

E-COMMERCE PRICE TRACKING SYSTEM

ABSTRACT

Online purchasing is gradually displacing traditional shopping methods in every manner. As a result of pandemic, it has gained traction and has become the new normal. From shoes to food, everything is now available on E-cart. People like to purchase online since there are so many possibilities in each area. Users can notice the price differences between websites and, as a result, the majority people will choose the service with the lowest price. So, in order to stay ahead of the competition, business minds are constantly devising new strategies in order to maximize the profit from each sale. They come up with enticing offers in order to attract more clients. In the E-commerce industry, dynamic pricing is currently the most popular method. Consumer may find it challenging to keep up with the pricing changes that occur every 10 minutes on average. They give the illusion that the goods are on sale on that particular by doing so. We offer a basic price tracker application in this paper that tracks the price of a product and notifies the customer when the price reaches the desired level. It will get the customer pricing from the browser and act in accordance with the customer's wishes.

CHAPTER 1

INTRODUCTION

Modern consumers are becoming increasingly aware of the prospects of alternative pricing schemes, which has made their buying decisions more sophisticated, thanks to significant improvements in the Electronic Commerce sector. According to recent research, internet customers are increasingly waiting for the best price offers before making purchases. This research seeks to track down prices and determine whether the offered offers are genuine savings or simply a marketing ploy to encourage sales. Many academics have proposed ideas to automate the bargaining process, but they are inaccurate. The cheapest price in history cannot be considered the best value for a product. The product's true worth fluctuates from day to day. As a result, we give the user the option of setting a price limit for the product. The user will be aware of the product's current price and will set the price based on that information. As a result, instead of waiting eternally for the lowest price, the application will know what price to wait for. Using HTML, CSS, and Bootstrap, we designed an E-cart website. Then we used Go DADDY to host it, and then we used C# and Java to construct an application that scrapes the pricing from the website and notifies the consumer.

1.1 LITERATURE SURVEY

PRICE COMPARISON WEBSITE

**HARISHCHANDRA MAURYA, KOMAL PATIL, SHREYA SAWANT, MRUDULA
THANGE, ASMITA MAHADIK**

Mobile apps have evolved to be more useful for regular use in recent years. The goal of this project is to give users an easy way to compare product availability and costs on various e-commerce websites. Users can easily compare prices from numerous sources by simply entering the product information into the program. To compare the product information found on several websites side by side, the application's databases are then searched. In order to ensure that they never miss out on a great offer, customers can also receive push notifications when things become available or go on sale.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

- Choosing the right existing system for e-commerce price tracking depends on your specific needs and budget.
- Some popular options include Price2Spy, Prisync, Competera, RepricerExpress, and Crayon.
- Take the time to explore each one, considering factors like the size of your business and the features offered.
- Reading reviews and trying out demos can help you make an informed decision based on your requirements.

2.1.1 DISADVANTAGES

- Inaccurate or outdated pricing information may lead to incorrect insights and affect users' decision-making.
- If the system is not adaptable to these changes, it may break or provide incomplete data, requiring frequent updates to maintain functionality.
- Disruptions in the scraping process can result in incomplete or delayed data updates, affecting the system's real-time capabilities and reliability.

2.2 PROPOSED SYSTEM

Maintain a centralized product database with current and historical price data. Automate price updates through APIs or web scraping. Enable user registration for personalized tracking and notifications. Implement customizable alerts for significant price changes. Provide access to historical price trends and analytics. Support multiple e-commerce platforms. Design a user-friendly interface with sorting and filtering options. Prioritize security measures and encryption. Ensure mobile compatibility for broader accessibility. Plan for regular updates to adapt to changes and add features.

2.2.1 ADVANTAGES

- The system provides automated, real-time updates on product prices, ensuring users have the most accurate and up-to-date information available. This real-time capability enhances the user's ability to make timely and informed purchasing decisions.
- User registration enables a personalized experience with customizable tracking and notification preferences. This feature allows users to tailor the system to their specific needs, receiving alerts for significant price changes or promotions that match their preferences.
- Access to historical price data allows users to analyze trends and patterns, providing valuable insights into how product prices have changed over time. This historical perspective empowers users to make informed decisions and strategically time their purchases based on past pricing trends.

2.3 ALGORITHM

APPROXIMATE TREE PATTERN MATCHING ALGORITHM

An Approach Using Swaps Cartesian tree pattern matching consists of finding all the factors of a text that have the same Cartesian tree than a given pattern. There already exist theoretical and practical solutions for the exact case. The Bitap algorithm is an approximate string matching algorithm. The algorithm indicates whether a given text contains a substring “approximately equal” to a given pattern, where approximate equality arises in terms of Levenshtein distance.

2.4 SYSTEM REQUIREMENTS

2.4.1 HARDWARE REQUIREMENTS

- ▣ **System** : Pentium i3 Processor
- ▣ **Hard Disk** : 500 GB
- ▣ **Monitor** : 15’’ LED
- ▣ **Input Devices** : Keyboard, Mouse
- ▣ **Ram** : 4 GB

2.4.2 SOFTWARE REQUIREMENTS

- ▣ **Operating system** : Windows 10.
- ▣ **Coding Language** : Python 3.8+
- ▣ **IDE** : pycharm

CHAPTER-3

MODULE DESCRIPTION

3.1 USER INPUT MODULE

The user input module refers to a component or module in a system or software application that handles and processes input provided by users. It is responsible for capturing and interpreting the data or commands entered by users through various input devices such as keyboards, mice, touch screens, or voice recognition systems.

3.2 JSON REQUEST MODULE

The JSON request module typically refers to a component or functionality in a software system that handles the sending and receiving of data using the JSON (JavaScript Object Notation) format. JSON is a lightweight data interchange format that is easy for humans to read and write, and easy for machines to parse and generate.

3.3 WEB SCRAPING MODULE

A web scraping module is a component within a software system designed to automate the extraction of data from websites. It typically involves sending HTTP requests to web servers, parsing the HTML or other structured data formats of web pages, and extracting specific information based on predefined rules. This module enables automated data gathering from online sources, commonly used in applications requiring up-to-date information from various websites.

CHAPTER-4

SYSTEM DESIGN

4.1 ARCHITECTURE DESIGN

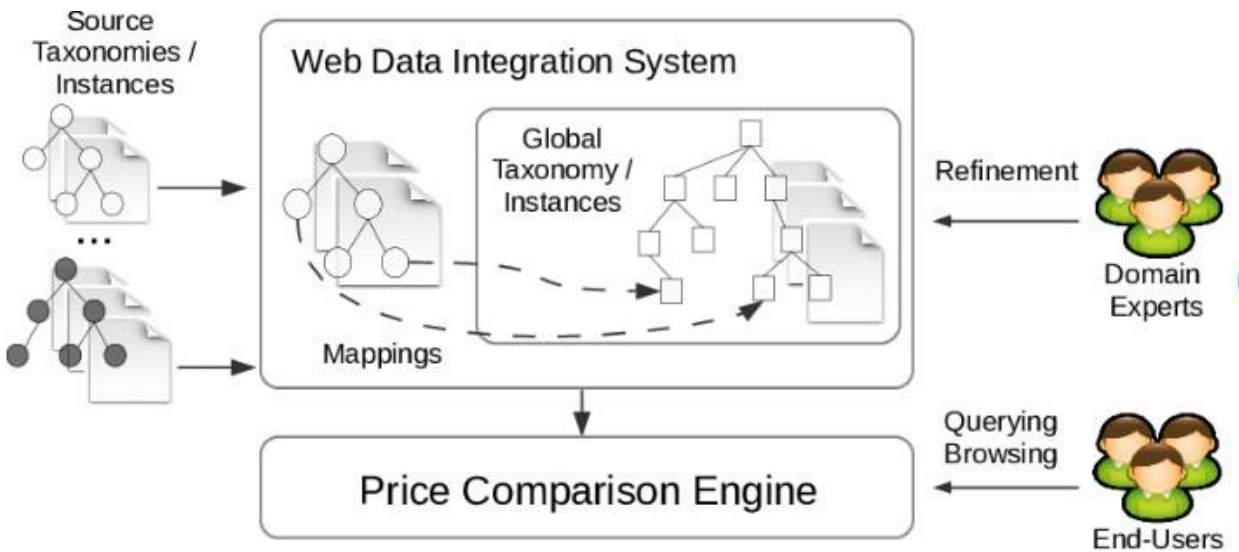


FIGURE: 1 ARCHITECTURE DESIGN

4.2 GUI INPUT DESIGN

4.2.1 SYSTEM DESIGN

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system

requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word “Quality”. Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer’s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

4.3 NORMALIZATION

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy i.e. repetition of data in the database, maintain data integrity as well as handling problems that can arise due to insertion, updation, deletion anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relation.

1. **Insertion anomaly:** Inability to add data to the database due to absence of other data.
2. **Deletion anomaly:** Unintended loss of data due to deletion of other data.
3. **Update anomaly:** Data inconsistency resulting from data redundancy and partial update
4. **Normal Forms:** These are the rules for structuring relations that eliminate anomalies.

4.3.1 FIRST NORMAL FORM:

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

4.3.2 SECOND NORMAL FORM:

A relation is said to be in second Normal form is it is in first normal form and it should satisfy any one of the following rules.

- 1) Primary key is a not a composite primary key
- 2) No non key attributes are present
- 3) Every non key attribute is fully functionally dependent on full set of primary key.

4.3.3 THIRD NORMAL FORM:

A relation is said to be in third normal form if their exits no transitive dependencies.

Transitive Dependency:

If two non key attributes depend on each other as well as on the primary key then they are said to be transitively dependent.

The above normalization principles were applied to decompose the data in multiple tables thereby making the data to be maintained in a consistent state.

4.4 E – R DIAGRAMS

- The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
- The entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.
- The set of primary components that are identified by the ERD are
 1. Data object
 2. Relationships
 3. Attributes
 4. Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships.

4.5 DATA FLOW DIAGRAMS

4.5.1 DFD DIAGRAMS

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are

developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name.

Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

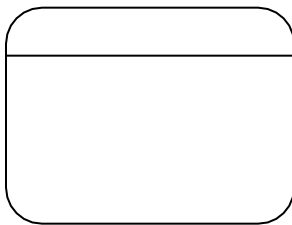
A DFD is also known as a “bubble Chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design.

So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

4.5.2 DFD SYMBOLS

In the DFD, there are four symbols

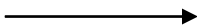
1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data



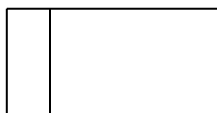
Process that transforms data flow.



Source or Destination of data



Data flow



Data Store

4.5.3 CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each word capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

4.5.4 SILENT FEATURES OF DFD'S

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

4.6 TYPES OF DATA FLOW DIAGRAMS

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

4.6.1 CURRENT PHYSICAL:

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

4.6.2 CURRENT LOGICAL:

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transform them regardless of actual physical form.

4.6.3 NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with he user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

4.6.4 NEW PHYSICAL:

The new physical represents only the physical implementation of the new system.

4.7 RULES GOVERNING THE DFD'S

4.7.1 PROCESS

- 1) No process can have only outputs.
- 2) No process can have only inputs. If an object has only inputs than it must be a sink.
- 3) A process has a verb phrase label.

4.7.2 DATA STORE

- 1) Data cannot move directly from one data store to another data store, a process must move data.
- 2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
- 3) A data store has a noun phrase label.

4.7.3 SOURCE OR SINK

The origin and / or destination of data.

- 1) Data cannot move direly from a source to sink it must be moved by a process
- 2) A source and /or sink has a noun phrase land.

4.7.4 DATA FLOW

- 1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
- 2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
- 3) A data flow cannot go directly back to the same process it leads. There must be atleast one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
- 4) A Data flow to a data store means update (delete or change).
- 5) A data Flow from a data store means retrieve or use.
- 6) A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

4.7.5 DATA DICTONARY

After carefully understanding the requirements of the client the the entire data storage requirements are divided into tables. The below tables are normalized to avoid any anomalies during the course of data entry

4.8 UML Diagram

4.8.1Actor:

A coherent set of roles that users of use cases play when interacting with the use cases.



FIGURE: 2 ACTOR

4.8.2 Use case:

A description of sequence of actions, including variants, that a system performs that yields an observable result of value of an actor.

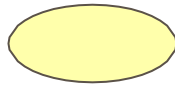


FIGURE: 3 USE CASE

UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

There are various kinds of methods in software design:

They are as follows:

- Use case Diagram
- Sequence Diagram
- Class diagram
- Activity Diagram

4.9 USECASE DIAGRAMS:

Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor. Use case diagram can be useful for getting an overall view of the system and clarifying who can do and more importantly what they can't do. Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

- The purpose is to show the interactions between the use case and actor.
- To represent the system requirements from user's perspective.
- An actor could be the end-user of the system or an external system.

CHAPTER 5

SYSTEM DEVELOPMENT

5.1 SOFTWARE DESCRIPTION

5.1.1 INTRODUCTION TO PYTHON:

Python is currently the most widely used multi-purpose, high-level programming language, which allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and the indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard library which can be used for the following:

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia
- Scientific computing

- Text processing and many more.

Python is currently the most widely used multi-purpose, high-level programming language, which allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and the indentation requirement of the language, makes them readable all the time.

Now that we know what makes Python a high level language, let us also have a look at some of the lesser-known features of the language listed below:-

- 1. Function annotations:** Python allows you to add annotations to function parameters and return types. These annotations are optional and do not affect the function's behaviour, but they can be used to provide additional information to developers working on the language.
- 2. Coroutines:** Python supports coroutines, which are functions that can be paused and resumed. Coroutines are useful for writing asynchronous code, such as in web servers or networking applications.
- 3. Enumerations:** Python has a built-in Enum class that allows you to define symbolic names for values. Enumerations are useful for improving the readability and maintainability of your code.
- 4. List comprehensions:** Python allows you to create lists in a concise and readable way using list comprehensions. For example, you can create a list of squares of numbers using `[x**2 for x in range(10)]`.

5. **Extended iterable unpacking:** Python allows you to unpack iterable (e.g., lists, tuples, and dictionaries) into variables using the `*` and `**` operators. This feature makes it easy to work with complex data structures.
6. **The with statement:** Python's with statement allows you to manage resources (such as files or network connections) cleanly and concisely. The with statement automatically opens and closes the resource, even in the case of exceptions.
7. **The walrus operator:** Python 3.8 introduced the walrus operator (`:=`), which allows you to assign a value to a variable as part of an expression. This feature is useful for simplifying code and reducing the number of lines.
8. **The slots attribute:** Python allows you to optimize memory usage by using the slots attribute in classes. This attribute tells Python to allocate memory only for the specified attributes, reducing memory overhead.

Python is a popular programming language for multiple reasons, some of which include:

- **Simplicity:** Python is easy to learn and read, making it a number one choice for beginners. Its syntax is straightforward and consistent, allowing developers to write code quickly and efficiently.
- **Versatility:** Python is a general-purpose language, which means it can be used for a wide range of applications, including web development, data analysis, machine learning, and artificial intelligence.
- **Large community:** Python has a large and active community of developers who contribute to its development and offer support to new users. This community has

created a vast array of libraries and frameworks that make development faster and easier.

- **Open-source:** Python is an open-source language, which means its source code is freely available and can be modified and distributed by anyone. This has led to the creation of many useful libraries and frameworks that have made Python even more popular.
- **Scalability:** Python can be used to build both small and large-scale applications. Its scalability makes it an attractive choice for startups and large organizations alike.

These are just a few of the lesser-known features of Python as a language. Python is a powerful and flexible language that offers many more features and capabilities that can help developers write efficient, maintainable, and scalable code with ease and efficiency.

5.2 LIBRARIES IN PYTHON

Normally, a library is a collection of books or is a room or place where many books are stored to be used later. Similarly, in the programming world, a library is a collection of precompiled codes that can be used later on in a program for some specific well-defined operations. Other than pre-compiled codes, a library may contain documentation, configuration data, message templates, classes, and values, etc.

A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs. Python libraries play a very vital role in fields of Machine Learning, Data Science, Data Visualization, etc.

5.3 WORKING OF PYTHON LIBRARY

As is stated above, a Python library is simply a collection of codes or modules of codes that we can use in a program for specific operations. We use libraries so that we don't need to write the code again in our program that is already available. But how it works. Actually, in the MS Windows environment, the library files have a DLL extension (Dynamic Load Libraries). When we link a library with our program and run that program, the linker automatically searches for that library. It extracts the functionalities of that library and interprets the program accordingly. That's how we use the methods of a library in our program. We will see further, how we bring in the libraries in our Python programs.

5.4 PYTHON STANDARD LIBRARY

The Python Standard Library contains the exact syntax, semantics, and tokens of Python. It contains built-in modules that provide access to basic system functionality like I/O and some other core modules. Most of the Python Libraries are written in the C programming language. The Python standard library consists of more than 200 core modules. All these work together to make Python a high-level programming language. Python Standard Library plays a very important role. Without it, the programmers can't have access to the functionalities of Python. But other than this, there are several other libraries in Python that make a programmer's life easier. Let's have a look at some of the commonly used libraries:

1. **TensorFlow:** This library was developed by Google in collaboration with the Brain Team. It is an open-source library used for high-level computations. It is also used

in machine learning and deep learning algorithms. It contains a large number of tensor operations. Researchers also use this Python library to solve complex computations in Mathematics and Physics.

2. **Matplotlib:** This library is responsible for plotting numerical data. And that's why it is used in data analysis. It is also an open-source library and plots high-defined figures like pie charts, histograms, scatterplots, graphs, etc.
3. **Pandas:** Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.
4. **Numpy:** The name "Numpy" stands for "Numerical Python". It is the commonly used library. It is a popular machine learning library that supports large matrices and multi-dimensional data. It consists of in-built mathematical functions for easy computations. Even libraries like TensorFlow use Numpy internally to perform several operations on tensors. Array Interface is one of the key features of this library.
5. **SciPy:** The name "SciPy" stands for "Scientific Python". It is an open-source library used for high-level scientific computations. This library is built over an extension of Numpy. It works with Numpy to handle complex computations. While Numpy allows sorting and indexing of array data, the numerical data code is stored in SciPy. It is also widely used by application developers and engineers.

6. **Scrapy:** It is an open-source library that is used for extracting data from websites. It provides very fast web crawling and high-level screen scraping. It can also be used for data mining and automated testing of data.
7. **Scikit-learn:** It is a famous Python library to work with complex data. Scikit-learn is an open-source library that supports machine learning. It supports variously supervised and unsupervised algorithms like linear regression, classification, clustering, etc. This library works in association with Numpy and SciPy.
8. **PyGame:** This library provides an easy interface to the Standard Directmedia Library (SDL) platform-independent graphics, audio, and input libraries. It is used for developing video games using computer graphics and audio libraries along with Python programming language.
9. **PyTorch:** PyTorch is the largest machine learning library that optimizes tensor computations. It has rich APIs to perform tensor computations with strong GPU acceleration. It also helps to solve application issues related to neural networks.
10. **PyBrain:** The name “PyBrain” stands for Python Based Reinforcement Learning, Artificial Intelligence, and Neural Networks library. It is an open-source library built for beginners in the field of Machine Learning. It provides fast and easy-to-use algorithms for machine learning tasks. It is so flexible and easily understandable and that’s why is really helpful for developers that are new in research fields.

There are many more libraries in Python. We can use a suitable library for our purposes.

5.5 USE OF LIBRARIES IN PYTHON

As we write large-size programs in Python, we want to maintain the code's modularity. For the easy maintenance of the code, we split the code into different parts and we can use that code later ever we need it. In Python, modules play that part. Instead of using the same code in different programs and making the code complex, we define mostly used functions in modules and we can just simply import them in a program wherever there is a requirement. We don't need to write that code but still, we can use its functionality by importing its module. Multiple interrelated modules are stored in a library. And whenever we need to use a module, we import it from its library. In Python, it's a very simple job to do due to its easy syntax. We just need to use import.

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000. Python 3.0, released in 2008, was a major revision not completely backward-compatible with earlier versions. Python 2.7.18, released in 2020, was the last release of Python 2. Python consistently ranks as one of the most popular programming languages.

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde&Informatica (CWI) in the Netherlands as a successor to the ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's "benevolent dictator for life", a title the Python community bestowed upon him to reflect

his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a five-member Steering Council to lead the project. Python 2.0 was released on 16 October 2000, with many major new features such as list comprehensions, cycle-detecting garbage, and Unicode support. Python 3.0, released on 3 December 2008, with many of its major features back ported to Python 2.6.x and 2.7.x. Releases of Python 3 include the 2to3 utility, which automates the translation of Python 2 code to Python 3.

Python 2.7's end-of-life was initially set for 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No further security patches or other improvements will be released for it. Currently only 3.7 and later are supported. In 2021, Python 3.9.2 and 3.8.8 were expedited as all versions of Python (including 2.7) had security issues leading to possible remote code execution and web cache poisoning.

In 2022, Python 3.10.4 and 3.9.12 were expedited and 3.8.13, and 3.7.13, because of many security issues. When Python 3.9.13 was released in May 2022, it was announced that the 3.9 series (joining the older series 3.8 and 3.7) would only receive security fixes in the future. On September 7, 2022, four new releases were made due to a potential denial-of-service attack: 3.10.7, 3.9.14, 3.8.14, and 3.7.14. As of November 2022, Python 3.11.0 is the current stable release. Notable changes from 3.10 include increased program execution speed and improved error reporting.

5.6 DESIGN PHILOSOPHY AND FEATURES

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of their features

support functional programming and aspect-oriented programming (including meta programming and meta objects). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It uses dynamic name resolution (late binding), which binds method and variable names during program execution.

Its design offers some support for functional programming in the Lisp tradition. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

Its core philosophy is summarized in the document The Zen of Python (PEP 20), which includes aphorisms such as:

- ❖ Beautiful is better than ugly.
- ❖ Explicit is better than implicit.
- ❖ Simple is better than complex.
- ❖ Complex is better than complicated.
- ❖ Readability counts.

Rather than building all of its functionality into its core, Python was designed to be highly extensible via modules. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one—and preferably only one—obvious way to do it" philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, wrote: "To describe something as 'clever' is not considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C; or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

Python's developers aim for it to be fun to use. This is reflected in its name—a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as the use of the terms "spam" and "eggs" (a reference to a Monty Python sketch) in examples, instead of the often-used "foo" and "bar".

A common neologism in the Python community is *pythonic*, which has a wide range of meanings related to program style. "Pythonic" code may use Python idioms well, be natural or show fluency in the language, or conform with Python's minimalist philosophy and emphasis on readability. Code that is difficult to understand or reads like a rough transcription from another programming language is called *unpythonic*.

5.7 SYNTAX AND SEMANTICS

Python is meant to be an easily readable language. Its formatting is visually uncluttered and often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are allowed but rarely used. It has fewer syntactic exceptions and special cases than C or Pascal.

Python uses whitespace indentation, rather than curly brackets or keywords, to delimit blocks. An increase in indentation comes after certain statements; a decrease in indentation signifies the end of the current block. Thus, the program's visual structure accurately represents its semantic structure. This feature is sometimes termed the off-side rule. Some other languages use indentation this way; but in most, indentation has no semantic meaning. The recommended indent size is four spaces.

Statements and control flow

Python's statements include:

- The assignment statement, using a single equals sign =
- The if statement, which conditionally executes a block of code, along with else and elif (a contraction of else-if)
- The for statement, which iterates over an iterable object, capturing each element to a local variable for use by the attached block
- The while statement, which executes a block of code as long as its condition is true
- The try statement, which allows exceptions raised in its attached code block to be caught and handled by except clauses (or new syntax except* in Python 3.11 for exception

groups[85]); it also ensures that clean-up code in a finally block is always run regardless of how the block exits

- The raise statement, used to raise a specified exception or re-raise a caught exception
- The class statement, which executes a block of code and attaches its local namespace to a class, for use in object-oriented programming
- The def statement, which defines a function or method
- The with statement, which encloses a code block within a context manager (for example, acquiring a lock before it is run, then releasing the lock; or opening and closing a file), allowing resource-acquisition-is-initialization (RAII)-like behavior and replacing a common try/finally idiom
- The break statement, which exits a loop
- The continue statement, which skips the rest of the current iteration and continues with the next
- The del statement, which removes a variable—deleting the reference from the name to the value, and producing an error if the variable is referred to before it is redefined
- The pass statement, serving as a NOP, syntactically needed to create an empty code block
- The assert statement, used in debugging to check for conditions that should apply
- The yield statement, which returns a value from a generator function (and also an operator); used to implement coroutines
- The return statement, used to return a value from a function
- The import and from statements, used to import modules whose functions or variables can be used in the current program

The assignment statement (=) binds a name as a reference to a separate, dynamically allocated object. Variables may subsequently be rebound at any time to any object. In Python, a variable name is a generic reference holder without a fixed data type; however, it always refers to some object with a type. This is called dynamic typing—in contrast to statically-typed languages, where each variable may contain only a value of a certain type.

Python does not support tail call optimization or first-class continuations, and, according to Van Rossum, it never will. However, better support for coroutine-like functionality is provided by extending Python's generators. Before 2.5, generators were lazy iterators; data was passed unidirectionally out of the generator. From Python 2.5 on, it is possible to pass data back into a generator function; and from version 3.3, it can be passed through multiple stack levels.

5.8 DEVELOPMENT

Python's development is conducted largely through the Python Enhancement Proposal (PEP) process, the primary mechanism for proposing major new features, collecting community input on issues, and documenting Python design decisions. Python coding style is covered in PEP 8 Outstanding PEPs are reviewed and commented on by the Python community and the steering council.

Enhancement of the language corresponds with the development of the CPython reference implementation. The mailing list python-dev is the primary forum for the language's development. Specific issues were originally discussed in the Roundup bug tracker hosted at by the foundation. In 2022, all issues and discussions were migrated

to GitHub. Development originally took place on a self-hosted source-code repository running Mercurial, until Python moved to GitHub in January 2017.

CPython's public releases come in three types, distinguished by which part of the version number is incremented:

- Backward-incompatible versions, where code is expected to break and needs to be manually ported. The first part of the version number is incremented. These releases happen infrequently—version 3.0 was released 8 years after 2.0. According to Guido van Rossum, a version 4.0 is very unlikely to ever happen. Major or "feature" releases are largely compatible with the previous version but introduce new features. The second part of the version number is incremented. Starting with Python 3.9, these releases are expected to happen annually. Each major version is supported by bug fixes for several years after its release.
- Bugfix releases, which introduce no new features, occur about every 3 months and are made when a sufficient number of bugs have been fixed upstream since the last release. Security vulnerabilities are also patched in these releases. The third and final part of the version number is incremented.

Many alpha, beta, and release-candidates are also released as previews and for testing before final releases. Although there is a rough schedule for each release, they are often delayed if the code is not ready. Python's development team monitors the state of the code by running the large unit test suite during development. The major academic conference on Python is PyCon. There are also special Python mentoring programs, such as Pyladies.

Python 3.10 deprecated `wstr` (to be removed in Python 3.12; meaning Python extensions need to be modified by then), and added pattern matching to the language.

5.9.1 API documentation generators

Tools that can generate documentation for Python API include `pydoc` (available as part of the standard library), `Sphinx`, `Pdoc` and its forks, `Doxygen` and `Graphviz`, among others.

5.9.2 Naming

Python's name is derived from the British comedy group `Monty Python`, whom Python creator Guido van Rossum enjoyed while developing the language. `Monty Python` references appear frequently in Python code and culture; for example, the metasyntactic variables often used in Python literature are `spam` and `eggs` instead of the traditional `foo` and `bar`. The official Python documentation also contains various references to `Monty Python` routines. The prefix `Py-` is used to show that something is related to Python. Examples of the use of this prefix in names of Python applications or libraries include `Pygame`, a binding of `SDL` to Python (commonly used to create games); `PyQt` and `PyGTK`, which bind `Qt` and `GTK` to Python respectively; and `PyPy`, a Python implementation originally written in Python.

Popular it Since 2003, Python has consistently ranked in the top ten most popular programming languages in the TIOBE Programming Community Index where as of December 2022 it was the most popular language (ahead of C, C++, and Java). It was selected Programming Language of the Year (for "the highest rise in ratings in a year") in 2007, 2010, 2018, and 2020 (the only language to have done so four times as of 2022).An

empirical study found that scripting languages, such as Python, are more productive than conventional languages, such as C and Java, for programming problems involving string manipulation and search in a dictionary.

5.9 OPEN CV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

5.10 OPENCV LIBRARY MODULES

Following are the main library modules of the OpenCV library.

5.11.1 Core Functionality

This module covers the basic data structures such as Scalar, Point, Range, etc., that are used to build OpenCV applications. In addition to these, it also includes the

multidimensional array Mat, which is used to store the images. In the Java library of OpenCV, this module is included as a package with the name org.opencv.core.

5.11.2 Image Processing

This module covers various image processing operations such as image filtering, geometrical image transformations, color space conversion, histograms, etc. In the Java library of OpenCV, this module is included as a package with the name org.opencv.imgproc.

5.11.3 Video

This module covers the video analysis concepts such as motion estimation, background subtraction, and object tracking. In the Java library of OpenCV, this module is included as a package with the name org.opencv.video.

5.11.4 Video I/O

This module explains the video capturing and video codecs using OpenCV library. In the Java library of OpenCV, this module is included as a package with the name org.opencv.videoio.

5.11.5 calib3d

This module includes algorithms regarding basic multiple-view geometry algorithms, single and stereo camera calibration, object pose estimation, stereo correspondence and elements of 3D reconstruction. In the Java library of OpenCV, this module is included as a package with the name org.opencv.calib3d.

5.11.6 features2d

This module includes the concepts of feature detection and description. In the Java library of OpenCV, this module is included as a package with the name `org.opencv.features2d`.

5.11.7 Objdetect

This module includes the detection of objects and instances of the predefined classes such as faces, eyes, mugs, people, cars, etc. In the Java library of OpenCV, this module is included as a package with the name `org.opencv.objdetect`.

5.11.8 Highgui

This is an easy-to-use interface with simple UI capabilities. In the Java library of OpenCV, the features of this module is included in two different packages namely, `org.opencv.imgcodecs` and `org.opencv.videoio`.

5.11 PILLOW

Digital Image processing means processing the image digitally with the help of a computer. Using image processing we can perform operations like enhancing the image, blurring the image, extracting text from images, and many more operations. There are various ways to process images digitally. Here we will discuss the Pillow module of Python.

Python Pillow is built on the top of PIL (Python Image Library) and is considered as the fork for the same as PIL has been discontinued from 2011.

Pillow supports many image file formats including BMP, PNG, JPEG, and TIFF. The library encourages adding support for newer formats in the library by creating new file decoders.

The Pillow module provides the `open()` and `show()` function to read and display the image respectively. For displaying the image Pillow first converts the image to a .png format (on Windows OS) and stores it in a temporary buffer and then displays it. Therefore, due to the conversion of the image format to .png some properties of the original image file format might be lost (like animation). Therefore, it is advised to use this method only for test purposes.

5.12 TENSORFLOW

TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML, and gives developers the ability to easily build and deploy ML-powered applications.

TensorFlow provides a collection of workflows with intuitive, high-level APIs for both beginners and experts to create machine learning models in numerous languages. Developers have the option to deploy models on a number of platforms such as on servers, in the cloud, on mobile and edge devices, in browsers, and on many other JavaScript platforms. This enables developers to go from model building and training to deployment much more easily.

5.13.1 Easy model building

Build and train ML models easily using intuitive high-level APIs like Keras with eager execution, which makes for immediate model iteration and easy debugging.

5.13.2 Robust ML production anywhere

Easily train and deploy models in the cloud, on-prem, in the browser, or on-device no matter what language you use.

5.13.3 Powerful experimentation for research

A simple and flexible architecture to take new ideas from concept to code, to state-of-the-art models, and to publication faster.

- **TensorFlow** was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well!

Let us first try to understand what the word **TensorFlow** actually mean! **TensorFlow** is basically a software library for numerical computation using **data flow graphs** where:

- **Nodes** in the graph represent mathematical operations.
- **Edges** in the graph represent the multidimensional data arrays (called **tensors**) communicated between them. (Please note that **tensor** is the central unit of data in TensorFlow).

5.13 TORCH

Deep Learning is a branch of Machine Learning where algorithms are written which mimic the functioning of a human brain. The most commonly used libraries in deep learning are Tensorflow and PyTorch. As there are various deep learning frameworks available, one

might wonder when to use PyTorch. Here are reasons why one might prefer using Pytorch for specific tasks.

Pytorch is an open-source deep learning framework available with a Python and C++ interface. Pytorch resides inside the torch module. In PyTorch, the data that has to be processed is input in the form of a tensor.

CHAPTER-6

SYSTEM TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of software specification, design and coding. The increasing visibility of software as a system element and the attendant “costs” associated with a software failure are motivating forces for conference management system project well planned, thorough testing. It is not unusual for conference management system project software. Development organization to expend 40 percent of total project effort on testing. Hence the importance of software testing and its implications with respect to software quality cannot be overemphasized. Different types of testing have been carried out for conference management system project this system, and they are briefly explained below.

6.1 TYPES OF TESTS

Testing is the process of trying to discover every conceivable fault or weakness in a work product. The different types of testing are given below:

6.1.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.1.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.1.3 VALIDATION TESTING

Tests were performed to find conformity with the requirements. Plans and procedures were designed to ensure that all functional requirements are satisfied. The software was alpha-tested there are two goals in preparing test plans. Firstly, a properly detailed test plan demonstrates that the program specifications are understood completely. Secondly, the test plan is used during program testing to prove the correctness of the program.

7.SOURCE CODE:

```
from abc import ABC, abstractmethod
from bs4 import BeautifulSoup
from datetime import datetime
import smtplib
from email.message import EmailMessage
import validators
from time import sleep
import re
from PIL import Image
import requests
import os
import streamlit as st
import base64

page_logo = Image.open('logo.png')
st.set_page_config(page_title="E-commerce Price Tracker", layout='wide',
page_icon=page_logo)
new_title = '<p style="font-family:sans-serif; color:#cc8e3d; text-shadow: -1px -1px 0 #fff,
1px -1px 0 #000, -1px 1px 0 #000, 1px 1px 0 #000; font-weight:600; font-size: 42px;
padding-left: 400px;">E-Commerce Price Tracker</p>'
st.markdown(new_title, unsafe_allow_html=True)
price_output = "

def add_bg_from_local(image_file):
    with open(image_file, "rb") as image_file:
        encoded_string = base64.b64encode(image_file.read())
    st.markdown(
        f"""
        <style>
        .stApp {{
            background-image: url(data:image/{ "jpg" };base64,{encoded_string.decode()});
            background-size: cover
        }}
        </style>
        """,
        unsafe_allow_html=True
    )

add_bg_from_local("bg3.jpg")
```

```

class PriceTracker(ABC):
    """
    Base class for Price Trackers.
    """
    removals = re.compile(r"₹|,|[$]")

    def __init__(self, product_name: str, product_url: str, desired_price: int) -> None:
        """
        Constructor.
        """
        if not validators.url(product_url):
            raise Exception("Invalid URL!")

        self.product_url = product_url
        self.set_price = desired_price
        self.product_name = product_name
        self.price = None

    @abstractmethod
    def get_price(self):
        """
        Fetches the latest price of the product.
        """
        pass

    @staticmethod
    def email_alert(subject: str, body: str):
        """
        Sends mail for the given price.
        """
        msg = EmailMessage()
        msg.set_content(body)
        msg['subject'] = subject
        your_email = "aaddictor1@gmail.com"
        msg['to'] = your_email
        msg['from'] = your_email
        your_password = "phkbpoyoerkcbzzkt"
        server = smtplib.SMTP('smtp.gmail.com', 587)
        server.starttls()
        server.login(your_email, your_password)
        server.send_message(msg)
        server.quit()
        print("Mail sent")

```

```

@abstractmethod
def write_to_file(self):
    """
    Write content to a text file.
    """
    pass

def track_price(self):
    """
    Tracks price for a given product against the set price.
    """
    while True:
        self.get_price()
        if self.price == None:
            print("Unable to fetch the latest price!")
            os._exit(1)

        if self.set_price >= self.price:
            print("Price low for", self.product_name)
            st.write(f"<p style='color:white; font-size:28px; padding-left:360px;'>Price low for
{self.product_name} ...</p>", unsafe_allow_html=True)
            self.write_to_file()
            self.email_alert(f"Price down for {self.product_name}", f"Dear sir,\nThe price for
`{self.product_name}` is now {self.price} which is less than or equal to what you desired,
{self.set_price}! Visit {self.product_url} for more info.")
            os._exit(0)

        else:
            self.write_to_file()
            sleep(60)

class FlipkartPriceTracker(PriceTracker):
    """
    Helps poor people by notifying them when the price of their favorite product on Flipkart is
    less than what they desired.
    """

    def get_price(self):
        """
        Fetches price for given product on Flipkart.
        """
        r = requests.get(self.product_url)

```

```

soup = BeautifulSoup(r.content, "html5lib")
price = soup.find("div", {"class": "_30jeq3 _16Jk6d"}).string
price = re.sub(self.removals, "", price)
self.price = int(price)

def write_to_file(self):
    """
    Writes datetime and price to a file.
    """
    now = datetime.now().strftime("%d-%m-%y %H:%M %p")
    content = f"{now} --> Flipkart --> Rs:{self.price}/-\n"
    print(content)
    with open(f"Price History of {self.product_name}.txt", "a") as f:
        f.write(content)
    price_output = f'At {now}, Updated price value \u20B9{self.price}/-'
    st.write(f'<p style="color:white; font-size:28px; padding-left:360px;">{price_output}</p>', unsafe_allow_html=True)

def run(self):
    self.track_price()

def check_internet():
    try:
        requests.get("https://google.com")

    except Exception as e:
        print("Make sure you're connected to the internet!")
        print(e)
        quit()

if __name__ == "__main__":
    check_internet()
    try:
        print("Welcome to the Ecommerce Price Tracker! A tool which you can use to track prices for a given product on Flipkart.\n")
        st.write("<p style='color:white; font-size:28px;'>Welcome to the Ecommerce Price Tracker! A tool which you can use to track prices for a given product on Flipkart ...</p>", unsafe_allow_html=True)
        product_name = st.text_input("", placeholder="Enter Product Name")
        product_url = st.text_input("", placeholder="Enter Product URL")
        desired_price = int(st.text_input("", placeholder="Enter Your Desired Price"))
        st.write("")

```



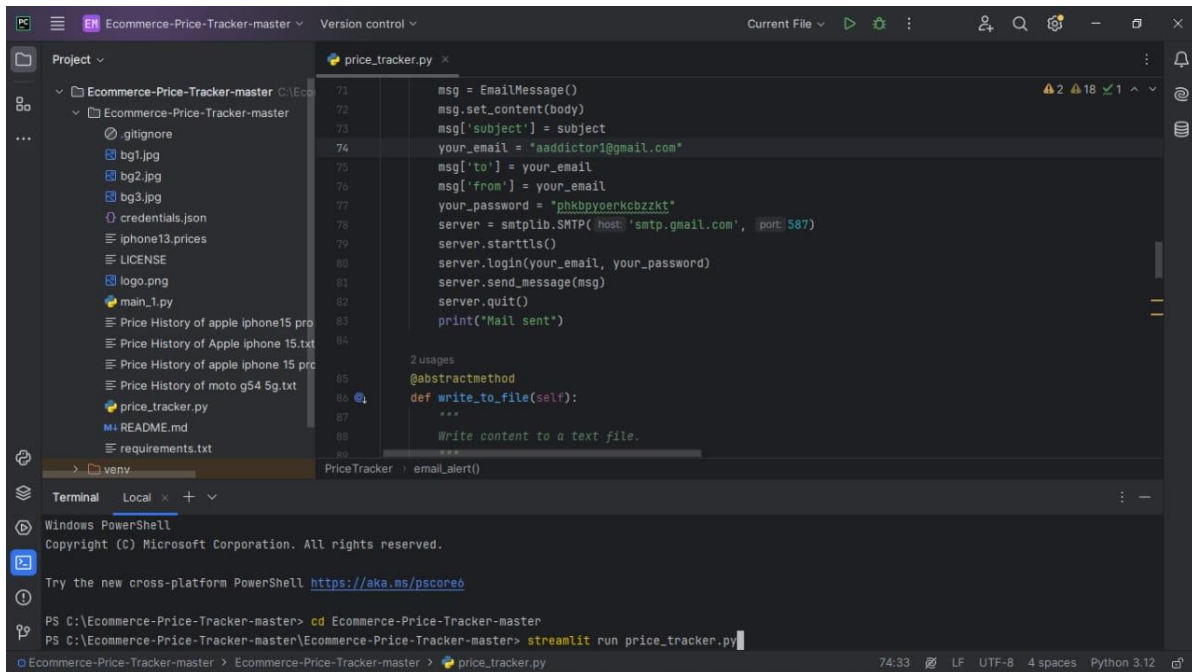
```

st.write("")
if st.button("Track Price"):
    st.write(f"<p style='color:orange; font-size:36px;'>Current Price of {product_name}
in Flipkart", unsafe_allow_html=True)
    fkt = FlipkartPriceTracker(product_name, product_url, desired_price)
    fkt.run()

except Exception as e:
    print("Some problem occurs ... ")
    print(e)

```

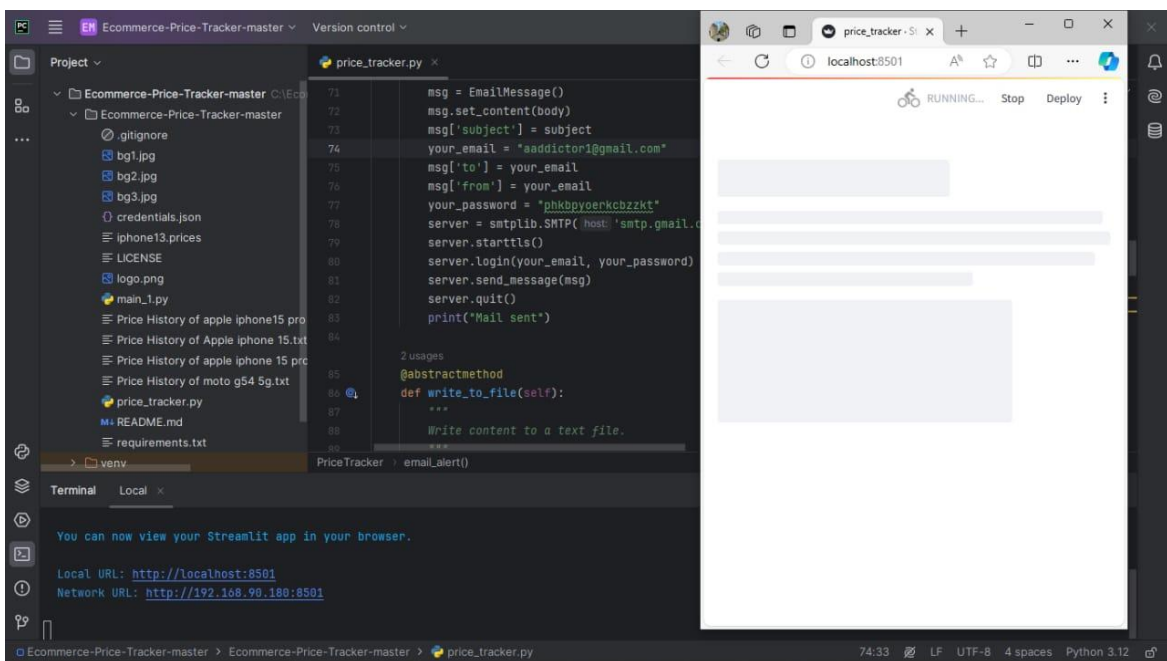
8.OUTPUT:



The screenshot shows a VS Code editor with the 'Ecommerce-Price-Tracker-master' project open. The file explorer on the left shows the project structure, including files like bg1.jpg, bg2.jpg, bg3.jpg, credentials.json, iphone13.prices, LICENSE, logo.png, main_1.py, Price History of apple iphone15 pro, Price History of Apple iphone 15.txt, Price History of apple iphone 15 pro, Price History of moto g54 5g.txt, price_tracker.py, README.md, requirements.txt, and .env. The main editor window shows the code for price_tracker.py, which includes an email alert function and a write_to_file method. The terminal window at the bottom shows the command 'streamlit run price_tracker.py' being executed.

```
71 msg = EmailMessage()
72 msg.set_content(body)
73 msg['subject'] = subject
74 your_email = "aaddictor1@gmail.com"
75 msg['to'] = your_email
76 msg['from'] = your_email
77 your_password = "phkbpvoerkcbzzkt"
78 server = smtplib.SMTP(host='smtp.gmail.com', port=587)
79 server.starttls()
80 server.login(your_email, your_password)
81 server.send_message(msg)
82 server.quit()
83 print("Mail sent")
84
85
86 @abstractmethod
87 def write_to_file(self):
88     """
89     Write content to a text file.
90     """
91
92 PriceTracker().email_alert()
```

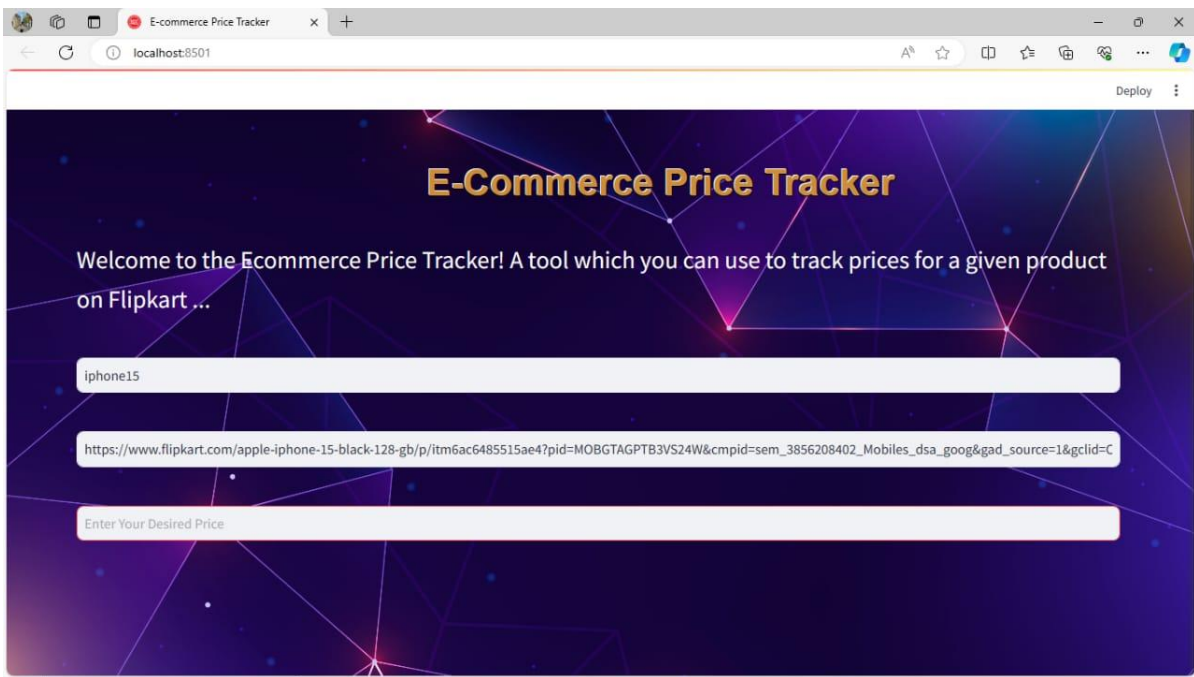
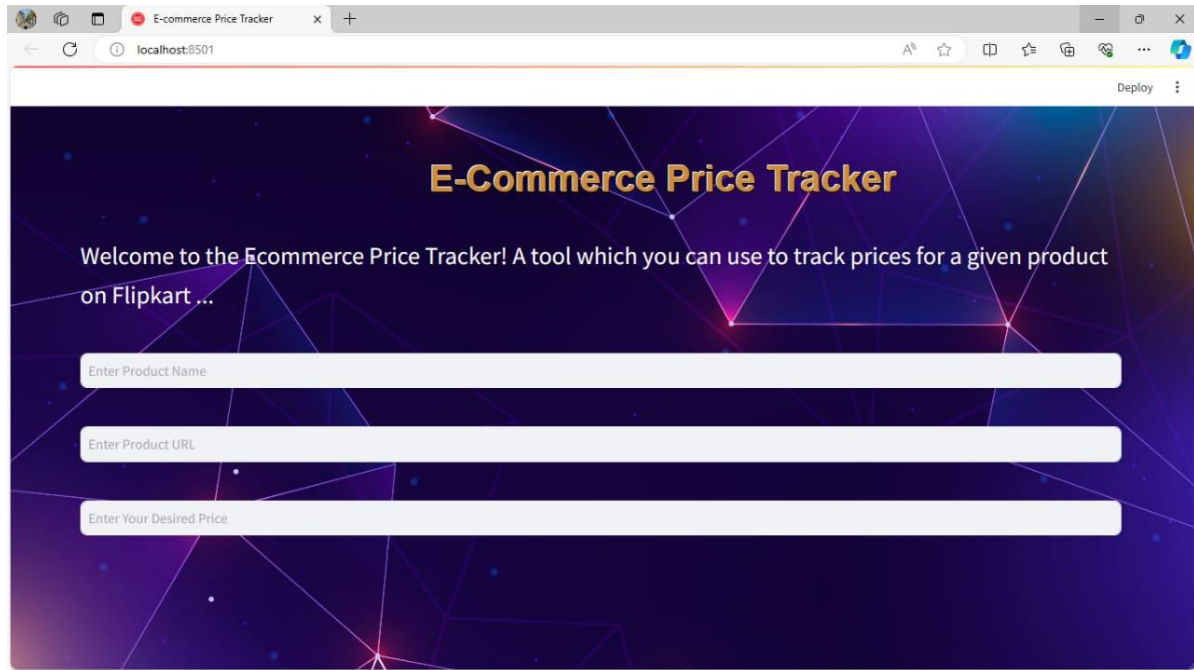
```
PS C:\Ecommerce-Price-Tracker-master> cd Ecommerce-Price-Tracker-master
PS C:\Ecommerce-Price-Tracker-master\Ecommerce-Price-Tracker-master> streamlit run price_tracker.py
```

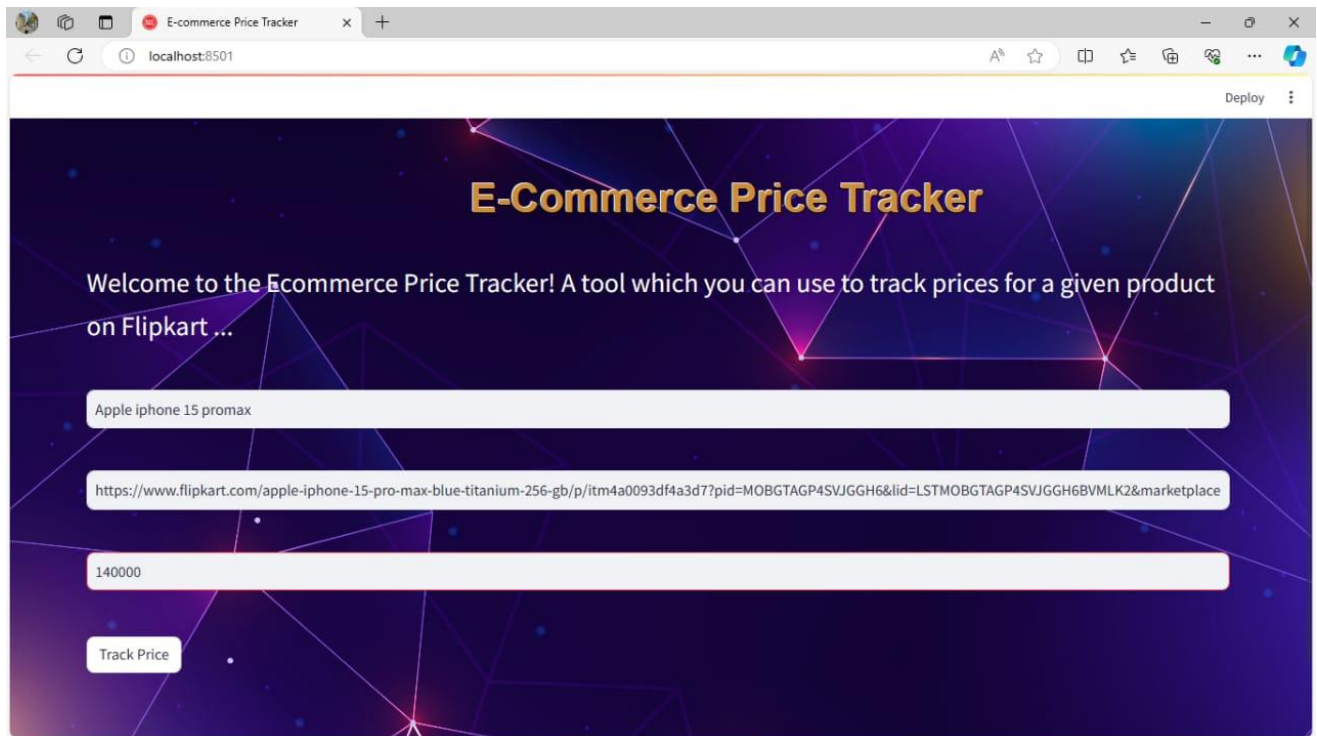
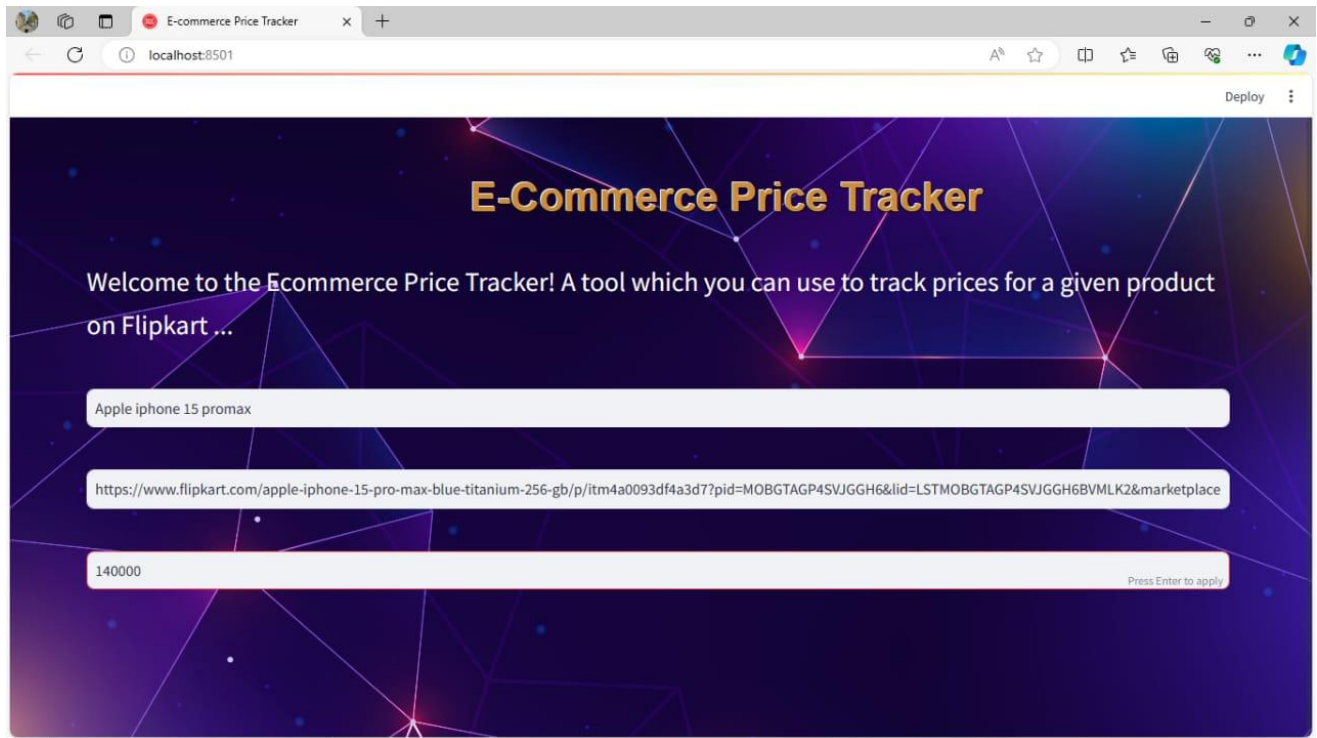


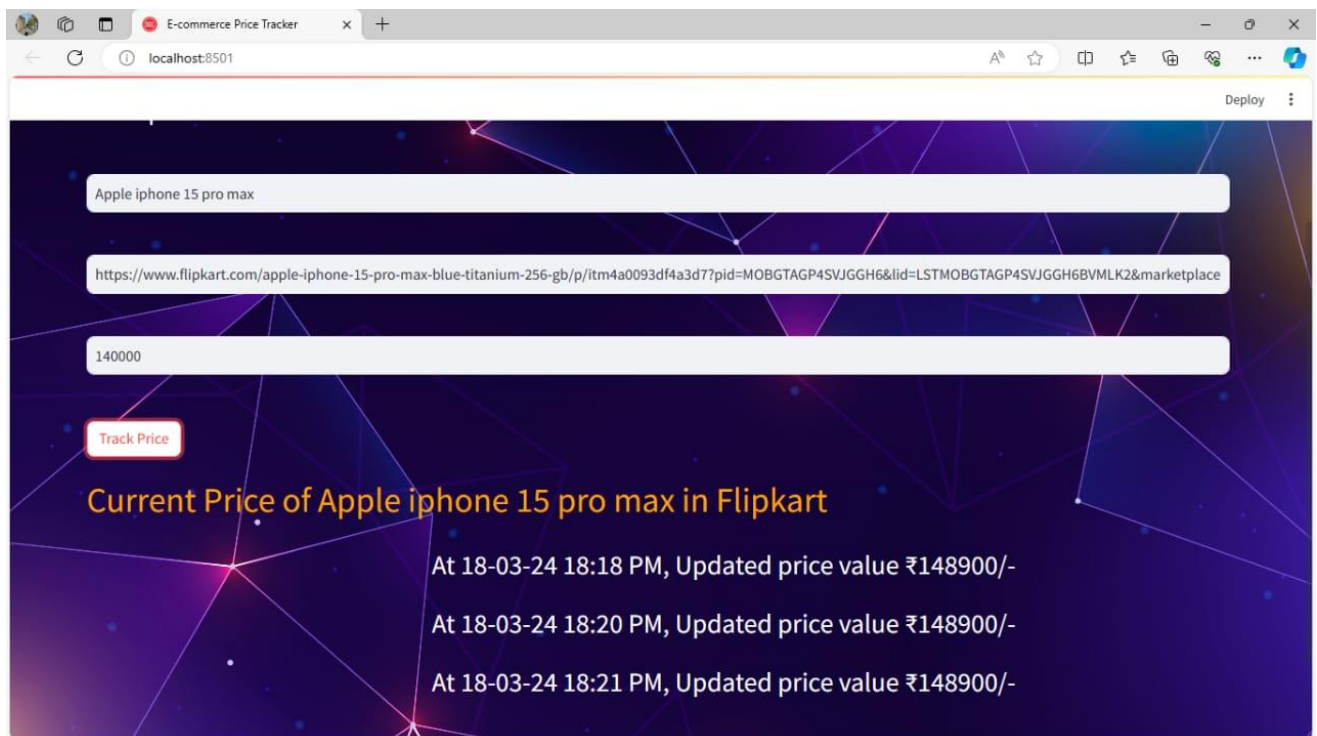
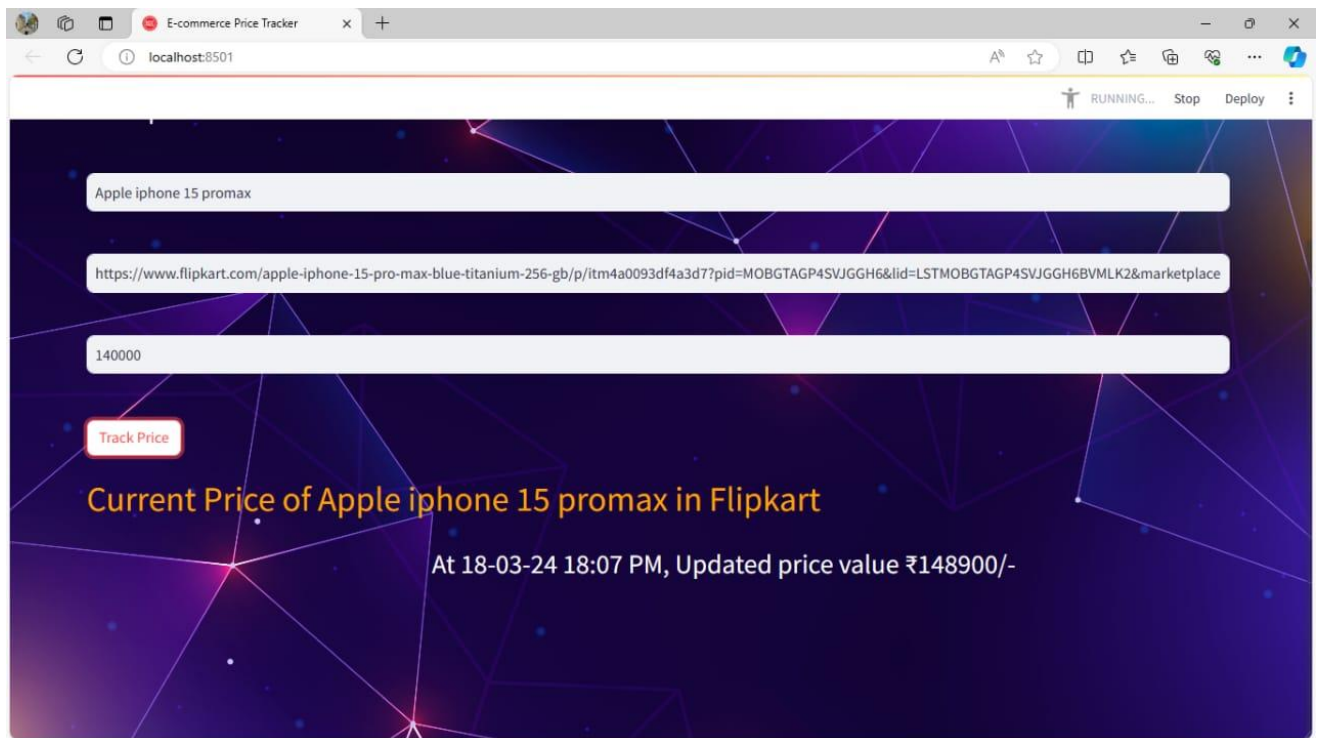
The screenshot shows the same VS Code editor setup as the previous one, but with a Streamlit app running in a browser window. The browser window displays the output of the Streamlit app, which includes a title bar with 'price_tracker - Sl' and a 'RUNNING...' status. The app content shows a list of items with their prices, including 'Price History of apple iphone15 pro', 'Price History of Apple iphone 15.txt', 'Price History of apple iphone 15 pro', and 'Price History of moto g54 5g.txt'. The terminal window at the bottom shows the command 'streamlit run price_tracker.py' being executed.

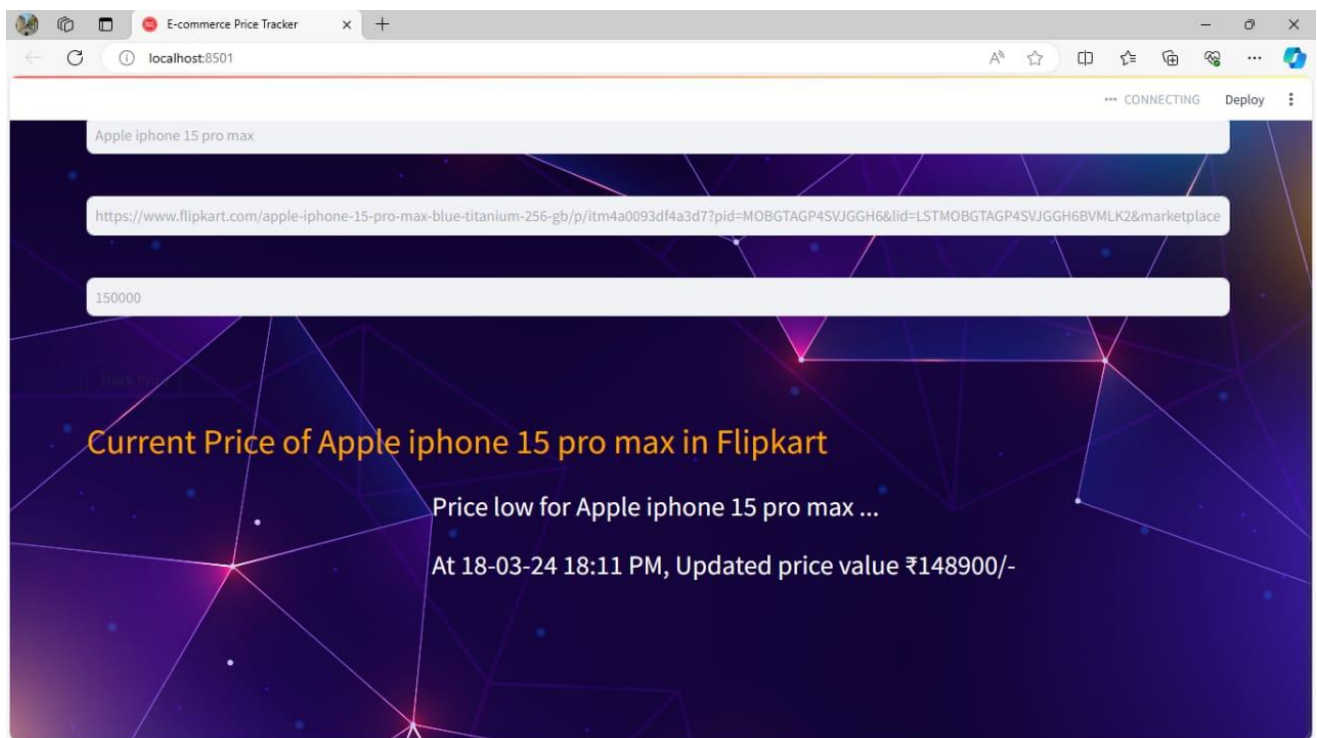
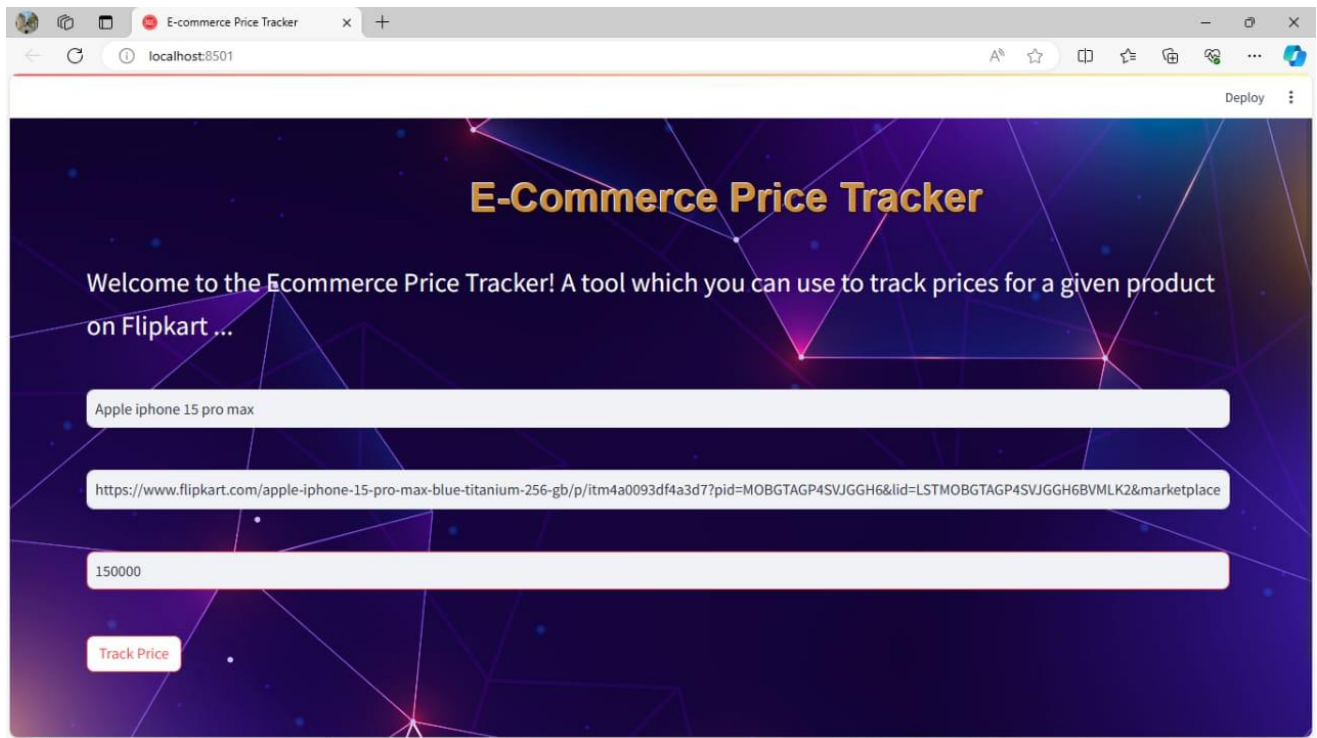
```
71 msg = EmailMessage()
72 msg.set_content(body)
73 msg['subject'] = subject
74 your_email = "aaddictor1@gmail.com"
75 msg['to'] = your_email
76 msg['from'] = your_email
77 your_password = "phkbpvoerkcbzzkt"
78 server = smtplib.SMTP(host='smtp.gmail.com', port=587)
79 server.starttls()
80 server.login(your_email, your_password)
81 server.send_message(msg)
82 server.quit()
83 print("Mail sent")
84
85
86 @abstractmethod
87 def write_to_file(self):
88     """
89     Write content to a text file.
90     """
91
92 PriceTracker().email_alert()
```

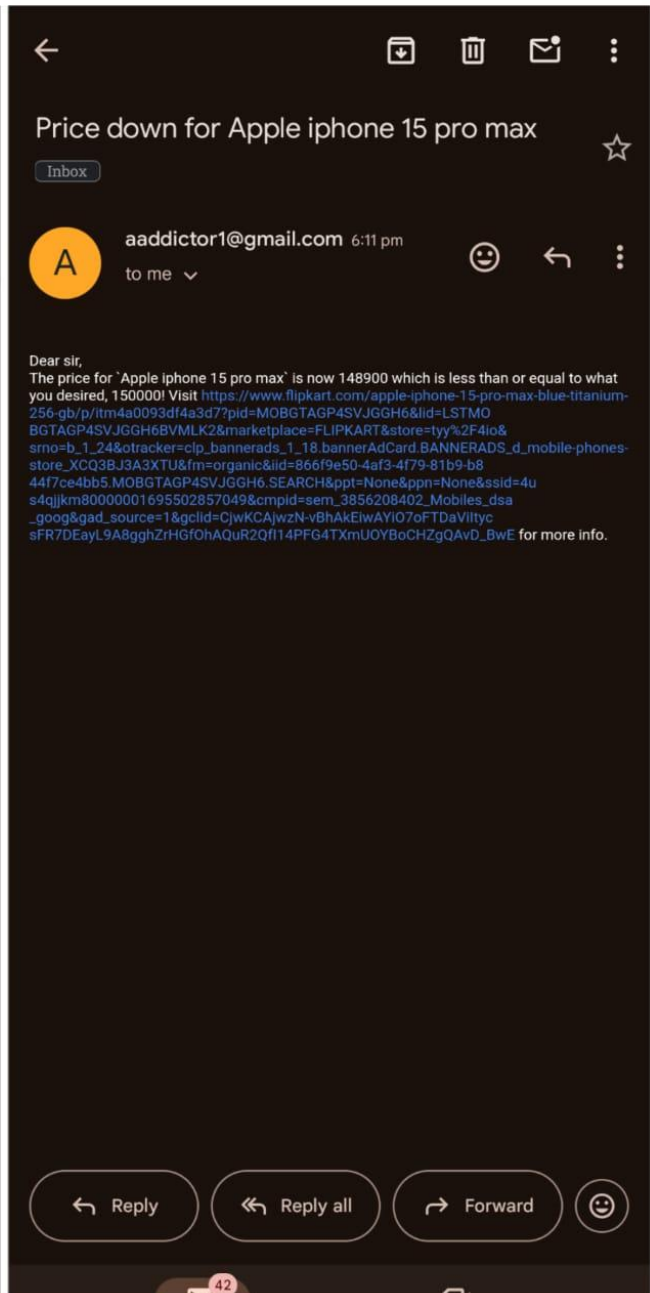
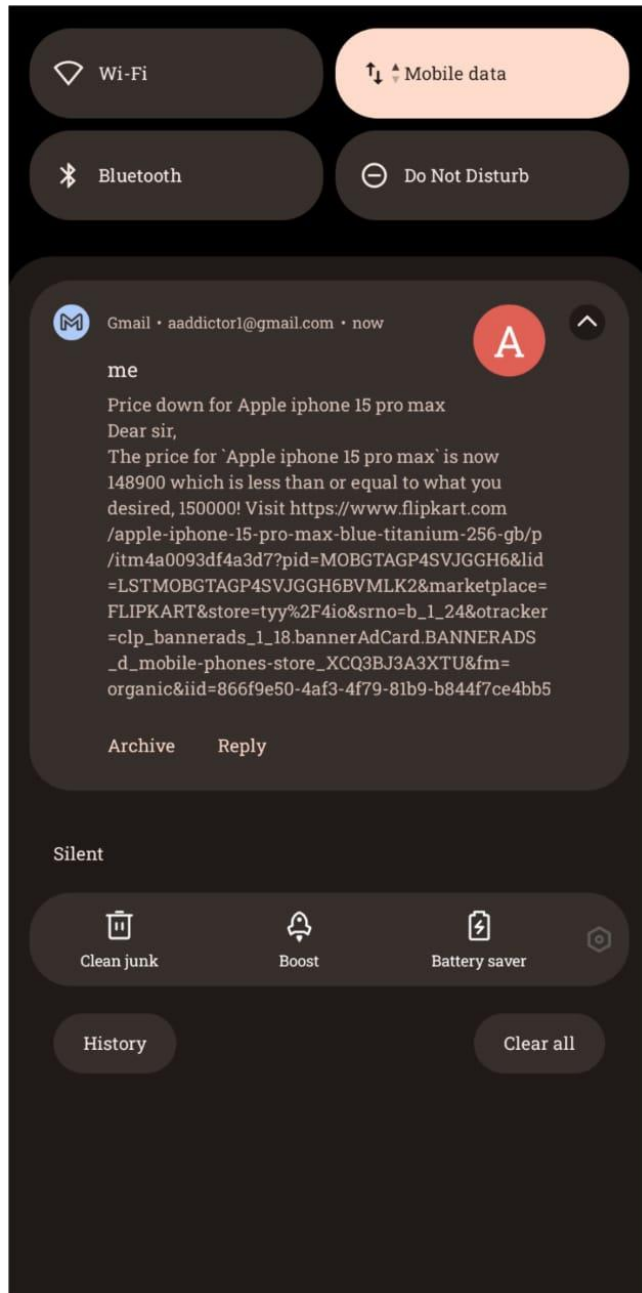
```
PS C:\Ecommerce-Price-Tracker-master> cd Ecommerce-Price-Tracker-master
PS C:\Ecommerce-Price-Tracker-master\Ecommerce-Price-Tracker-master> streamlit run price_tracker.py
```











CHAPTER-9

CONCLUSION

Before the development of this project. There are many loopholes in the process of taking attendance using the old method which caused many troubles to most of the institutions. Therefore, the facial recognition feature embedded in the attendance monitoring system can not only ensure attendance to be taken accurately and also eliminated the flaws in the previous system. By using technology to conquer the defects cannot merely save resources but also reduces human intervention in the whole process by handling all the complicated task to the machine. The only cost to this solution is to have sufficient space in to store all the faces into the database storage. Fortunately, there is such existence of micro SD that can compensate with the volume of the data. In this project, the face database is successfully built. Apart from that, the face recognizing system is also working well. At the end, the system not only resolve troubles that exist in the old model but also provide convenience to the user to access the information collected by mailing the attendance sheet to the respected faculty.

10.FUTURE ENHANCEMENT:

1.Product Price Tracking:

Allow users to track the prices of specific products from various e-commerce websites over time.

2. Price Alerts:

Enable users to set up price alerts for products, receiving notifications when prices drop below a specified threshold.

3. Product Comparison:

Provide tools for users to compare prices of the same product across different e-commerce platforms.

4. Historical Price Data:

Display historical price data for products, allowing users to analyze trends and make informed purchasing decisions.

5. User Profiles:

Allow users to create profiles to save their tracked products, preferences, and settings.

6. Search and Filters:

Implement robust search functionality and filters to help users quickly find the products they are interested in tracking.

7. Price History Charts:

Present price history charts for tracked products, giving users a visual representation of price fluctuations over time.

8. Price Drop Notifications:

Notify users when the price of a tracked product drops significantly, helping them capitalize on savings.

9. Integration with E-commerce Platforms:

Integrate with popular e-commerce platforms like Amazon, eBay, and Walmart to access a wide range of products.

10. Email Notifications:

Allow users to receive price alerts and updates via email for added convenience.

11. Customizable Alerts:

Enable users to customize their price alerts based on criteria such as percentage drops, specific price thresholds, or availability of discounts.

12. User Reviews and Ratings:

Integrate user reviews and ratings for products, helping users make informed purchasing decisions.

13. Mobile Compatibility:

Ensure the website is mobile-friendly, allowing users to track prices and receive alerts on their smartphones and tablets.

14. Security Features:

Implement robust security measures to protect user data and ensure safe transactions.

15. Customer Support:

Offer responsive customer support channels to assist users with any inquiries or issues they may Encounter.

11.BIBLIOGRAPHY:

1. Research on the price Forecast without Complete data based on Web Mining. Published in 10th international Symposium on Distributed Computing and Applications to, Business Engineering and Science (2010).
2. Dynamic pricing; different schemes, related research survey and evaluation. Published on 9th International Renewable Energy Congress (IREC) in the year 2018.
3. A price comparison system based on IOT. Published on 8th International Conference on Computer Science and Education(2013).
4. Prediction of prices for using regression models. Published on 5th International Conference on Business and Industrial Research (2018).
5. Technical help document – HeidiSQL <https://www.heidisql.com/help.php>
6. Documentation | HTML agility pack <https://html-agility-pack.net/documentation>

