HOMEWORK ASSIGNMENT 2

CSCI 571 – Fall 2023

Abstract

Server-side Scripting using Python, Flask, JSON, AJAX, and the eBay API

Marco Papa

papa@usc.edu

Homework 2: Server-side Scripting using Python Flask, JSON and the eBay API

1. Objectives

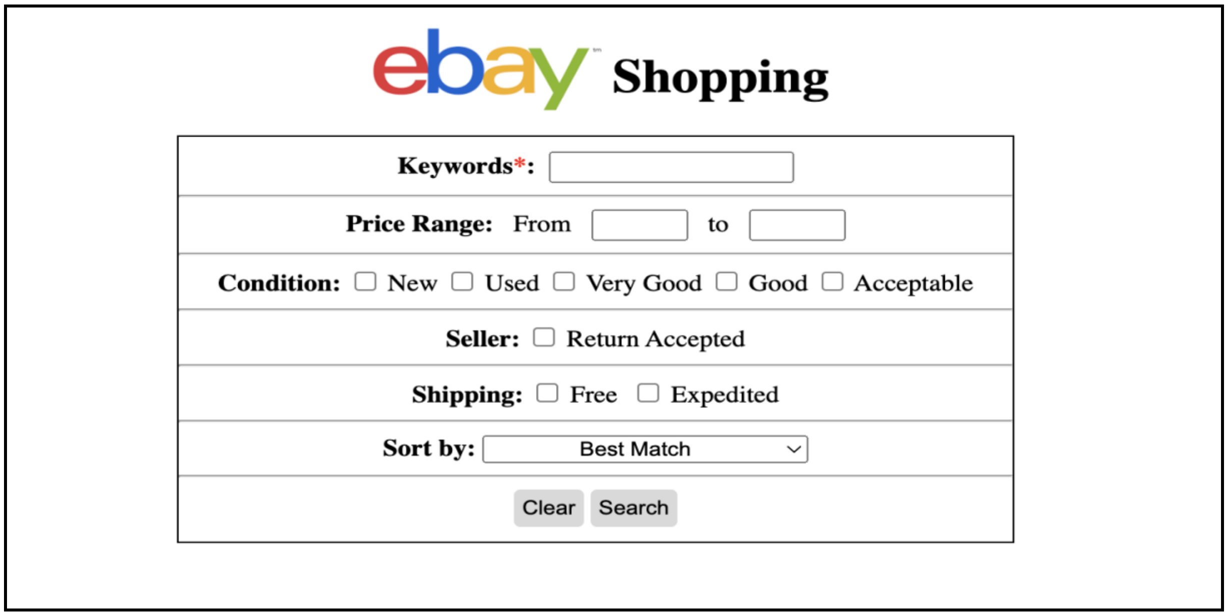
- Get experience with Python programming language and Flask framework.
- Get experience creating web pages using HTML, CSS, JavaScript, HTML DOM and XMLHTTPRequest object.
- Get experience with the eBay Finding API.
- Get experience using JSON format.
- Getting hands-on experience in GCP, AWS or Azure.

1.1 Cloud Exercise

- The backend of this homework must be implemented in the cloud on GCP, AWS or Azure using Python.
- See *Cloud Setup (Python)* for the installation of needed components on GCP, AWS or Azure.
- See the hints (section 3) at the bottom; a lot of reference material is given to you.
- For Python and Flask kickstart please refer to the Lecture slides on the class website.
- You must refer to the grading guidelines, the video, the specs and Piazza. Styling is graded this time and the points breakup is mentioned in guidelines.

2. Description

In this exercise, you are asked to create a webpage that allows users to search items for sale on eBay.com using their API, and the results will be displayed in a tabular format. Instructions on how to use the API are given in section 3.1. The user first opens a page as shown below in **Figure 1**, where they can enter a query in the keywords textbox.



ebay Shopping

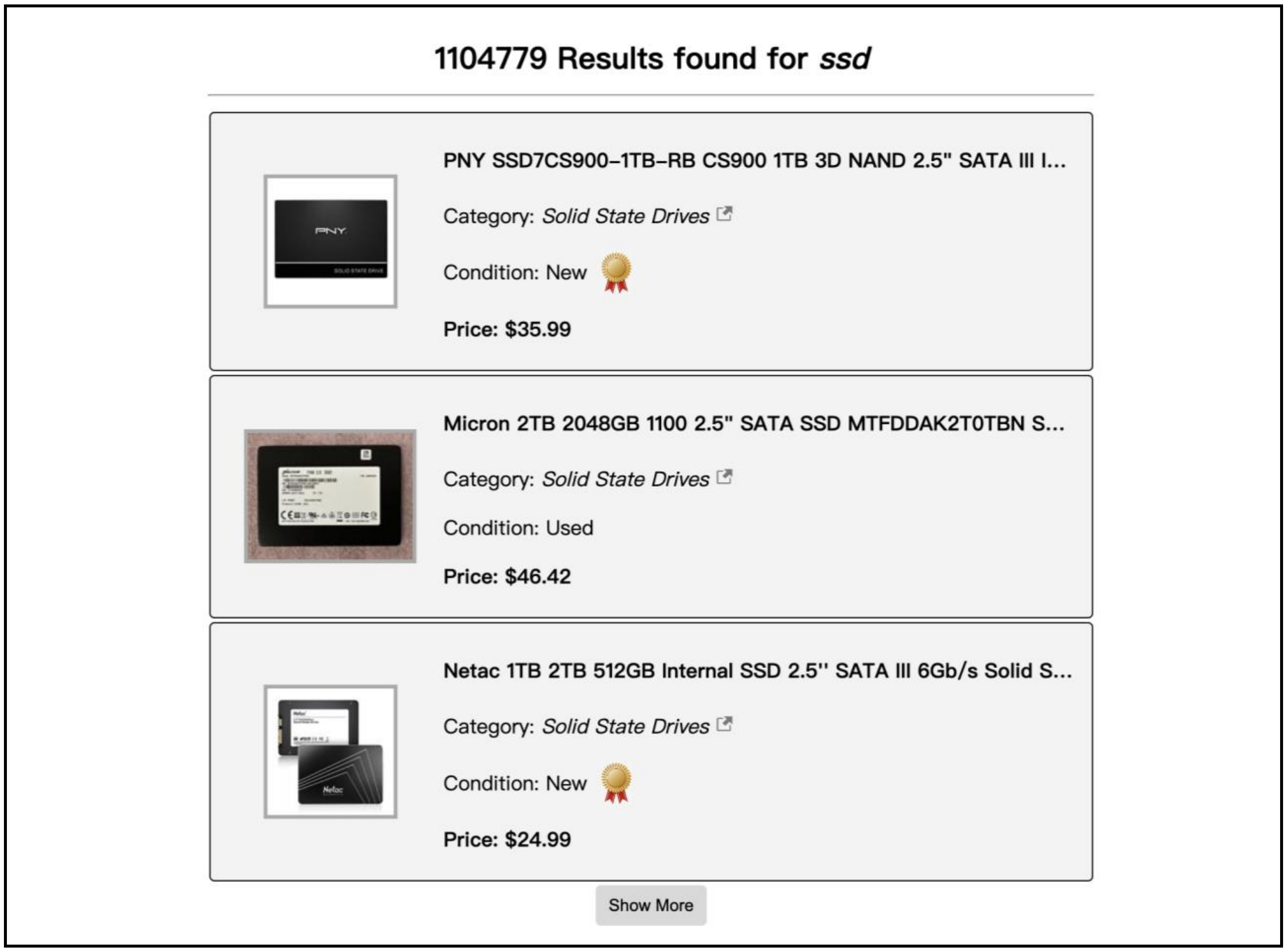
Keywords*:	
Price Range:	From ☐ to ☐
Condition:	☐ New ☐ Used ☐ Very Good ☐ Good ☐ Acceptable
Seller:	☐ Return Accepted
Shipping:	☐ Free ☐ Expedited
Sort by:	Best Match

Figure 1: Initial Search Form

Once the user has provided valid data (a keyword is required), your script will make a request to your web server providing it with the form data that was entered. You **must** use **GET** to transfer the form data to your web server (**do not** use **POST,** as you would be unable to provide a sample link to your cloud services). A python script using Flask will retrieve the data and send it to the *eBay API Web Service* "***findItemsAdvanced***". The API call to eBay is done simply using a URL REST request. For example, if you are just searching for "harry potter" and you want to retrieve the results sorted by ascending price (i.e., from low to high price including shipping cost) in JSON format, the URL will be:

> https://svcs.eBay.com/services/search/FindingService/v1?**OPERATION-NAME=findItemsAdvanced**&SERVICE-VERSION=1.0.0&**SECURITY-APPNAME=**YourEBayAppKey&**RESPONSE-DATA-FORMAT=**JSON&**keywords=**harry%20potter&**sortOrder**=PricePlusShippingLowest

A sample result snippet of 3 results is shown below in **Figure 2.**



1104779 Results found for *ssd*

PNY SSD7CS900–1TB–RB CS900 1TB 3D NAND 2.5" SATA III I...

Category: *Solid State Drives*

Condition: New

Price: $35.99

Micron 2TB 2048GB 1100 2.5" SATA SSD MTFDDAK2T0TBN S...

Category: *Solid State Drives*

Condition: Used

Price: $46.42

Netac 1TB 2TB 512GB Internal SSD 2.5'' SATA III 6Gb/s Solid S...

Category: *Solid State Drives*

Condition: New

Price: $24.99

Figure 2. Valid Search Results

2.1. Description of the Search Form

The following is a brief description of the fields in the search form (see **Figure 1**) which you need to implement in your homework.

1. **Key Words**: This is a text box, which enables the user to search for matching items by entering keywords. This must be a non-empty string. If the **search** button is clicked while this text box is empty, a tooltip should inform the user that this is a mandatory field (see **Figure 3**), and no query to the server should be performed.

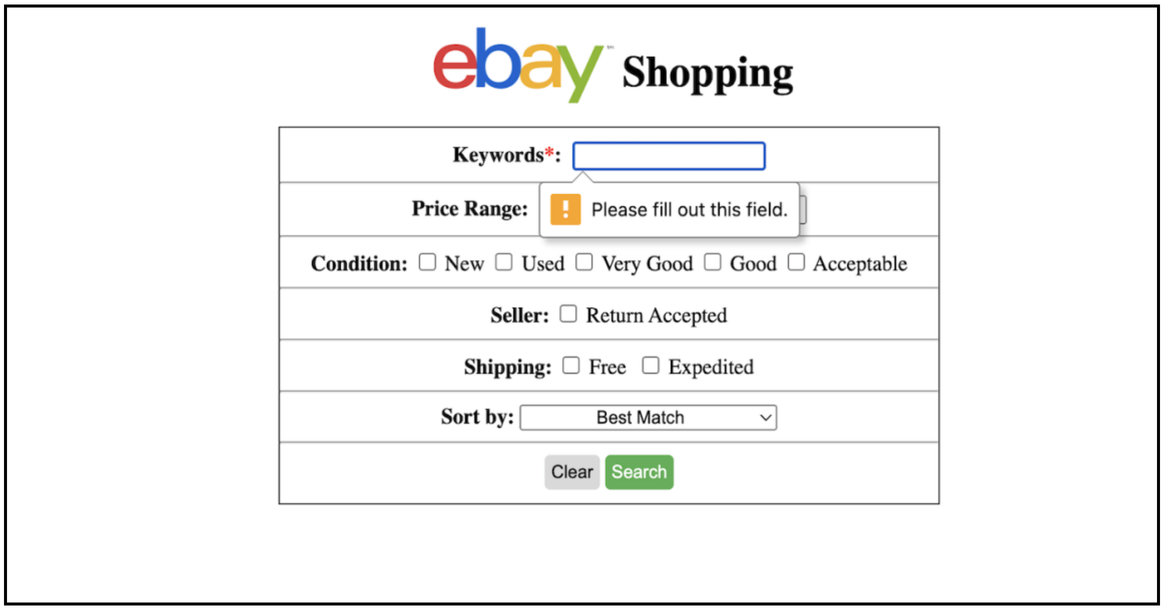
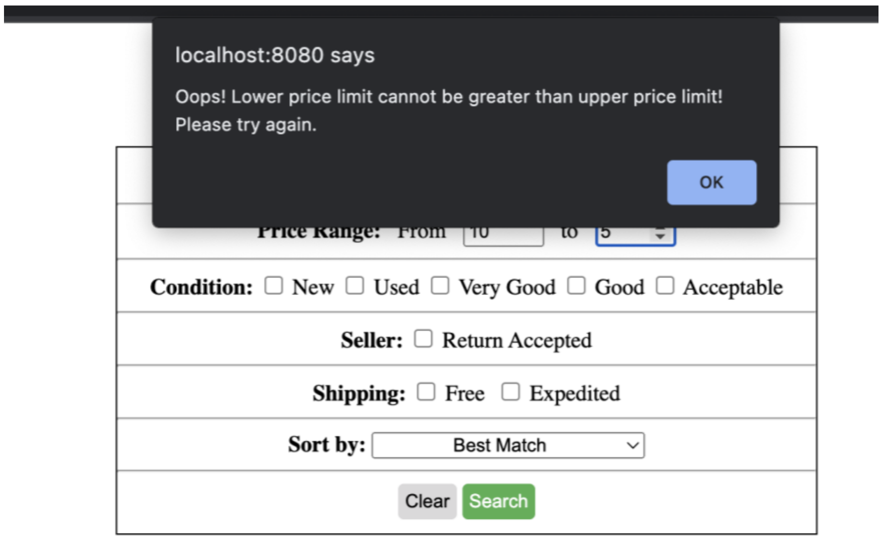


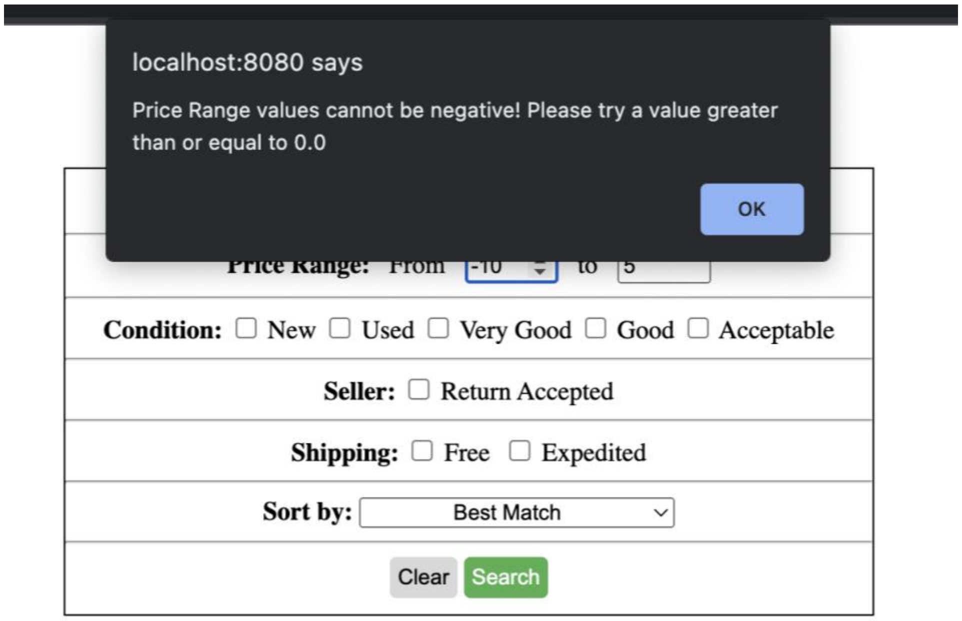
Figure 3. Empty Query

2. **Price Range:** These are two boxes, displaying numeric values, which enable the user to assign a price range for the matching items. These can be positive integers or decimal numbers [0.0, ∞). You must ensure that **Minimum Price <= Maximum Price and MinimumPrice, Maximum Price >=0.0.** For specifying the price range**,** it is possible to assign only the lower price range or the higher price range, or both of them, or none of them. If the minimum price > maximum price, an error message should be displayed as shown in **Figure 4**. An error message for negative price value should be displayed as shown in **Figure 5**.


localhost:8080 says

Oops! Lower price limit cannot be greater than upper price limit! Please try again.

Figure 4. Incorrect Price Range


localhost:8080 says

Price Range values cannot be negative! Please try a value greater than or equal to 0.0

Figure 5. Negative Price Value

3. **Condition**: These are five checkboxes "*New*, *Used*, *Very Good*, *Good*, and *Acceptable*" which enable the user to control the item condition retrieved in the results. The user can select one of the options or a combination of them to search for items in specific conditions.
4. **Seller**: Specifies if the user wants only items sold by a seller who accepts returns.
5. **Shipping**: Specify if only "free shipping" items should be returned, if only items available with expedited shipping should be returned or combination of both. Default is not specified which means "any" filter.
6. **Sort By**: This specifies the ordering of the results table, which can be one of:
    a. Best Match - ***THIS IS THE DEFAULT***
    b. Price: highest first
    c. Price + Shipping: highest first
    d. Price + Shipping: lowest first

The search form has two buttons:

1. **SEARCH** button**:** this button validates whether the user provided all the mandatory field (i.e., key words) and verifies the correctness of the provided data in the fields (the price range is valid). The validation should be implemented in a JavaScript function. Then, an HTTP request is made to your web server in the cloud providing it with the form data that was entered. You need to use **GET** to transfer the form data as query/path parameters to the web server.
2. **CLEAR** button: This button **must** clear the result area, all text fields, uncheck all checkboxes and reset the “sort by” field to its default value mentioned above. The clear operation is done using a JavaScript function.

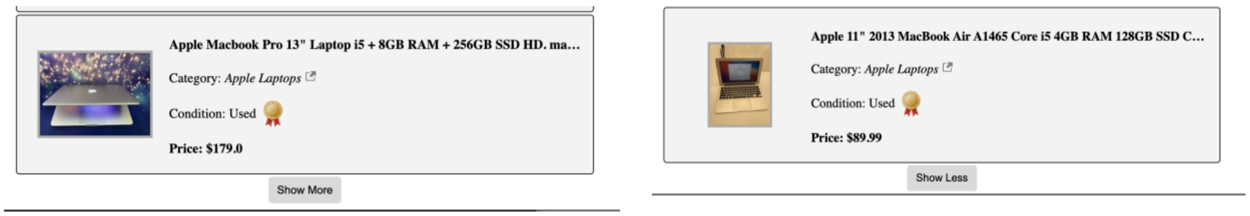
NOTE: In the video, you should notice the change in styling for the two buttons when hovering. Also, notice that the selected sort criteria in the Sort By filter is center aligned.

2.2. Displaying Results

The list of results should be displayed as shown in **Figure 2** along with a headline indicating the number of results, i.e., the total number of entries on eBay retrieved for a given keyword, and the corresponding filters. The headline is separated from the list of results by a horizontal line. Items are to be stacked vertically (1 item per row). Inside the item card, to the left, is the item’s display image. On the top right, the item title is displayed in bold. Below it, the Category (in which the item is tagged) is displayed in the second line, “condition” of the item is displayed in the third line and lastly the “Price” in USD is displayed. Additionally, if applicable, you must also indicate whether the item is top rated to the right of the Category field using the image *topRatedImage.jpg*.

NOTE: If any attribute is missing or empty or blank, do not display the item to the user.

If more than 3 items are returned in the results, you should display the first 3 items along with the **Show More** button underneath. On clicking the button, a maximum of 7 more items (for a total of 10 items) should be displayed and the **Show More** button changes to **Show Less** as shown in **Figure 6**. After clicking the **Show More** button, all the items will be displayed and a **Show Less** button will be displayed. Clicking the **Show Less** button will again display just 3 results. Additionally, when the **Show More** button is clicked, the page should automatically scroll to the bottom, and it should scroll to the top of the page when the **Show Less** button is clicked. The **Show More** and **Show Less** buttons should have the same styling as the clear and search buttons of the form. See the behavior in the video.



Apple Macbook Pro 13" Laptop i5 + 8GB RAM + 256GB SSD HD. ma...

Category: *Apple Laptops*

Condition: Used

Price: $179.0

Apple 11" 2013 MacBook Air A1465 Core i5 4GB RAM 128GB SSD C...

Category: *Apple Laptops*

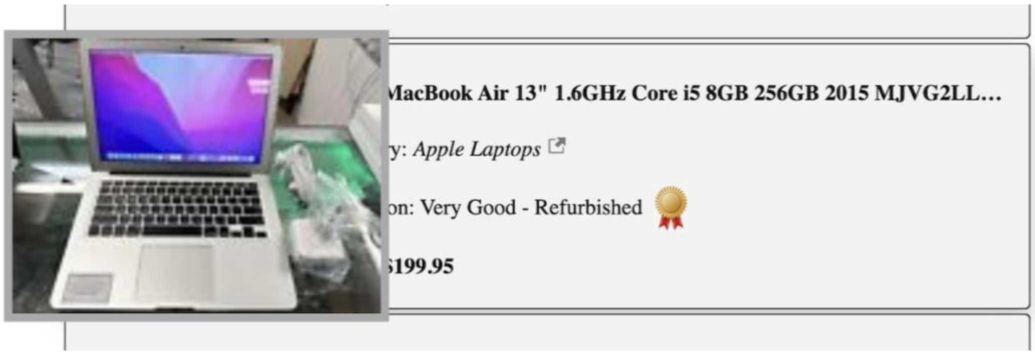
Condition: Used

Price: $89.99

Figure 6: Show More and Show Less Buttons

In case the item does not contain images, the eBay server returns a default image with the URI as https://thumbs1.ebaystatic.com/ pict/04040_0.jpg. Replace this image with the **Item Default Image** given to you (See section 5, Images).

Additionally, the item image magnifies when the user hovers on the image and has a transition while magnifying, as shown in **Figure 7**. See the video for the effect.



MacBook Air 13" 1.6GHz Core i5 8GB 256GB 2015 MJVG2LL...
y: Apple Laptops
on: Very Good - Refurbished
199.95

Figure 7: Zoomed-in image on mouseover

The item is initially in the summary state as shown in **Figure 7**. The fields used from the JSON Response for creating the item summary card and the number of results headlines are shown in **Table 1** below.

Item Information in the Card	**Tags in eBay JSON response**
Total Results Found	*paginationOutput->totalEntries*
Item's image URL	*searchResult->item->galleryURL*
Item Title	*searchResult->item->title*
Item Category tag	*searchResult->item->primaryCategory->categoryName*
eBay product link for redirection	*searchResult->item->viewItemURL*
Condition of the item	*searchResult->item->condition->conditionDisplayName*
Item Top Rated Image (If the current listing is top rated for the given item)	if the value of *searchResult-->item->topRatedListing* is true, display the image topRatedImage.jpg. You can find the image at **URL Link**
Item Price	The price of the item is displayed in the format: *Price: $x (+ $y for shipping)* The item price value is read from *searchResult->item->sellingStatus-> convertedCurrentPrice* The shipping cost is read from *searchResult->item->shippingInfo->shippingServiceCost*

	The shipping cost phrase (+ *$YY for shipping*) is mentioned if and only if the value of ***shippingServiceCost*** *field* is greater than or equal to 0.01 dollars

Table 1. Summary Card information

Note: All the numeric values in the JSON response are strings and need to be converted into JavaScript Number type for comparisons.

2.3. Displaying Item Details

In the search result card, if the user clicks on the card, the page should make a request for the detailed information using the eBay shopping API documented at:
https://developer.ebay.com/devzone/shopping/docs/CallRef/GetSingleItem.html

After receiving the JSON from Flask, you should use JavaScript to replace the search card, item count and show more/show less button with an individual item table using the JSON object and display the results in a similar format as **Figure 8A**.

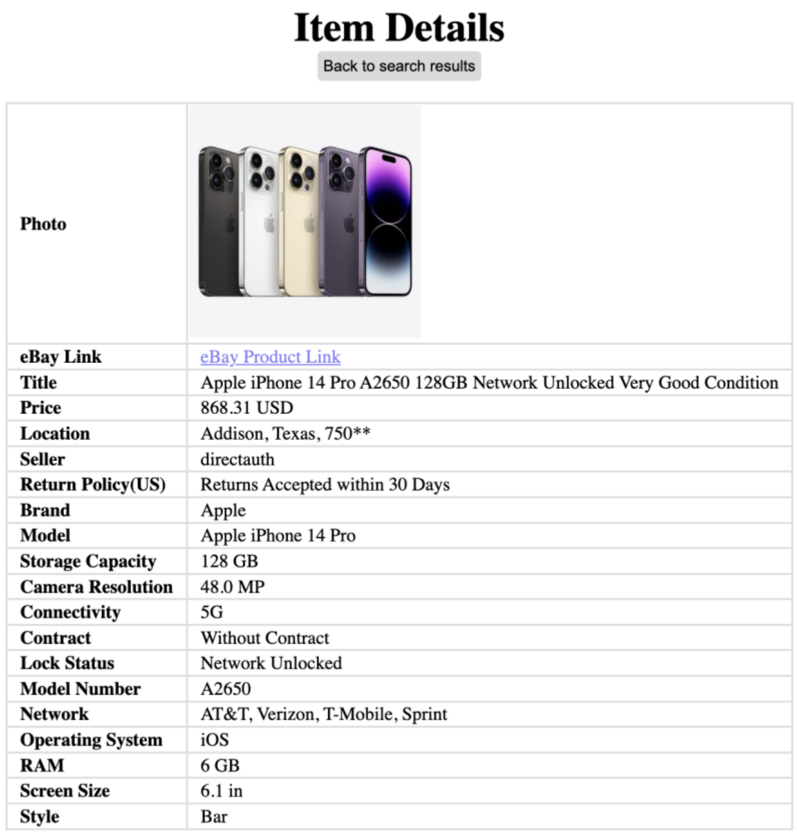
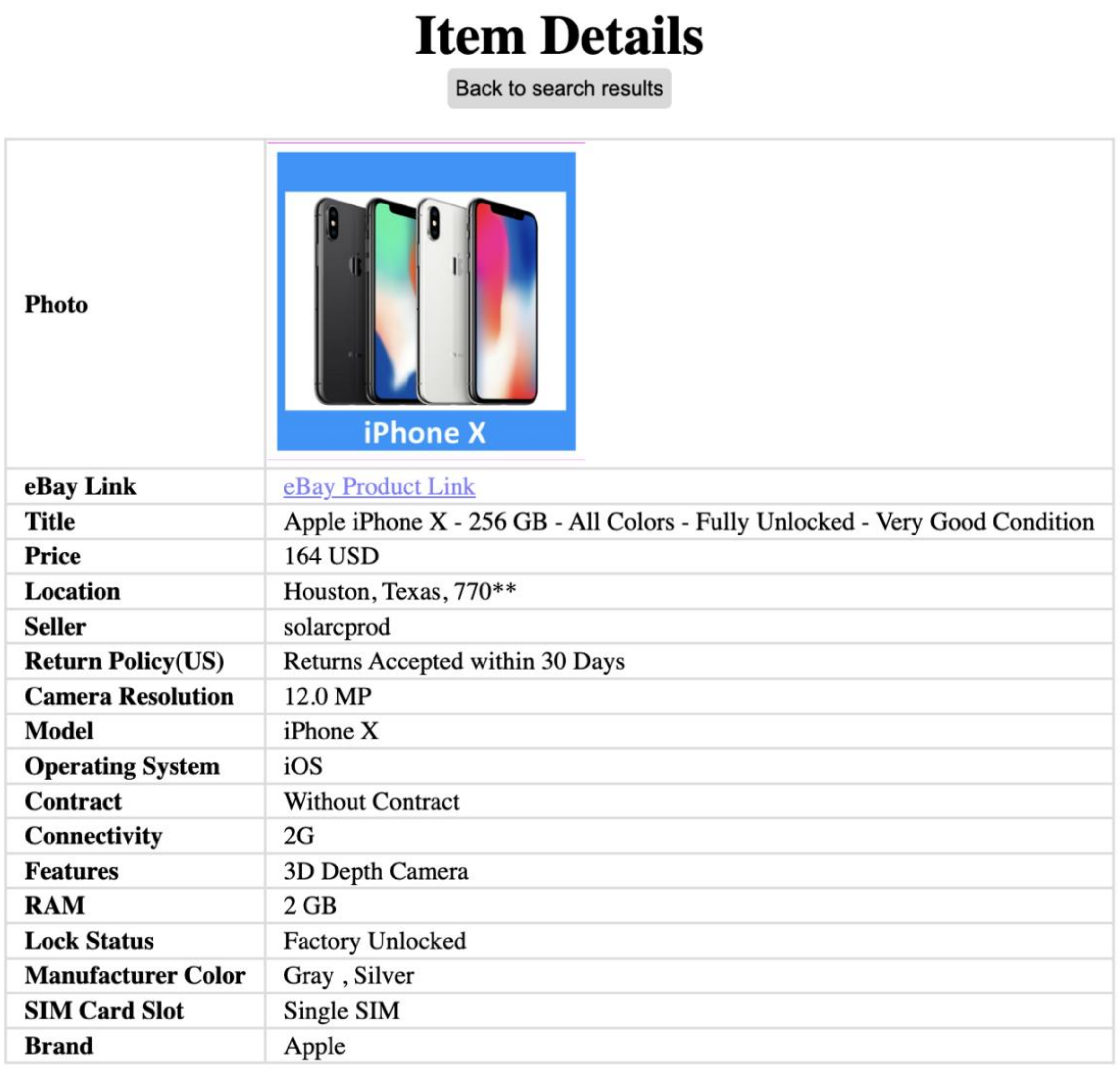


Photo	
eBay Link	eBay Product Link
Title	Apple iPhone 14 Pro A2650 128GB Network Unlocked Very Good Condition
Price	868.31 USD
Location	Addison, Texas, 750**
Seller	directauth
Return Policy(US)	Returns Accepted within 30 Days
Brand	Apple
Model	Apple iPhone 14 Pro
Storage Capacity	128 GB
Camera Resolution	48.0 MP
Connectivity	5G
Contract	Without Contract
Lock Status	Network Unlocked
Model Number	A2650
Network	AT&T, Verizon, T-Mobile, Sprint
Operating System	iOS
RAM	6 GB
Screen Size	6.1 in
Style	Bar

Figure 8A: Detailed Item with Full Field

If the returned JSON stream doesn't contain certain fields, those fields will not appear on the detail page. A sample output is shown in **Figure 9**. **Figure 8A** shows a result with all fields. **Figure 8B** shows a result with missing fields such as "Processor" and other item specific fields.



Item Details

Photo	
eBay Link	eBay Product Link
Title	Apple iPhone X - 256 GB - All Colors - Fully Unlocked - Very Good Condition
Price	164 USD
Location	Houston, Texas, 770**
Seller	solarcprod
Return Policy(US)	Returns Accepted within 30 Days
Camera Resolution	12.0 MP
Model	iPhone X
Operating System	iOS
Contract	Without Contract
Connectivity	2G
Features	3D Depth Camera
RAM	2 GB
Lock Status	Factory Unlocked
Manufacturer Color	Gray , Silver
SIM Card Slot	Single SIM
Brand	Apple

Figure 8B: Item with Missing field.

When the search result contains at least one field, you need to map the data extracted from the API result to render the HTML result table as described in **Table 3**.

HTML Key	API service response
Photo	The value of the "PictureURL" attribute that is part of the "Item" object.
eBay Link	*searchResult -> item -> ViewItemURLForNaturalSearch*
Title	The value of the "Title" attribute that is part of the "Item" object.

SubTitle	The value of the “Subtitle” attribute that is part of the “Item” object.
Price	The value of the “price” attribute that is part of the “currentPrice” object inside the “Item” object.
Location	The value of the “Location” attribute that is part of the “Item” object along with its “postalcode” which is also a part of the “Item” object.
Seller	The value of the “UserId” attribute that is part of the “Seller” object inside the “Item” object.
Return Policy (US)	The value of the “ReturnPolicy” attribute that is part of the “Item” object.
ItemSpecifics (Name)	The value of the all the "NameValueList" array corresponding to that name inside the “ItemSpecifics” object in the “Item” object.

Table 3: Mapping the result from eBay Shopping API into HTML Table

To get back to the search result, a back to search button will be added above the item table. Then click the button and it should display back to the search result card. The button should have the same CSS style as submit and reset.

2.4. Saving Previous Inputs

In addition to displaying the results, your page should maintain the provided filter values while displaying the current results. For example, if one searches for “Free Shipping” items for “harry potter”, one should see what was provided in the search form and the corresponding results. Same goes for **all** fields and input types. It follows that you need to keep the whole search box/input fields and buttons, even while displaying results/errors. Also, the keyword text field should provide previous keywords searched as suggestions (default behavior for text fields but can be toggled to enable this feature).

2.4. No Results

In cases where the eBay API returns no items for a specific keyword such as “*alcnkajcsnkacnka*”, you should display “No results found” or any similar message as shown in **Figure 9**.

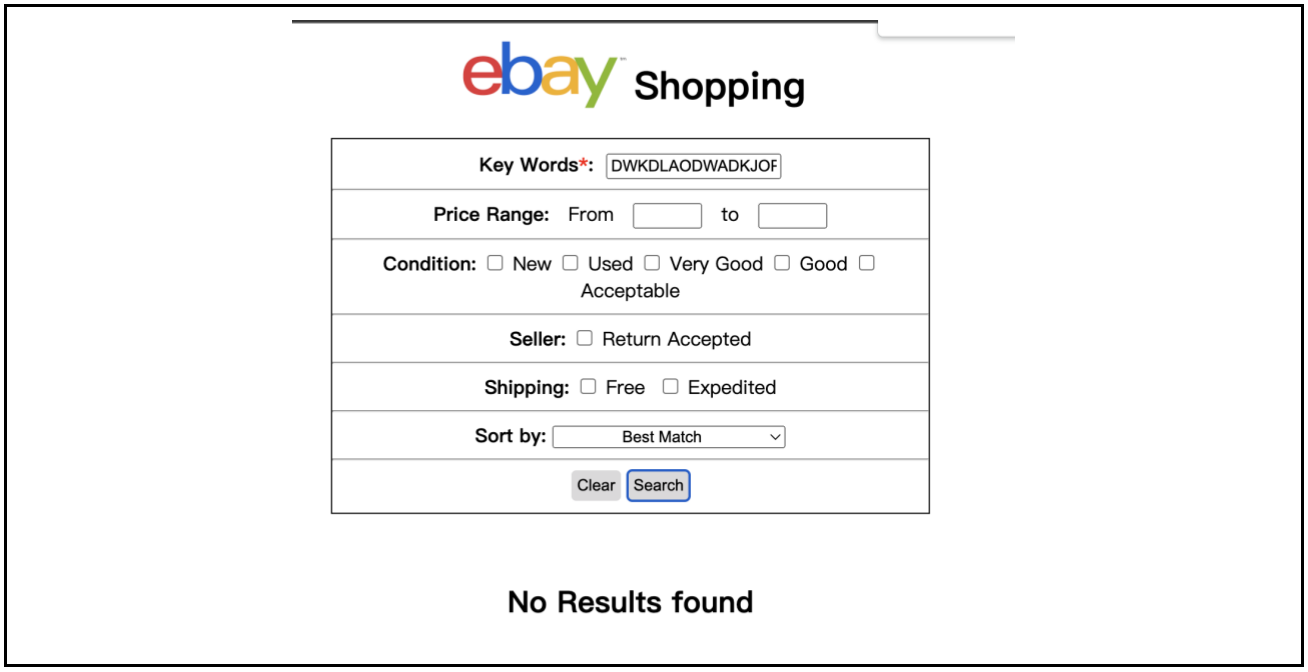

ebay Shopping
Key Words*: DWKDLAODWADKJOF
Price Range: From to
Condition: ☐ New ☐ Used ☐ Very Good ☐ Good ☐ Acceptable
Seller: ☐ Return Accepted
Shipping: ☐ Free ☐ Expedited
Sort by: Best Match
Clear
Search
No Results found

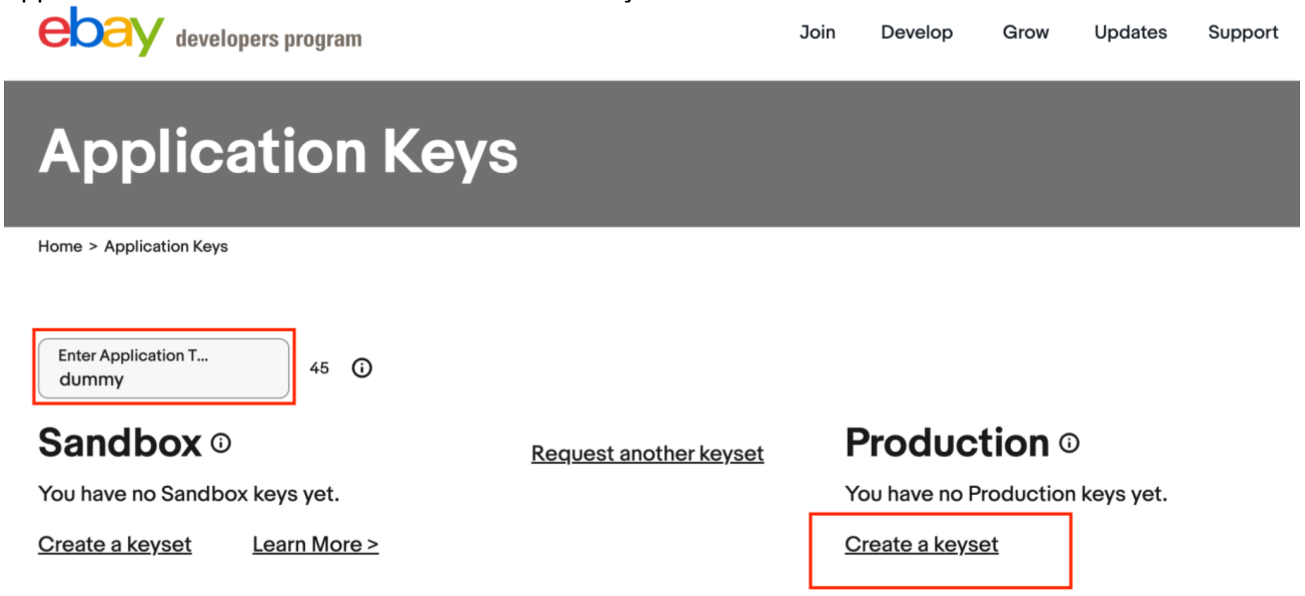
Figure 9: An example when No Results are found

3. How to Use the eBay "findItemsAdvanced" API

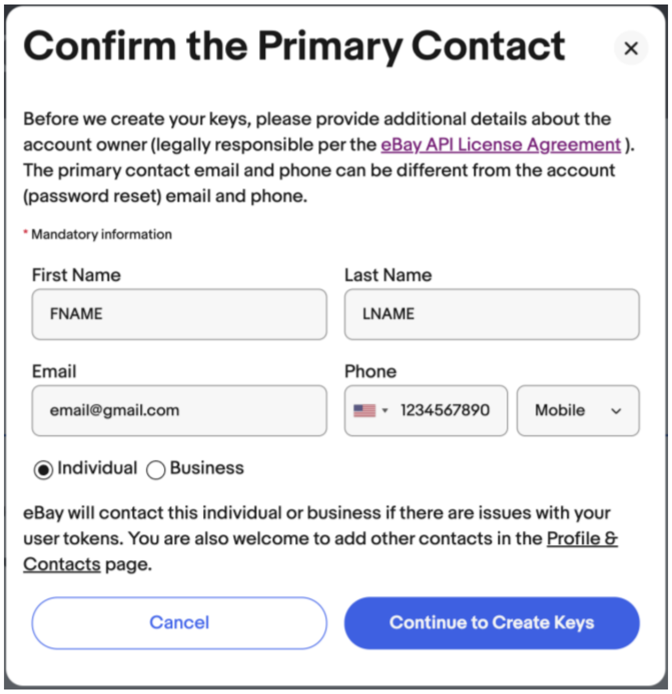
3.1. HOW TO CREATE THE eBay App ID

To use any of the eBay APIs, you should first register for membership in the eBay Developer Program at https://developer.ebay.com/signin?tab=register. **The activation of the account takes 1 business day so register early to avoid delays.**

1. After registration, login into the developer portal and create an application by entering your application name and then click on "Create a keyset" as shown below



2. You might get a pop-up to confirm your developer account details. Enter the details and "Continue to Create Keys"



Confirm the Primary Contact

Before we create your keys, please provide additional details about the account owner (legally responsible per the eBay API License Agreement). The primary contact email and phone can be different from the account (password reset) email and phone.

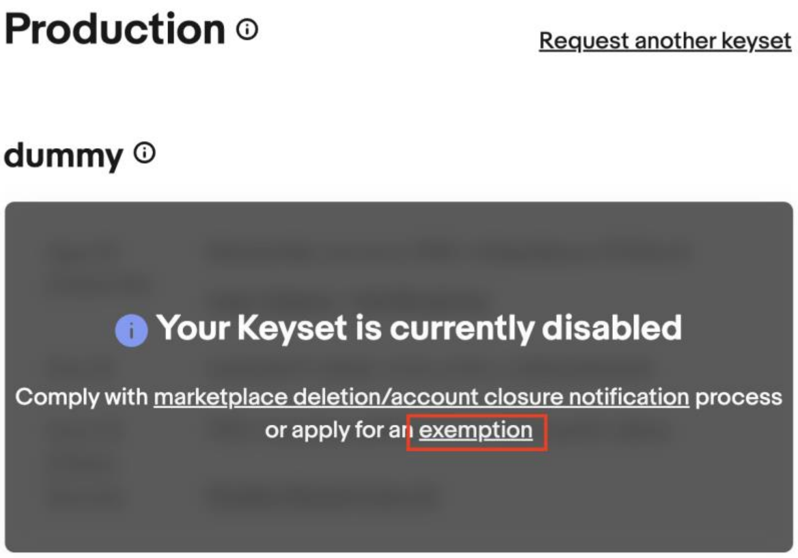
* Mandatory information

First Name: FNAME	Last Name: LNAME
Email: email@gmail.com	Phone: 1234567890 Mobile

[✓ Individual] Business

eBay will contact this individual or business if there are issues with your user tokens. You are also welcome to add other contacts in the Profile & Contacts page.

3. You might now see a message like this. To enable your Keyset, apply for an exemption by clicking on the link shown in the image below. (Since we don't persist any personal data - When creating a production key, we can submit an Exemption request)



Production

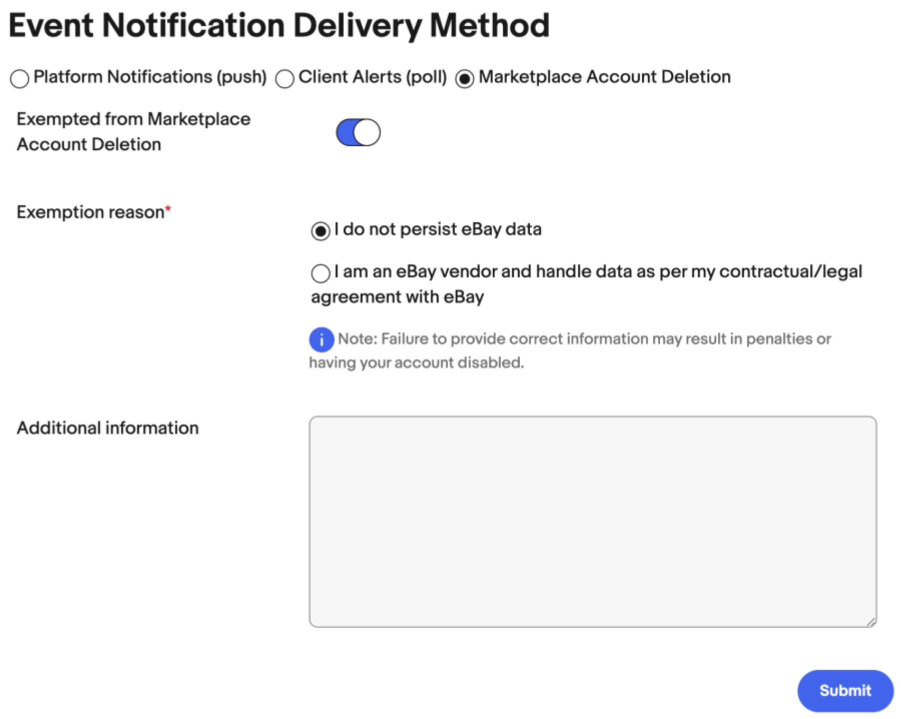
Request another keyset

dummy

Your Keyset is currently disabled

Comply with marketplace deletion/account closure notification process or apply for an exemption

4. Fill the exemption form as shown below and submit it.



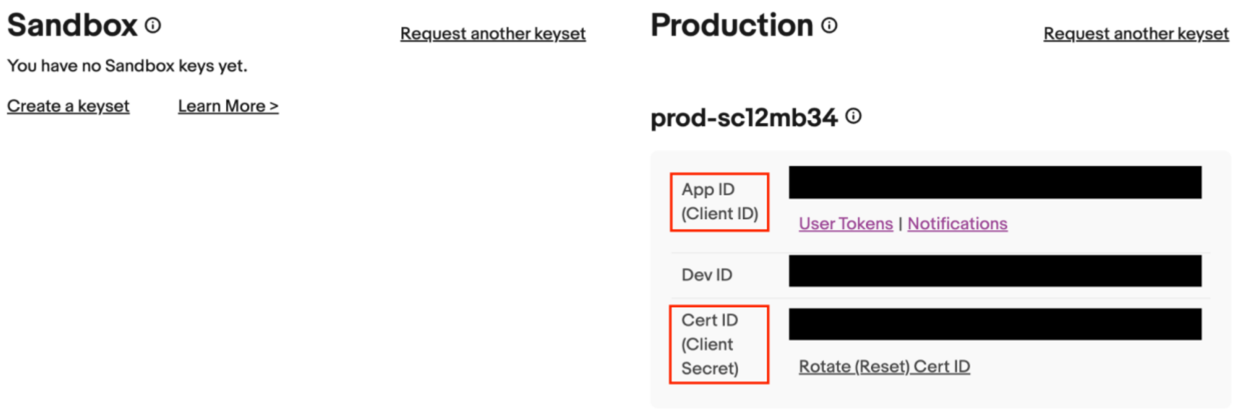
Event Notification Delivery Method

○ Platform Notifications (push) ○ Client Alerts (poll) [✓ ● Marketplace Account Deletion]

Exempted from Marketplace Account Deletion	On
Exemption reason*	[✓ ● I do not persist eBay data] ○ I am an eBay vendor and handle data as per my contractual/legal agreement with eBay
Additional information	

Note: Failure to provide correct information may result in penalties or having your account disabled.

5. This will create a “Production Keyset”. Do not create more than one Key Set (set of 3 keys, DEVID, AppID, CertID). When you have created your Key Set, your account page may look something like below (ignore the “Sandbox Keys”)



Sandbox Request another keyset

You have no Sandbox keys yet.

Create a keyset Learn More >

Production Request another keyset

prod-sc12mb34

App ID (Client ID)	[redacted] User Tokens | Notifications
Dev ID	[redacted]
Cert ID (Client Secret)	[redacted] Rotate (Reset) Cert ID

Figure 10A: Application Keys

Under the “Application Keys” section, “Production” sub-section, you’ll find your AppID. In **Figure 10A**, the application ID is marked in red. When constructing the URL to call the eBay APIs, your Application ID should be provided as a value for the parameter SECURITY-APPNAME. Each AppID only allows 5000 API calls a day. Please ensure you do not exceed the daily usage limit. After reaching the limit, any subsequent calls will fail for 24 hours. You will need the “Client ID” and “Client Secret” highlighted in Figure 10A for the “**GetSingleItem**” API. Keep them handy.

3.2 Constructing URL For eBay API Calls

In this homework, we will use the eBay ***"findItemsAdvanced"*** API. A comprehensive reference about this API is available at:

http://developer.eBay.com/DevZone/finding/CallRef/findItemsAdvanced.html.

Table 4 shows the mapping between the search fields and the URL parameters to call the eBay API. Some of the search fields are handled through ItemFilter. The item filter is specified using two parameters: *itemFilterName* and *itemFilterValue*. The Condition parameter can take multiple values. The reference for the URL parameters can be found at:

https://developer.ebay.com/DevZone/finding/CallRef/extra/fnditmsadvncd.rqst.tmfltr.nm.html

Search Field	**URL parameter in the API Call**
Key words	The name of the query parameter is *Keywords*
Sort by (A drop-down list displaying: 1. Best Match 2. Price: highest first 3. Price + Shipping: highest first 4. Price + Shipping: lowest first)	*sortOrder.* The possible values are: 1. BestMatch 2. CurrentPriceHighest 3. PricePlusShippingHighest 4. PricePlusShippingLowest
Price Range From	The name of the **item filter** is *MinPrice*. The value of the filter is the value of "Price Range From" field. Values have to be decimal and greater than or equal to 0.0. Additional parameters to pass with this filter are **paramName=Currency** and **paramValue=USD**
Price Range To	The name of the **item filter** is *MaxPrice*. The value of the filter is the value of "Price Range To" field. Values have to be decimal and greater than or equal to 0.0. Additional parameters to pass with this filter are **paramName=Currency** and **paramValue=USD**
Seller - Returns Accepted	The name of the **item filter** is *ReturnsAcceptedOnly*. The possible values are true or false.
Shipping - Free Shipping	The name of the item filter is FreeShippingOnly. The value can be true/false.
Shipping - Expedited shipping available	The name of the item filter is ExpeditedShippingType. One of the allowed values is: Expedited. If checked, pass Expedited value else do not apply this filter in the API call.
Condition (a set of check boxes: New, Used, Very Good, Good, and Acceptable)	The name of the **item filter** is Condition. This filter defaults to OR logic if multiple values are provided. The possible values are: 1. 1000 (means New) 2. 3000 (means Used) 3. 4000 (means Very Good) 4. 5000 (means Good) 5. 6000 (means Acceptable)

	If no value is checked, do not apply this filter in the API call.

Table 4: Mapping between the search fields and the URL parameters

In addition to the parameters mentioned in the above table, there are five parameters which should be included in every call. The name of these parameters and their values are listed below:

- OPERATION-NAME=findItemsAdvanced
- SERVICE-VERSION=1.0.0
- SECURITY-APPNAME=**YourEBayAppKey**
- RESPONSE-DATA-FORMAT=JSON
- REST-PAYLOAD

Every item filter should have two parameters (name and value). When listing the filters, they should be indexed starting from ZERO. An Example of listing three parameters:

itemFilter**(0)**.name=*filter1NAME*&itemFilter**(0)**.value=*filter1Value*&itemFilter**(1)**.name=*filter2NAME*&
&itemFilter**(1)**.value=*filter2Value*&itemFilter**(2)**.name=*filter3NAME*&itemFilter**(2)**.value=*filter3Value*

If the filter is assigned multiple values, the values should be mentioned in a list of parameters and the list of the values should be indexed from ZERO. For example, in order to filter items based on their *Condition* to be *NEW* or *USED* or *Very Good* can be written as:

itemFilter**(X)**.name=Condition**&**itemFilter**(X)**.value**(0)**=1000**&**itemFilter**(X)**.value**(1)**=3000**&**itemFilter**(
X)**.value**(2)**=4000

For more information about filters, you can read this page:

http://developer.ebay.com/DevZone/finding/CallRef/types/ItemFilterType.html

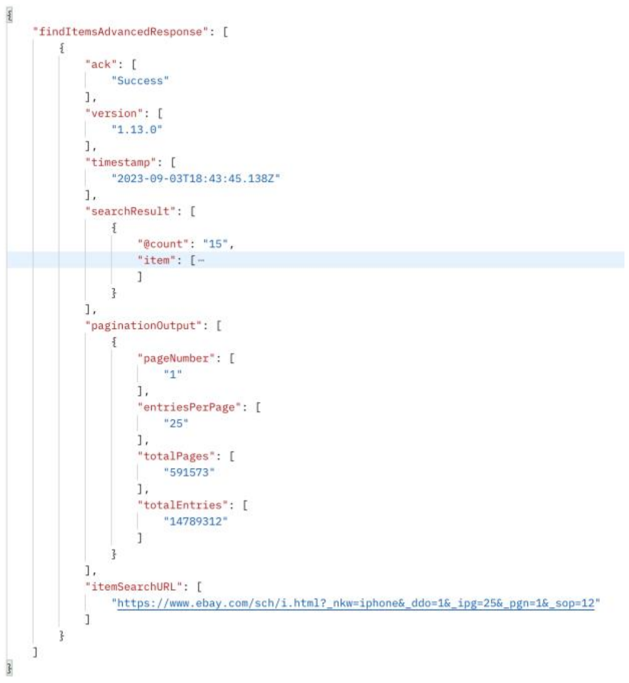
3.3 FindingService API

An example URL constructed from the parameters will look like:

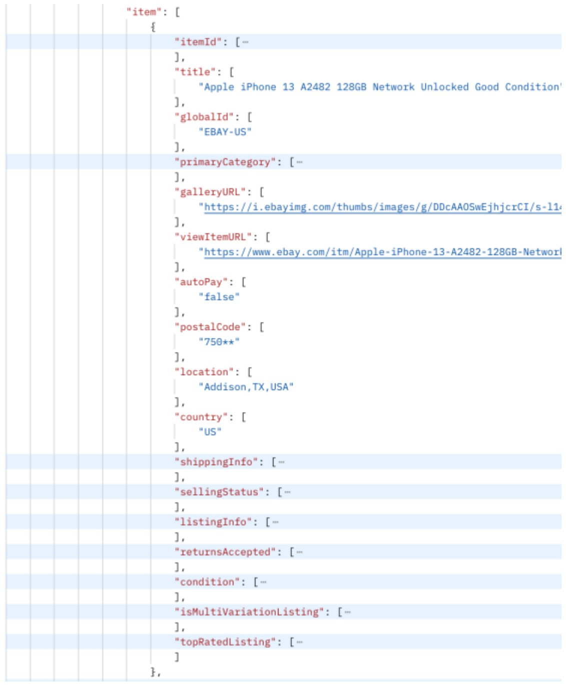
https://svcs.ebay.com/services/search/FindingService/v1?OPERATION-NAME=findItemsAdvanced&SERVICE-VERSION=1.0.0&SECURITY-APPNAME=**YourAppID**&RESPONSE-DATA-FORMAT=JSON&REST-PAYLOAD&keywords=iphone&paginationInput.entriesPerPage=10&sortOrder=BestMatch&itemFilter(0).name=MaxPrice&itemFilter(0).value=25&itemFilter(0).paramName=Currency&itemFilter(0).paramValue=USD

In the example URL above, the **paginationInput.entriesPerPage** parameter can be used to reduce the number of items returned by the eBay API. This parameter is **optional** and if omitted, the eBay API will return a maximum of 100 items by default. You need to only display a maximum of 10 items on the web page but might need to request more items to ensure having 10 items even when some listings must be removed due to missing fields or errors.

Figure 11 A and B show an example of the JSON Response returned by the FindingService API.



```
{
    "findItemsAdvancedResponse": [
        {
            "ack": [
                "Success"
            ],
            "version": [
                "1.13.0"
            ],
            "timestamp": [
                "2023-09-03T18:43:45.138Z"
            ],
            "searchResult": [
                {
                    "@count": "15",
                    "item": [⋯
                    ]
                }
            ],
            "paginationOutput": [
                {
                    "pageNumber": [
                        "1"
                    ],
                    "entriesPerPage": [
                        "25"
                    ],
                    "totalPages": [
                        "591573"
                    ],
                    "totalEntries": [
                        "14789312"
                    ]
                }
            ],
            "itemSearchURL": [
                "https://www.ebay.com/sch/i.html?_nkw=iphone&_ddo=1&_ipg=25&_pgn=1&_sop=12"
            ]
        }
    ]
}
```



```
"item": [
    {
        "itemId": [⋯
        ],
        "title": [
            "Apple iPhone 13 A2482 128GB Network Unlocked Good Condition'
        ],
        "globalId": [
            "EBAY-US"
        ],
        "primaryCategory": [⋯
        ],
        "galleryURL": [
            "https://i.ebayimg.com/thumbs/images/g/DDcAAOSwEjhjcrCI/s-l1
        ],
        "viewItemURL": [
            "https://www.ebay.com/itm/Apple-iPhone-13-A2482-128GB-Networ
        ],
        "autoPay": [
            "false"
        ],
        "postalCode": [
            "750**"
        ],
        "location": [
            "Addison,TX,USA"
        ],
        "country": [
            "US"
        ],
        "shippingInfo": [⋯
        ],
        "sellingStatus": [⋯
        ],
        "listingInfo": [⋯
        ],
        "returnsAccepted": [⋯
        ],
        "condition": [⋯
        ],
        "isMultiVariationListing": [⋯
        ],
        "topRatedListing": [⋯
        ]
    },
```

Figure 11A and **B:** JSON for FindingService API and Single Item JSON

3.4 GetSingleItem API

<u>To retrieve the details of a single item, the request needs the following parameters (output should be JSON):</u>

- **callName**: Set it to "**GetSingleItem**" to get information for a specific product.
- **responseencoding**: Set it to "JSON" to get a JSON response.
- **appid**: Your application's API key. This key identifies your application for purposes of quota management.
- **siteid**: Set it to '0' for siteId purposes.
- **version**: Set it to '967' for API version purposes.
- **ItemId**: It is the "itemId" of the product the user clicked.
- **IncludeSelector**: Set it to "Description,Details,ItemSpecifics" to get required fields for that product.

Make sure to set the **X-EBAY-API-IAF-TOKEN** header in your API requests as described in the next section.

<u>NOTE</u> - Developers using "GetSingleItem" API calls must authenticate with an OAuth application access token in the HTTP header X-EBAY-API-IAF-TOKEN. Follow the steps below to create an OAuth token.

1. Download the `ebay_oauth_token.py` file shared with this assignment. The file should be placed in your backend on GCP, AWS or Azure.

2. Use the following code in your API server Python file:

```
from ebay_oauth_token import OAuthToken
client_id = "same_as_your_app_id"
client_secret = "cert_id_or_client_secret"
# Create an instance of the OAuthUtility class
oauth_utility = OAuthToken(client_id, client_secret)
# Get the application token
application_token = oauth_utility.getApplicationToken()
```

3. Add the following header to your **get** call:

```
headers = {
"X-EBAY-API-IAF-TOKEN": oauth_utility.getApplicationToken()
}
response = requests.get(get_single_item_url, headers=headers)
```

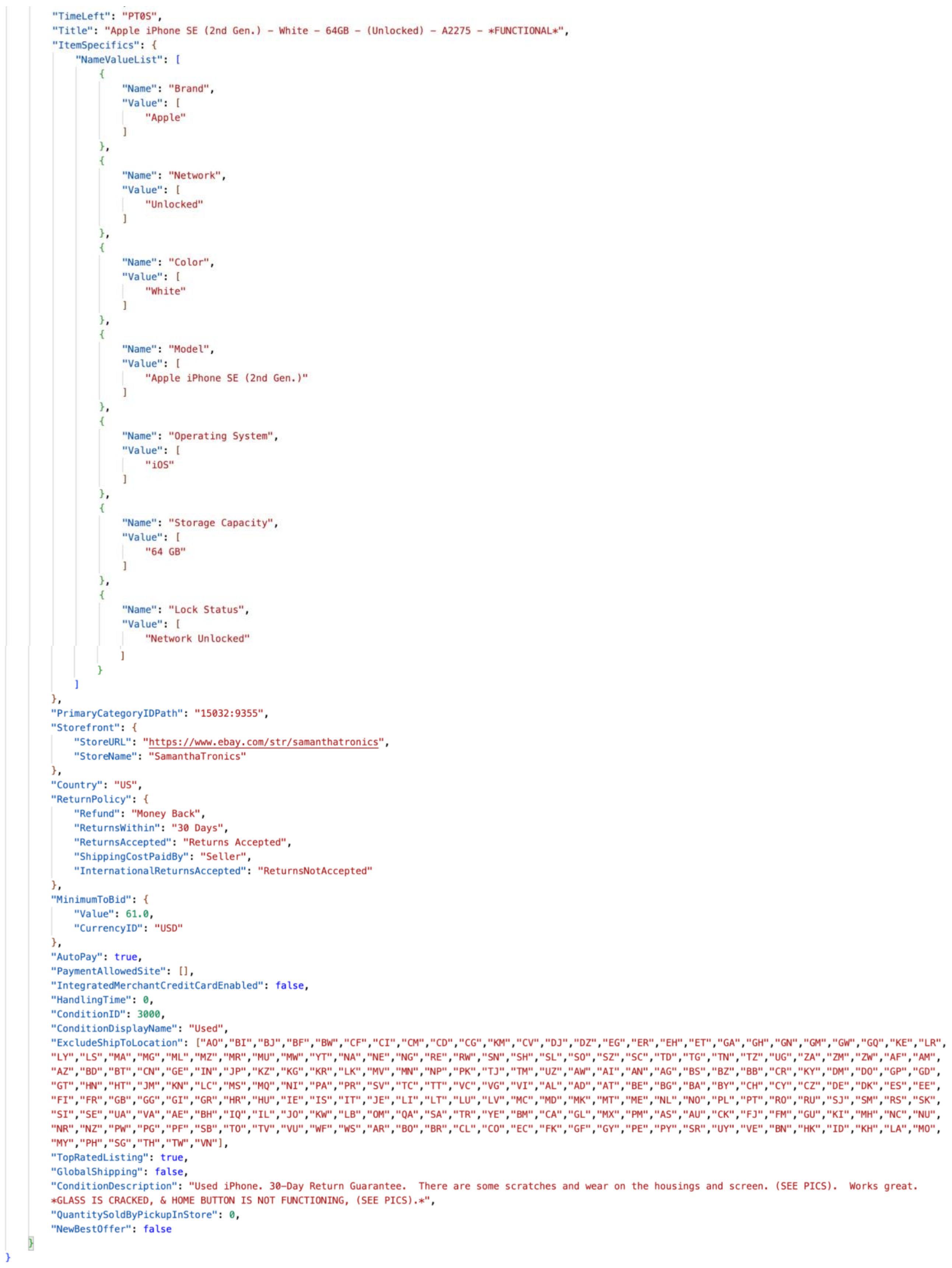
More details - https://developer.ebay.com/devzone/shopping/docs/callref/getsingleitem.html

An example of an HTTP request for single item using the *GetSingleItem* API is shown below:

http://open.api.ebay.com/shopping?callname=GetSingleItem&responseencoding=JSON&appid=[APPID]&siteid=0&version=967&ItemID=[ITEMID]&IncludeSelector=Description,Details,ItemSpecifics

Figure 12 shows an example of the JSON Response returned by the GetSingleItem API.

```
{
    "Timestamp": "2023-09-05T01:29:17.685Z",
    "Ack": "Success",
    "Build": "1299_CORE_APILW_19146596_R1",
    "Version": "1299",
    "Item": {
        "BestOfferEnabled": false,
        "BuyItNowPrice": {
            "Value": 104.99,
            "CurrencyID": "USD"
        },
        "Description": "<div id=\"ds_div\"><div id=\"ds_div\"><div id=\"ds_div\"><div id=\"ds_div\">\n\t\t\t\t\t\t\t<span
        id=\"docs-internal-guid-6d2d306f-7fff-ef74-9f9c-96444dcf357f\"><div id=\"ds_div\">\n\t\t\t\t\t\t\t<span
        id=\"docs-internal-guid-6d2d306f-7fff-ef74-9f9c-96444dcf357f\"><p style=\"margin: 0px; font-stretch: normal; line-height: normal;\"><font size=\"4\"
        face=\"Arial\">Used iPhone. 30-Day Return Guarantee.&nbsp;&nbsp;There are some scratches and wear on the housings and screen. (SEE PICS).&nbsp;&nbsp;Works
        great.&nbsp;&nbsp;Clear ESN/MEID/IMEI to activate, no iCloud lock. Charger not included, but you might have one already.&nbsp;&nbsp;This one uses the
        standard lightning charger. If you do not have one you can get one here on eBay for under $5. Payment due within 3 days of auction close or you agree to
        cancel order and item will be relisted.<\/font><\/p><div><font size=\"4\" face=\"Arial\"><b><br><\/b><\/font><\/div><div><font face=\"Arial\" size=\"5\"><b
        style=\"\">*GLASS IS CRACKED, &amp; HOME BUTTON IS NOT FUNCTIONING, (SEE PICS).*<\/b><\/font><\/div><div><font size=\"4\" face=\"Arial\"><br><\/font><\/
        div><div><font size=\"4\" face=\"Arial\">Shipping\n Policy: I offer free shipping (to USA addresses only) on most of the items I sell. I pay for \nthe
        shipping fees so to save costs I will always do combined shipping \nfor multiple wins paid for within 24 hours.<\/font><\/div><p dir=\"ltr\"
        style=\"line-height: 1.38; margin-top: 0pt; margin-bottom: 0pt;\"><font size=\"4\" face=\"Arial\"><b><br><\/b><\/font><\/p><p dir=\"ltr\"
        style=\"line-height: 1.38; margin-top: 0pt; margin-bottom: 0pt;\"><font size=\"4\" face=\"Arial\"><b>IMEI: 356848112284015<\/b><br><\/font><br><\/p><\/
        span><\/div><\/span><\/div><p style=\"margin: 0px; font-stretch: normal;\">iPhone Charger available <a href=\"https://www.ebay.com/itm/334736808459\">
        HERE<\/a><\/p><\/div><\/div><div id=\"desc\" style=\"font-family: Arial, Helvetica, sans-serif;\"><div><p><\/p><\/div><\/div><span style=\"color: red;
        \"><strong><\/strong><\/span><\/div><div id=\"ds_div\"><font style=\"font-size: 14pt; font-family: Arial;\"><table style=\"color: rgb(0, 0, 0);
        border-spacing: 0px; width: 1379px;\" align=\"center\"><\/table><\/font><table style=\"font-family: Arial; color: rgb(0, 0, 0); border-spacing: 0px; width:
        1379px;\" align=\"center\"><\/table><\/div><div id=\"desc\" style=\"font-family: Arial, Helvetica, sans-serif;\"><div><p><\/p><\/div><\/div><span
        style=\"color: red;\"><strong><\/strong><\/span><div id=\"ds_div\"><p style=\"margin: 0px; font-stretch: normal;\"><br><\/p><\/div><div id=\"desc\"
        style=\"font-family: Arial, Helvetica, sans-serif;\"><p><strong><br><\/strong><\/p><p><strong><br><\/strong><\/p><p><strong><br><\/strong><\/
        p><p><strong><br><\/strong><\/p><p><br><\/p><p><\/p><\/div><span style=\"color: red;\"><strong><\/strong><\/span><p style=\"margin: 0px; font-stretch:
        normal;\"><br><\/p><font style=\"font-size: 14pt; font-family: Arial;\"><table style=\"color: rgb(0, 0, 0); border-spacing: 0px; width: 1379px;\"
        align=\"center\"><\/table><\/font><table style=\"font-family: Arial; color: rgb(0, 0, 0); border-spacing: 0px; width: 1379px;\" align=\"center\"><\/
        table><div id=\"ds_div\"><div id=\"ds_div\"><div id=\"ds_div\"><div id=\"ds_div\"><div id=\"ds_div\"><div id=\"desc\"><div><strong style=\"font-family:
        Arial, Helvetica, sans-serif;\"><\/strong><\/div><\/div><\/div><\/div><\/div><\/div><font style=\"\" face=\"Arial\" size=\"4\" color=\"#9e2048\"><i><b>no<\/
        b><\/i><\/font><font style=\"font-size:14pt; font-family:Arial;\"><i><font color=\"#9e2048\"><b>a<\/b><\/font><\/i><br><table style=\"color:rgb(0, 0, 0);
        border-spacing:0px; width:1379px;\" align=\"center\"><tbody><\/tbody><\/table><\/font><\/div><div id=\"desc\" style=\"font-family: Arial, Helvetica,
        sans-serif;\"><div><span><p><\/p><\/span><\/div><\/div><span style=\"color: red;\"><strong><\/strong><\/span>",
        "ItemID": "335007021375",
        "BuyItNowAvailable": true,
        "ConvertedBuyItNowPrice": {
            "Value": 104.99,
            "CurrencyID": "USD"
        },
        "EndTime": "2023-09-04T16:31:11.000Z",
        "StartTime": "2023-08-28T16:31:11.000Z",
        "ViewItemURLForNaturalSearch": "https://www.ebay.com/itm/Apple-iPhone-SE-2nd-Gen-White-64GB-Unlocked-A2275-FUNCTIONAL-/335007021375",
        "ListingType": "Chinese",
        "Location": "Boise, Idaho",
        "PaymentMethods": [],
        "PictureURL": [
            "https://i.ebayimg.com/00/s/MTYwMFgxMjAw/z/azEAAOSwJFRk3YBU/$_57.JPG?set_id=880000500F",
            "https://i.ebayimg.com/00/s/MTYwMFgxMjAw/z/ugwAAOSwu8lk3YBU/$_57.JPG?set_id=880000500F",
            "https://i.ebayimg.com/00/s/MTYwMFgxMjAw/z/C28AAOSwGPpk3YBU/$_57.JPG?set_id=880000500F",
            "https://i.ebayimg.com/00/s/MTYwMFgxMjAw/z/txMAAOSwPrJk3YBU/$_57.JPG?set_id=880000500F",
            "https://i.ebayimg.com/00/s/MTYwMFgxMjAw/z/6J0AAOSwrkJk3YBU/$_57.JPG?set_id=880000500F",
            "https://i.ebayimg.com/00/s/MTE1MVgxNjAw/z/d3MAAOSwA9hk3YBV/$_57.JPG?set_id=880000500F"
        ],
        "PostalCode": "837**",
        "PrimaryCategoryID": "9355",
        "PrimaryCategoryName": "Cell Phones & Accessories:Cell Phones & Smartphones",
        "Quantity": 1,
        "Seller": {
            "UserID": "samanthatronics",
            "FeedbackRatingStar": "YellowShooting",
            "FeedbackScore": 15224,
            "PositiveFeedbackPercent": 99.4,
            "TopRatedSeller": true
        },
        "BidCount": 29,
        "ConvertedCurrentPrice": {
            "Value": 60.0,
            "CurrencyID": "USD"
        },
        "CurrentPrice": {
            "Value": 60.0,
            "CurrencyID": "USD"
        },
        "HighBidder": {
            "UserID": "s***h",
            "FeedbackPrivate": false,
            "FeedbackRatingStar": "Blue",
            "FeedbackScore": 56
        },
        "ListingStatus": "Completed",
        "QuantitySold": 1,
        "ShipToLocations": [
            "US"
        ],
        "Site": "US",
```



```
        "TimeLeft": "PT0S",
        "Title": "Apple iPhone SE (2nd Gen.) - White - 64GB - (Unlocked) - A2275 - *FUNCTIONAL*",
        "ItemSpecifics": {
            "NameValueList": [
                {
                    "Name": "Brand",
                    "Value": [
                        "Apple"
                    ]
                },
                {
                    "Name": "Network",
                    "Value": [
                        "Unlocked"
                    ]
                },
                {
                    "Name": "Color",
                    "Value": [
                        "White"
                    ]
                },
                {
                    "Name": "Model",
                    "Value": [
                        "Apple iPhone SE (2nd Gen.)"
                    ]
                },
                {
                    "Name": "Operating System",
                    "Value": [
                        "iOS"
                    ]
                },
                {
                    "Name": "Storage Capacity",
                    "Value": [
                        "64 GB"
                    ]
                },
                {
                    "Name": "Lock Status",
                    "Value": [
                        "Network Unlocked"
                    ]
                }
            ]
        },
        "PrimaryCategoryIDPath": "15032:9355",
        "Storefront": {
            "StoreURL": "https://www.ebay.com/str/samanthatronics",
            "StoreName": "SamanthaTronics"
        },
        "Country": "US",
        "ReturnPolicy": {
            "Refund": "Money Back",
            "ReturnsWithin": "30 Days",
            "ReturnsAccepted": "Returns Accepted",
            "ShippingCostPaidBy": "Seller",
            "InternationalReturnsAccepted": "ReturnsNotAccepted"
        },
        "MinimumToBid": {
            "Value": 61.0,
            "CurrencyID": "USD"
        },
        "AutoPay": true,
        "PaymentAllowedSite": [],
        "IntegratedMerchantCreditCardEnabled": false,
        "HandlingTime": 0,
        "ConditionID": 3000,
        "ConditionDisplayName": "Used",
        "ExcludeShipToLocation": ["AO","BI","BJ","BF","BW","CF","CI","CM","CD","CG","KM","CV","DJ","DZ","EG","ER","EH","ET","GA","GH","GN","GM","GW","GQ","KE","LR",
        "LY","LS","MA","MG","ML","MZ","MR","MU","MW","YT","NA","NE","NG","RE","RW","SN","SH","SL","SO","SZ","SC","TD","TG","TN","TZ","UG","ZA","ZM","ZW","AF","AM",
        "AZ","BD","BT","CN","GE","IN","JP","KZ","KG","KR","LK","MV","MN","NP","PK","TJ","TM","UZ","AW","AI","AN","AG","BS","BZ","BB","CR","KY","DM","DO","GP","GD",
        "GT","HN","HT","JM","KN","LC","MS","MQ","NI","PA","PR","SV","TC","TT","VC","VG","VI","AL","AD","AT","BE","BG","BA","BY","CH","CY","CZ","DE","DK","ES","EE",
        "FI","FR","GB","GG","GI","GR","HR","HU","IE","IS","IT","JE","LI","LT","LU","LV","MC","MD","MK","MT","ME","NL","NO","PL","PT","RO","RU","SJ","SM","RS","SK",
        "SI","SE","UA","VA","AE","BH","IQ","IL","JO","KW","LB","OM","QA","SA","TR","YE","BM","CA","GL","MX","PM","AS","AU","CK","FJ","FM","GU","KI","MH","NC","NU",
        "NR","NZ","PW","PG","PF","SB","TO","TV","VU","WF","WS","AR","BO","BR","CL","CO","EC","FK","GF","GY","PE","PY","SR","UY","VE","BN","HK","ID","KH","LA","MO",
        "MY","PH","SG","TH","TW","VN"],
        "TopRatedListing": true,
        "GlobalShipping": false,
        "ConditionDescription": "Used iPhone. 30-Day Return Guarantee.  There are some scratches and wear on the housings and screen. (SEE PICS).  Works great.
        *GLASS IS CRACKED, & HOME BUTTON IS NOT FUNCTIONING, (SEE PICS).*",
        "QuantitySoldByPickupInStore": 0,
        "NewBestOffer": false
    }
}
```

Figure 12: JSON Response returned by the GetSingleItem API

4. Hints

- For Flask/Python Backend, refer to:
  - Setting up a server - https://flask.palletsprojects.com/en/2.3.x/
  - Making requests from Python to the eBay APIs, use the Requests module - https://requests.readthedocs.io/en/master/
- Styling hints
  - Button styling - https://www.w3schools.com/css/css3_buttons.asp
  - Shadow on a card - https://www.w3schools.com/howto/tryit.asp?filename=tryhow_css_cards2
  - For close button - https://wesbos.com/times-html-entity-close-button
- Flask/Python
  1. Flask should be used to implement RESTful services to eBay
  2. Flask templates should not be used
  3. Flask send_static_file() or send_from_directory() should be used to send "static" HTML, CSS and JavaScript
  4. Helpful page to serve static page with python: https://stackoverflow.com/questions/20646822/how-to-serve-static-files-in-flask
- GCP, AWS, Azure
  1. Follow the steps mentioned in *Cloud Setup (Python)* to deploy your web application on GCP, AWS or Azure.

5. Images

eBay Logo: https://www.csci571.com/hw/hw6/images/eBayLogo.png
Top Rate Image: https://www.csci571.com/hw/hw6/images/topRatedImage.png
Item Default Image*:* https://www.csci571.com/hw/hw6/images/ebay_default.jpg
Redirect: https://www.csci571.com/hw/hw6/images/redirect.png

5. Files to Submit

In your course homework page, you should update the **Assignment 2** link to refer to your new initial search page for this exercise (for example, **ebay.html**). Your files must be hosted on GCP, AWS, or the Azure cloud service. Graders will verify that this link is indeed pointing to a cloud service.

Also, submit your source code files to DEN D2L. Submit a **single ZIP file** containing both front-end and back-end code, in separate folders, plus any additional files needed to build your app (e.g., yaml file).

****IMPORTANT**:**

- All discussions and explanations in Piazza related to this homework are part of the homework description and grading guidelines. So please review all Piazza threads, before finishing the assignment. If there is a clarification on Piazza that conflicts with this description and/or the grading guidelines, **Piazza always rules**. In most cases, the clarification is related to an error in the description, or missing information from the description, but **<u>no additional functionality</u>**.
- You can use jQuery for this assignment, but its use is not required.
- You can use FaaS like *Google Cloud Functions*, *Amazon Lambda* and *Azure Functions*, in lieu of building a monolithic application.
- You **should not call any of the eBay APIs directly from JavaScript**, bypassing the Python proxy. Implementing any one of them in JavaScript instead of Python will result in a **4-point penalty.** Other APIs can be called from JavaScript.