Developing a Graphical User Interface to Organize Ecommerce Product Attributes

Literature Review

- Retailers offer various “deals”, attractive discounts on commodities to incentivize consumers
  - Ecommerce industry boomed due to pandemic
    - Upwards of 15% of total US retail sales (Monterio)
  - A new norm of ecommerce and technology leads to worsening economic inequality (Anderson, Janna, et al.)
- Pricing is one of the most important factors in determining consumer purchases among ecommerce retailers (Rajani and Nakhat)
  - Reviews are also important factors
    - Many online retailers are not descriptive enough
    - Review systems often have both false positive and negative reviews, which misleads consumers (Kolhar)
- The process of searching for deals remains tedious
  - Existing forum communities, such as Slickdeals, rely heavily on user contributions
    - “Users share, confirm, and comment on deals so that everyone benefits.” (*Slickdeals)*
      - “People make great deals possible”
      - “[Deals are] sourced by real people”
- Tkinter is a toolkit built into Python used to create Graphical User Interfaces (GUI)

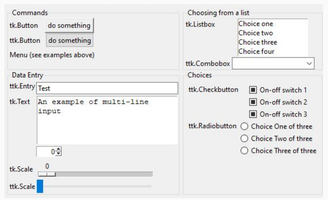


***Figure 1:** Example of the various widgets available in Tkinter as shown on a GUI.*

- eBay offers various Application Programming Interfaces (API) to externally access product attributes
  - Characteristics that describe a product, such as price
- eBay Developer Program
  - Free for individual developers with extensive APIs
  - Generous API call limits
    - Frequency of updating product information
  - Ecommerce retailer with wide range of products
- Other retail APIs are restrictive or have unreasonable costs for individuals long-term (*Amazon Web Services)*

Problem Statement

The purpose of this project was to develop a graphical user interface using the Python programming language in order to organize and display ecommerce product attributes.

Past Accomplishments

This project was a continuation from a previous year’s research. In past research, a Python program was created to extract product attributes from the eBay API given a search query. API results were displayed a Python dictionary and stored in a MySQL database. This program functioned successfully in communicating with the eBay API and MySQL database. It served as the backend program, a program that creates behind-the-scenes functionality, in conjunction with the currently developed frontend GUI.

Criteria

- Used binary project evaluation to track progress
  - Did an iteration meet the criterion? (Yes/No)
- Prompted user input? Store input to be used for API call?
- Command called when search button was clicked?
- Established communication (input/output) with API?
- Transferred API output into treeview widget?
- Treeview was displayed with all requested information?
- Established communication with MySQL database?
- Stored product attributes into database?

Experimental Design

1) Background Research
- Python programming to create GUIs and integrate APIs
  - Research on required Python Libraries
    - Tkinter (GUI) and PyMySQL (database communication)
  - Used online resources and documentations

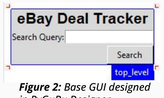


***Figure 2:** Base GUI designed in PyGuBu Designer*

2) Design and Preparation
- Used a Python GUI Builder (PyGuBu) for Tkinter
  - Designed and customized a base GUI (*Figures 2-4*)
  - Exported to raw code for further development

3) Development
- Python Programming language
  - Used Tkinter widgets to add features and functionality
    - On-button-clicked commands
    - Stored product attributes in MySQL database
    - Tkinter treeview widget
      - Widget to display data in a tabular structure
      - Selection of specific rows and values
- Application Testing
  - Repetitively and thoroughly tested the GUI
  - Ensure most, if not all, criteria were met

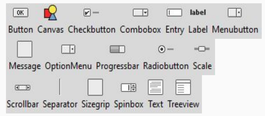


***Figure 3:** The various widgets available in Tkinter (Tkinter.ttk)*

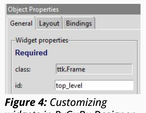


***Figure 4:** Customizing widgets in PyGuBu Designer*

4) Backend Integration - Pulling Information from eBay API
- Previously developed Python program
  - Used eBay Finding API to pull product attributes given a search query
    - eBay Python Software Development Kit (SDK) to communicate with API (*Figure 5*)
- Integrated via an on-button-clicked command

```
# Returns eBay API results
def search(keyword):
    api = Finding(domain='svcs.sandbox.ebay.com', appid=key, config_file=None)
    out = api.execute('findItemsAdvanced', {'keywords': keyword})
    return out.dict()
```

***Figure 5:** Defined function using eBay Python SDK to call eBay Finding API and return results as a Python dictionary.*

5) Evaluation and Reiteration
- Determined whether criteria were met, if not already
  - Binary definitions of success
- Determined usability and scarcity of bugs (problems)
  - Possible quality-of-life improvements
- Further development, improvement, and evaluation
  - Changes in code tracked using Git version control

6) Discussion with Mentor
- Mentor led discussion on student’s progress
  - Mentor provided insight and ideas for improvement
  - Updated criteria and methods if needed
- Further reiteration sparked from mentor discussions

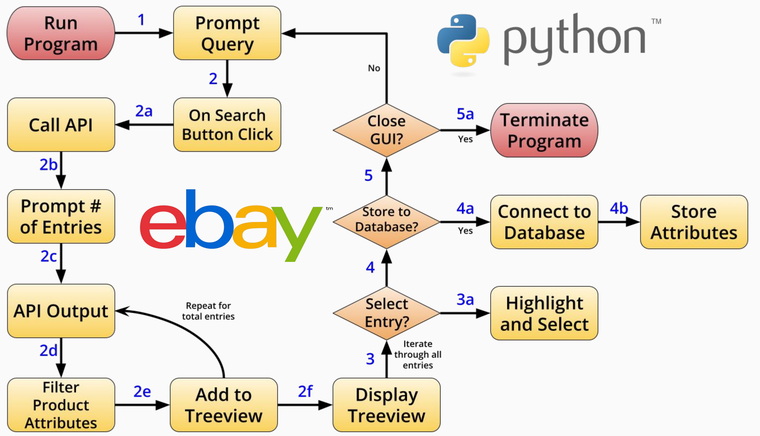
GUI Process Flowchart

1) User enters search query in text box (*Figure 6*)
2) Search button clicked (*Figure 7*)
- **a)** Calls eBay Finding API
- **b)** Prompts user to specify number of entries to display, out of the total found (*Figure 8*)
- **c)** API output stored as a Python dictionary
- **d)** Product attributes are filtered to the item name, item ID, price, and product link
- **e)** Filtered product attributes added to Treeview
- **f)** Treeview widget is displayed (*Figure 9*)

3) User can select specific entries in treeview
- **a)** Entries are highlighted and can be copied

4) Select entries can be stored to MySQL database
- **a)** Connect and and login to database
- **b)** Copy and store data to database

5) User can continue or terminate the program by closing the GUI
- **a)** Program terminated by user



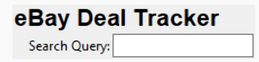


***Figure 6:** Initial GUI menu prompting the user to enter a search query in a textbox*

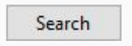


***Figure 7:** Tkinter button widget to call Python search function (Figure 5)*

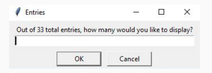


***Figure 8:** Gives the total number of entries found for a search query. Prompts the user to specify how many to display*

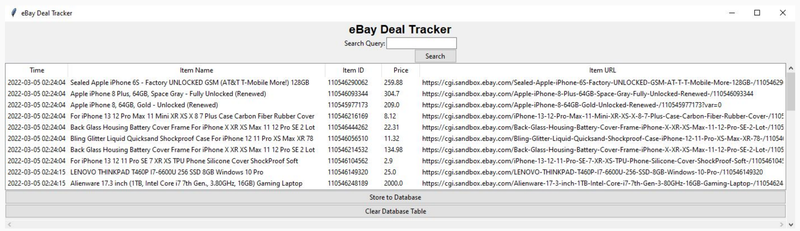


***Figure 9:** The entire GUI of the eBay Deal Tracker, which contains one “top_level” frame and many sub-level container frames for groups of widgets. User events include searching with a given query, terminating the program by closing the GUI, storing specific rows to the MySQL database, or clearing the database table.*

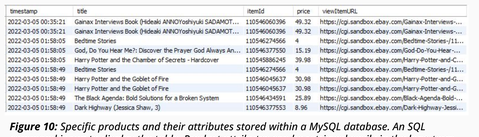


***Figure 10:** Specific products and their attributes stored within a MySQL database. An SQL command is run to display the table. Product attributes can be retrieved easily in the long-term*

Conclusion

- The GUI successfully displayed an interactive window
  - Successfully prompted and stored search query
- Previously developed backend program required minimal changes and functioned successfully
- Exceptions were handled successfully in any user inputs

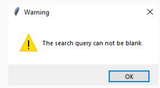


***Figure 11:** Warning displayed when user attempts to execute a blank search query (Figures 7-8)*

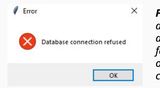


***Figure 12:** Error displayed MySQL database connection failed upon on-button-click command (Figure 9)*

- API output was neatly displayed in a treeview widget
  - Handled user interactions successfully
- eBay’s developer program limits access to production API
  - Used eBay Sandbox, which did not reflect products on the production website in real time
- Finding API has a call limit of 5000 calls per day
  - Would restrict how often prices can be updated, especially with various unique search queries
- Finding API has inaccuracies on eBay’s end
  - Could not distinguish “auction” and “buy-it-now” listings
    - Collection of average prices skewed
  - API retrieved other products containing query
    - Ex. searching “iphone” also gave iphone accessories

Future Research

- Further reiterations to both GUI and backend program to add features and fix bugs
  - Context menus and sub-menus (on-mouse-click)
  - Multiple frames to display more information or run multiple instances

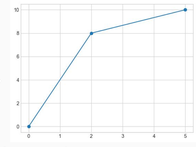


***Figure 13:** Warning displayed when user attempts to execute a blank search query (Figures 7-8)*

  - Matplotlib library to visually graph data (*Figure 13)*
    - Graphing historical price over time for products
- Export the GUI and backend functionality to a more widely accessible medium, such as a website
  - Ability to allow other individuals to test program
    - Quality assurance of good user experience
    - End user feedback to inspires further reiterations

Implications

- Providing deals to a greater audience with minimal hassle
  - Potentially saving money for the average consumer
- Adding automation to existing methods of searching, sorting, and sharing deals
- Helping consumers adapt to a new norm of ecommerce with widespread use of deals
  - “Fake deals” occur when products are marked-up and discounted back to their original price (Passy)
- Organization of product attributes can benefit the retailer
  - Retailers can catalog their Stock Keeping Units (SKUs)

References

- "Amazon API Gateway Pricing." *Amazon Web Services*, aws.amazon.com/api-gateway/pricing/.
- Anderson, Janna, et al. "Experts Say the 'New Normal' in 2025 Will Be Far More Tech-Driven, Presenting More Big Challenges." *Pew Research Center*, 18 Feb. 2021,www.pewresearch.org/internet/2021/02/18/experts-say-the-new-normal-in-2025-will-be-far-more-tech-driven-presenting-more-big-challenges/.
- *eBay Developers Program*. developer.ebay.com/.
- Kolhar, Manjur. "E-Commerce Review System to Detect False Reviews." Science & Engineering Ethics, vol. 24, no. 5, Oct. 2018, p.1577.EBSCOhost, doi:10.1007/s11948-017-9959-2.
- Miller, Brad, and David Ranum. How to Think Like a Computer Scientist: Interactive Edition.
- Monterio, Ana. "Covid E-Commerce Boom Sees U.S. Retailers Hunt for Warehouses." *Bloomberg*, 11 Jan. 2022, www.bloomberg.com/news/newsletters/2022-01-11/supply-chain-latest-covid-e-commerce-boom-sees-warehouse-demand-soar.
- Passy, Jacob. "Fake discounts in stores fool some shoppers more than others—here's why." *MarketWatch*, 20 Oct. 2018, www.marketwatch.com/story/fake-discounts-in-stores-fool-some-shoppers-more-than-othersheres-why-2018-08-24.
- Rajani, Vishal Tushar, and Preeti Nakhat. "Consumer Behavior in Online Shopping: What They Think Before They Buy." Journal of Psychosocial Research, vol. 14, no. 2, July 2019, pp. 377–384. EBSCOhost, doi:10.32381/JPR.2019.14.02.15.
- *Slickdeals*. slickdeals.net/corp/about-us.html.
- *TkDocs*. tkdocs.com/about.html.
- "Tkinter." *Python Documentation*, docs.python.org/3/library/tkinter.html.