

Name: Yihua Yang

Date: 4/26/2022

CMPSC 431W Project Phase 2 README

Introduction

In Project Phase 2 submission, there are some HTML files and Python files. Descriptions regarding these files and the functions in the files will be provided in this README file. For a brief description of the functions, please see the comments in the app.py code. **All functions which should be considered as extra-credits functions are highlighted and listed at the end.**

The names of these HTML files and Python files (although one of it technically is python notebook file) are listed below in alphabetical order:

HTML files (14 files in total):

- categoryHierarchy.html
- checkingInfo.html
- Delete_patient.html
- fail.html
- index.html
- input.html
- login.html
- Main_webpage.html
- placeOrder.html
- publishProductListing.html
- searching.html
- shoppingCart.html
- success.html
- viewOrders.html

Python files (2 files in total):

- app.py

- `Create_Insert.ipynb`

Although there are some other files in the submission, those files that not mentioned here are not important to Project Phase 2, so the explanations for them will be ignored. Now, let us first discuss the `Create_Insert.ipynb`, which should be the first function to execute in chronological order.

Create_Insert.ipynb:

In general, this .ipynb file creates all the tables in the database for the Project Phase 2. In the 1st cell of this python notebook file, all needed modules for this file are imported. In the 2nd cell, it reads data from `User.csv` and stores the data in lists for later use. The code in these two cells is short and should be easy to read.

In the 3rd cell, it connects to the database and uses a cursor to execute the commands for inserting tables. All necessary tables for `app.py` are created here mainly based on the given schema, although there are some small modifications, such as the extra columns (`Active_period` and `Status`) were added in the `Product_Listings` table. In this cell, an extra table called `Shopping_cart` is created for the function about shopping cart.

In the 4th cell, it iterates the lists created in the 2nd cell for hashing the passwords from `User.csv` file and then insert the hashed password to database with corresponding user email of the password.

In the 5th cell, it reads data from `Categories.csv` and inserts the data to the `Categories` table created in the 3rd cell. Here, it basically does the same thing as the 4th cell but without hash anything.

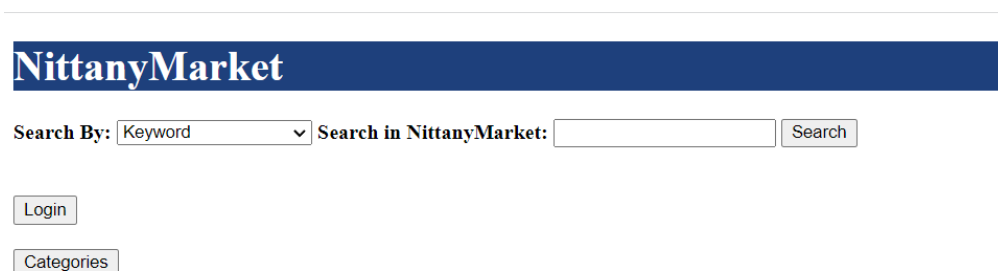
The description for each code cell in `Create_Insert` are explained above. In the next part, details of the most important file, `app.py`, will be elaborated.

app.py:

This file, app.py, was adapted from the given template code from Project Phase 1 and provides most of critical functions for Phase 2. At the beginning of the file, all needed modules are imported. The session secret key and host name are also defined here. Let us discuss each function in the file with corresponding HTML pages specifically.

- index():

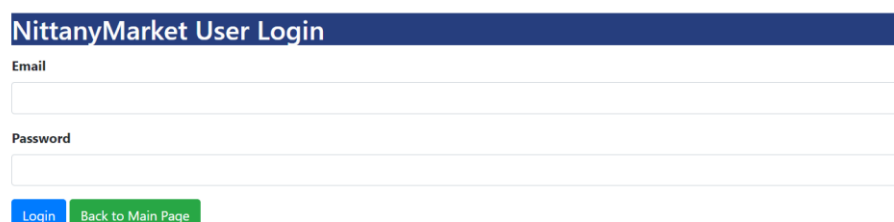
When this program starts, it will return the Main_webpage.html to users. The user can choose to login, search items based on keyword, and view all categories in the main page. The appearance of the main page will look like the below image:



The screenshot shows the NittanyMarket homepage. At the top is a dark blue header with the text "NittanyMarket" in white. Below the header is a search bar with the text "Search By: Keyword" and a dropdown arrow, followed by "Search in NittanyMarket:" and a text input field, and a "Search" button. Below the search bar are two buttons: "Login" and "Categories".

- login():

When a user clicks the “login”, the user will be redirected to the login.html page, which was modified based on the given HTML page in Phase1 and will look like the below image:



The screenshot shows the NittanyMarket User Login page. At the top is a dark blue header with the text "NittanyMarket User Login" in white. Below the header are two input fields: "Email" and "Password". Below the input fields are two buttons: "Login" (blue) and "Back to Main Page" (green).

The user will be asked to enter the email and the corresponding password here. After the user has entered the email and password, he/she can click the “Login” button. Then, the login.html will send a request form with method “POST” to login(). Next, in login(), it will get the email and password entered by the user from the form and

use `add_name(email, password)` to determine whether the password is correct or not. If the password is correct, the user type, user email, and user information will be stored in session for later use, and the user will be redirected to `checkingInfo.html` page. If the password is incorrect, it will redirect the user to `fail.html` page, which looks like:

Login failed!

Please check if email and password are entered correctly!

Note that it is required to enter some contents in both the textbox for Email and the textbox of Password. Otherwise, it will show a popup box saying “Please fill out this field” like the below image:

Also, the user can choose to click the “Back to Main Page” button to the main page.

- `add_name(email, password):`

In this function, it will first hash the given password, then connects to the database and uses cursor with SELECT SQL statement to decide if the email and password (hashed) are match. It returns 1 if email and password match and returns 0 if they do not match.

- `getInfo(email):`

In this function, it connects to the database and uses cursor with SQL SELECT statement to get the user’s information stored in the table Buyers. Specifically, it returns email ID, name, age, gender, home address, billing address, last 4 digits of credit card(s) of a buyer/seller(buyer) if it can find the result.

- `getUserType(email):`

This function returns the user type (buyer, seller(buyer), local_vendor) based on the given email. In this function, it will connect to the database and decide the type of the user (email) based on the data stored in tables Local_Vendors and Sellers.

- getSellerInfo(email, user_type):

This function returns the information (routing number, account number, balance) for both seller(buyer) and local_vendor. For local vendors, it will additionally return extra information (business name, business address, customer service number).

- checkingInfo():

After the user successfully login, the user will be redirected to the checkingInfo.html page so that the user can check the information he/she has. The look of the checkingInfo.html page will be different for different users.

For buyers, the checkingInfo.html page looks like (Figure 1):

This is checkingInfo page!

Hello, User:

arubertelli0@nsu.edu

Your information is listed below:

Email	First Name	Last Name	Gender	Age	Home Address	Billing Address	last 4 digits of credit card(s)
arubertelli0@nsu.edu	Ileana	Ziehms	Female	49	Fordem Court, Lockwood, NY 14859	Lakewood Pass, Schaumburg, IL 60193	['8663']

New Password:

Figure 1 (buyer's page)

For seller(buyer), the checkingInfo.html page contains the buyer information and seller information, and it looks like (Figure 2):

This is checkingInfo page!

Hello, User:

nrideoutmi@nsu.edu

Your information is listed below:

Email	First Name	Last Name	Gender	Age	Home Address	Billing Address	last 4 digits of credit card(s)
nrideoutmi@nsu.edu	Tedman	Sammonds	Bigender	46	Boyd Road, Tahuya, WA 98588	Boyd Road, Tahuya, WA 98588	[0367]

[View Orders](#)

[View Shopping Cart](#)

Email	Routing Number	Account Number	Balance (\$)
nrideoutmi@nsu.edu	0011-4959-0	31923667	195293

[Publish Product](#)

New Password:

[Change Password](#)

[Back to Login](#)

[Back to Main Page](#)

Figure 2 (seller(buyer)'s page)

For local vendors, this page will show the seller information and local-vendor information (business name, business address, and customer service number). The page will look like (Figure 3):

This is checkingInfo page!

Hello, User:

rdichee@adobe.com

Your information is listed below:

Email	Routing Number	Account Number	Balance (\$)	Business Name	Customer Service Number	Street	City	State	Zip Code
rdichee@adobe.com	0011-4959-0	35390965	762104	Pfeffer LLC	689-458-7547	Fordem Court	Lockwood	NY	14859

[Publish Product](#)

[View Orders](#)

New Password:

[Change Password](#)

[Back to Login](#)

[Back to Main Page](#)

Figure 3 (local_vendor's page)

Notice that there are some differences in the page among the users. For buyers and seller(buyer), they will have buyer information and the “View Shopping Cart” button on the page, as they can buy product because they are buyers. For seller(buyer) and local_vendor, they will see their seller information in this page and can publish

products as they are sellers. The description of “View Shopping Cart” and “View Orders” will be given later. However, all of them can change their password in this page, and click “Back to Login”/” Back to Main Page” to go to the login page and the main page. If a user enters the new password he/she wants to change and clicks the “Change Password” button, there will be a flash message shown in the page to tell the user that his/her password has been changed, as the below image shows.

New Password:

Your password has been changed!

- changePassword(email, newPassword):

This function will first hash the new password entered by user and then connect to the database to update the hashed new password for the given user. At the end, it returns 0.

- categoryHierarchy():

When a user clicks the “Categories” button on the main page, the user will be redirected to the catgoryHierarchy.html page. The basic look of this page will like the below image:

Select a specific category/subcategory to browse:

In categoryHierarchy(), it will first get the categories/subcategories from the database to fill out the selection box. Then the user can click the selection box to select a specific category/subcategory to browse. The content in the selection box will look like:

Select a specific category/subcategory to browse:

All

View

All
Beauty Products
- Makeup
--- Brushes & Applicators
--- Face
--- Lip
Clothing
- Bottoms
--- Jeans
--- Pants
--- Skirts
- Sleepwear
--- Bath Robes
- Tops
--- Bodysuits
--- Long Sleeves
--- T-Shirts
Electrical Supplies
- Cell Phones
--- Iphone

Note that all subcategories begin with “- “ and all subcategories of a subcategory begin with “--- “, which will help user to distinguish category and subcategory. After the user selected a specific category/subcategory and clicks view, the html page will send a request form with method “POST” to categoryHierarchy(), and it will show the products and the details of the products in the given category/subcategory/subcategory of subcategory by searching in the database and will tell what the category/subcategory the user searched. An example of that will like the image listed below:

Select a specific category/subcategory to browse:

All

View

Back to Main Page

Now showing category/subcategory:

Clothing

Title	Product Name	Product Description	Price	Quantity	Sold By		
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	1	asimonsen7z@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	1	asimonsen7z@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	bbartkowiak6r@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	bbartkowiak6r@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	1	bdeavenell7o@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	bmalbonee@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	bmalbonee@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	bspottiswoodhm@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	bspottiswoodhm@nsu.edu	Buy	Add to Cart
skinny	Skinny High Jeans	Men's Jeans	\$30	4	cdullardbp@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	4	cpaulone4@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	4	cpaulone4@nsu.edu	Buy	Add to Cart
Oversized T-shirt	Skeleton Hand Print Oversized tee	Attractive tee	\$110	1	dgurge15@jimdo.com	Buy	Add to Cart
Oversized T-shirt	Skeleton Hand Print Oversized tee	Attractive tee	\$110	1	dgurge15@jimdo.com	Buy	Add to Cart
skinny	Skinny High Jeans	Men's Jeans	\$30	1	dnaughton9f@nsu.edu	Buy	Add to Cart
leather	Ankle-length Leather Pants	Leather pants	\$30	4	dparietkw@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	3	ebullardig@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	3	ebullardig@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	2	feeles4r@nsu.edu	Buy	Add to Cart
leather	Ankle-length Leather Pants	Leather pants	\$30	5	fistodarddu@nsu.edu	Buy	Add to Cart
leather	Ankle-length Leather Pants	Leather pants	\$30	4	gcordall73@nsu.edu	Buy	Add to Cart
leather	Ankle-length Leather Pants	Leather pants	\$30	4	gcordall73@nsu.edu	Buy	Add to Cart
Ribbed Turtleneck Bodysuit	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	Turtleneck tee	\$120	3	gescalanteg4@nsu.edu	Buy	Add to Cart
Waffled Bathrobe	Mainstays Waffled Bathrobe	Brobe	\$40	2	gescalanteg4@nsu.edu	Buy	Add to Cart
Ribbed Turtleneck Bodysuit	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	Turtleneck tee	\$120	3	gescalanteg4@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	1	ghalvorsen95@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	1	gmoscone9r@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	gpendry79@nsu.edu	Buy	Add to Cart
short skirts	Selby Mini Skirt	short skirts - Classic	\$90	5	gpendry79@nsu.edu	Buy	Add to Cart

Notice that the products with same product names and details from same sellers are not merged. The user can click “Buy” if he/she wants to buy the product now or “Add to Cart” if he/she may buy the product later.

- getCategories():

This function will get all categories/subcategories/subcategories of a subcategory from the database. BFS technique applied here. It will first traverse all subcategories under a specific category and stores all the subcategories in a list. Then it will find all subcategories under the subcategories in the list. At the end, the list will like: [category1, subcategory1-1, sub_subcategory1-1-1, sub_subcategory2-1-1, ... subcategory2-1, ..., category2, subcategory1-2, ...]. Then it will just search all products under the categories/subcategories in the list and return the result.

- show_product(category):

In general, it will return a list showing the names and details of products which belong to the given category/subcategory. It will first trim the “- “ and “--- “ in front of the given category if “- “ and “--- “ exist. Similar to getCategories(), BFS technique also applied here. That is, first find all subcategories under the given category/subcategory and then continue to find the subcategories in the next level if needed. Then all subcategories will be added to a list, and the list will be traversed to find the corresponding products. At the end, all products in the subcategories/subcategories of the subcategories of the given category input will be returned in a result list.

- publishProductListing():

When a seller (including both seller(buyer) and local_vendor) clicks the “Publish Product” button on his/her checkingInfo page, the seller will be redirected to publishProductListing.html page. Basically, this page will like the image shown below:

This is PublishProductListing page!

Hello, Seller: arideoutm@asu.edu

Please provide below information to publish a new product listing:

Product Title:

Product Name:

Product Description:

Price:

Quantity:

Active Period (in Days):

Product Category/Subcategory/Subsubcategory:

All your product listings are listed below:

Listing ID	Category	Title	Product Name	Product Description	Price	Quantity	Active Period	Status	Operation
270	Bodyuits	Ribbed Turtleneck Bodysuit	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	Turtleneck tee	\$120	2	None	None	Change Status
315	Brushes & Applicators	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	4	None	None	Change Status
708	Toys	Monster Jam El Toro	MJ Mega El Toro	Kid's toys	\$70	2	None	None	Change Status
711	Health Care	Luden's Deliciously Soothing Throat Drops	Luden's Throat Drops	Soothing	\$2	5	None	None	Change Status
1353	Electrical Supplies	Insignia? - 65" Class F30 Series LED 4K UHD Smart Fire TV"	Insignia Fire	Smart TV	\$550	2	None	None	Change Status
1356	Kitchen Faucets & Sinks	Dual control kitchen faucet	DCF	Faucets	\$240	2	None	None	Change Status
1601	Brushes & Applicators	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	4	None	None	Change Status
1605	Bodyuits	Ribbed Turtleneck Bodysuit	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	Turtleneck tee	\$120	2	None	None	Change Status
2461	Toys	Monster Jam El Toro	MJ Mega El Toro	Kid's toys	\$70	2	None	None	Change Status
2465	Health Care	Luden's Deliciously Soothing Throat Drops	Luden's Throat Drops	Soothing	\$2	5	None	None	Change Status
2621	Playstation	111	111	test place order	\$5	0	7	1	Change Status
2650	All	111	111asda	test	\$5	1	7	0	Change Status

To publish a product, the seller needs to fill out the title, name, description, price, quantity, active period (in days), category of the product. After all needed information provided, the seller can click the “Add Product Listing” button to publish the product.

Product Category/Subcategory/Subsubcategory:

All your product listings are listed below:

Listing ID	Category	Title	Product Name	Product Description	Price	Quantity	Active Period	Status	Operation
270	Bodyuits	Ribbed Turtleneck Bodysuit	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	Turtleneck tee	\$120	2	None	None	Change Status
315	Brushes & Applicators	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	4	None	None	Change Status
708	Toys	Monster Jam El Toro	MJ Mega El Toro	Kid's toys	\$70	2	None	None	Change Status
711	Health Care	Luden's Deliciously Soothing Throat Drops	Luden's Throat Drops	Soothing	\$2	5	None	None	Change Status
1353	Electrical Supplies	Insignia? - 65" Class F30 Series LED 4K UHD Smart Fire TV"	Insignia Fire	Smart TV	\$550	2	None	None	Change Status
1356	Kitchen Faucets & Sinks	Dual control kitchen faucet	DCF	Faucets	\$240	2	None	None	Change Status
1601	Brushes & Applicators	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	4	None	None	Change Status
1605	Bodyuits	Ribbed Turtleneck Bodysuit	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	Turtleneck tee	\$120	2	None	None	Change Status
2461	Toys	Monster Jam El Toro	MJ Mega El Toro	Kid's toys	\$70	2	None	None	Change Status
2465	Health Care	Luden's Deliciously Soothing Throat Drops	Luden's Throat Drops	Soothing	\$2	5	None	None	Change Status
2621	Playstation	111	111	test place order	\$5	0	7	1	Change Status
2650	All	111	111asda	test	\$5	1	7	0	Change Status
2655	Face	aaa	s	test	\$5	5	6	1	Change Status

The product listing has been added!

After the button is clicked, there will be a flash message saying that “The product listing has been added” at the end of the page. At the same time, the table showing all product listings published by the seller will be refreshed to show the newest product listing. In this table, seller can change the status of a product listing from active (status=None or status=1) to inactive (status=0) or from inactive to active. If the active period of the listing becomes 0 or the status of the listing is 0, then this listing will not be shown to others, while the seller can still see the inactive product listing in this table. The seller can also go back to checkingInfo page by clicking the “Back” button.

- addListing(listingInfo):

This function will insert a product listing based on the given listingInfo list into the Product_Listings table. The Listing_ID is determined by the largest listing ID plus the quantity of the listings the seller wants to add. It will return 1 at the end.

- getProductListings(seller):

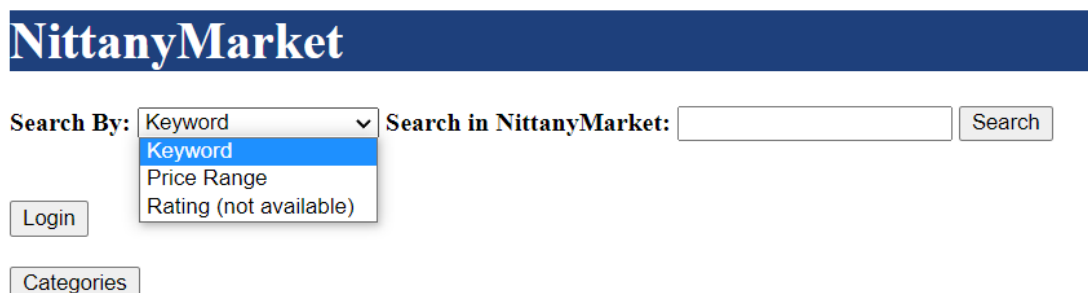
This function will return all product listings published by the given seller. The format of the return result will look like: [(Listing_ID, Category, Title, Product_Name, Product_Description, Price, Quantity, Active_period, Status)].

- changeStatus(listingID, seller):

It changes the status of a listing from 1/NULL (active) to 0 (inactive) or from 0 (inactive) to 1 (active) and returns 0 after the execution.

- searching():

When the user enters some keywords in the textbox on the main page and hits the “Search” button, the searching.html page will send a POST request to this function. In this function, it will collect the searchBy value from the selection box and the content from the textbox and search the contents related to the keyword in the Product_Listing table.



NittanyMarket

Search By: Keyword ▼ Search in NittanyMarket:

As the above image shows, although there are “Price Range” and “Rating” in the “Search By” selection box, only the “keyword” option is available to use; other two conditions are not implemented yet due to the time limits. If the user clicks “Search” button without entering any values in the textbox, it will show “Please fill out this field”

to the user, like the below image shows:

If the user entered some values, all products containing the entered keyword will be shown. For example, if keyword “iphone” entered in the textbox, after clicking the “Search” button, the user will be redirected to the searching.html page, and it will look like:

Search result of **iphone** is listed below:

Back to Main Page

Category	Title	Product Name	Product Description	Price	Quantity	Sold By	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	abithellon@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	abithellon@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	dnaughton9f@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	dnaughton9f@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	4	glibbie7q@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	4	glibbie7q@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	jcarvil9@economist.com	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	jcarvil9@economist.com	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	lcootesj9@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	lcootesj9@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	mamymo@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	mamymo@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	2	mfairhamf0@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	2	mfairhamf0@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	4	mlempertz10@yandex.ru	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	4	mlempertz10@yandex.ru	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	mromayniq@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	mromayniq@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	oflahiveif@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	oflahiveif@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	oflahiveif@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	oflahiveif@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	3	oreppaportr0@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	4	shaydones@nsu.edu	Buy	
Iphone	iPhone 13 Pro Max	New iPhone Model	\$1,100	4	shaydones@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	bstrangeri3@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	bstrangeri3@nsu.edu	Buy	
Electrical Supplies	iPhone 13 Pro Max	New iPhone Model	\$1,100	1	ccaldeiropf@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	2	cceaser3h@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	2	cceaser3h@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	1	cdutnallr1@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	1	cdutnallr1@nsu.edu	Buy	
Electrical Supplies	iPhone 13 Pro Max	New iPhone Model	\$1,100	1	eovenden5y@nsu.edu	Buy	
Cell Phones	iPhone 13 Pro Max	New iPhone Model	\$1,100	5	gdowtrv4i@nsu.edu	Buy	

All product listings whose categories, titles, product names, descriptions, seller email contain the keyword will be shown in this page. The user can click “Back to Main Page”

to go back to the main page or click the “Buy” button if he/she wants to buy the product. If the user clicks “Buy”, the listing ID of the product will be stored in the session. Note that it is different from the categoryHierarchy.html page that there is no “Add to Cart” button here.

- search_in_ProductListings(keyword):

This function returns all product listings whose seller’s email, category, title, name, description containing the keyword.

- placeOrderHTML():

When a buyer clicks “Buy” button from categoryHierarchy.html or searching.html, the buyer will be redirected to placeOrder.html page. Technically, a buyer needs to login before he/she clicks the “Buy” button, otherwise the webpage will crash, because no user information can be obtained by this function. If everything goes well, the look of this page will like:

Hello, Buyer: nrideoutmi@nsu.edu !

Category	Title	Product Name	Product Description	Price	Quantity	Sold By
Iphone	iPhone	13 Pro Max	New iPhone Model	\$1,100	3	abithellon@nsu.edu

Enter the quantity you want to buy: Using credit card: ['0367']

To buy the product, the buyer needs to enter a proper quantity number in the textbox. On one hand, if the buyer enters a number that is larger than the quantity of this listing and clicks “Confirm to Buy” after he/she sees his/her credit card number, a flash message will be shown to the user, just like the below image:

Hello, Buyer: nrideoutmi@nsu.edu !

Category	Title	Product Name	Product Description	Price	Quantity	Sold By
Iphone	iPhone	13 Pro Max	New iPhone Model	\$1,100	3	abithellon@nsu.edu

Enter the quantity you want to buy: Using credit card:

The quantity you entered is greater than the quantity in the listing!

On the other hand, if the buyer enters a valid number and clicks “Confirm to Buy”, the user will be redirected to success.html page, which looks like below:

Your order has been placed sucessfully!

Total Order Price: \$1100

The buyer can either clicks “Back to Main Page” to go back to the main page or clicks “View Your Orders” to see the order he/she bought (for buyer and seller(buyer)). **Note that the balance of the seller who sells this product will be added according to the price of the order.**

Notice that local vendors are not allowed to buy products on NittanyMarket because they lack credit cards information. If a local vendor clicks the “Confirm to Buy”, the page will like:

Hello, Buyer: rdichee@adobe.com !

Category	Title	Product Name	Product Description	Price	Quantity	Sold By
Brushes & Applicators	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	3	abartoszek16@reuters.com

Enter the quantity you want to buy: Using credit card:

You are a local vendor! You cannot buy a product!

In the placeOrder.html page, it will POST the quantity the buyer wants to buy to placeOrderHTML(), and placeOrderHTML() will get the listing ID of the product via

the session value.

- build_order_page(listing_id):

This function will provide needed information to build a placeOrder page based on the given listing_id. Basically, it uses cursor to fetch the result from Product_Listings table based on the listing_id and the status of the product listings.

- placeOrder(buyer, listing_id, quantity):

In this function, it will basically do three things: 1. inserts the new order to table Orders if the given quantity is valid; 2. decreases the quantity of the product in the Product_Listing table; 3. adds balance to the seller who sells this product according to the price of the order. The transaction ID is assigned based on the greatest original transaction ID plus one. This function will return the total price of this order as the result.

- viewOrder():

For all types of users, all of them have a “View Orders” in their checkingInfo.html page. For buyer and seller(buyer), they can view the orders they bought; for seller(buyer) and local_vendor, they can view the orders they sold. The looks of the viewOrder.html page are listed below:

Your Orders						
Bought:						
Transaction ID	Sold by	Product Name	Bought by	Date	Quantity	Total Payment
1260	abartoszek16@reuters.com	Rare Beauty Liquid Touch Brush	arubertelli0@nsu.edu	4/26/22	1	15
172	nrideoutmi@nsu.edu	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	arubertelli0@nsu.edu	7/31/21	3	360
1255	nrideoutmi@nsu.edu	111	arubertelli0@nsu.edu	4/25/22	1	5
1254	nrideoutmi@nsu.edu	111	arubertelli0@nsu.edu	4/25/22	1	5
1257	rdichee@adobe.com	bbbb	arubertelli0@nsu.edu	4/25/22	1	4
1256	rdichee@adobe.com	bbbb	arubertelli0@nsu.edu	4/25/22	4	16

[Back](#)

Figure 4 (buyer's page)

Your Orders

Bought:

Transaction ID	Sold by	Product Name	Bought by	Date	Quantity	Total Payment
1261	abithellon@nsu.edu	13 Pro Max	nrideoutmi@nsu.edu	4/26/22	1	1100
1253	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5
1252	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5
1251	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5
1250	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5

Sold:

Transaction ID	Sold by	Product Name	Bought by	Date	Quantity	Total Payment
172	nrideoutmi@nsu.edu	Wilde Black Ribbed Long Sleeve Mock Neck Bodysuit	arubertelli0@nsu.edu	7/31/21	3	360
89	nrideoutmi@nsu.edu	Rare Beauty Liquid Touch Brush	dvolett@nsu.edu	10/26/21	3	45
1016	nrideoutmi@nsu.edu	MJ Mega El Toro	mreeves46@nsu.edu	11/12/21	5	350
1017	nrideoutmi@nsu.edu	Luden's Throat Drops	rkorneev47@nsu.edu	2/1/22	2	4
1231	nrideoutmi@nsu.edu	Insignia Fire	dclearie54@nsu.edu	2/16/22	2	1100
1232	nrideoutmi@nsu.edu	DCKF	mreeves46@nsu.edu	3/15/21	4	960
1255	nrideoutmi@nsu.edu	111	arubertelli0@nsu.edu	4/25/22	1	5
1254	nrideoutmi@nsu.edu	111	arubertelli0@nsu.edu	4/25/22	1	5
1253	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5
1252	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5
1251	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5
1250	nrideoutmi@nsu.edu	111	nrideoutmi@nsu.edu	4/25/22	1	5

[Back](#)

Figure 5 (seller(buyer)'s page)

Note that the seller(buyer) can buy the product he/she sells.

Your Orders

Sold:

Transaction ID	Sold by	Product Name	Bought by	Date	Quantity	Total Payment
315	rdichee@adobe.com	Double boiler insert_for_all	igetley3@nsu.edu	3/31/21	2	70
1022	rdichee@adobe.com	ER: MMO	fleere4c@nsu.edu	3/4/21	1	150
1237	rdichee@adobe.com	Skinny High Jeans	tevangelines4b@nsu.edu	5/29/21	3	90
1257	rdichee@adobe.com	bbbb	arubertelli0@nsu.edu	4/25/22	1	4
1256	rdichee@adobe.com	bbbb	arubertelli0@nsu.edu	4/25/22	4	16

[Back](#)

Figure 6 (local_vender's page)

This function should be considered as an extra-credit function!

If no sold order or no bough order, the page will like:

Your Orders

Bought:

Transaction ID	Sold by	Product Name	Bought by	Date	Quantity	Total Payment
820	rbuchankl@nsu.edu	13 Pro Max	mgudemman44@nsu.edu	11/22/21	3	3300

Sold:

Transaction ID	Sold by	Product Name	Bought by	Date	Quantity	Total Payment
----------------	---------	--------------	-----------	------	----------	---------------

[Back](#)

- getOrders(user, user_type):

This function is used in the viewOrder(). It will return orders bought/sold by the given user and the given user_type. That is, for buyer, it will return orders(bought); for seller(buyer), it returns orders(bought) and orders(sold); for local vendor, it returns orders(sold).

- shopping_Cart():

In this function, it assumes that the user has login. Of course, the users can only access to the shopping cart via the button on the checkingInfo.html page. After a buyer or a seller(buyer) clicks the “Shopping Cart” button in the checkingInfo.html page, it will show:

Shopping Cart								
Shopping Cart ID	Category	Title	Product Name	Product Description	Price	Quantity	Sold By	Operation
3	Brushes & Applicators	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	3	abartoszek16@reuters.com	Buy Remove

[Back](#)

In this page, the user can either click the “Buy” button, which will redirect the user to the placeOrder.html, just like clicking the “Buy” button in the categoryHierarchy.html page, or click the “Remove” button to remove a specific product in the user’s shopping cart. The Listing_ID of the product in the shopping cart will be sent to placeOrder.html via request form. The product in the shopping cart will be not available for the buyer to see if its quantity becomes 0, status changes to 0, or the active period is 0. If the shopping cart of a user is empty, the page will look like:

Shopping Cart	
No items in your shopping cart!	
Back	

The buyer/seller(buyer) can click “Back” to go back to the checkingInfo.html page here. Note that all data regarding the shopping cart is stored on an extra table called Shopping_Cart.

If the user clicks “Add to Cart” button on the categoryHierarchy.html page, it will redirect the user to the placeOrder.html page with a flash message, just like the below image:

Hello, Buyer: mgudemman44@nsu.edu !

Category	Title	Product Name	Product Description	Price	Quantity	Sold By
Makeup	Liquid Touch Concealer Brush	Rare Beauty Liquid Touch Brush	Makeup Brush	\$15	4	cfridpm@nsu.edu

Enter the quantity you want to buy: Using credit card: ['9645', '8275']

Added to your shopping cart!

- showShoppingCart(buyer):

This function will show the buyer’s shopping cart based on the given buyer email. The shopping cart information gained from this function will look like: [(Shopping_Cart_ID, Category, Title, Product_Name, Product_Description, Price, Quantity, Seller_Email, Listing_ID)]. Notice that the product that quantity is 0 or status is 0 or active period is 0 will not be shown to the buyer.

- remove_from_cart(shopping_cart_ID):

After the user clicks the “Remove” button on the shoppingCart.html page, this function will run to remove the product from the buyer’s shopping cart based on the given listing_id from the request from sent by the “Remove” button.

Conclusion

placeOrder, viewOrder, searching, shoppingCart should be considered as extra-credit functions!

For a brief or a detailed description of these above functions, please check the comments in app.py file and the Create_Insert.ipynb file!

Thank you for reading this long README!