

PROJECT REPORT

ON

Track My Deal

BY

Akshay Jegaonkar

**SAVITRIBAI PHULE PUNE UNIVERSITY
MASTER IN COMPUTER APPLICATION
MAHARASHTRA EDUCATION SOCIETY's
INSTITUTE OF MANAGEMENT AND CAREER COURSES
(IMCC), PUNE-411038
2024-25**



MAHARASHTRA EDUCATION SOCIETY'S
(SINCE 1860)

INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC) (AUTONOMOUS)

Approved by AICTE and Recognized by Savitribai Phule Pune University, Pune
IMCC Campus, 131, Mayur Colony, Kothrud, Pune-411038, Maharashtra, India | Ph.: 020-2546 6271 / 73 | e-mail: info.imcc@mespune.in | <https://imcc.mespune.in>

NAAC Accredited with Grade A+

Ref. No. MES IMCC /345/2024-25

Date: 03/04/2025

CERTIFICATE

This is to certify that the Project Report entitled

"TRACKMYDEAL"

is prepared by

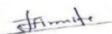
Akshay Jegaonkar

M.C.A. Semester IV Course for the Academic Year 2024-25 at M.E. Society's Institute of Management & Career Courses (IMCC), Pune - 411038.

M.C.A Course is affiliated to Savitribai Phule Pune University.

To the best of our knowledge, this is original study done by the said student and important sources used by him have been duly acknowledged in this report.

The report is submitted in partial fulfillment of M.C.A Course for the Academic Year 2024-25 as per the rules and prescribed guidelines of Savitribai Phule Pune University.


Dr. Ravikant Zirmite
Head, Dept of MCA MES IMCC


Dr. Santosh Deshpande
Director, MES IMCC

Internal Examiner

External Examiner

Head Office: 'MES Bhavan', 1214-1215, Sadashiv Peth, Pune - 411030, Maharashtra, India. | Ph.: +91-020-41038100 | www.mespune.in

CERTIFICATE

This is to certify that the student **Akshay Jegaonkar** has completed the project work entitled "**TRACKMYDEAL**" under my guidance. The report is submitted in partial fulfillment of M.C.A. Course for the Academic Year 2024-2025 as per the rules & prescribed guidelines of Savitribai Phule Pune University.

His work is found to be satisfactory and complete in all respects.

Ms. Kalpana Dhende
(Internal Project Guide)

Acknowledgement

I take this opportunity to express my heartfelt gratitude to all those who supported and guided me throughout the development of my project titled “**Track My Deal.**”

First and foremost, I would like to thank my college guide, **Ms. Kalpana Dhende**, for her invaluable guidance, constant encouragement, and insightful suggestions at every stage of the project. Her support was instrumental in shaping the direction and success of this work.

I am also sincerely thankful to my external guide, **Mr. Nitin Kawade**, for his practical insights, technical expertise, and unwavering support, which greatly contributed to the successful implementation of the application.

My heartfelt thanks to our Head of Department, **Prof. Ravikant Zirmite**, for providing me with the necessary resources, academic support, and motivation throughout this project.

I extend my sincere gratitude to our Director, **Prof. Dr. Santosh Deshpande**, for his continuous encouragement, guidance, and support that served as a great source of inspiration.

I would also like to thank all the teaching and non-teaching staff of our college for their timely help, cooperation, and support throughout this journey.

Lastly, I am grateful to my friends and classmates for their valuable feedback, encouragement, and consistent motivation, which played a vital role in keeping me focused and driven during the entire course of this project.

Index

Chapter No		Details	Page No
1		Introduction	1
	1.1	Institute Profile	3
	1.2	Abstract	4
	1.3	Existing System and Need for System	6
	1.4	Scope of System	8
	1.5	Operating Environment - Hardware and Software	10
	1.6	Brief Description of Technology Used	12
2		Proposed System	16
	2.1	Study of Similar Systems	18
	2.2	Feasibility Study	20
	2.3	Objectives of Proposed System	22
	2.4	Users of System	24
3		Analysis and Design	
	3.1	System Requirements (Functional and Non-Functional requirements)	26-29
	3.2	Entity Relationship Diagram (ERD)	30

	3.3	Table Structure	31
	3.4	Use Case Diagrams	32
	3.5	Class Diagram	33
	3.6	Activity Diagram	34
	3.7	Sequence Diagram	35
	3.8	Module Hierarchy Diagram	36
	3.9	Sample Input and Output Screen	37-44
4		Coding	
	4.1	Algorithms	45-47
	4.2	Code snippets	48-52
5		Testing	
	5.1	Test Strategy	53-55
	5.2	Unit Test Plan	56-59
	5.3	Acceptance Test Plan	60-61
	5.4	Test Case	62-63
6		Limitations of Proposed System	64-65
7		Conclusion	66
8		Bibliography	
9		User Manual	

CHAPTER 1: INTRODUCTION

Introduction

In today's highly competitive and digitally driven market, online shopping has become a preferred choice for millions of consumers. With the rise of platforms like Amazon, Flipkart, and eBay, users have access to a vast array of products at varying prices. However, keeping track of product prices across multiple platforms can be overwhelming and time-consuming. Price fluctuations happen frequently due to offers, sales, and seasonal discounts, and users often miss out on the best deals simply because they cannot monitor them constantly.

TrackMyDeal is a web-based application developed to solve this issue by allowing users to track the price history of products across various e-commerce platforms and receive alerts when their desired product reaches a target price. The application is designed with a focus on user convenience, affordability, and automation. It serves as a digital assistant for online shoppers, enabling them to make smarter, more cost-effective purchasing decisions without the need to manually revisit multiple websites.

The system provides a clean and intuitive user interface where users can register and log in securely, search for products, set price drop alerts, and get notified via email when their selected product hits the desired price. It utilizes web scraping and external APIs to fetch real-time product data, while background schedulers track

pricing changes periodically. Notification services like SMTP are integrated to ensure users are informed instantly when deals become available.

One of the distinguishing features of **TrackMyDeal** is its ability to build and present a price trend chart over time, allowing users to view the historical pricing pattern of a product. This feature empowers users with insights to decide the best time to buy a product. Additionally, the system promotes affordability by helping users avoid overpaying and encouraging price-conscious shopping behavior.

Ultimately, **TrackMyDeal** bridges the gap between online buyers and the overwhelming amount of pricing information available on the internet. By simplifying the decision-making process and enhancing the online shopping experience, this application provides a practical and scalable solution for modern consumers who value both convenience and savings.

1.1 Institute Profile

The Institute popularly known as IMCC was established in 1983 by M. E. Society for providing quality education and technical expertise at the Post Graduation Level in the Fields of Computers and Management. The Institute is recognized by SPPU under Section 46 of Pune University Act, 1974 and Section 85 of Maharashtra University Act, 1994. The Institute is located at 131, Mayur Colony, Kothrud, Pune-411 038.

The main motto of the Institute is to instil the concepts of total personality development in the students. The emphasis is laid on 'Teacher Disciple Relationship' in place of 'Boss Subordinate'

The pre-amble of IMCC "FACTA-NON-VERBA" lucidly means that the Institute produces the new breed of professionals, who's deeds will speak and there could be no requirement of pomposity. The zooming enthusiastic, rationale and excellent external endeavours are being imbibed in the students to prove their mettle. The conducive milieu of the Institute moulds the budding managers to reveal in managing flexibility, integration, change and transformation. These 'would be' professionals are channelised in such a way to 'orchestrate' and deploy business and technological management skills in a synergistic manner to grab the tangible success.

1.2 Abstract

The **TrackMyDeal** application is a web-based platform designed to assist users in tracking product prices across various ecommerce platforms and to notify them when a product reaches their desired target price. The primary goal of this project is to simplify the process of monitoring price changes and help users make informed purchasing decisions, all while ensuring a smooth and automated user experience.

In today's e-commerce-driven market, users frequently encounter the challenge of fluctuating product prices and limited-time offers. Monitoring these price changes manually across multiple platforms is tedious and inefficient. **TrackMyDeal** addresses this issue by offering a centralized solution that allows users to search for products, set price alerts, and receive real-time notifications when deals become available.

The system is developed using Python and Django for the backend, with HTML, CSS, and JavaScript for the frontend, and integrated email services for timely notifications. It uses web scraping or APIs to collect real-time pricing information, stores product history in a database, and presents it in a user-friendly dashboard. Users can view historical price trends, set target price thresholds, and be notified automatically via email when a product matches their conditions.

Beyond individual convenience, **TrackMyDeal** promotes smarter and more economical shopping behavior by helping users avoid overpaying and make purchases at the most opportune time. It enhances the e-commerce experience with data-driven insights and automation, enabling users to shop with confidence.

This project exemplifies the practical application of web technologies and automation to solve real-world problems, offering a scalable, efficient, and user-centric solution to modern online shopping challenges.

1.3 Existing System and Need for System

In the current digital marketplace, consumers frequently struggle with monitoring the fluctuating prices of products on various ecommerce websites such as Amazon, Flipkart, and others. The traditional approach often involves bookmarking product pages or manually checking prices daily. This process is not only tedious and time-consuming but also inefficient, as users may miss out on significant price drops or flash sales due to the lack of timely updates.

Although browser extensions and limited price tracking websites exist, they typically offer restricted functionality, support only specific platforms, or lack customization options such as target price alerts. Moreover, many of these tools are not tailored for the Indian e-commerce ecosystem and often come with complex interfaces or paid subscriptions that limit their accessibility.

From the consumer's perspective, there is a clear demand for a system that can provide accurate, real-time product pricing information and timely alerts without requiring constant manual intervention. On the other hand, for e-commerce platforms and affiliate marketers, such a system offers an opportunity to drive traffic and conversions through targeted deal promotion.

TrackMyDeal is designed to fill this gap by offering an automated, centralized platform that allows users to add product URLs, set a desired target price, and receive email notifications when the product price drops to or below that target. The platform also tracks the historical pricing of products, offering users a data driven way to determine the best time to make a purchase.

In addition, **TrackMyDeal** supports affiliate integration, enabling monetization through redirection to partner sites when users choose to make a purchase. This creates a win-win situation by not only benefiting consumers but also generating potential revenue through referrals.

In conclusion, the current ecosystem lacks an accessible, intuitive, and fully automated platform tailored for Indian users to track product prices and grab the best deals. The need for a solution like **TrackMyDeal** is critical in helping users make informed and cost-effective buying decisions while simplifying their online shopping experience.

1.4 Scope Of System

The **TrackMyDeal** system is developed with scalability, automation, and user convenience in mind. It includes a set of well-defined features for both consumers and administrators, ensuring an efficient price tracking experience and potential integration with affiliate and promotional ecosystems.

For Users:

- Register and log in securely via Email, Google, or Facebook.
- Add product URLs from e-commerce platforms such as Amazon and Flipkart.
- Set a target price for each tracked product.
- Automatically receive email alerts when the product price drops to or below the target.
- View a history of price changes over time for better decision making.
- Access a personal dashboard to manage tracked items.
- Update profile information such as name, email, and preferences.

For Admin:

- Access and manage all registered user accounts.
- Monitor system activity such as product tracking trends and price alert triggers.
- Manage affiliate redirection links and promotional banners.
- Moderate the platform, ensuring security and performance efficiency.

System Features and Future Expansion:

- The platform has been designed to easily support future enhancements such as:
- **WhatsApp and Telegram Notifications** for real-time alerts.
- **Mobile Application** (Android/iOS) for on-the-go tracking and notifications.
- **AI-Based Price Predictions** to suggest the best time to buy a product.
- **Multi-Site Price Comparison** to compare deals across various platforms.
- **Admin Dashboard with Analytics** to track active users, trending products, and performance

Overall, the system aims to bridge the gap between smart shopping and user convenience by offering automated, accurate, and scalable price tracking functionality that can adapt to growing user demands and evolving e-commerce trends.

1.5 Operating Environment

The development and deployment of the **TrackMyDeal** system require specific hardware and software configurations to ensure efficient operation, testing, and user access. The platform is designed as a web-based system and can be accessed via browsers on desktops, laptops, and mobile devices.

- **Hardware Requirements:**

- **Client-Side (User Device):**

- Desktop, Laptop, Tablet, or Mobile Phone
- Minimum 2 GB RAM
- Internet Connectivity (Wi-Fi or Mobile Data)
- Screen Resolution: 720p or higher

- **Development & Server Machine:**

- Windows/Linux/macOS Operating System
- Processor: Intel i5 / AMD Ryzen 5 or above
- RAM: Minimum 8 GB
- Storage: At least 100 GB free disk space
- Network: Reliable broadband connection for deployment and testing

- **Software Requirements:**
 - **Development Environment:**
 - **Operating System:** Windows 11 / Ubuntu 22.04 / macOS Ventura (for developers)
 - **Backend Framework:** Python 3.11+ with Django 4.x
 - **Frontend:** React.js with Tailwind CSS or Bootstrap
- **Database:** SQLite (for development), PostgreSQL (for production deployment)
- **Web Scraping Libraries:** BeautifulSoup, Requests, or Scrapy
- **Notification Services:** Gmail SMTP, SendGrid API (for email), Twilio (for WhatsApp - optional)
- **Version Control:** Git and GitHub
- **Testing Tools:** Postman, Pytest, Browser Dev Tool
- **Deployment Options:**
 - **Cloud Platform:** Render, Railway, or Heroku (for Django & PostgreSQL deployment) o
 - **Frontend Hosting:** Netlify or Vercel

1.6 Brief Description of Technology Used

1.6.1 Operating System Used – Windows-11:

- Windows 11 was used as the primary development operating system throughout the development of **TrackMyDeal**. Its modern design, improved system performance, and advanced multitasking features made it an ideal environment for web development. It provided seamless integration with tools such as Visual Studio Code, Python, Git, Postman, and Docker. Windows 11's support for WSL (Windows Subsystem for Linux) also made it easier to manage backend environments, enabling developers to switch between Windows and Linux-based operations during deployment and testing.

1.6.2 Python (Django Framework):

- Python was selected as the core backend programming language due to its simplicity, readability, and vast ecosystem of libraries. The Django framework, built on Python, was used to develop the backend of TrackMyDeal. Django follows the Model-View-Template (MVT) architecture and includes built-in tools for authentication, database ORM, security, and admin management. It allowed for rapid development, with scalability and flexibility. Django REST Framework (DRF) was utilized to build secure and reusable APIs to connect with the fronten

1.6.3 React.js:

- React.js was used to create the frontend interface of TrackMyDeal. This JavaScript library enables the development of dynamic and responsive user interfaces with component-based architecture. React's virtual DOM, unidirectional data flow, and support for reusable components ensured a fast and maintainable UI. Tailwind CSS was used alongside React to style the application and provide a clean and modern look.

1.6.4 SQLite and PostgreSQL:

- For initial development and testing, SQLite was used due to its lightweight and serverless nature. It is integrated natively with Django and ideal for small-scale development environments. For production, PostgreSQL was preferred as it is a powerful, scalable, and secure relational database system. It supports advanced queries, indexing, and performance optimization features necessary for large-scale data handling.

1.6.5 BeautifulSoup/Scrapy (Web Scraping):

- BeautifulSoup and Scrapy were used to extract product data from e-commerce platforms. These Python-based libraries are powerful tools for parsing HTML/XML and retrieving structured data such as product titles, prices, images, and descriptions. This enabled TrackMyDeal to periodically fetch updated product prices from userprovided URLs.

1.6.6 SMTP (Gmail Integration):

- Gmail's SMTP service was integrated to handle email notifications. Whenever a product's price dropped to or below the user's set target, the system triggered an automated email alert. Django's built-in EmailMessage module was configured to send transactional emails securely via SMTP authentication.

1.6.7 Git and GitHub:

- Git was used as the version control system throughout the project lifecycle, ensuring proper code management, backup, and collaboration. GitHub hosted the central repository, making it easy to track changes, manage branches, and collaborate efficiently with team members.

1.6.8 Postman and Browser Dev. Tools:

- Postman was utilized for API testing, allowing verification of request/response structures and endpoint performance. Additionally, browser developer tools were used to debug frontend issues, test responsiveness, and monitor live interactions between frontend and backend component

1.6.9 Hosting Tools – Netlify and Render:

- The frontend (React.js) was deployed on **Netlify**, which provided continuous integration with GitHub and HTTPS support. The Django backend and PostgreSQL database were deployed on **Render**, a scalable cloud platform that supports Python applications with custom build and deploy pipelines.

This technology stack ensures that **TrackMyDeal** is robust, scalable, and ready for both development and production use, while also allowing for future feature integration and performance optimization.

CHAPTER 2: PROPOSED SYSTEM

Proposed System

The proposed system, **TrackMyDeal**, is a smart, web-based application designed to help online shoppers track product prices and receive instant notifications when their desired products reach a target price. It provides an intuitive and secure platform where users can register, input product URLs from supported ecommerce websites (like Amazon and Flipkart), and set a target price to be notified when deals become available.

Unlike traditional deal-hunting approaches that involve manual price checking or browser extensions with limited scope, **TrackMyDeal** offers a fully automated and centralized system that regularly scrapes product prices and maintains a price history. Once a product hits the user-defined price point, the system automatically sends an email alert, allowing users to make informed and timely purchases.

The system supports secure user authentication through Email, Google, and Facebook logins, along with a personalized dashboard to manage tracked products, view past alerts, and monitor price trends. On the backend, Django handles the logic and API interactions, while data is securely stored in SQLite (development) or PostgreSQL (production).

Beyond the convenience of automation, **TrackMyDeal** promotes smarter, budget-conscious shopping and aims to integrate affiliate marketing features in the future. This allows redirection to ecommerce platforms via affiliate links—generating commission revenue without affecting user experience or price transparency. In addition to enhancing the online shopping experience, the platform is built with future scalability in mind. It is designed to integrate additional features such as WhatsApp notifications, Telegram alerts, AI-powered deal predictions, and a mobile friendly interface for users on the go.

By combining real-time product tracking, historical price analytics, and notification mechanisms, **TrackMyDeal** presents a reliable, secure, and user-centric solution that simplifies online deal hunting while supporting e-commerce ecosystem efficiency.

2.1 Study of Similar Systems

In the current digital commerce landscape, several tools and platforms exist that offer basic price tracking features, such as **CamelCamelCamel**, **Keepa**, and **Price Tracker for Amazon**. These systems are primarily browser-based or limited to specific platforms like Amazon and do not support broader, crossplatform tracking functionality. Moreover, many of these solutions offer limited customization for setting target price alerts and lack seamless integration with email notifications or affiliate marketing capabilities.

Some extensions and apps are difficult to use for the average consumer due to cluttered interfaces or paid-only features. Additionally, many of these systems are developed with a global market in mind and lack optimization for Indian e-commerce platforms such as Flipkart, Myntra, or Ajio, limiting their utility for Indian users.

TrackMyDeal is proposed to address these shortcomings by offering a **fully web-based, platform-agnostic**, and **userfriendly** price tracking system specifically tailored for Indian online shoppers. It allows users to track prices across multiple websites, view historical price trends, and receive email alerts when a product's price drops to a predefined level.

Unlike many existing systems, **TrackMyDeal** is designed to be lightweight, intuitive, and accessible across all devices without needing any browser extension or app installation. It emphasizes automation, real-time communication, and local adaptability while also being scalable for affiliate marketing and future AI-based price forecasting integrations.

By focusing on simplicity, affordability, and Indian-market compatibility, **TrackMyDeal** offers a more inclusive and practical solution than currently available alternatives.

2.2 Feasibility Study

A feasibility study was carried out to evaluate the technical, economic, and operational viability of developing and deploying the **TrackMyDeal** web application:

- **Technical Feasibility:**
 - The project was developed using **Python (Django)** for the backend, **React.js** for the frontend, and tools like **BeautifulSoup** and **Scrapy** for web scraping. All selected technologies are open-source, stable, and widely used in the industry. The Django framework ensures robust backend operations, including API handling, authentication, and database management.
 - React enables the creation of responsive and dynamic user interfaces, while SMTP services (like Gmail or SendGrid) manage notification delivery. Since the system is web-based, it is accessible across all major browsers on desktops, laptops, and mobile devices, making it widely usable and technically viable.

- **Economic Feasibility:**

- The project leverages free and open-source tools, which significantly reduces development costs. The initial version uses SQLite for the database and Gmail SMTP for email delivery, both of which are free. As the platform scales, it can be migrated to PostgreSQL and hosted on platforms like Render or Heroku, which also offer free tiers for small applications.
- No significant investment is needed other than basic hosting services and internet connectivity. If affiliate marketing is successfully integrated, the system can even generate revenue to offset hosting costs.

- **Operational Feasibility:**

- TrackMyDeal is designed with a user-first approach, featuring a clean, minimalistic UI and a highly intuitive workflow. Users can easily register, log in, add product URLs, set target prices, and receive alerts without requiring any technical knowledge.
- Administrators can manage users, product entries, and system logs through backend tools. The system is flexible and scalable, allowing enhancements such as WhatsApp alerts, Telegram bots, and a mobile application in the future.
- The application requires minimal training or technical support, making it feasible to operate and maintain over time.

2.3 Objectives of Proposed System

The main goals of the **TrackMyDeal** web application are:

- **To help users track product prices** from popular ecommerce platforms such as Amazon and Flipkart in an automated and efficient manner.
- **To allow users to set custom target prices** for products they wish to purchase and receive real-time alerts when those prices are met.
- **To provide a secure and simple user authentication system** using Django's built-in authentication and third-party options like Google and Facebook logins.
- **To offer an intuitive dashboard interface** where users can view, manage, and delete their tracked products with ease.
- **To generate email notifications** using SMTP when the tracked product's price falls below or matches the set threshold, ensuring timely communication.
- **To maintain historical price data** using a clean and structured database, allowing users to analyze price trends before making a purchase decision.

- **To reduce the time and effort spent on manual price checking** and promote smarter shopping behaviour among users.
- **To support future scalability** through affiliate link integration, WhatsApp notifications, AI-based price predictions, and mobile platform extensions.
- **To create a lightweight, responsive, and browser compatible system** accessible from any modern device without requiring downloads or installations.

2.4 Users of System

The **TrackMyDeal** system is designed for two primary types of users:

1. End Users (Shoppers):

- Individuals looking to purchase products online at the best possible price.
- Can register or log in securely using email, Google, or Facebook.
- Add product URLs from e-commerce platforms like Amazon and Flipkart to their personal tracking dashboard.
- Set a target price for each product and receive notifications when the product reaches or drops below that price.
- View product details, price history, and manage tracked items.
- Get redirected to the merchant site via affiliate links to make a purchase.

2. Administrator (System Admin):

- Responsible for managing the overall platform and user base.
- Can monitor user activities and maintain database integrity.
- Manages affiliate link redirection, notification schedules, and data security settings.
- Has access to backend logs, price update processes, and can trigger manual scraping or batch price checks when needed.
- Handles system maintenance, debugging, and future upgrades.

Each user type has access to features and interfaces specifically tailored to their role. While end users interact through a responsive web dashboard, administrators work within a backend control panel to ensure smooth system performance and user satisfaction.

CHAPTER 3: ANALYSIS AND DESIGN

3.1 System Requirements

➤ Functional Requirements:

These are the core features and functionalities that the system must perform:

- **User Registration and Login**

- Users can register using email and password.
- OAuthbased login options through Google and Facebook are supported.
- Django Authentication is used to manage secure user sessions.

- **Profile Management:**

- Users can update their personal information, such as name, email, and preferences.
- User profiles are stored securely in the database and can be modified from the dashboard.

- **Product Tracking System:**

- Users can enter product URLs from supported ecommerce websites such as Amazon and Flipkart.
- The system extracts product details like title, price, and image using web scraping techniques.

Users can set a desired target price for each product.

- **Price Drop Notification System**

- A scheduled background process checks prices at regular intervals (e.g., every 6 to 12 hours).
- When a product's price drops to or below the target price, an email notification is sent automatically to the user.
- Notifications are handled via Gmail SMTP or similar email services.

- **Affiliate Redirection:**

- When a user chooses to purchase a product, the system redirects them to the merchant site through an embedded affiliate link.
- These redirects are tracked and logged for reporting and revenue generation.

- **Price History and Analytics:**

- The application stores historical price data for each tracked product.
- Users can view graphs or lists displaying how the product's price has changed over time.
- Helps users determine the best time to make a purchase.

- **Admin Dashboard:**

- Admins can view all users, products, and notification logs.
- Admins can manually trigger scraping routines, manage affiliate URLs, and resolve system errors.

➤ **Non-Functional Requirements:**

These define how the system performs and behaves under different conditions:

- **Security:**

- All user passwords are securely hashed and stored using Django's authentication system.
- Sessions are protected using JWT or CSRF tokens to prevent unauthorized access.
- Direct product URLs and affiliate links are protected from hijacking or abuse.

- **Scalability:**

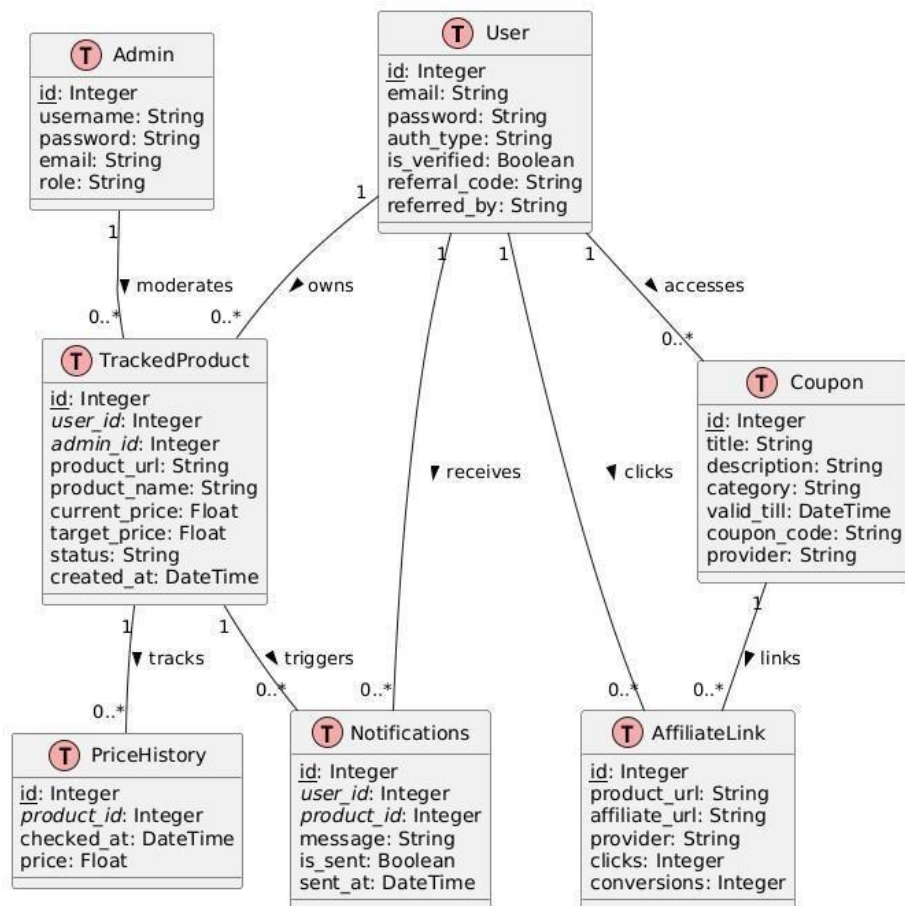
- The system is designed to begin with a lightweight database (SQLite) and can scale to PostgreSQL or other RDBMS as user demand increases.
- Scraping and notification services can be scaled using task queues or scheduled jobs.

- **Usability and Accessibility:**

- The user interface is designed using React.js and Tailwind CSS for responsiveness and simplicity.
- The application is accessible on desktops, tablets, and mobile browsers.

- Users of all technical backgrounds can navigate and use the platform with minimal guidance.
- **Performance and Reliability:**
 - Background jobs (such as price checks and email alerts) are designed to run efficiently without overloading the system.
 - Retry mechanisms ensure delivery of email notifications even in case of temporary failures.
 - Data redundancy and logging ensure high reliability and traceability.

3.2 Entity Relationship Diagram



3.3 Table Structure

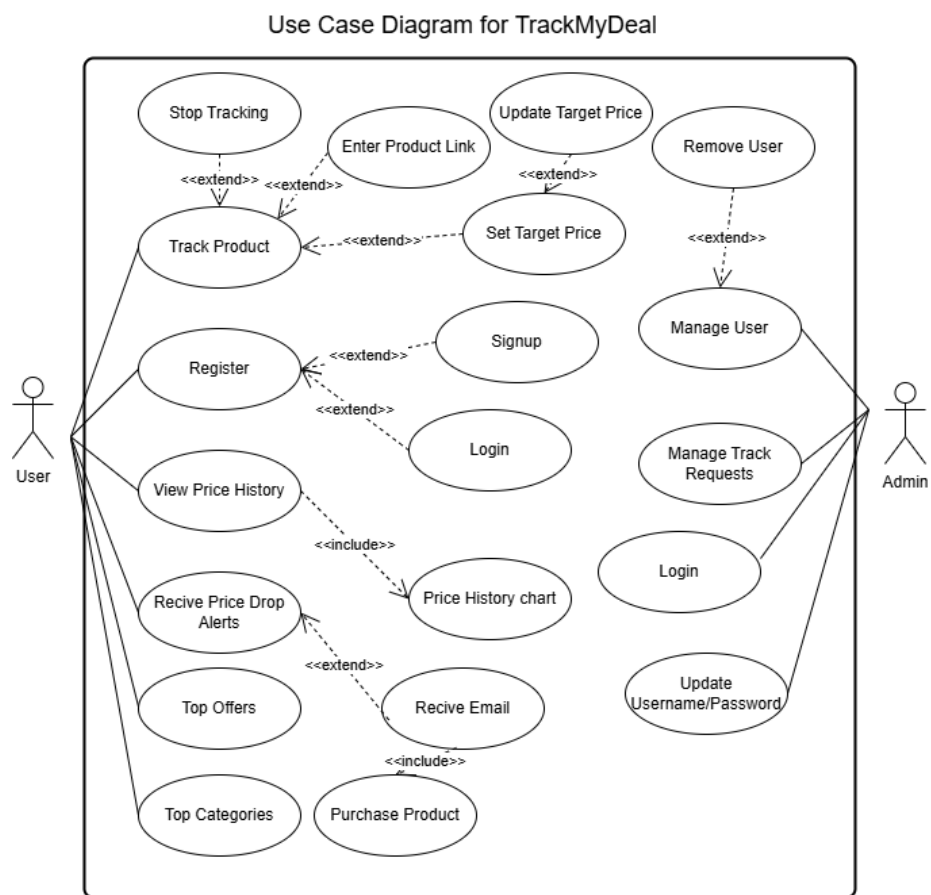
➤ User Table:

Field Name	Data Type	Description
role	String	Role of the user
email	String	User's email address
username	String	Username of the user
password	String	Password of the user
mobile	String	Contact number of the user

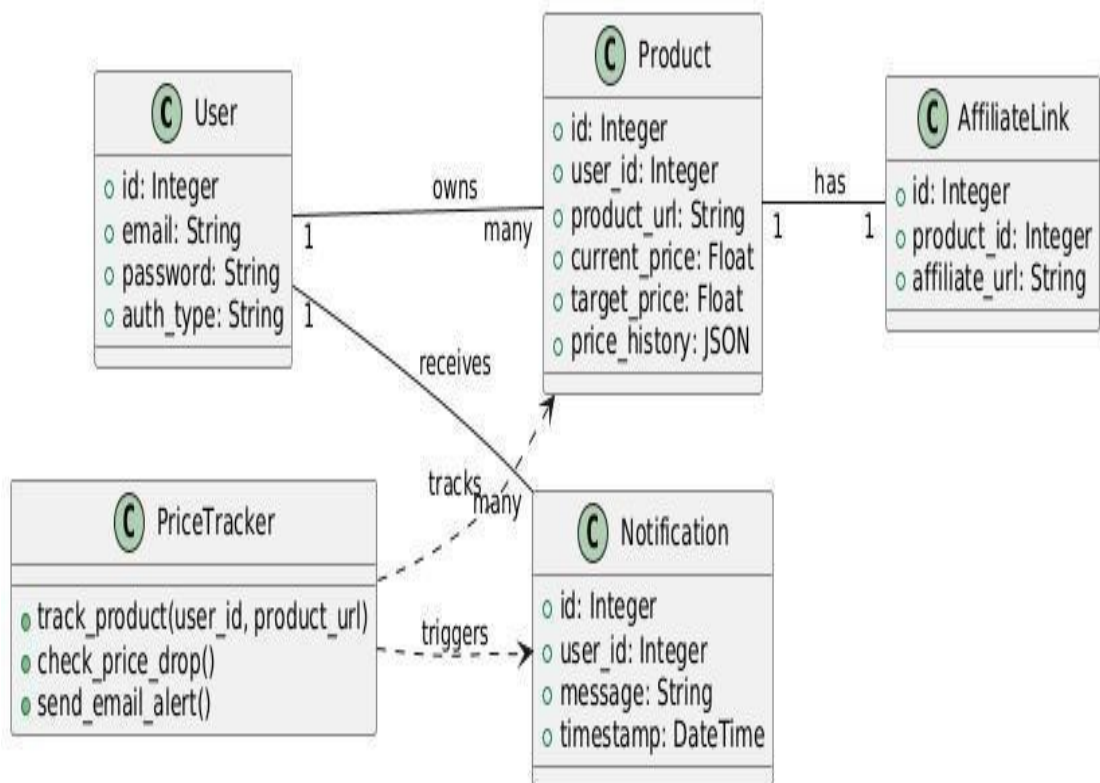
➤ Tracked Product Table:

Field Name	Data Type	Description
id	Integer	Primary Key, Unique Product ID
userid	Integer	Foreign Key – Associated User ID
product_url	String	URL of the product
current_price	Float	Latest fetched price
target_price	Float	User-defined target price

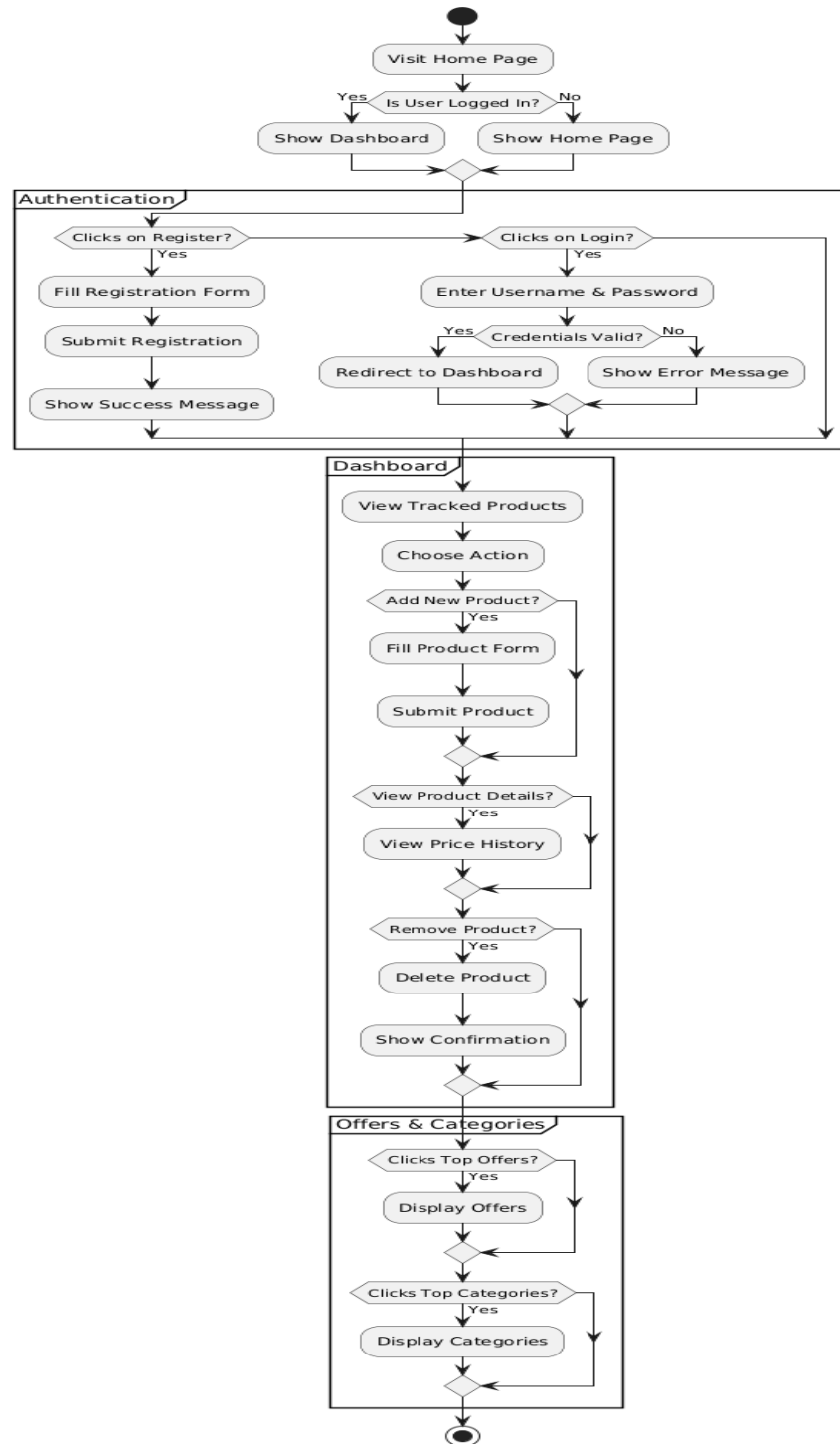
3.4 Use Case Diagram



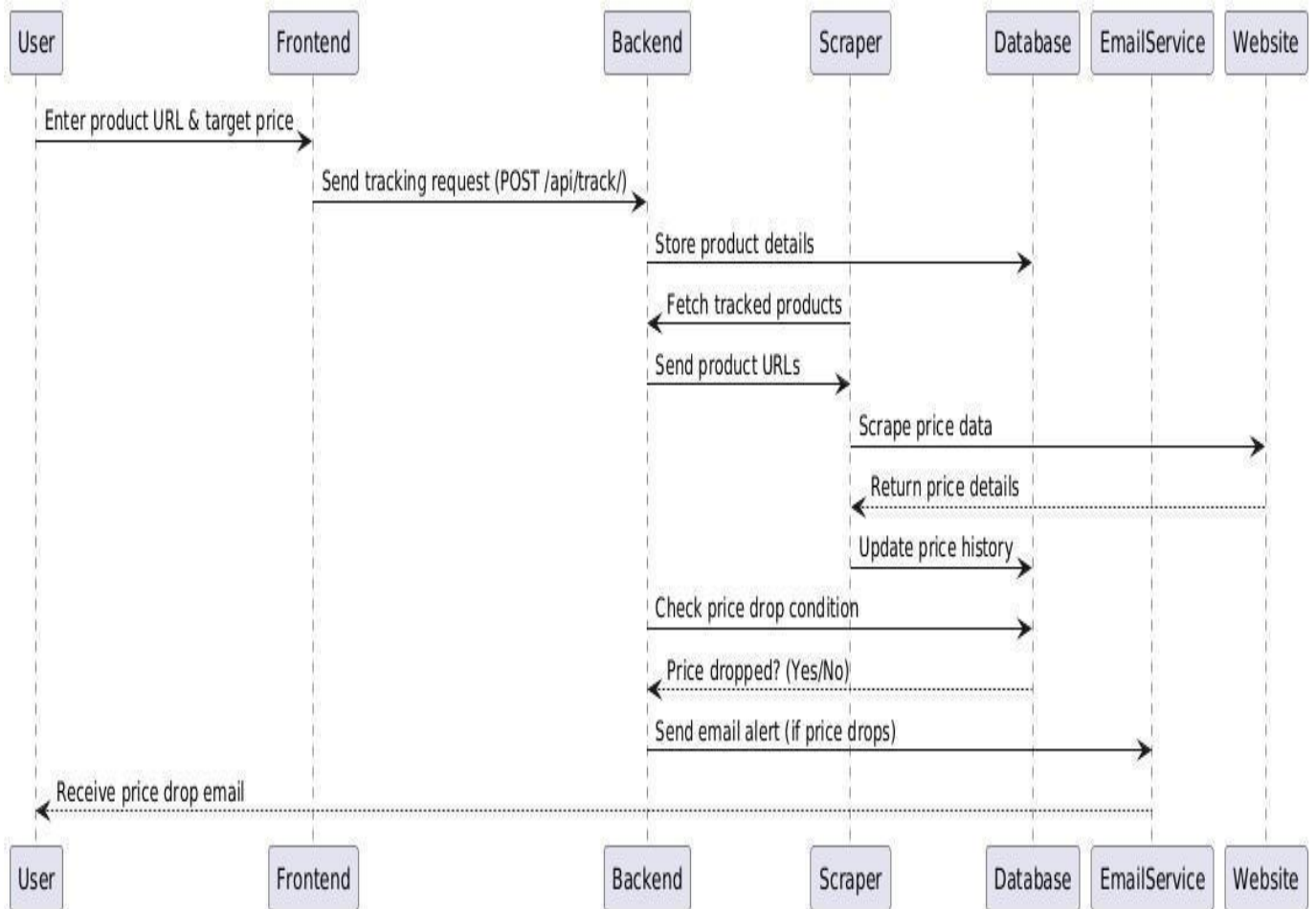
3.5 Class Diagram



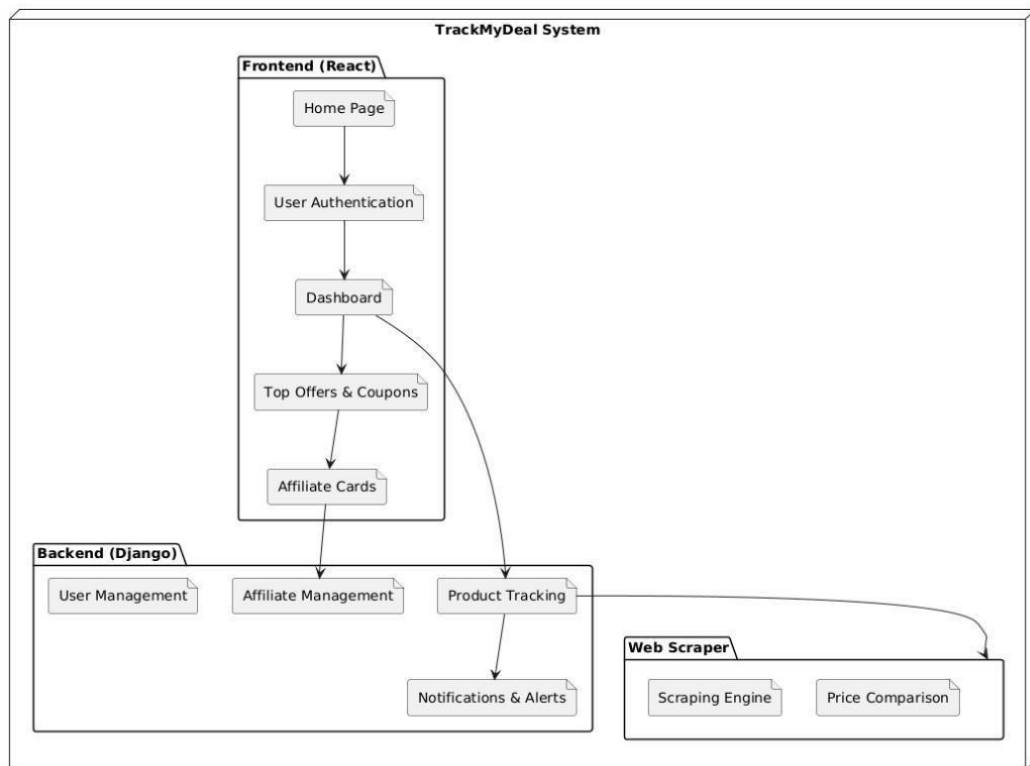
3.6 Activity Diagram



3.7 Sequence Diagram



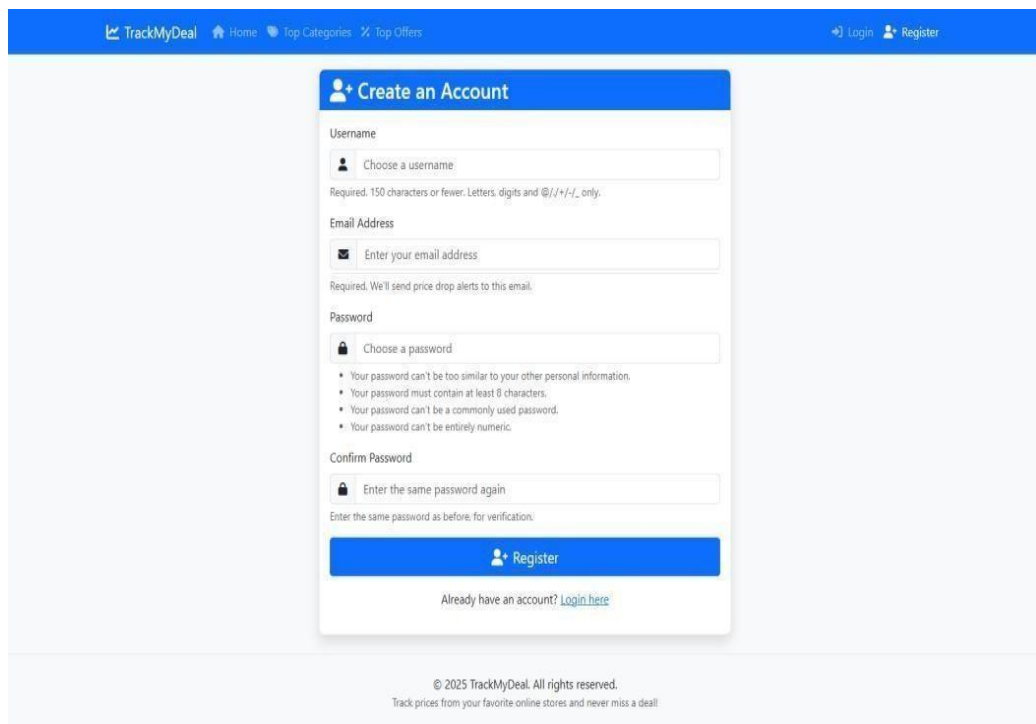
3.8 Module Hierarchy Diagram



3.9 Sample Input and Output Diagram

➤ Track My Deal Web Application:

1. Sign-Up Page:



The screenshot displays the 'Create an Account' registration form on the TrackMyDeal website. The form is centered on a light gray background. At the top, a blue navigation bar contains the 'TrackMyDeal' logo, 'Home', 'Top Categories', 'Top Offers', 'Login', and 'Register' links. The form itself has a white background with a blue header bar that reads 'Create an Account' with a user icon. It includes four input fields: 'Username' (with a placeholder 'Choose a username' and a note 'Required, 150 characters or fewer. Letters, digits and @/+/+/_ only.'), 'Email Address' (with a placeholder 'Enter your email address' and a note 'Required, We'll send price drop alerts to this email.'), 'Password' (with a placeholder 'Choose a password' and a list of password requirements), and 'Confirm Password' (with a placeholder 'Enter the same password again' and a note 'Enter the same password as before, for verification:'). A blue 'Register' button with a user icon is positioned below the fields. At the bottom of the form, a link says 'Already have an account? [Login here](#)'. The footer of the page contains the copyright notice '© 2025 TrackMyDeal. All rights reserved.' and the tagline 'Track prices from your favorite online stores and never miss a deal!'.

TrackMyDeal Home Top Categories Top Offers Login Register

Create an Account

Username
Choose a username
Required, 150 characters or fewer. Letters, digits and @/+/+/_ only.

Email Address
Enter your email address
Required, We'll send price drop alerts to this email.

Password
Choose a password

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

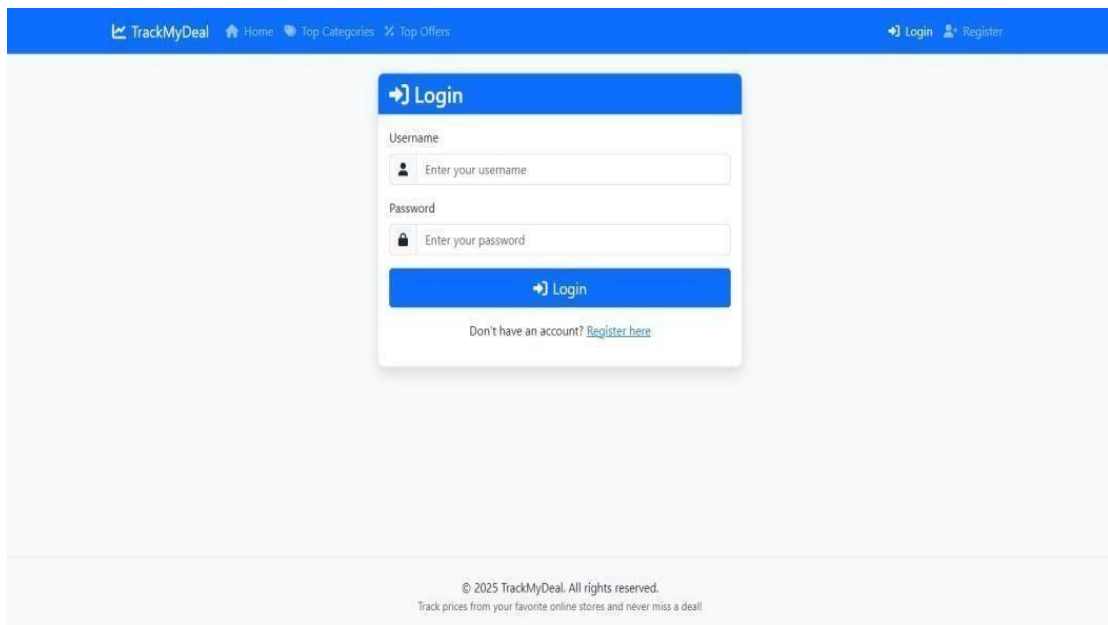
Confirm Password
Enter the same password again
Enter the same password as before, for verification:

[Register](#)

Already have an account? [Login here](#)

© 2025 TrackMyDeal. All rights reserved.
Track prices from your favorite online stores and never miss a deal!

2. Login Page:



The screenshot displays the login interface of the TrackMyDeal website. At the top, a blue navigation bar contains the site logo and links for Home, Top Categories, Top Offers, Login, and Register. The main content area features a central login box with a blue header labeled 'Login'. Inside the box, there are input fields for 'Username' and 'Password', each preceded by a small icon (a person for username and a lock for password). Below these fields is a blue 'Login' button. At the bottom of the box, a link for users without an account is provided. The footer of the page includes copyright information and a brief description of the site's purpose.

TrackMyDeal Home Top Categories Top Offers Login Register

Login

Username
Enter your username

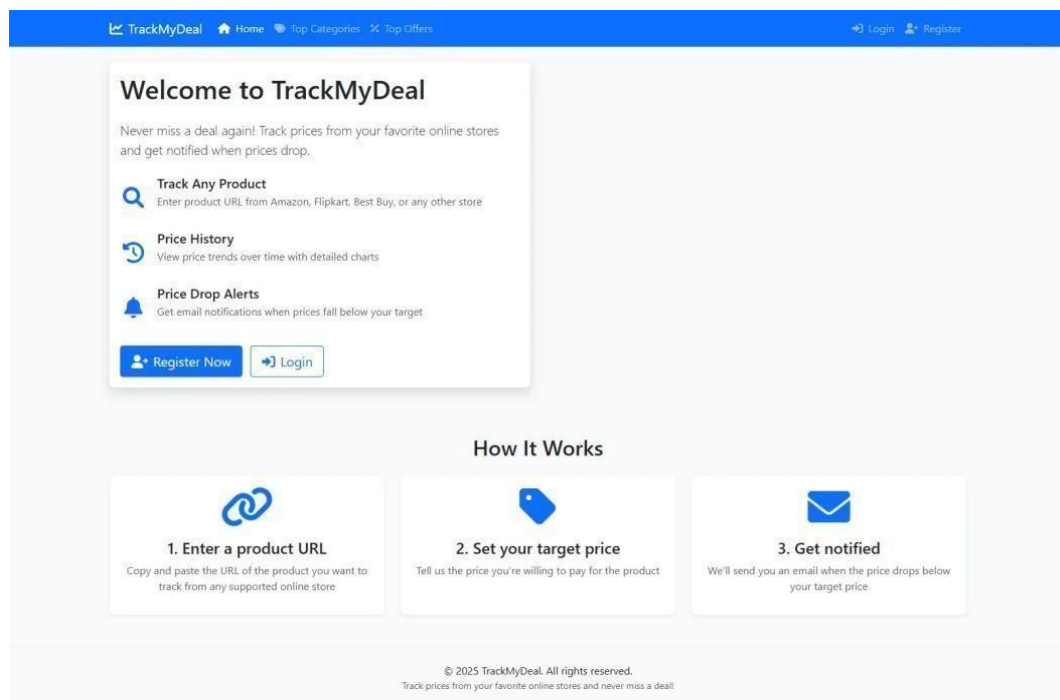
Password
Enter your password

Login

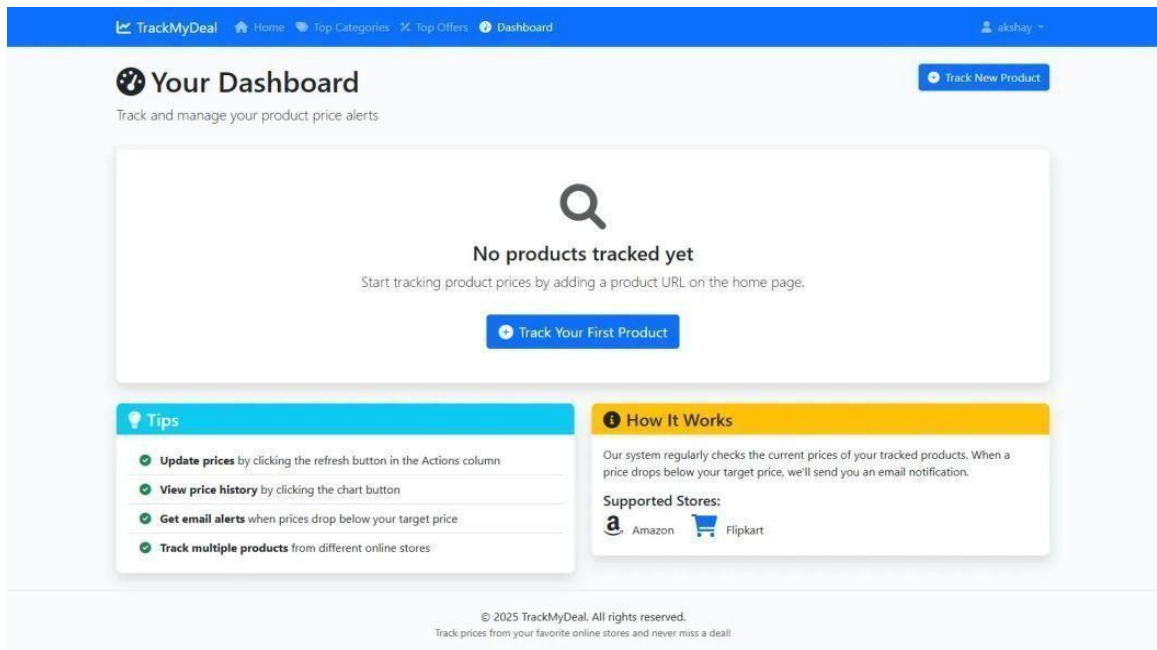
Don't have an account? [Register here](#)

© 2025 TrackMyDeal. All rights reserved.
Track prices from your favorite online stores and never miss a deal!

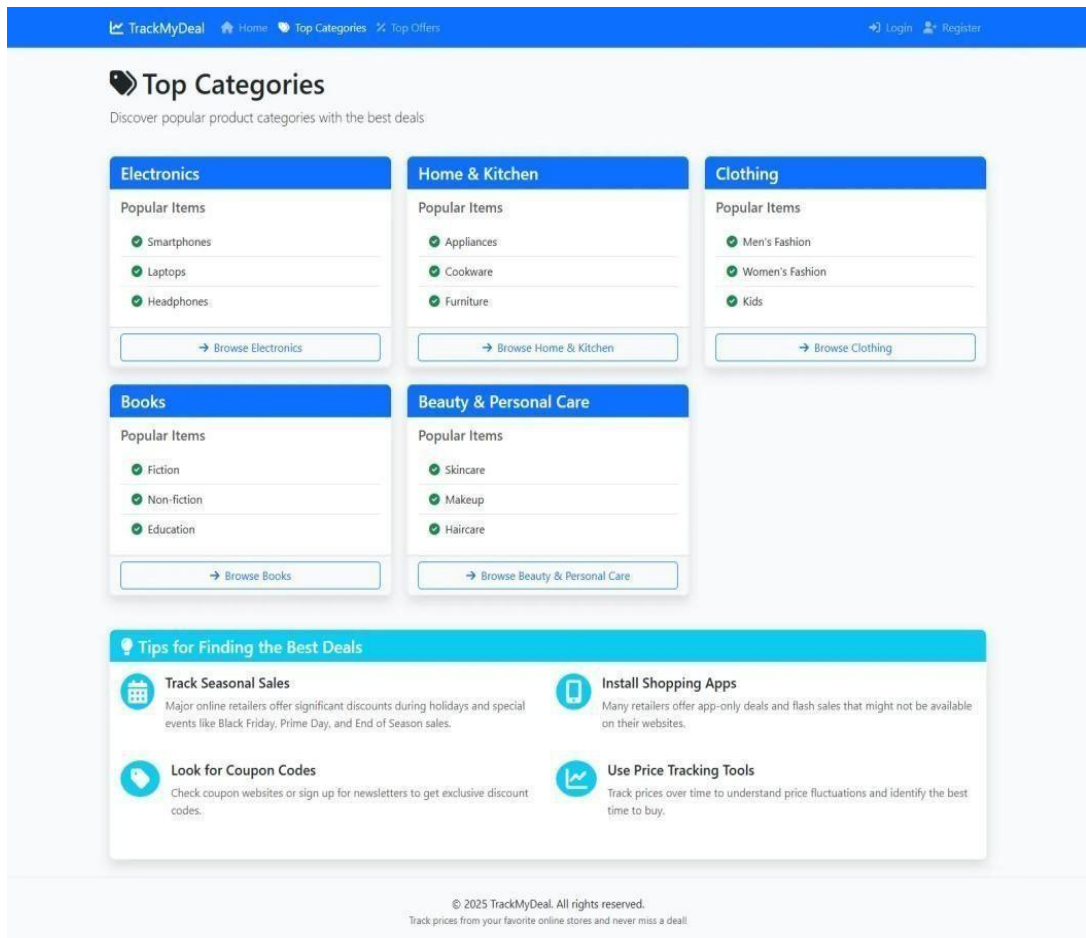
3. Home Page:



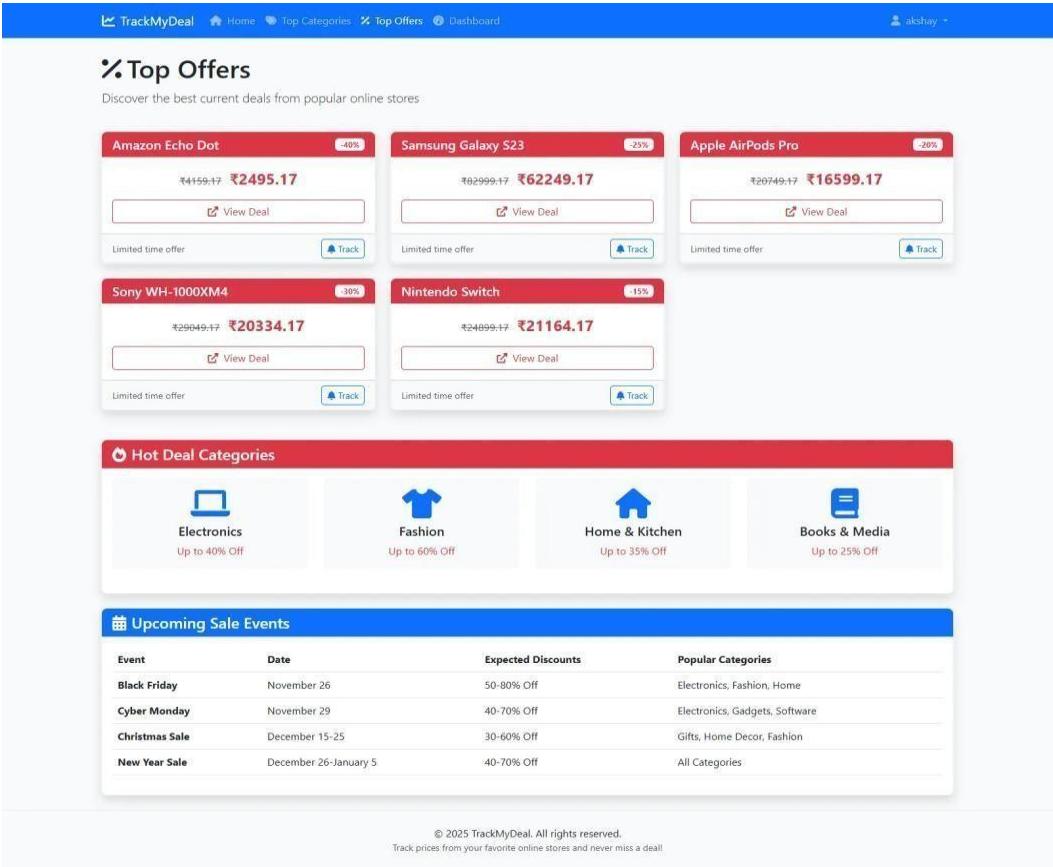
4. Dashboard Page:



5. Top-Categories Page:



6. Top Offers Page:



7. Price History Page:


TrackMyDeal

HomeTop CategoriesTop OffersDashboard

akshay

Back to Dashboard

Product Details



Sparx Mens SM 9039 | Stylish,
Comfortable | Black Sneaker - 6 UK
(SM 9039)

[View on store](#)

Current Price	Target Price
₹499.00	₹500.00

Original Price	Price Change
₹499.00	No change

Date Added

Mar 19, 2025

Last Updated

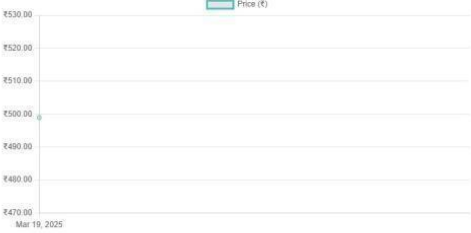
Mar 19, 2025 05:56

Update Price

Stop Tracking

Price History

Price (₹)



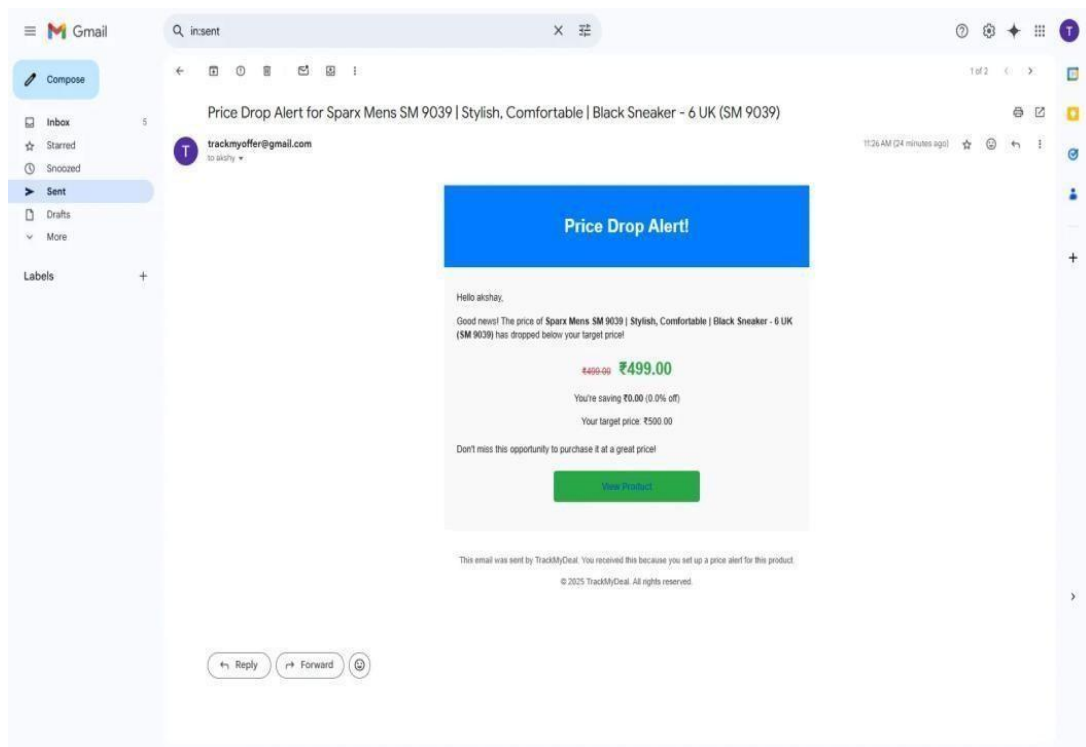
Date	Price	Change
Mar 19, 2025 05:56	₹499.00	-

© 2025 TrackMyDeal. All rights reserved.

Track prices from your favorite online stores and never miss a deal!

43

8. Email Response Page:



CHAPTER 4: CODING

4.1 Algorithms

- **Home:**
 - IF user is logged in THEN
 - Create a blank product tracking form
 - Render home page with form
 - ELSE
 - Render home page without form
- **Register:**
 - IF request is POST THEN
 - Bind submitted data to registration form
 - IF form is valid THEN
 - Save user account
 - Show success message
 - Redirect to login page
 - ELSE
 - Show empty registration form
 - Render registration template with form
- **Dashboard:**
 - Get products tracked by current user, ordered by newest
 - Paginate the list (10 items per page)
 - Render dashboard page with paginated products

- **Track_Product:**

IF request is POST THEN Bind submitted data to tracking form IF form is valid THEN Create new TrackedProduct without saving Assign current user to product

TRY:

Scrape product data (initial price)

Save product

Show success message

Redirect to dashboard

EXCEPT:

Show error message Return to home page with form

ELSE

Show empty tracking form Render home page with form

- **Untrack_Product:**

Find product by ID and current user

Delete the product

Show success message

Redirect to dashboard

- **Check_Price:**
Get product for current user by ID TRY:
Update product price using scraping
Show success message EXCEPT:
Show error message Redirect to dashboard
- **Product_Details:**
Fetch tracked product by ID
Get all price history entries for this product
Render product details page with product info and history
- **Top_Categories:**
Define static list of top product categories and sub-items
Render template with this category data
- **Top_Offers:**
Define static list of current top offers (product name, discount, price) Render template with this offer data

4.2 Code Snippets

```
from django.shortcuts import render, redirect, get_object_or_404
from django.contrib.auth.decorators import login_required
from django.contrib import messages
from django.http import JsonResponse
from django.core.paginator import Paginator
```

```
from forms import UserRegistrationForm, TrackedProductForm
from models import TrackedProduct, PriceHistory
from services import update_product_price
```

```
def home(request):
    """Home page view with product tracking form for logged-in
    users"""
    if request.user.is_authenticated:
        form = TrackedProductForm()
        return render(request, 'tracker/home.html', {'form': form})
    return render(request, 'tracker/home.html')
```

```
def register(request):
    """User registration view"""
    if request.method == 'POST':
        form = UserRegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            messages.success(request, 'Account created successfully!
You can now log in.')
            return redirect('login')
    else:
        form = UserRegistrationForm()
        return render(request, 'tracker/register.html', {'form': form})
```

```
@login_required
def dashboard(request)
```

```

        """User dashboard showing tracked products"""
        user_products =
TrackedProduct.objects.filter(user=request.user).order_by('-
date_added')

        # Paginate results - 10 products per page
        paginator = Paginator(user_products, 10)
        page_number = request.GET.get('page')
        page_obj = paginator.get_page(page_number)

        return render(request, 'tracker/dashboard.html', {'page_obj':
page_obj})

@login_required
def track_product(request):
    """Add a new product to track"""
    if request.method == 'POST':
        form = TrackedProductForm(request.POST)
        if form.is_valid():
            # Create but don't save the product instance yet
            product = form.save(commit=False)
            product.user = request.user

            try:
                # Fetch initial product data
                update_product_price(product)
                messages.success(request, f'Successfully added
{product.product_name} to your tracking list!')
                return redirect('dashboard')
            except Exception as e:
                messages.error(request, f'Error tracking product:
{str(e)}')
            return render(request, 'tracker/home.html', {'form':
form})
        else:
            form = TrackedProductForm()

```

```

        return render(request, 'tracker/home.html', {'form': form})

@login_required
def untrack_product(request, product_id):
    """Remove a product from tracking"""
    product = get_object_or_404(TrackedProduct, id=product_id,
user=request.user)
    product_name = product.product_name
    product.delete()
    messages.success(request, f'{product_name} has been
removed from your tracking list.")
    return redirect('dashboard')

@login_required
def check_price(request, product_id):
    """Manually check price for a product"""
    product = get_object_or_404(TrackedProduct, id=product_id,
user=request.user)

    try:
        update_product_price(product)
        messages.success(request, f"Price updated for
{product.product_name}")
    except Exception as e:
        messages.error(request, f"Error updating price: {str(e)}")

    return redirect('dashboard')

@login_required
def product_details(request, product_id):
    """View product details and price history"""
    product = get_object_or_404(TrackedProduct, id=product_id,
user=request.user)
    price_history = product.price_history.all()

    return render(request, 'tracker/product_details.html', {

```

```

        'product': product,
        'price_history': price_history
    })

def top_categories(request):
    """View top product categories"""
    # In a real-world scenario, this would be dynamic data from a
    database
    # For this example, we'll use static categories
    categories = [
        {'name': 'Electronics', 'popular_items': ['Smartphones',
'Laptops', 'Headphones']},
        {'name': 'Home & Kitchen', 'popular_items': ['Appliances',
'Cookware', 'Furniture']},
        {'name': 'Clothing', 'popular_items': ['Men\'s Fashion',
'Women\'s Fashion', 'Kids']},
        {'name': 'Books', 'popular_items': ['Fiction', 'Non-fiction',
'Education']},
        {'name': 'Beauty & Personal Care', 'popular_items':
['Skincare', 'Makeup', 'Haircare']}
    ]

    return render(request, 'tracker/top_categories.html',
{'categories': categories})

def top_offers(request):
    """View top current offers"""
    # In a real-world scenario, this would be dynamic data from a
    database or API
    # For this example, we'll use static offers
    offers = [
        {
            "name": "Amazon Echo Dot",
            "discount": "40%",
            "original_price": 4159.17,
            "current_price": 2495.17
        }
    ]

```

```

    },
    {
      "name": "Samsung Galaxy S23",
      "discount": "25%",
      "original_price": 82999.17,
      "current_price": 62249.17
    },
    {
      "name": "Apple AirPods Pro",
      "discount": "20%",
      "original_price": 20749.17,
      "current_price": 16599.17
    },
    {
      "name": "Sony WH-1000XM4",
      "discount": "30%",
      "original_price": 29049.17,
      "current_price": 20334.17
    },
    {
      "name": "Nintendo Switch",
      "discount": "15%",
      "original_price": 24899.17,
      "current_price": 21164.17
    }
  ]

  return render(request, 'tracker/top_offers.html', {'offers':
offers})

```


CHAPTER 5: TESTING

5.1 Test Strategy

1. Objectives of Testing:

- To validate that all functional and non-functional requirements are met.
- To identify defects and usability issues before deployment.
- To ensure that both user and service provider modules interact seamlessly via ORM.
- To confirm real-time communication, request handling, and database integrity.

3. Testing Types:

- Unit Testing: Test individual components such as user login, input validation, and request creation using Django's built-in testing tools and React component-level testing frameworks (e.g., Jest, React Testing Library).
- Integration Testing: Ensure seamless interaction between the frontend (React) and backend (Django REST Framework), verifying API requests, authentication, and data persistence in the database.
- System Testing: Validate the complete end-to-end functionality of the application, including product tracking, user registration and profile management, and email alert.
- UI/UX Testing: Evaluate layout consistency, responsiveness across devices, and intuitive navigation using browser-based tools (e.g., Chrome DevTools, Lighthouse).

- Usability Testing: Assess how intuitive and user-friendly the application is for new users, focusing on workflows like onboarding, product search, and alert setup.
- Regression Testing: Re-test critical features after each update to confirm that new changes haven't broken existing functionalities.
- Performance Testing: Measure the application's response time and stability under normal and high-traffic conditions, particularly focusing on dynamic price updates and real-time product availability checks.

3. Test Environment:

- Network Dependency: The TrackMyDeal platform requires a stable internet connection for key operations such as user authentication, product data retrieval via web scraping, and delivery of email notifications. Limited or unstable internet may result in delayed responses or temporary loss of access to features.
- Tools Used:
 - Postman – for testing REST API endpoints (Django backend)
 - Django Test Client – for backend unit and integration tests
 - React Testing Library / Jest – for frontend component testing
 - Browser DevTools & Lighthouse – for UI/UX and performance analysis

- Test Platforms & Devices:
 - Desktop and laptop browsers (Chrome, Firefox, Edge)
 - Responsive testing across screen sizes using browser developer tools
- Database: PostgreSQL (or SQLite during development/testing phases)
- Authentication: Django's built-in authentication system (email/password) with optional email verification and password reset functionality

4. Test Data:

- Multiple test accounts for both users and service providers. Dummy service categories, profile data, and request scenarios.
- Sample profile images and service images.

5. Test Deliverables:

- Test Plan Document
- Test Cases and Test Scenarios
- Bug Report
- Final Test Summary Report The testing process for the TrackMyDeal system is structured to ensure all modules function as expected, real-time updates are reliable, and the application provides a secure and seamless user experience across all supported devices and browsers.

5.2 Unit Test Plan

➤ Unit Test Plan – Track My Deal User Application:

This unit test plan outlines individual functional components of the TrackMyDeal system to ensure that each part performs as expected when tested in isolation.

1. Login System Validation

- Verify that users can log in successfully with valid credentials.
- Test login failure scenarios such as incorrect email, wrong password, or blank fields.
- Check the "Forgot Password" link functionality (if implemented), ensuring it triggers email flow.
- Ensure session timeout occurs after a predefined period of user inactivity.
- Confirm that Django Authentication properly validates user sessions and permissions.

2. User Registration and Profile Management:

- Validate that all required fields (email, password, username, mobile) are enforced during sign-up.
- Reject email addresses with incorrect format (e.g., "user@.com" or missing '@' symbol).

- Enforce password rules: minimum 6 characters, must contain alphanumeric values.
- Test file upload feature for profile pictures and ensure the image is stored securely.
- Check that profile updates (name, email, password) are saved correctly in the database.

3. Product URL Submission and Price Tracking

- Ensure only valid product URLs are accepted from supported platforms (Amazon, Flipkart).
- Test form validation for empty fields or unsupported URLs.
- Verify that the product details (name, price, image) are correctly scraped and displayed.
- Confirm that product data is stored under the corresponding user profile.

4. Price Drop Notification Flow:

- Simulate a price drop and confirm that email alerts are sent to the correct user.
- Ensure notifications are triggered only when current price \leq target price.
- Verify that duplicate notifications are not sent unless explicitly allowed.
- Confirm SMTP email integration works without delays or failure under normal conditions.

5. Real-Time Price Monitoring and History Logging

- Check periodic scraping schedules (e.g., every 6–12 hours) are functioning correctly.
- Ensure product price history is updated and appended rather than overwritten.
- Confirm historical data is displayed accurately in the dashboard using charts or lists.
- Simulate failed scrapes and ensure the system handles it gracefully with retry or error log.

6. Authentication and Security

- Validate secure login sessions using Django's session management.
- Attempt accessing dashboard without login and confirm redirection to login screen.
- Test restricted access: a user cannot access another user's tracked product list.
- Confirm that passwords are securely stored using Django's hashing (not plain text).
- Check CSRF tokens are enforced on all form submissions to prevent cross-site request forgery.

7. UI/UX and Responsiveness:

- Test UI layout and responsiveness across screen sizes (mobile, tablet, desktop).
- Ensure all modals, alerts, and loaders are functioning correctly.
- Verify alignment and visibility of forms, tables, buttons, and error messages.
- Test dashboard usability with multiple tracked products and price histories.

8. Additional Functional Tests

- Verify logout functionality clears session and redirects user to login page.
- Simulate network OFF/ON toggle and test how the frontend reacts to failed fetches.
- Confirm proper error and success messages appear during all CRUD operations.
- Check how the application handles simultaneous tracking of multiple products per user.

5.3 Acceptance Test Plan

1. Test Case 1:

- Ensure that a user can successfully register and log in to the application.
- Expected Result:
User account is created and stored in the database. Upon successful login, user is redirected to the dashboard.

2. Test Case 2:

- Verify that a user can add a product URL with a target price for tracking.
- Expected Result:
Product data is scraped successfully. The product is saved and displayed on the user's dashboard with initial price and tracking status.

3. Test Case 3:

- Ensure that the system sends an email when a price drops below the user's target.
- Expected Result:
Email is triggered via Gmail SMTP. The user receives an email with the product name, current price, and link.

4. Test Case 4:

- Verify that the "Top Offers" page displays current trending deals.
- Expected Result:
Offers are displayed in card format with name, discount, and price comparison.

5. Test Case 5:

- Check that “Top Categories” are shown with sample product types.
- Expected Result:
Categories such as Electronics, Fashion, and Books are visible. Clicking them redirects to relevant brand or deal pages.

6. Test Case 6:

- Ensure that the user can remove a tracked product from the dashboard.
- Expected Result:
Tracked product is deleted from the database and is no longer shown in the UI.

7. Test Case 7:

- Test that a user cannot track a product without being logged in.
- Expected Result:
System redirects unauthenticated users to the login page with a message.

8. Test Case 8:

- Verify error handling when an invalid or unsupported product URL is entered.
- Expected Result:
An error message is shown: “Error tracking product: Invalid URL or site not supported.”

9. Test Case 9:

- Ensure pagination works on the dashboard when tracking many products.
- Expected Result:
Only 10 products are shown per page; next/prev buttons navigate pages correctly.

5.4 Test Case

➤ Test Cases for Track My Deal Web Application:

Test Case ID	Module	Test Description	Expected Output	Status
TC-001	User Registration	Verify that users can register with valid details	User successfully registered	Pass
TC-002	User Login	Check login with correct credentials	Login successful	Pass
TC-003	Password Recovery	Ensure password reset link is sent	Email received with reset link	Pass
TC-004	Product Search	Check search results for a valid product name	Relevant products displayed	Pass
TC-005	Product Search	Invalid product	Found message shown	Pass
TC-006	Product Tracking	Verify product is added to the watchlist	Product added successfully	Pass
TC-007	Product Tracking	Attempt to track a product already in the list	Duplicate entry not allowed	Pass

TC-008	Price Update	Ensure system updates product price regularly	Price updated in the database
TC-009	Price Drop Notification	Validate email notification on price drop	Email received with updated price
TC-010	Price Drop Notification	Ensure SMS notification is sent (if enabled)	SMS received by user
TC-011	Wishlist	Check if users can remove a product from the list	Product removed successfully
TC-012	Wishlist	Check if wishlist	Wishlist displayed correctly
TC-013	Checkout	Verify redirection to the merchant site	User is redirected correctly
TC-014	Checkout	Attempt checkout with an invalid link	Error message displayed
TC-015	UI/UX	Ensure navigation across pages works correctly	No broken links

CHAPTER 6: LIMITATIONS OF PROPOSED SYSTEM

Limitations of Proposed System

- **Limited Support for E-commerce Platforms:**

Currently, the system supports price tracking only for major platforms like Amazon and Flipkart. Users looking to track products on regional or niche websites may not be able to do so until future integrations are developed.

- **No Real-Time Price Monitoring:**

TrackMyDeal monitors product prices at fixed intervals (e.g., every 6 to 12 hours). Due to the absence of real-time API integrations with e-commerce platforms, users may miss flash deals or time-limited discounts that occur between scraping intervals.

- **No In-App Purchase or Checkout Integration:**

The platform redirects users to merchant websites via affiliate links but does not support actual transactions or cart integration. All purchases must be completed on third-party sites, which limits end-to-end user control within the application.

- **No Mobile Application (Currently):**

As of now, TrackMyDeal is a web-based system. The absence of a dedicated Android or iOS mobile app restricts users from receiving push notifications or accessing the platform in a mobile-optimized app interface.

- **Lack of Admin Dashboard for Moderation:**

The system does not currently offer a centralized admin panel to monitor scraping activity, user registrations, affiliate link performance, or issue management. This limits the ability to scale operations or respond to misuse and errors efficiently.

- **No In-Built Notification Options Beyond Email:**

The current implementation supports email alerts for price drops. However, real-time messaging through WhatsApp, Telegram, or browser notifications is not yet integrated, which may affect the immediacy and reach of alerts.

- **Data Accuracy and Scraping Limitations:**

Since the system relies on HTML scraping to fetch product information, any major structural changes to the source websites (e.g., Amazon or Flipkart) may break the scraping logic and affect the accuracy of price tracking unless promptly updated.

CHAPTER 7: CONCLUSION

Conclusion

TrackMyDeal is a comprehensive and efficient platform developed to assist users in making smarter, more cost-effective online purchasing decisions. By enabling users to track the prices of their favorite products and receive timely email notifications when prices fall below a specified target, the system eliminates the hassle of manual monitoring and enhances the digital shopping experience. With support for secure user authentication through email and Google login, the platform ensures both accessibility and data protection.

Developed using a reliable tech stack including Django for the backend, SQLite for lightweight data storage, and BeautifulSoup for automated web scraping, TrackMyDeal is built for stability and ease of expansion. Its modular design allows for future enhancements such as WhatsApp alerts, referral rewards, and more advanced pricing analytics. Overall, TrackMyDeal addresses a relevant user need with a practical and scalable solution, making it an ideal foundation for ongoing development in the e-commerce support space.

CHAPTER 8: BIBLIOGRAPHY

Bibliography

- **W3Schools:**

Used for basic tutorials on HTML, CSS, JavaScript, and responsive layouts.

<https://www.w3schools.com>

- **MDN Web Docs**

Reference for frontend development best practices including HTML form validation, JavaScript DOM manipulation, and more.

<https://developer.mozilla.org>

- **Bootstrap Official Documentation**

Used for building responsive navbar, cards, and layout for the dashboard using Bootstrap 5.

<https://getbootstrap.com/docs>

- **SQLite Documentation**

Reference for the default Django database system used in this project.

<https://www.sqlite.org/docs.html> /

- **Django Documentation:**

Used for authentication system, ORM queries, and creating class-based and function-based views.

<https://docs.djangoproject.com/en>

- **GeeksforGeeks:**

Helped in understanding Python programming patterns, Django tips, and email integration using SMTP.

<https://www.geeksforgeeks.org>

- **YouTube Tutorials**

Videos used to understand the flow of Django projects, creating user authentication and integrating email alerts.

CHAPTER 9: USER MANUAL

User Manual

➤ User Registration (Sign-Up) Page:

The user must create an account to start using the price tracker service.

Fields & Criteria:

- Username: Minimum 5 characters, can include letters, numbers, and underscores. Ex: akshay_123
- Password: Must include at least:
 - 1 Uppercase letter
 - 1 Lowercase letter
 - 1 Digit
 - 1 Special character
 - Minimum 8 characters Ex: Akshay@123
 - Confirm Password: Must match the password field.

The screenshot shows the 'Create an Account' registration form on the TrackMyDeal website. The form is titled 'Create an Account' and includes the following fields and instructions:

- Username:** A text input field with a placeholder 'Choose a username'. Below it, a note states: 'Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.'
- Email Address:** A text input field with a placeholder 'Enter your email address'. Below it, a note states: 'Required. We'll send price drop alerts to this email.'
- Password:** A text input field with a placeholder 'Choose a password'. Below it, a list of requirements is provided:
 - Your password can't be too similar to your other personal information.
 - Your password must contain at least 8 characters.
 - Your password can't be a commonly used password.
 - Your password can't be entirely numeric.
- Confirm Password:** A text input field with a placeholder 'Enter the same password again'. Below it, a note states: 'Enter the same password as before, for verification.'

At the bottom of the form is a blue 'Register' button. Below the button, there is a link: 'Already have an account? [Login here](#)'.

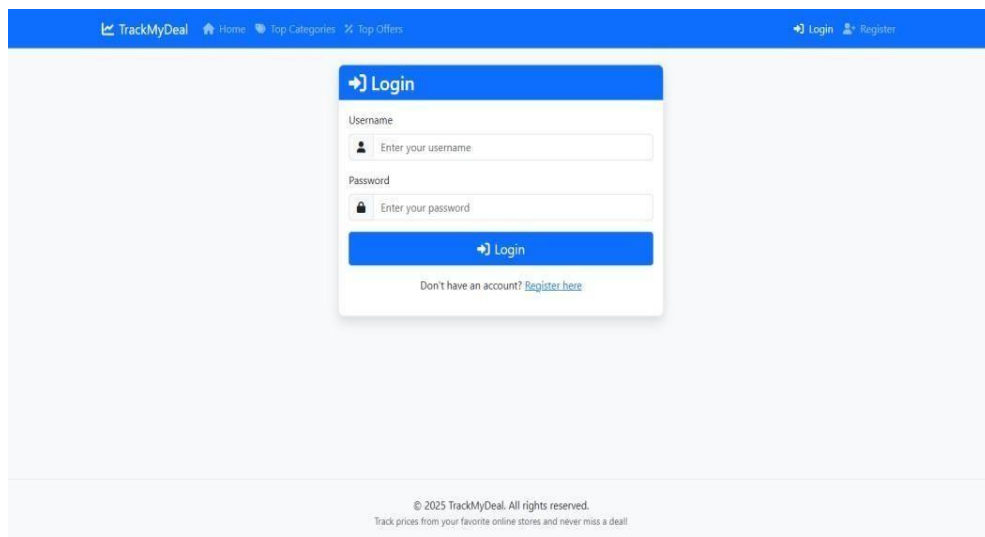
The footer of the page contains the copyright notice: '© 2025 TrackMyDeal. All rights reserved.' and the tagline: 'Track prices from your favorite online stores and never miss a deal!'.

➤ **User Login Page:**

Users with existing accounts can sign in.

Credentials:

- o Correct username and password must be provided.
- o If either is incorrect, display: “Wrong login credentials.”

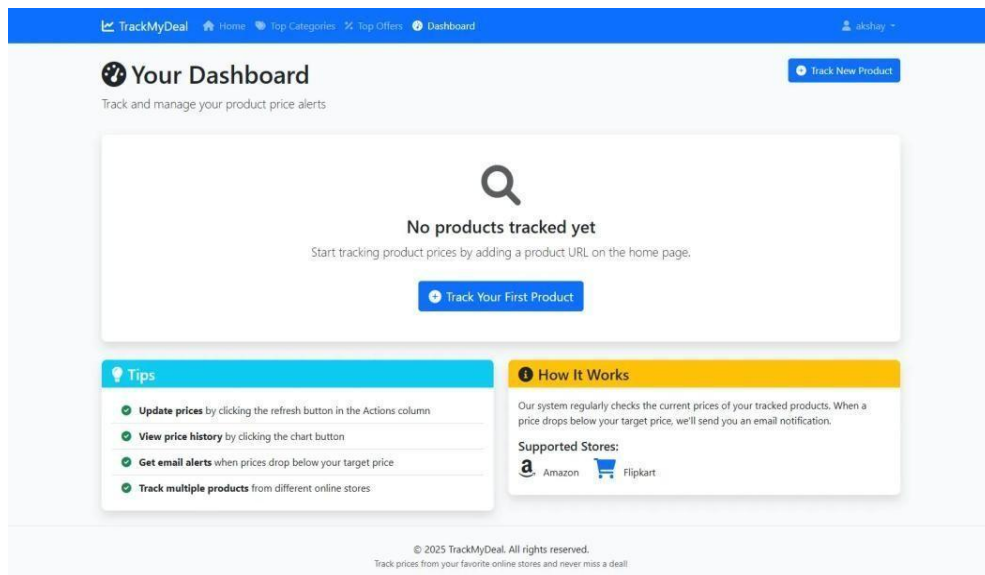


The screenshot displays the TrackMyDeal website's login interface. At the top, a blue navigation bar contains the site logo, a home icon, and links to 'Top Categories' and 'Top Offers'. On the right side of the bar are links for 'Login' and 'Register'. The main content area features a central white login box with a blue header labeled 'Login'. Inside the box, there are two input fields: 'Username' with a user icon and 'Password' with a lock icon. Below these fields is a blue 'Login' button. A link for 'Register here' is positioned at the bottom of the login box. The footer of the page includes the copyright notice '© 2025 TrackMyDeal. All rights reserved.' and a tagline: 'Track prices from your favorite online stores and never miss a deal!'.

➤ **User Dashboard (Tracking Section):**

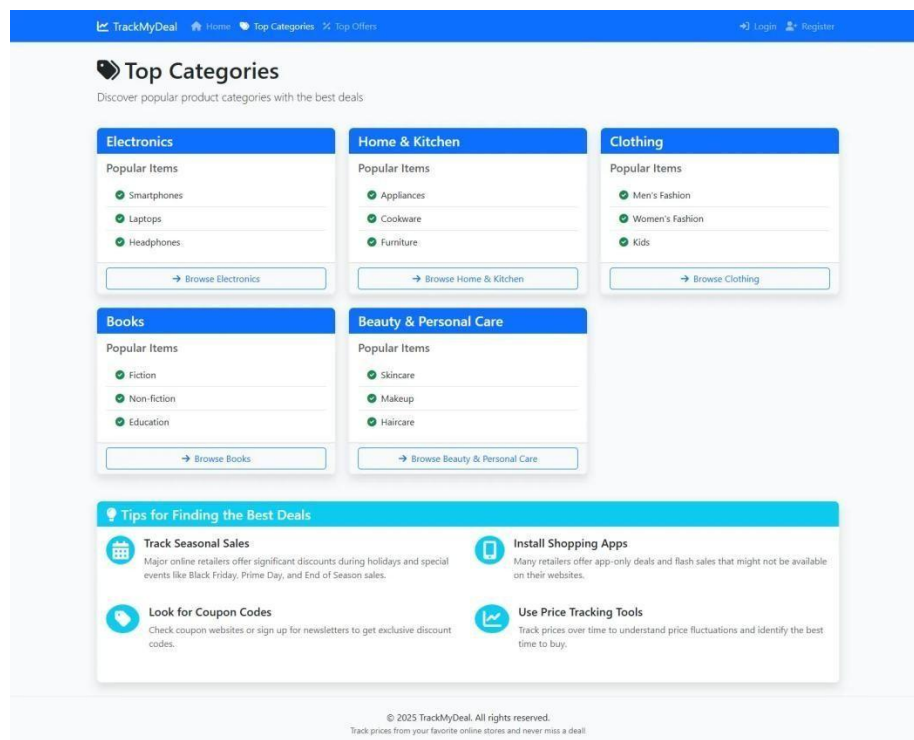
Once logged in, the user is taken to their dashboard:

- Can add product URLs with a target price.
- Can see the current price (fetched automatically).
- Can delete or manually refresh product price.
- Pagination used for better UX.



➤ **Top Categories Section:**

- Users can view trending categories like:
 - Electronics
 - Home & Kitchen
 - Books, etc.



- **Top Offers Section:**
- Static list of top deals available shown on a separate page.

TrackMyDeal

Home

Top Categories

Top Offers

Dashboard

akshay

🔥 Top Offers

Discover the best current deals from popular online stores

Amazon Echo Dot

40%

~~₹4159.17~~ ₹2495.17

View Deal

Limited time offer

Track

Samsung Galaxy S23

25%

~~₹82999.17~~ ₹62249.17

View Deal

Limited time offer

Track

Apple AirPods Pro

20%

~~₹20749.17~~ ₹16599.17

View Deal

Limited time offer

Track

Sony WH-1000XM4

38%

~~₹25049.17~~ ₹20334.17

View Deal

Limited time offer

Track

Nintendo Switch

13%

~~₹24899.17~~ ₹21164.17

View Deal

Limited time offer

Track

Hot Deal Categories

Electronics

Up to 40% Off

Fashion

Up to 60% Off

Home & Kitchen

Up to 35% Off

Books & Media

Up to 25% Off

Upcoming Sale Events

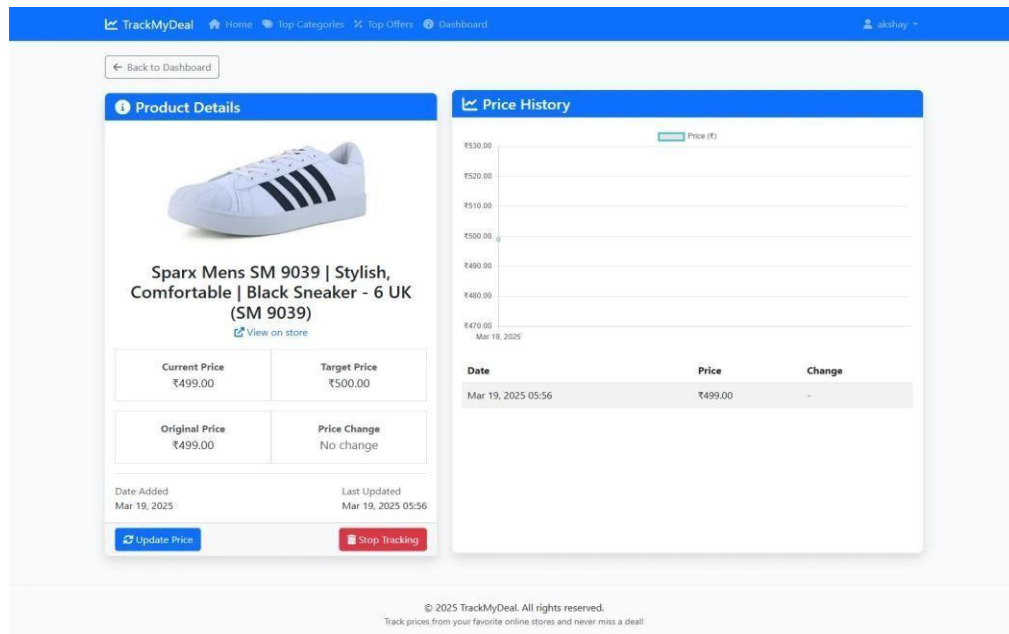
Event	Date	Expected Discounts	Popular Categories
Black Friday	November 26	50-80% Off	Electronics, Fashion, Home
Cyber Monday	November 29	40-70% Off	Electronics, Gadgets, Software
Christmas Sale	December 15-25	30-60% Off	Gifts, Home Decor, Fashion
New Year Sale	December 26-January 5	40-70% Off	All Categories

© 2025 TrackMyDeal. All rights reserved.

Track prices from your favorite online stores and never miss a deal!

➤ **Tracking System:**

- Users submit a product URL and a target price.
- Price history is fetched and stored using ORM.
- If the current price drops below the target price:
 - An email alert is sent to the user.

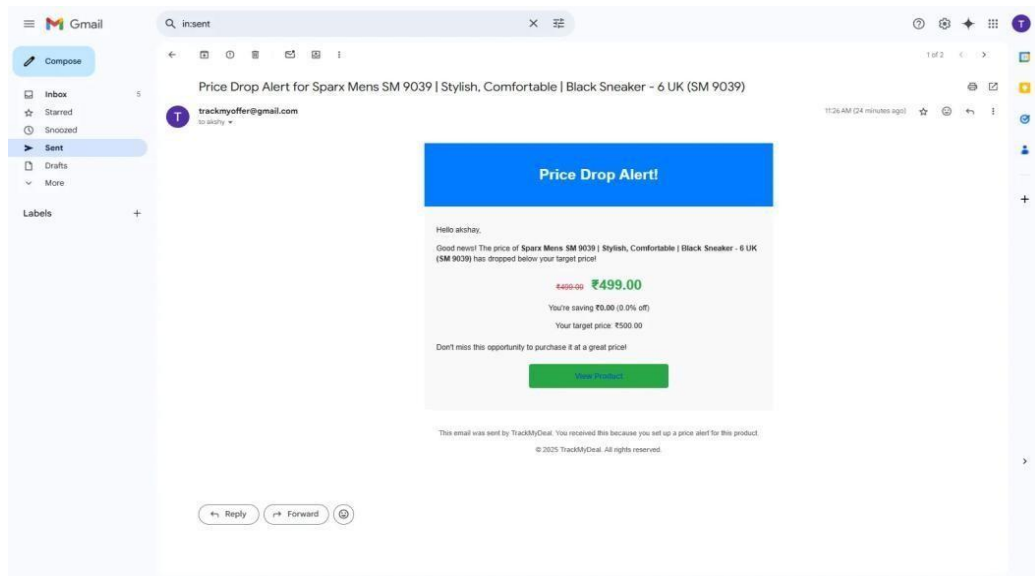


➤ **Price Drop Alert Email Notification:**

This is an automated email notification sent when a tracked product drops below the user's target price.

Users can:

- Receive instant alerts via email for tracked product price drops.
- View the product name, price change, and the amount saved.
- Click the “View Product” button to directly access the product page and make a purchase.
- Stay informed and make timely buying decisions based on their preferred price thresholds.

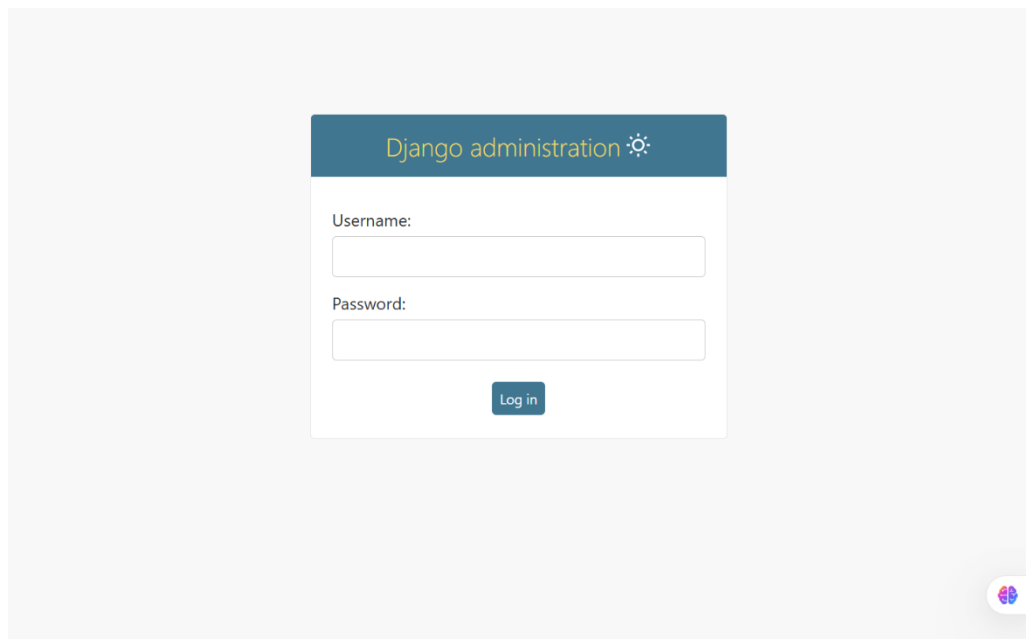


➤ **Admin Login Page:**

This is the default Django admin login interface.

Admins can:

- Securely log in using their registered credentials (username and password).
- Gain access to the backend management system after authentication.
- Be redirected to the admin dashboard upon successful login

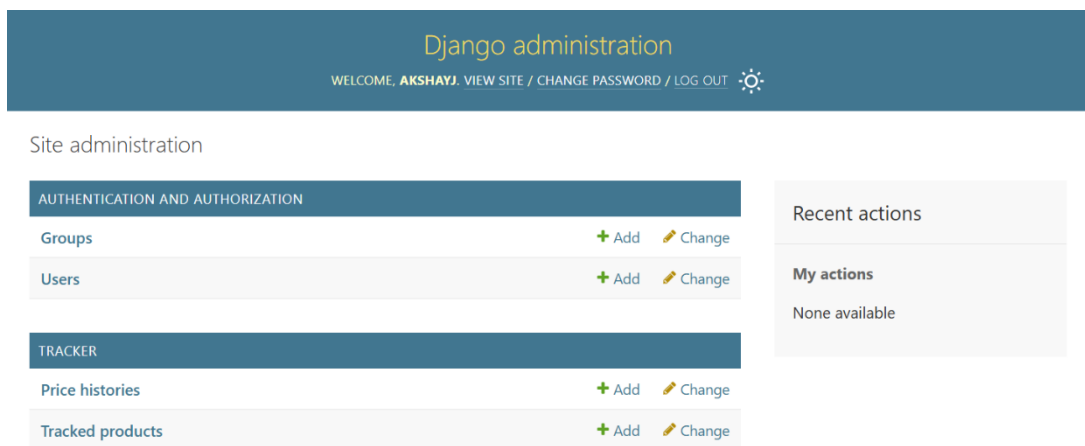


➤ **Admin Home/Dashboard Page:**

This is the main administrative control panel that appears after logging in.

Admins can:

- View and manage all registered models in the project.
- Add, edit, or delete records for any model listed (e.g., users, products, posts).
- Access advanced filters, search, and bulk actions for streamlined management.
- Customize the admin layout or interface if needed for business requirements.



➤ **Admin Password Change Page:**

This section allows logged-in administrators to change their current password securely.

Admins can:

- Enter their old password followed by the new one and confirmation.
- Enhance account security by updating credentials periodically.
- Prevent unauthorized access in case of suspected compromise.

Django administration
WELCOME, AKSHAYJ. CHANGE PASSWORD / LOG OUT

Home > Password change

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

TRACKER

Price histories + Add

Tracked products + Add

«

Password change

Please enter your old password, for security's sake, and then enter your new password twice so we can verify you typed it in correctly.

Old password:

New password:

Your password can't be too similar to your other personal information.
Your password must contain at least 8 characters.
Your password can't be a commonly used password.
Your password can't be entirely numeric.

New password confirmation:

Enter the same password as before, for verification.

CHANGE MY PASSWORD