found\_order, user=user, order\_product=order\_product) |
|  | else: |
|  | return 'Order is not found' |
|  |  |
|  | # Lịch sử ship đơn hàng |
|  | @app.route('/shipped\_history') |
|  | def historyShipper(): |
|  | # những đơn shipper đã giao thành công và user xác nhận đã nhận được hàng |
|  | # thực hiện tìm kiếm tất cả các đơn đã giao của shipper hiện tại |
|  | all\_order = Order.find({'shipper\_id': session['shipper\_id'], 'status': "User đã nhận hàng"}) |
|  | # đếm số lượng đơn order |
|  | c = int(all\_order.count()) |
|  | # nếu tồn tại thì thực hiện truyền dữ liệu sang html và hiển thị |
|  | if c > 0: |
|  | return render\_template('shipper/shipped-history.html', all\_order=all\_order, template=1) |
|  | else: |
|  | return render\_template('shipper/shipped-history.html', template=0) |
|  |  |
|  | # Shipper nhận đơn hàng |
|  | @app.route('/shipper\_accepted\_order/<order\_id>') |
|  | def shipper\_accepted\_order(order\_id): # nhận đơn hàng khi user gửi yêu cầu với id đơn order |
|  | # thực hiện tìm kiếm đơn order |
|  | found\_order = Order.find\_one({"\_id": ObjectId(order\_id)}) |
|  | if found\_order is not None: |
|  | # lấy ra id user |
|  | id\_user = found\_order['user\_id'] |
|  | # thực hiện tìm kiếm user |
|  | found\_user = Login.find\_one({"\_id": ObjectId(id\_user)}) |
|  | # Thêm ID của Shipper vào đơn và đổi trạng thái đơn |
|  | update = {"$set": {"shipper\_id": session['shipper\_id'], |
|  | "status": "Shipper đã nhận đơn, đang tiến hành lấy hàng"}} |
|  | Order.update\_one(found\_order, update) |
|  | # Gửi Mail cho User |
|  | gmail = GMail('ngovankhanh108@gmail.com', 'KhanhNgo108') |
|  | html\_content = '''Đơn hàng của bạn đã được Shipper chấp nhận giao hàng, vui lòng kiểm tra tình trạng đơn. |
|  | Cảm ơn bạn đã sử dụng dịch vụ của chúng tôi''' |
|  | msg = Message( |
|  | 'NHẬN ĐƠN', |
|  | to=found\_user['email'], |
|  | html=html\_content |
|  | ) |
|  | gmail.send(msg) |
|  | return redirect('/ship\_request') |
|  | else: |
|  | return 'Order is not found' |
|  |  |
|  | # Xem thông tin đơn hiện tại Shipper chấp nhận vận chuyển |
|  | @app.route('/ship\_status/<status>') |
|  | def ship\_status(status): |
|  | # những đơn hàng má shipper nhận sẽ có 2 trạng thái |
|  | # trạng thái 1: Shipper đã nhận đơn, đang tiến hành lấy hàng |
|  | # trạng thái 2: Shipper đã nhận hàng, bắt đầu tiến hành vận chuyển |
|  | # thực hiện tìm kiếm status với trạng thái trên |
|  | found\_order = Order.find({'shipper\_id': session['shipper\_id'], 'status': status}) |
|  | c = int(found\_order.count()) |
|  | if c > 0: |
|  | return render\_template('shipper/ship-status.html', found\_order=found\_order, template=1) |
|  | else: |
|  | return render\_template('shipper/ship-status.html', template=0) |
|  |  |
|  | # Shipper xác nhận đã lấy hàng |
|  | @app.route('/product\_request/<order\_id>') |
|  | def product\_request(order\_id): |
|  | # trạng thái thứ 2 khi shipper đã xác nhận lấy hàng và bắt đầu vận chuyển |
|  | # tìm đơn order hiện tại |
|  | found\_order = Order.find\_one({"\_id": ObjectId(order\_id)}) |
|  | if found\_order is not None: |
|  | # lấy thông tin id user |
|  | id\_user = found\_order['user\_id'] |
|  | # tìm thông tin user |
|  | found\_user = Login.find\_one({"\_id": ObjectId(id\_user)}) |
|  | # thực hiện update trạng thái giao hàng trên database |
|  | update = { |
|  | "$set": {"status": "Shipper đã nhận hàng, bắt đầu tiến hành vận chuyển"}} |
|  | Order.update\_one(found\_order, update) |
|  | # gửi email thông báo |
|  | gmail = GMail('ngovankhanh108@gmail.com', 'KhanhNgo108') |
|  | msg = Message( |
|  | 'LẤY HÀNG', to=found\_user['email'], html="Shipper đã lấy được hàng, bắt đầu vận chuyển đến bạn") |
|  | gmail.send(msg) |
|  | return redirect('/shipped\_history') |
|  | else: |
|  | return 'Order is not found' |
|  |  |
|  |  |
|  | # TODO Admin |
|  | # Chức năng của admin |
|  | # Information Admin |
|  | @app.route('/admin\_information') |
|  | def adminInformation(): |
|  | # thông tin cá nhận |
|  | login = Login.find\_one({"\_id": ObjectId(session["admin\_id"])}) |
|  | if login is not None: |
|  | return render\_template('admin/information.html', found\_admin=login) |
|  | else: |
|  | return 'Login is not found' |
|  |  |
|  | # Update thông tin admin |
|  | @app.route('/admin\_update/<id>', methods=['GET', 'POST']) |
|  | def updateInfoAdmin(id): |
|  | login = Login.find\_one({"\_id": ObjectId(id)}) |
|  | if request.method == 'GET': |
|  | return render\_template('admin/admin-update.html', update\_admin=login) |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  |  |
|  | password = form['password'] |
|  |  |
|  | update\_login = {"$set": { |
|  | "password": password, |
|  | }} |
|  | Login.update\_one(login, update\_login) |
|  | return redirect('/homeLogin') |
|  |  |
|  | # Hiển thị sản phẩm |
|  | @app.route('/all\_product\_admin') |
|  | def allProduct(): |
|  | # tìm kiếm tất cả các sản phẩm có trên database |
|  | all\_product = Product.find() |
|  | return render\_template('admin/all\_product.html', all\_product=all\_product) |
|  |  |
|  | # Xóa sản phẩm |
|  | @app.route('/product/delete/<id>') |
|  | def deleteProduct(id): |
|  | # thực hiện xóa 1 sản phẩm khi 1 id sản phẩm được truyền vào hàm |
|  | product = Product.find\_one({"\_id": ObjectId(id)}) |
|  | Product.delete\_one(product) |
|  | return redirect('/all\_product\_admin') |
|  |  |
|  | # Sửa sản phẩm |
|  | @app.route('/all\_product/update\_product/<id>', methods=['GET', 'POST']) |
|  | def updateProduct(id): |
|  | product = Product.find\_one({"\_id": ObjectId(id)}) |
|  | if request.method == 'GET': |
|  | return render\_template('admin/update\_product.html', product=product) |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  |  |
|  | image = form['image'] |
|  | name = form['name'] |
|  | description = form['description'] |
|  | status = form['status'] |
|  | price = form['price'] |
|  | brand = form['brand'] |
|  | product\_type = form['product\_type'] |
|  | product\_gender = form['product\_gender'] |
|  | product\_kids = form['product\_kids'] |
|  |  |
|  | update\_product = {"$set": { |
|  | "image": image, |
|  | "name": name, |
|  | "description": description, |
|  | "price": price, |
|  | "status": status, |
|  | "brand": brand, |
|  | "product\_type": product\_type, |
|  | "product\_gender": product\_gender, |
|  | "product\_kids": product\_kids, |
|  | }} |
|  | Product.update\_one(product, update\_product) |
|  | return redirect('/all\_product\_admin') |
|  |  |
|  | # Thêm sản phẩm |
|  | @app.route('/add\_product', methods=['GET', 'POST']) |
|  | def addProduct(): |
|  | if request.method == 'GET': |
|  | return render\_template('admin/add\_product.html') |
|  | elif request.method == 'POST': |
|  | form = request.form |
|  |  |
|  | image = form['image'] |
|  | name = form['name'] |
|  | description = form['description'] |
|  | status = form['status'] |
|  | price = form['price'] |
|  | brand = form['brand'] |
|  | product\_type = form['product\_type'] |
|  | product\_gender = form['product\_gender'] |
|  | product\_kids = form['product\_kids'] |
|  |  |
|  | new\_product = { |
|  | "image": image, |
|  | "name": name, |
|  | "description": description, |
|  | "status": status, |
|  | "price": price, |
|  | "brand": brand, |
|  | "product\_type": product\_type, |
|  | "product\_gender": product\_gender, |
|  | "product\_kids": product\_kids, |
|  | "view":1, |
|  | "sold\_count":1, |
|  | } |
|  |  |
|  | Product.insert\_one(new\_product) |
|  | return redirect('/all\_product\_admin') |
|  |  |
|  | # Quản lý tài khoản |
|  | @app.route('/all\_login') |
|  | def allLogin(): |
|  | # tìm kiếm tất cả các tài khoản hiện có |
|  | all\_login = Login.find() |
|  | return render\_template('admin/all\_login.html', all\_login=all\_login) |
|  |  |
|  | # Xóa tài khoản |
|  | @app.route('/all\_login/delete/<id>') |
|  | def deleteLogin(id): |
|  | # thực hiện xóa tải khoản khi admin thực hiện |
|  | login = Login.find\_one({"\_id": ObjectId(id)}) |
|  | Login.delete\_one(login) |
|  | return redirect('/all\_login') |
|  |  |
|  | # Sửa tài khoản |
|  | @app.route('/edit\_logins\_admin/<id>', methods=['GET', 'POST']) |
|  | def editLoginsAdmin(id): |
|  | # sửa các tài khoản hiện có trên database( quyền admin) |
|  | login = Login.find\_one({"\_id":ObjectId(id)}) |
|  | if request.method == "GET": |
|  | return render\_template('admin/update\_account.html',login = login) |
|  | elif request.method =="POST": |
|  | form = request.form |
|  |  |
|  | update = {"$set":{ |
|  | 'fullname' : form['fullname'], |
|  | 'username' : form['username'], |
|  | 'password' : form['password'], |
|  | 'level ': form['level'], |
|  | 'email' : form['email'], |
|  | 'phone' : form['phone'], |
|  | 'address' : form['address'], |
|  | 'balance' : form['balance'], |
|  | }} |
|  |  |
|  | Login.update\_one(login,update) |
|  | return redirect('/all\_login') |
|  |  |
|  | # Xuất file Excel Account |
|  | @app.route('/output\_excel') |
|  | def output\_excel(): |
|  | # lấy tất cả tài khoản |
|  | all\_login = Login.find() |
|  | # tạo mảng để lưu các trường |
|  | data = [] |
|  | i = 0 |
|  | for login in all\_login: |
|  | list\_login = {} |
|  | # lấy giá trị lưu vào biến |
|  | index = i |
|  | fullname = login['fullname'] |
|  | username = login['username'] |
|  | password = login['password'] |
|  | level = login['level'] |
|  | email = login['email'] |
|  | phone = login['phone'] |
|  | balance = login['balance'] |
|  | # Key |
|  | list\_login['STT'] = index |
|  | list\_login['Fullname'] = fullname |
|  | list\_login['Username'] = username |
|  | list\_login['Password'] = password |
|  | list\_login['Level'] = level |
|  | list\_login['Email'] = email |
|  | list\_login['Phone'] = phone |
|  | list\_login['Balance'] = balance |
|  | i = i + 1 |
|  | # |
|  | # thêm 1 list mới vào mảng |
|  | data.append(list\_login) |
|  | # thực hiện xuất file excel |
|  | pyexcel.save\_as(records=data, dest\_file\_name="Logins.xlsx") |
|  | return "Done" |
|  |  |
|  | # Xuất file Excel products |
|  | @app.route('/output\_product\_excel') |
|  | def output\_product\_excel(): |
|  | all\_product = Product.find() |
|  | data = [] |
|  | i = 0 |
|  | for product in all\_product: |
|  | list\_product = {} |
|  | # |
|  | index = i |
|  | \_id = product['\_id'] |
|  | image = product['image'] |
|  | name = product['name'] |
|  | description = product['description'] |
|  | price = product['price'] |
|  | status = product['status'] |
|  | brand = product['brand'] |
|  | product\_type = product['product\_type'] |
|  | product\_gender = product['product\_gender'] |
|  | # |
|  | list\_product['STT'] = index |
|  | list\_product['ID'] = \_id |
|  | list\_product['Name'] = name |
|  | list\_product['Description'] = description |
|  | list\_product['Price'] = price |
|  | list\_product['Status'] = status |
|  | list\_product['Brand'] = brand |
|  | list\_product['Type'] = product\_type |
|  | list\_product['Gender'] = product\_gender |
|  | i = i + 1 |
|  | # |
|  | data.append(list\_product) |
|  | pyexcel.save\_as(records=data, dest\_file\_name="Products.xlsx") |
|  | return "Done" |
|  |  |
|  | # Xuất file Order |
|  | @app.route('/output\_orders\_excel') |
|  | def output\_order\_excel(): |
|  | all\_order = Order.find() |
|  | data = [] |
|  | i = 0 |
|  | for order in all\_order: |
|  | list\_order = {} |
|  | # |
|  | index = i |
|  | \_id = order['\_id'] |
|  | user\_id = order['user\_id'] |
|  | address = order['address'] |
|  | order\_time = order['order\_time'] |
|  | order\_fee = order['order\_fee'] |
|  | ship\_fee = order['ship\_fee'] |
|  | is\_ordered = order['is\_ordered'] |
|  | status = order['status'] |
|  | shipper\_id = order['shipper\_id'] |
|  | # |
|  | list\_order['STT'] = index |
|  | list\_order['ID'] = \_id |
|  | list\_order['User ID'] = user\_id |
|  | list\_order['Address'] = address |
|  | list\_order['Order Time'] = order\_time |
|  | list\_order['Order Fee'] = order\_fee |
|  | list\_order['Ship Fee'] = ship\_fee |
|  | list\_order['Is Ordered'] = is\_ordered |
|  | list\_order['Status'] = status |
|  | list\_order['Shipper ID'] = shipper\_id |
|  | i = i + 1 |
|  |  |
|  | # |
|  | data.append(list\_order) |
|  | pyexcel.save\_as(records=data, dest\_file\_name="Orders.xlsx") |
|  |  |
|  | return "Done" |
|  |  |
|  | # Xuất hóa đơn ra Word |
|  | @app.route('/output\_bill\_word/<id>') |
|  | def output\_bill\_word(id): |
|  | # tìm đơn order muốn xuất |
|  | order = Order.find\_one({"\_id":ObjectId(id)}) |
|  | # lấy thông tin các sản phẩm |
|  | all\_product =order['product\_id'] |
|  | # tìm thông tin user |
|  | user = Login.find\_one({"\_id":ObjectId(order['user\_id'])}) |
|  | # khởi tạo đối tượng |
|  | document = Document() |
|  | # add tiêu đều |
|  | document.add\_heading('Hóa đơn bán hàng', 0) |
|  | # lấy ngày giờ order |
|  | date = order['order\_time'] |
|  | day = str(date.day) |
|  | month = str(date.month) |
|  | year = str(date.year) |
|  | # thêm ngày giờ vào hóa đơn |
|  | document.add\_heading('Ngày ' + day+"/"+month+"/"+year, 0) |
|  | # lấy tên khách hàng từ database |
|  | document.add\_heading("Tên khách hàng: " + user['fullname'], level=1) |
|  | # duyết mảng sản phẩm để lấy thông tin sản phẩm |
|  | index = 1 |
|  | for i in all\_product: |
|  | name = i['name'] |
|  | price = str(i['price']) |
|  | s = "Sản phẩm " + str(index) + " : " + name + " " + price+"$" |
|  | document.add\_paragraph(s, style='Intense Quote') |
|  | index += 1 |
|  | # lấy thông tin địa chỉ từ đơn order |
|  | document.add\_paragraph("Địa chỉ: " + order['address'], style='Intense Quote') |
|  | # lấy phí ship từ đơn hàng |
|  | document.add\_paragraph( |
|  | "Phí ship: " + str(order['ship\_fee']), style='Intense Quote') |
|  | # tổng tiền hàng |
|  | document.add\_paragraph("Tổng tiền hàng : " + |
|  | str(order['order\_fee']), style='Intense Quote') |
|  | # tổng tiền của dơn order phí ship + tiền hàng |
|  | totalPrice = int(order['ship\_fee']) + int(order['order\_fee']) |
|  | document.add\_paragraph("Tổng thanh toán: " + |
|  | str(totalPrice), style='Intense Quote') |
|  |  |
|  | document.add\_page\_break() |
|  | # xuất file word |
|  | document.save('HoaDonWord.docx') |
|  | return "done" |
|  |  |
|  | # TODO |
|  | @app.route('/all\_order') |
|  | def allOrder(): |
|  | all\_order = Order.find() |
|  | return render\_template('admin/all\_order.html',all\_order = all\_order) |
|  |  |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | app.run(debug=True) |

1. Công nghệ
   1. Database: NoSql, MongoDB
   2. Query: Pymongo <https://pypi.org/project/pymongo/>
   3. Front-End: HTML, CSS, Bootstrap, Jinja2, Font Awesome
   4. Back-End: Python 3.7: <https://www.anaconda.com/distribution/>
   5. Flask: <https://pypi.org/project/Flask/>
   6. Gmail: <https://pypi.org/project/gmail/>
2. Hướng dẫn chạy app

* Cài đặt VSCode: <https://code.visualstudio.com/>
* Python 3.7: <https://www.anaconda.com/distribution/>

Chú ý khi cài đặt Anaconda( python 3)



* Các thư viện cần cài đặt: nhấn tổ hợp phím “Ctrl” + ”~” để mở terminal trong VS Code:

+ Chạy câu lệnh: pip install pymongo

+ pip install flask

+ pip install pyexcel

+ pip install xlsx

+ pip install python-docx

+ pip install gmail

+ pip install dnspython

* Các extension trong Vscode:

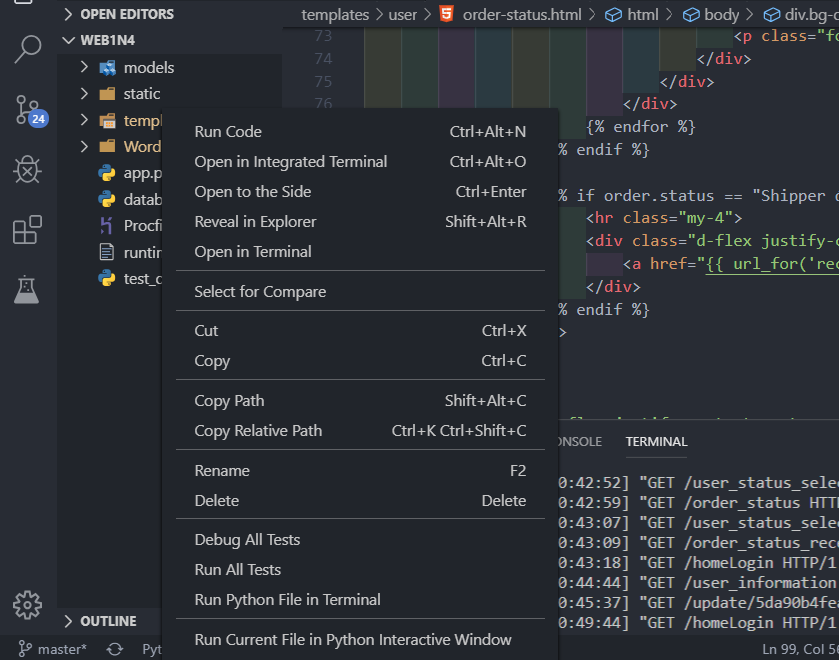
+ Flask-snippets: <https://marketplace.visualstudio.com/items?itemName=cstrap.flask-snippets>

+ Python: <https://marketplace.visualstudio.com/items?itemName=ms-python.python>

+ Terminal: <https://marketplace.visualstudio.com/items?itemName=formulahendry.terminal>

+ Sau khi cài đặt các thư viện cần thiết:

+ Chọn File trên thanh công cụ chọn Open Folder : chọn thư mục chứa code

+ Chọn chuột phải vào file app.py : chọn Open in integrated terminal

Gõ: python app.py để chạy chương trình

1. Đăng web lên host( Heroku)
   * + - Chưa thành công.....(Đang tìm hiểu)
2. Tài liệu tham khảo
   1. Python MongoDB:

<https://www.w3schools.com/python/python_mongodb_create_db.asp>

<https://viblo.asia/p/python-and-mongodb-oOVlYBna58W>

<https://viblo.asia/p/mongodb-co-ban-phan-1-l5XRBVN3RqPe>

<https://api.mongodb.com/python/current/tutorial.html>

<https://docs.mongodb.com/manual/reference/operator/update-array/>

<https://realpython.com/introduction-to-mongodb-and-python/>

* 1. Github

<https://viblo.asia/p/githubio-hien-thuc-hoa-uoc-mo-thoi-tho-au-0bDM6wkXG2X4>

* 1. Flask

<https://csc.edu.vn/lap-trinh-va-csdl/tin-tuc/kien-thuc-lap-trinh/7-ly-do-ban-nen-chon-Flask-Framework-4130>