INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 01, January 2024, pp : 138-142

e-ISSN : 2583-1062

Impact Factor : 5.725

DEALCRAKER: A PRODUCT PRICE MONITORING APPLICATION

Mrs. Rikeeta Mahajan[1], Mr. Swayam Dusing[2], Mr. Siddhant Dane[3], Mr. Manas Patil[4], Mr. Smit Pandit[5]

[1]Professor, Computer Engineering, Sandip Foundation's Sandip Polytechnic, Nashik, Maharashtra, India.

[2,3,4,5]student, Computer Engineering, Sandip Foundation's Sandip Polytechnic, Nashik, Maharashtra, India.

ABSTRACT

An Effective Price Monitoring Application is essential for today's E-Commerce websites. By studying some factors that impact customers during online shopping we found that the price of a product can vary across various platforms and sometimes this E-Commerce platform provides flash sales for very short periods of time. These factors make it difficult for customers to manually track product prices and may result in them missing out on good deals. So, in-order to resolve this concern, we have designed an Android Application which enables users to easily track product prices across different online E-commerce websites like Amazon and Flipkart. Our application requires the user to provide the URL link of a product and set his/her desired price, in future if the product achieved this desired price, then the user will be notified via in-app alerts. This enables users to make informed purchasing decisions and take advantage of cost-saving opportunities thanks to this real time notification mechanism. This Price Monitoring Application is a useful tool for enhancing online shopping experience and optimizing financial decisions as E-Commerce continues to Develop.

Keywords: Price Tracking, Web Scraping, Product URL, E-Commerce, Web Page Content Extraction

1. INTRODUCTION

The Core Objective of our project is to develop a Price Monitoring Application in the form of a user-friendly mobile application. This Application will empower users to track and compare the prices of products available on E-Commerce websites like Amazon and Flipkart, allowing them to make informed purchasing decisions and potentially save money. Following are the key features of our application:

- Product URL Input: Users will be able to input the URL of a specific product they are interested in purchasing. This URL will serve as the reference point for tracking the price of that product.
- Multi-Platform Availability: Our application can accept URLs from both Amazon and Flipkart website.
- Price Tracking: Once the user has added a product, our system will regularly monitor its price on the selected E-Commerce website. This automated tracking process will provide users with up-to-date information regarding any changes in the product's price.
- Price Drop Notifications: The application will notify users whenever there is a price drop for the monitored product. These Notifications can be delivered through various channels, such as in-app alerts, emails, or push notification depending on user preferences.
- Historical Price Data: Users will have access to historical price data for the products they are monitoring.
- User-Friendly Interface: The user interface will be intuitive and easy to navigate, making it accessible to users with varying levels of technical expertise.

2. METHODOLOGY

The Methodology behind the Deal Craker typically involves a combination of concept of Web Scraping and Android Development that works together to fetch product data from E-Commerce platforms and alert users regarding the fluctuation in product prices.

2.1 Project Scope: The project scope encompasses thorough requirements gathering and ongoing user feedback sessions to ensure alignment with user expectations. The backend, implemented in Python, employs libraries like requests and selectorlib for efficient HTML data extraction from e-commerce websites, providing a modular and scalable solution. Integration involves the development of an Android application that seamlessly interacts with the backend, empowering users to input product URLs and access scraped e-commerce data with a focus on optimal usability and responsiveness.

2.2 Data Collection and Scraping: Our backend infrastructure is implemented in Python, leveraging essential libraries such as requests and selectorlib for the purpose of scraping HTML data from e-commerce websites. This approach allows us to efficiently retrieve and process product information from the targeted platforms. The script dynamically determines the e-commerce site based on the input URL and employs selector files to extract relevant data. This Python script employs the FastAPI framework to create a web service for scraping product data from e-commerce websites such as Flipkart and Amazon. It defines one main endpoint, /scrape, utilizing the selectorlib library for data extraction based on YAML selector files. The scrape function dynamically determines the e-commerce site from the input URL and fetches data accordingly and returns the data in JSON format. Additionally, there is a special case in the scrape function for a specific URL pattern ("SMSSPRODUCT"), which returns pre-loaded data from a JSON file. The Objective behind SMSSPRODUCT is to load the data of a sample product available on our application, this also helps the user to understand the process of using our app. The script sets up appropriate HTTP headers to mimic a browser, and the FastAPI application can be run using the uvicorn server. Overall, it provides a modular and extensible solution for extracting product information from diverse online platforms. The backend program can be hosted either on a dedicated server or utilizing the NGROK service, which effectively transforms our localhost into a globally accessible host.

2.3 Integration with Android Application: This project not only focuses on creating a robust backend infrastructure but also includes building an Android application. The Android app acts as an essential interface, communicating easily with the Python-based backend. Its primary function is to empower users to initiate background queries for efficient data recovery. By creating an easy-to-use interface, the app greatly improves the overall user experience and provides an easy way to access e-commerce data through the Scraping Application. The Android application is conceived into five distinct screens, each serving a specific purpose. The first screen acts as the home page, providing an overview of available features. The second screen allows you to search for products and link directly to popular e-commerce platforms like Amazon and Flipkart. Going to the third screen, users can see a list of content they are currently tracking. The fourth screen acts as a container, making it easy to enter detailed product information and allowing users to set price alerts. Finally, the fifth panel integrates the WebView option, allowing users to browse eCommerce platforms within the application, which allow user to select the products from the platforms and directly add the product to track within the app. Application adopts MVVM architecture, using ViewModels for business logic and Views for user interface The user interface is built using the JETPACK Compose framework with Kotlin. Asynchronous operations are best handled through Kotlin coroutins, while Coil is used for asynchronous image loading. In-house libraries are used for local database management, ensuring easy seamless storage of essential data. Networking and HTTP requests are done through the OkayHTTP library, while background processing is handled by WorkManager. The LOTTIE library is used to add interesting graphics to the user interface, enhancing the overall visual appeal of the application. Webview Functionality: Users can open WebView within the application, allowing them to select and initiate tracking of objects. The URL of the selected object is sent to the custom API backend server, which responds with product data in JSON format. This data is then fed into the product loader screen, where users can enter a value for the alert. The input data is stored locally on the device using the Room library. Background processing and cost management: Background processing managed by WorkManager continuously runs on the machine, ensuring seamless management. The service retrieves data from a local database and makes API calls for each object using the OkayHTTP library. The new value of the item is compared to the user-defined expected value. If the new value matches or falls below the expected value, a notification is triggered to alert the user.

3. MODELING AND ANALYSIS

The activity diagram visually represents the workflow and dynamic aspects of a project, illustrating the sequential flow of activities and interactions among components to enhance understanding and facilitate efficient project management.

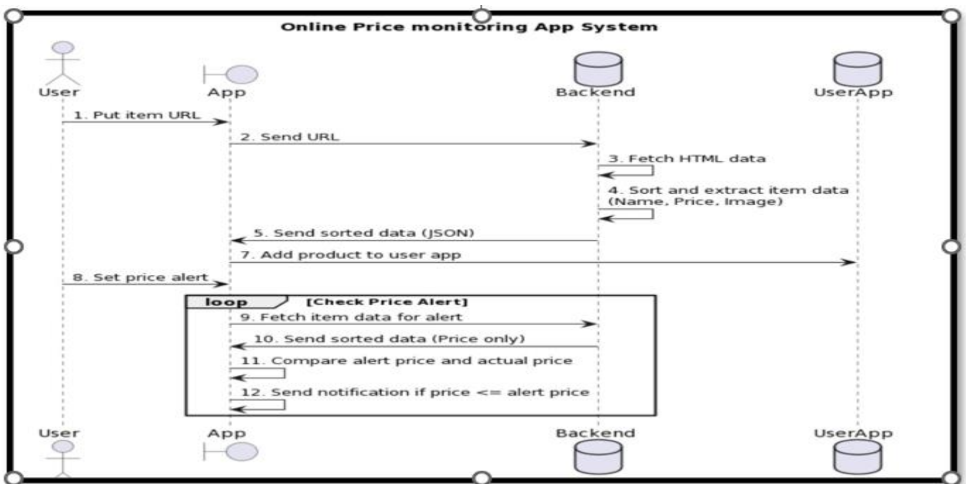

Online Price monitoring App System
User
App
Backend
UserApp
1. Put item URL
2. Send URL
3. Fetch HTML data
4. Sort and extract item data (Name, Price, Image)
5. Send sorted data (JSON)
7. Add product to user app
8. Set price alert
loop
[Check Price Alert]
9. Fetch item data for alert
10. Send sorted data (Price only)
11. Compare alert price and actual price
12. Send notification if price <= alert price
User
App
Backend
UserApp

Figure 1: Activity Diagram

.A use case diagram visually represents how users interact with a system, outlining the system's functionalities and the various ways users can engage with it.

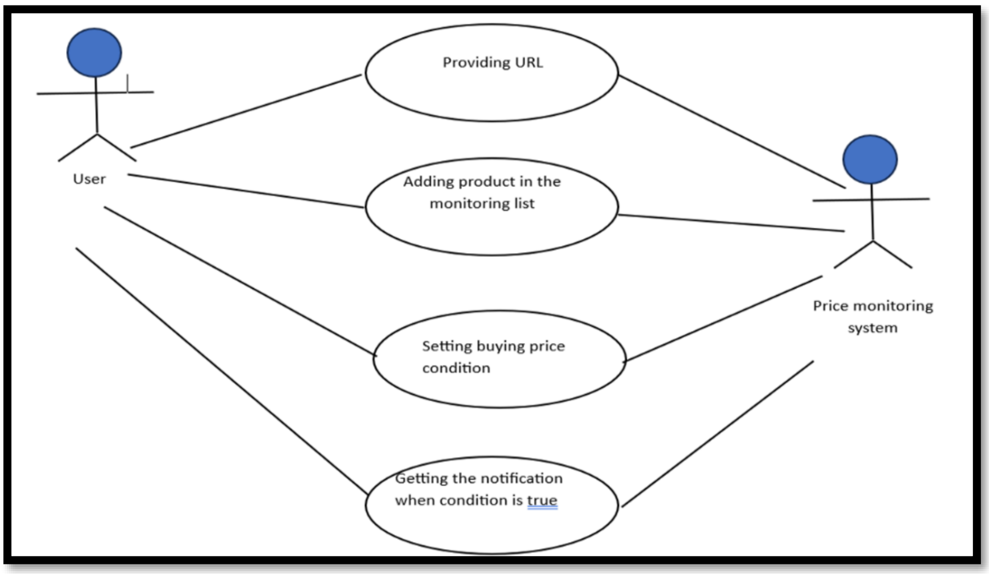
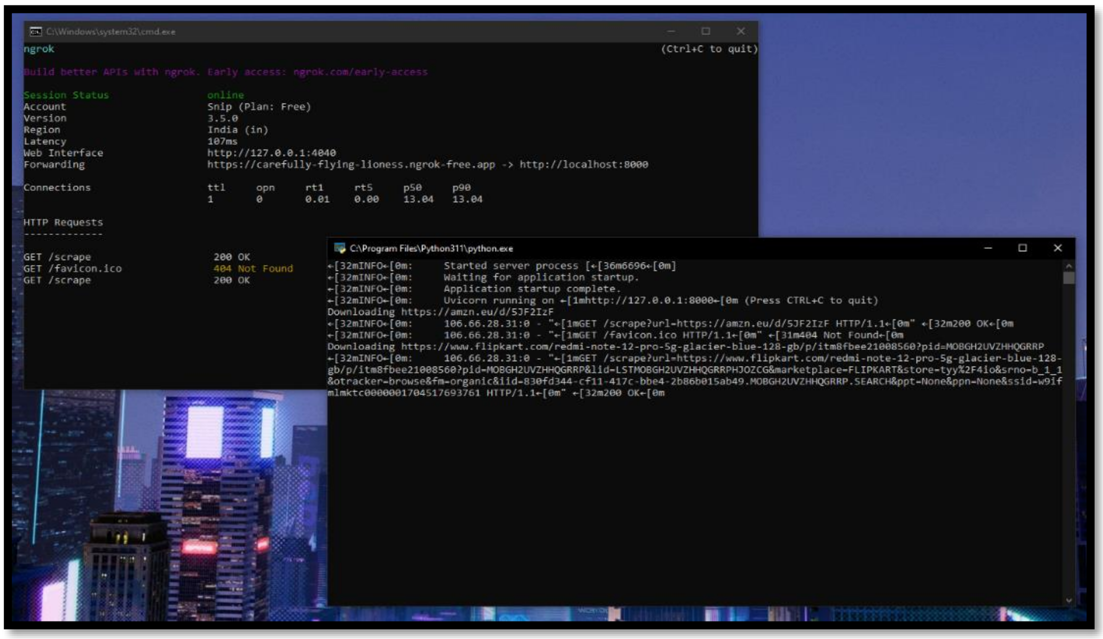

User
Providing URL
Adding product in the monitoring list
Setting buying price condition
Getting the notification when condition is true
Price monitoring system

Figure 2: Use Case Diagram

4. RESULTS AND DISCUSSION

The below image showcases the simultaneous operation of the NGROK service and a localhost server. NGROK plays a crucial role in elevating the accessibility of the localhost server to a global scale, enabling universal access to the hosted program.



```
C:\Windows\system32\cmd.exe
ngrok                                                                                    (Ctrl+C to quit)

Build better APIs with ngrok. Early access: ngrok.com/early-access

Session Status                online
Account                       Snip (Plan: Free)
Version                       3.5.0
Region                        India (in)
Latency                       107ms
Web Interface                 http://127.0.0.1:4040
Forwarding                    https://carefully-flying-lioness.ngrok-free.app -> http://localhost:8000

Connections                   ttl     opn     rt1     rt5     p50     p90
                              1       0       0.01    0.00    13.04   13.04

HTTP Requests
-------------

GET /scrape                    200 OK
GET /favicon.ico               404 Not Found
GET /scrape                    200 OK
```

```
C:\Program Files\Python311\python.exe
←[32mINFO←[0m:     Started server process [←[36m6696←[0m]
←[32mINFO←[0m:     Waiting for application startup.
←[32mINFO←[0m:     Application startup complete.
←[32mINFO←[0m:     Uvicorn running on ←[1mhttp://127.0.0.1:8000←[0m (Press CTRL+C to quit)
Downloading https://amzn.eu/d/5JF2IzF
←[32mINFO←[0m:     106.66.28.31:0 - "←[1mGET /scrape?url=https://amzn.eu/d/5JF2IzF HTTP/1.1←[0m" ←[32m200 OK←[0m
←[32mINFO←[0m:     106.66.28.31:0 - "←[1mGET /favicon.ico HTTP/1.1←[0m" ←[31m404 Not Found←[0m
Downloading https://www.flipkart.com/redmi-note-12-pro-5g-glacier-blue-128-gb/p/itm8fbee21008560?pid=MOBGH2UVZHHQGRRP
←[32mINFO←[0m:     106.66.28.31:0 - "←[1mGET /scrape?url=https://www.flipkart.com/redmi-note-12-pro-5g-glacier-blue-128-
gb/p/itm8fbee21008560?pid=MOBGH2UVZHHQGRRP&lid=LSTMOBGH2UVZHHQGRRPHJOZCG&marketplace=FLIPKART&store=tyy%2F4io&srno=b_1_1
&otracker=browse&fm=organic&iid=830fd344-cf11-417c-bbe4-2b86b015ab49.MOBGH2UVZHHQGRRP.SEARCH&ppt=None&ppn=None&ssid=w91f
mlmktc0000001704517693761 HTTP/1.1←[0m" ←[32m200 OK←[0m
```

Figure 1. Execution of NGROK service and localhost

The below image illustrates the data returned in JSON format after sending a request to the hosted program through RestAPI. The request is sent to the hosted program using a link containing the NGROK subdomain name and a product link. The hosted program identifies the product's platform based on the link, proceeds to download, and scrape the product data, and subsequently returns the extracted information in JSON format.

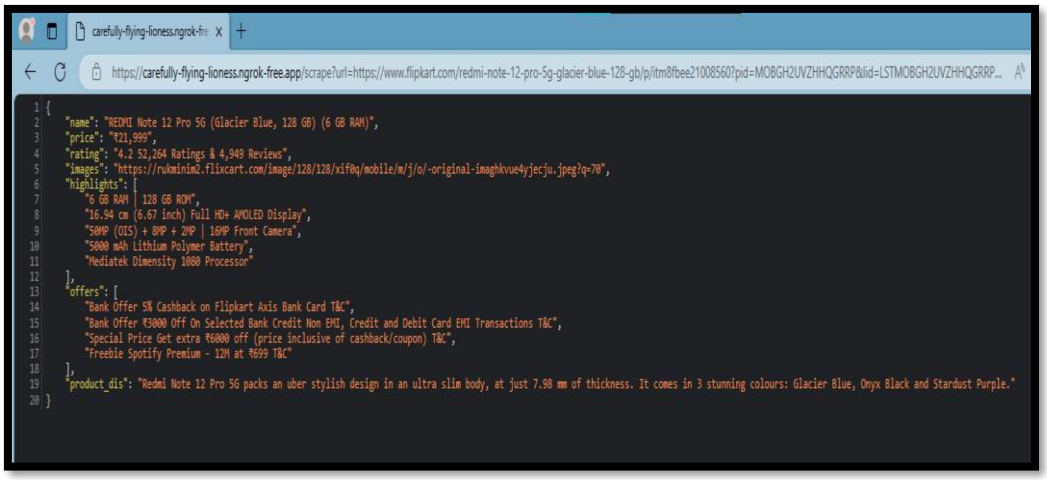
RestApi link: https://carefully-flying-lioness.ngrok-free.app/scrape?url=https://amzn.eu/d/5JF2IzF

Here,

https://carefully-flying-lioness.ngrok-free.app is a NGROK subdomain "carefully-flying-lioness".

/scrape?url= is a Path and Query Parameter.

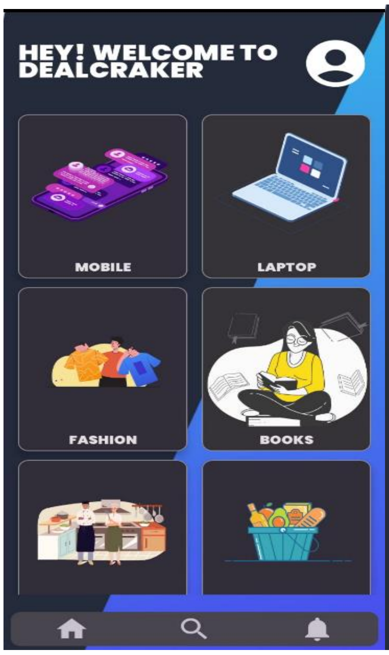
https://amzn.eu/d/5JF2IzF is a link of product in Amazon Website.

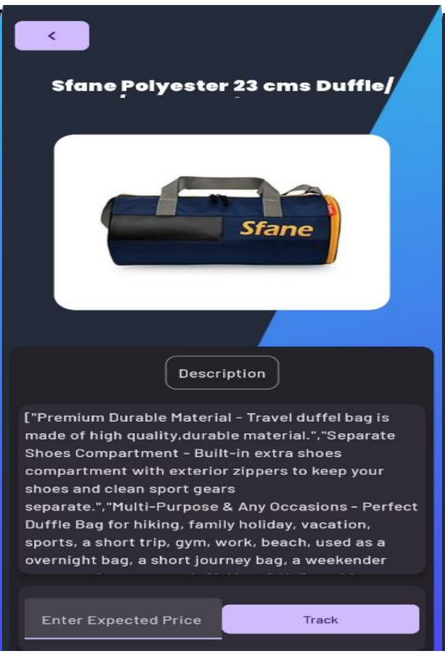

carefully-flying-lioness.ngrok-fre
https://carefully-flying-lioness.ngrok-free.app/scrape?url=https://www.flipkart.com/redmi-note-12-pro-5g-glacier-blue-128-gb/p/itm8fbee21008560?pid=MOBGH2UVZHHQGRRP&lid=LSTMOBGH2UVZHHQGRRP...

```
{
    "name": "REDMI Note 12 Pro 5G (Glacier Blue, 128 GB) (6 GB RAM)",
    "price": "₹21,999",
    "rating": "4.2 52,264 Ratings & 4,949 Reviews",
    "images": "https://rukminim2.flixcart.com/image/128/128/xif0q/mobile/m/j/o/-original-imaghkvue4yjecju.jpeg?q=70",
    "highlights": [
        "6 GB RAM | 128 GB ROM",
        "16.94 cm (6.67 inch) Full HD+ AMOLED Display",
        "50MP (OIS) + 8MP + 2MP | 16MP Front Camera",
        "5000 mAh Lithium Polymer Battery",
        "Mediatek Dimensity 1080 Processor"
    ],
    "offers": [
        "Bank Offer 5% Cashback on Flipkart Axis Bank Card T&C",
        "Bank Offer ₹3000 Off On Selected Bank Credit Non EMI, Credit and Debit Card EMI Transactions T&C",
        "Special Price Get extra ₹6000 off (price inclusive of cashback/coupon) T&C",
        "Freebie Spotify Premium - 12M at ₹699 T&C"
    ],
    "product_dis": "Redmi Note 12 Pro 5G packs an uber stylish design in an ultra slim body, at just 7.98 mm of thickness. It comes in 3 stunning colours: Glacier Blue, Onyx Black and Stardust Purple."
}
```

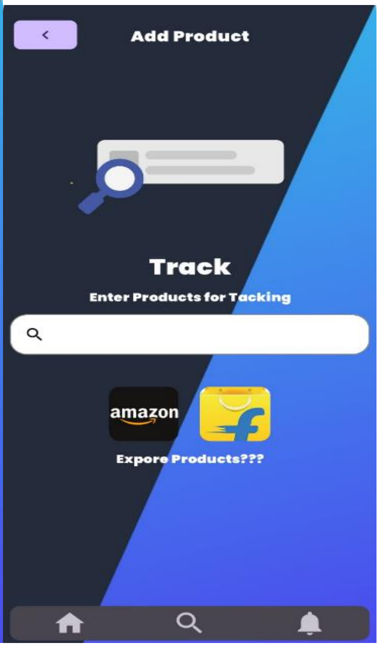
Figure 2. Product Data in JSON format is returned after scraping

Following are the snapshots of Deal Craker Application:

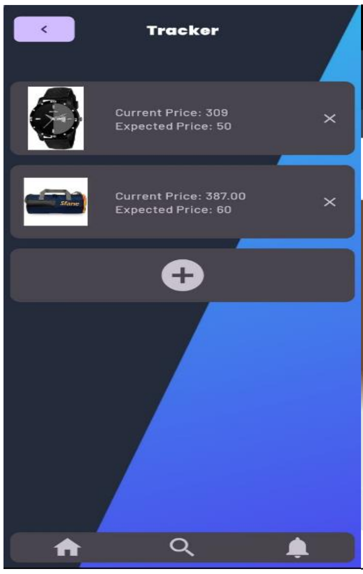


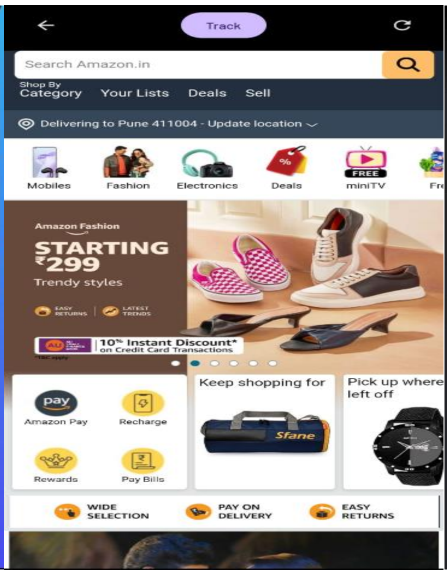


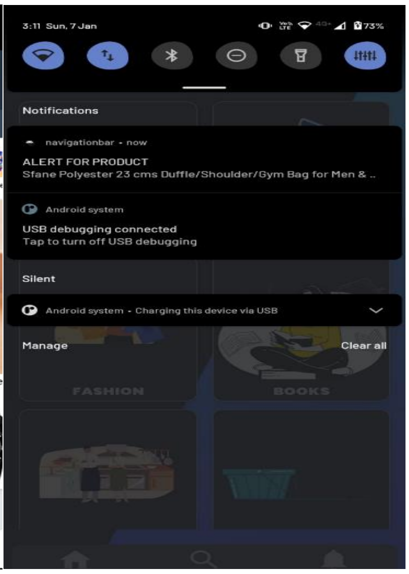
["Premium Durable Material - Travel duffel bag is made of high quality,durable material.","Separate Shoes Compartment - Built-in extra shoes compartment with exterior zippers to keep your shoes and clean sport gears separate.","Multi-Purpose & Any Occasions - Perfect Duffle Bag for hiking, family holiday, vacation, sports, a short trip, gym, work, beach, used as a overnight bag, a short journey bag, a weekender



Home Page Search Page Product Loader Page







ALERT FOR PRODUCT
Sfane Polyester 23 cms Duffle/Shoulder/Gym Bag for Men & ..

USB debugging connected
Tap to turn off USB debugging

List of Product Being Tracked WebView for Amazon and Flipkart Notification after Set Price achieved.

5. CONCLUSION

In Conclusion, the need of applications like Deal Craker is increasing rapidly as more and more people are now shifting to Online Shopping. In future the number of online shopping platforms will increase which will cause headaches to customers to find a perfect deal among various platforms. So mobile application empowers users or customers to effortlessly track and compare product prices across major E-commerce platforms. In summary, our Price Monitoring Application stands as a testament to the successful integration of advanced features, user-centric design, and functionality, ultimately empowering users to make informed and cost-effective purchasing decisions in the dynamic landscape of online shopping.

ACKNOWLEDGEMENTS

With a deep sense of gratitude, we would like to thank all the people who have lit our path with their kind guidance. We are very grateful to these intellectuals who did their best to help during our project work planning. It is our proud privilege to express a deep sense of gratitude to Prof. P.M. Dharmadhikari, Principal of Sandip Polytechnic, Nashik, for his comments and kind permission to complete this project work planning. We remain indebted to Prof. G. K. Gaikwad, H.O.D, Computer Engineering Department for their timely suggestion and valuable guidance. Our special gratitude goes to my guide Prof. Rikeeta C. Mahajan and staff members, technical staff members of Computer Engineering Department for their expensive, excellent, and precious guidance in completion of this work planning. We thank all our colleagues for their appreciable help for our project work planning. With various industry owners or lab technicians to help, it has been our endeavor to throughout our work to cover the entire project work planning. And lastly, we thanks to our all friends and the people who are directly or indirectly related to our project work planning.

6. REFERENCES

[1] https://chat.openai.com/ for learning and understanding.
[2] https://www.youtube.com/ @PhilippLackner for android development
[3] https://github.com for uploading our project files.
[4] https://ngrok.com for NGROK Service
[5] https://flask.palletsprojects.com/en/3.0.x/ for flask service
[6] https://fastapi.tiangolo.com for fastapi service
[7] https://pypi.org/project/selectorlib/ for python selectorlib library
[8] https://pypi.org/project/requests/ for python request library
[9] https://www.amazon.in for product link and data
[10] https://www.flipkart.com for product link and data
[11] https://www.figma.com for UI/UX reference
[12] https://developer.android.com/courses/jetpack-compose/course for android development
[13] https://square.github.io/okhttp/ for http handling in android
[14] https://developer.android.com/topic/libraries/architecture/workmanager for scheduling tasks in android
[15] https://developer.android.com/reference/android/arch/persistence/room/RoomDatabase for database in android
[16] https://developer.android.com/codelabs/kotlin-coroutines for kotlin-coroutines
[17] https://developer.android.com/topic/libraries/architecture/viewmodel for view models
[18] https://www.geeksforgeeks.org/how-to-use-coil-image-loader-library-in-android-apps/ for image loading