Shoe Collection Manager Documentation

Joshua Joji: 240328553

Initiation and Design:

Introduction:

The Shoe Collection Manager is a web-based tool to organise, visualise and manage a collection of shoes. The application allows users to filter, add, edit, and view detailed information about individual shoes and visualise trends with the interactive dashboard.

Key Features:

- View entire collection or filtered subsets with corresponding picture of shoe.
- Add new shoes to the collection.
- Edit details of the existing shoes.
- Visualise data through the charts and pivot table.
- Simple, intuitive user interface.

Technologies:

The Shoe collection manager was developed using Python as the primary programming language for its robust and versatile capabilities. Libraries such as Pandas were used for efficient data handling and analysis, while Plotly was utilised for creating interactive and visually appealing data visualisations. The user interface was built using ipywidgets, allowing for intuitive and dynamic dropdowns and selections. Additionally, Mermaid was used for generating charts and diagrams to illustrate the system's architecture and relationships. To design the application's layout and interface, Figma was used, enabling a well-thought-out and user-friendly design.

Use of Object-Oriented Programming:

I have used 4 classes in this project which includes ShoeCollectionManager, ShoeDetailEditor, ShoeAdder, ShoeDashboard. The ShoeCollectionManager class manages the main functionality of the displaying and filtering the shoe collection. This class also abstracts the logic for viewing and filtering shoes, hiding the implementation details from the user. Users interact with dropdowns and see results without seeing how filters applied. Users can edit details of individual shoes already in the collection, without interacting directly with the DataFrame using the ShoeDetailEditor. If users want to add additional shoes into the collection, the ShoeAdder class allows them to, without showing the process, which includes saving images and appending data to the collection. The Shoe Dashboard provides visualisations of the collection to users, for example to see how many shoes they have of each brand, without exposing the underlying Plotly and Pandas logic. Each class is responsible for a single task so they can operate independently while still contributing to the overall functionality. For example, the ShoeDashboard can generate visualisations without relying on ShoeAdder or ShoeDetailEditor.

Furthermore, these classes can be extended in the future. The ShoeCollectionManager can be extended to add more filters or additional data sources.

Requirements:

The application interface should be intuitive and easy to navigate. It should be accessible to users with different levels of technical expertise, providing clear feedback. The application should handle moderate-sized datasets without significant delays. It should also run on multiple platforms through Jupyter Notebook and Voila. The application should be accessible and responsive to user's needs. It must allow user to view the entire collection and filter by Brand, Colour, Size, Category. It should enable multi-level filtering, for example, selecting a brand and then filter by size or colour, through a dropdown-based interface. The application must allow users to add new shoes by providing details like Brand, Model, Size, Colour, Category and include an option to upload image of shoe and save to designated folder. The application should also enable users to select an existing shoe from the collection and edit its attributes, where the changes are reflected immediately in the DataFrame. It should also provide summary statistics including a pie chart and pivot table, which should update dynamically as the data changes. Finally, the shoe collection should load data from the Excel File at startup and save any changes.

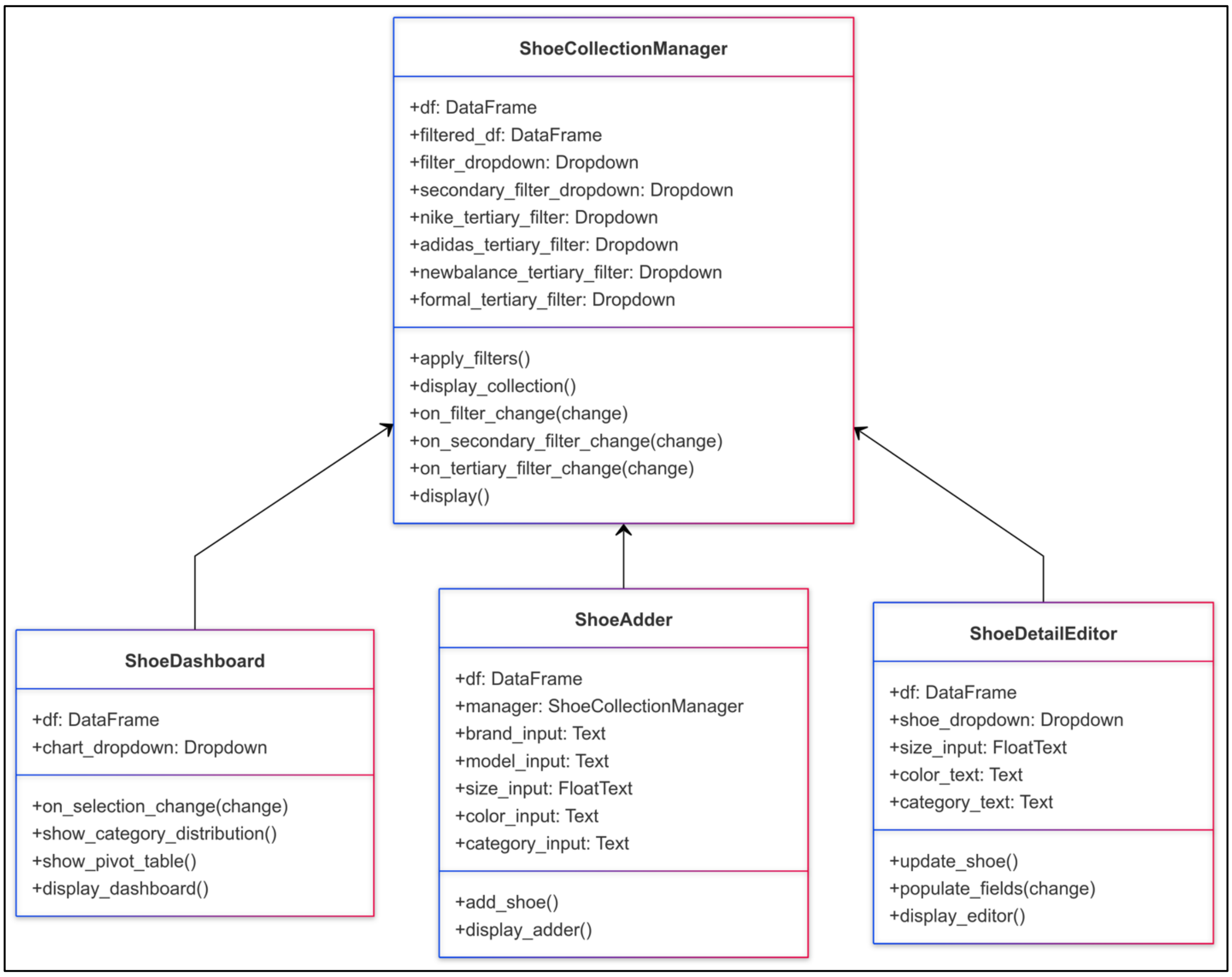
Design and Architecture:

User Personas:

1. **Joshua (Primary User):**
   A 19-year-old sneakerhead who collects shoes as a hobby. He wants to maintain a record of his collections and visualise trends. He needs a simple, intuitive interface with powerful filtering and visualisation options.
2. **Bob (Shoe Store Manager):**
   A 30-year-old manager who wants to manage his inventory in his shoe store. He wants to be able to store all his stock on the collection manager, track trends, and make data-driven decisions. He needs a robust dashboard with statistics and category-wise breakdowns.
3. **Christina (Casual User):**
   A 70-year-old woman who enjoys keeping a small collection of shoes organized but struggles with memory loss due to dementia. She often forgets which shoes she owns and their specific details, so she needs a tool that helps her keep track of her collection. The Collection Manager should be designed with simplicity and accessibility in mind, offering a minimal setup and an easy-to-navigate interface. Key features like adding, editing, and viewing shoe details should be intuitive and straightforward. Quick access to essential filters and a clear, clutter-free layout is needed.

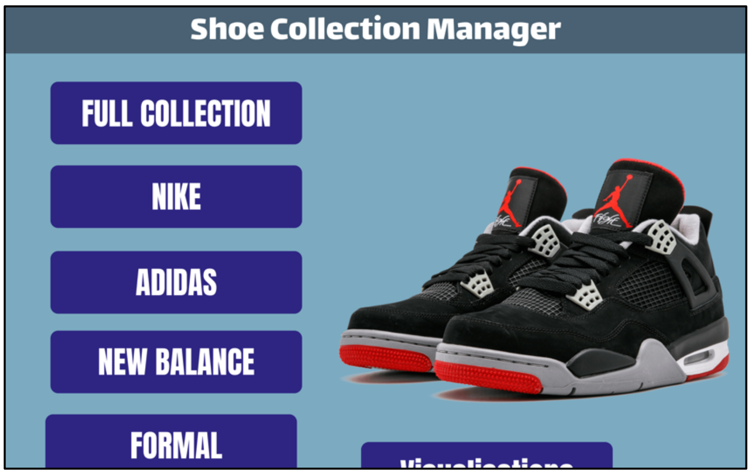
System Design:

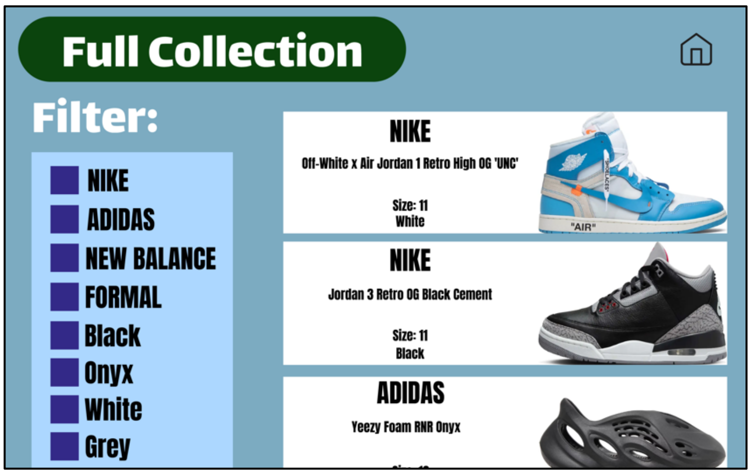
Unified Modelling Language Class Diagram using Mermaid:



User Interface:

I created a prototype user interface design using Figma: Joshua's Shoe Collection Mock UI, to visualize how the Shoe Collection Manager application would look and function. This design aims to provide a user-friendly, visually appealing layout that caters to various user needs, including filtering, viewing, and interacting with the shoe collection.

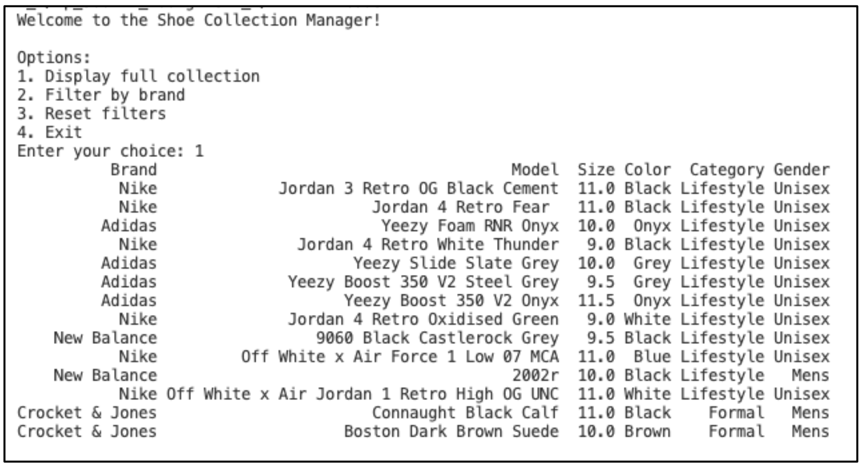




Implementation

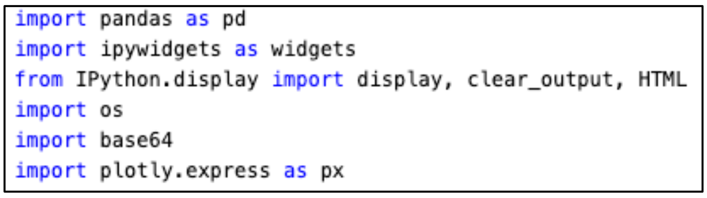
Initial Application

This is the initial application, built to run in a command-line interface. It allows for basic filtering but not a very user-friendly and accessible design. The interface provides text-based interaction for managing the shoe collection and provides the core features needed. However, it does not allow for additional filters like size and colour. Users are not able to add or update shoe details directly from the interface. Furthermore, this does not provide visualisations to users to gather details about their collection. It provides a basic CRUD workflow.



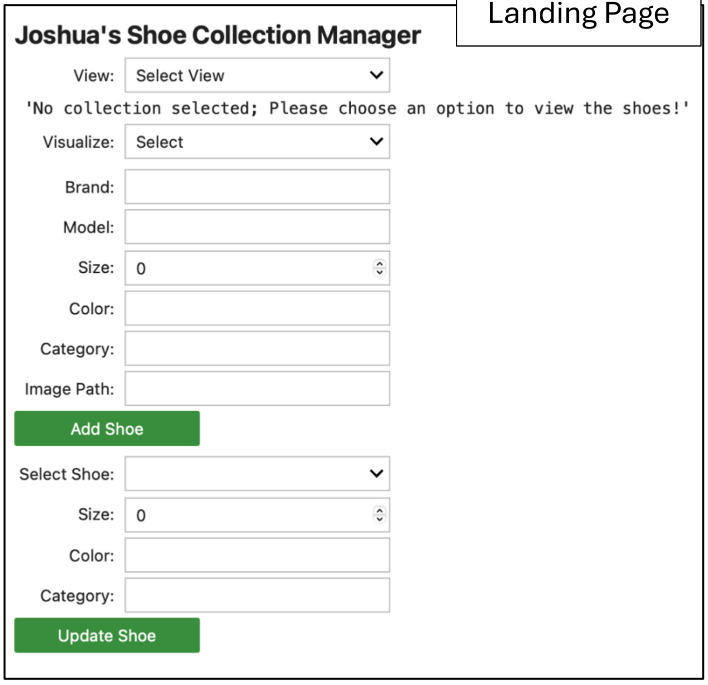
```
Welcome to the Shoe Collection Manager!

Options:
1. Display full collection
2. Filter by brand
3. Reset filters
4. Exit
Enter your choice: 1
             Brand                                  Model  Size Color  Category Gender
              Nike              Jordan 3 Retro OG Black Cement  11.0 Black Lifestyle Unisex
              Nike                        Jordan 4 Retro Fear  11.0 Black Lifestyle Unisex
            Adidas                        Yeezy Foam RNR Onyx  10.0  Onyx Lifestyle Unisex
              Nike                Jordan 4 Retro White Thunder   9.0 Black Lifestyle Unisex
            Adidas                     Yeezy Slide Slate Grey  10.0  Grey Lifestyle Unisex
            Adidas                 Yeezy Boost 350 V2 Steel Grey   9.5  Grey Lifestyle Unisex
            Adidas                       Yeezy Boost 350 V2 Onyx  11.5  Onyx Lifestyle Unisex
              Nike                 Jordan 4 Retro Oxidised Green   9.0 White Lifestyle Unisex
       New Balance                 9060 Black Castlerock Grey   9.5 Black Lifestyle Unisex
              Nike            Off White x Air Force 1 Low 07 MCA  11.0  Blue Lifestyle Unisex
       New Balance                                      2002r  10.0 Black Lifestyle   Mens
              Nike Off White x Air Jordan 1 Retro High OG UNC  11.0 White Lifestyle Unisex
   Crocket & Jones                       Connaught Black Calf  11.0 Black    Formal   Mens
   Crocket & Jones                     Boston Dark Brown Suede  10.0 Brown    Formal   Mens
```

Final Application



```python
import pandas as pd
import ipywidgets as widgets
from IPython.display import display, clear_output, HTML
import os
import base64
import plotly.express as px
```

Landing Page



Joshua's Shoe Collection Manager

View: Select View

'No collection selected; Please choose an option to view the shoes!'

Visualize: Select

Brand:

Model:

Size: 0

Color:

Category:

Image Path:

Add Shoe

Select Shoe:

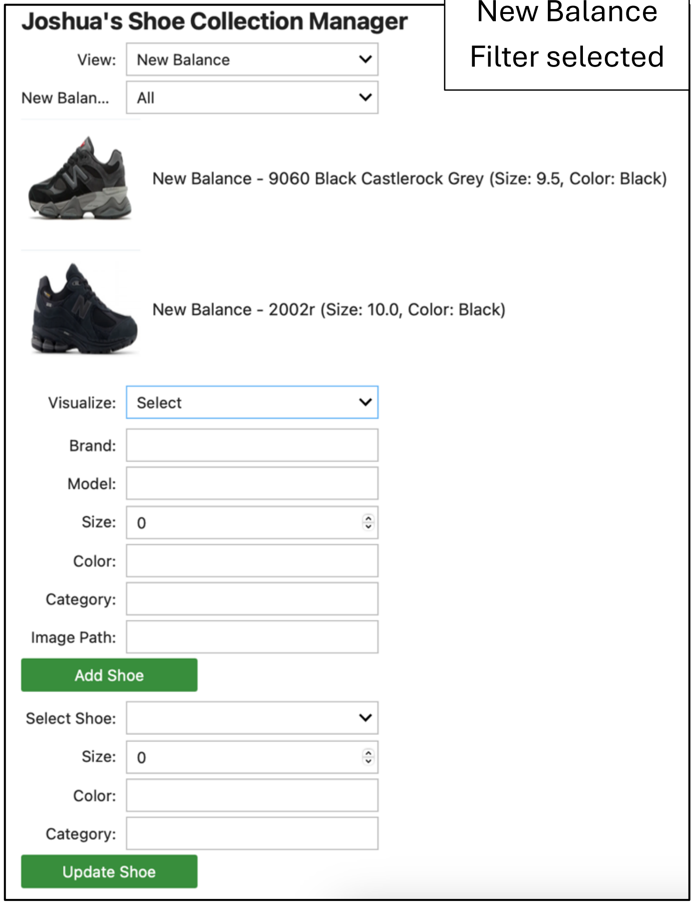
Size: 0

Color:

Category:

Update Shoe

New Balance Filter selected



Joshua's Shoe Collection Manager

View: New Balance

New Balan...: All

New Balance - 9060 Black Castlerock Grey (Size: 9.5, Color: Black)

New Balance - 2002r (Size: 10.0, Color: Black)

Visualize: Select

Brand:

Model:

Size: 0

Color:

Category:

Image Path:

Add Shoe

Select Shoe:

Size: 0

Color:

Category:

Update Shoe

This final application design was built using Python Version 3.12 and the libraries I downloaded was using the command "pip install pandas ipywidgets ipython plotly openpyxl", and these dependencies were added into the 'requirements.txt' file. Other libraries are part of Python standard library, so no installation is needed.

This final design provides a GUI-based Shoe collection manager. Using the ipywidgets library, I was able to establish core operations like displaying, filtering and resetting. These add more interactivity, visuals and features for an enhanced user experience.

Users can view between full collection or other brands. They can filter by size and colour. They can visualise the details of the collection with pivot table and pie chart, allowing them to get summary statistics. Users can also add a shoe to the collection with the image, which will update the 'Shoe Pics' folder and the 'Shoe Collection.xlsx' file. Furthermore, they can edit shoes already in the collection.

Development Process

This programme was created on Visual Studio Code, which provided the aid of Copilot to increase efficiency and streamline processes. This application was deployed on Voila as an extension to the Jupyter server, which runs on 'localhost:8866'.

<u>Reflection and Debugging:</u>

Manual Testing

I conducted this test to examine whether the shoe collection manager functions as intended and provides a seamless user experience. This involved me manually interacting with the application and testing different scenarios to validate core functionalities, edge cases and usability.

Test Case ID	Feature	Input	Expected Output	Actual Output	Result
TC001	Filter Functionality	Primary Filter: "Full Collection"	Displays all shoes in collection	Correctly displays all shoes in collection.	Pass
TC002	Filter Functionality	Secondary Filter: "Nike"	Displays all shoes with the brand "Nike".	Correctly displays only Nike shoes.	Pass
TC003	Filter Functionality	Tertiary Filter: "Size: 11"	Displays all Nike shoes with size 11.	Correctly displays Nike shoes with size 11.	Pass
TC004	Add Shoe	Brand: Nike, Model: Air Max 95, Size 11, Colour: Black	Shoe added successfully to the collection	Shoe added successfully	Pass

TC005	Edit Shoe	Jordan 4 Retro Oxidised, Size 10	Shoe details updated	Shoe updated in collection	Pass
TC006	Visualisation	Select "Pie Chart: Shoe Categories"	Pie chart displayed	Pie chart rendered successfully	Pass
TC007	Add Shoe Image	Image Path: /Users/joshuajoji/Downloads/Air Max 95.png	Image added successfully to the collection	Error saving image: PermissionError [Errno 1]	Fail

Failed Test Cases

Test Case ID: **TC007**

Feature: Add Shoe Image

Issue: A PermissionError [Errno 1] occurred when saving image file to the target directory

Recommended Fix: Update folder permissions for the Shoe Pics directory or save images to an accessible directory

User Acceptance Testing

I conducted a user acceptance test on the application, where the actual users tested it to verify if it works as intended and fulfils their requirements. This involved presenting users with specific scenarios and asking them to interact with the system while providing feedback.

User	Scenario	Steps	Expected Outcome	Feedback	Result
Joshua	Add a new shoe to the collection	Use the "Add Shoe" feature to add a new sneaker with details.	The sneaker is successfully added and appears in the full collection view.	"The process was straightforward, but I'd love a bulk upload feature for multiple shoes at once."	Pass
Bob	Bulk add shoes to inventory	Attempt to add multiple shoes using a bulk-upload feature	Multiple shoes are added successfully and visible in the collection.	"A bulk-upload feature would save a lot of time for large inventories."	Fail
Joshua	Visualise trends by category	View the "Pie Chart: Shoe Categories" in the dashboard.	A pie chart showing the distribution of shoes by category is displayed.	"The visualisation is clear and informative. Adding a bar chart comparison might be helpful."	Pass

Bob	Filter stock by brand and category	Apply primary filter "Adidas" and secondary filter "Casual."	Only Adidas casual shoes are displayed.	"Filters worked well, but a toggle for 'in stock' or 'out of stock' would be useful for inventory."	Pass
Christina	Add a shoe	Use the "Add Shoe" feature to add a new shoe to the collection with minimal details.	The shoe is successfully added and visible in the full collection view.	"Adding shoes was simple, but an option to add notes (e.g., 'for special occasions') would be great."	Pass

Future Improvements

For the future, to improve the functionality, usability, and scalability of the Shoe Collection Manager, I would make the following enhancements:

1. **Bulk Upload Feature** – To enable users to upload multiple shoes simultaneously by importing data from a CSV or Excel file. This would save time for users like Bob, who manage large inventories. Add a "Bulk upload" button and a file input option to process data from CSV/Excel file and update collection in single action.
2. **Enhanced Image Handling –** Resolve permission issues so users can upload images to specified directories. To not encounter PermissionError, I would add an optional directory setup feature.
3. **Advanced Visualisations –** Add more visualisation options like bar charts for brand comparisons or time-series analysis to track trends over time. Use Plotly for interactive and dynamic chart creation
4. **Notes field for shoes –** For users like Christina, a "Notes" feature would be beneficial to add contextual information to individual shoes.

Reflection

Overall, working on this Shoe Collection Manager project has been an extremely rewarding experience, allowing me to combine technical skills with creative problem-solving. I particularly enjoyed designing the user interface, as it required thinking from various perspective and user personas, ensuring the application met their unique needs. Implementing OOP principles to create modular classes was both challenging and satisfying, as it enhanced the structure and scalability of the application. The process of visualising the collection was another highlight, as it brought the collection to life and provided insights. Debugging and improving the application through iterative testing taught me the importance of attention to detail and the value of user feedback. This project has improved my technical abilities in Python, data handling and visualisation and has deepened my appreciation for user-centred design. It was exciting to see this application evolve from a basic command-line interface to a dynamic, interactive tool and I look forward to developing this application further.