Kaunas University of Technology

Faculty of Informatics

Developing a Comprehensive Machine Learning System for E-Commerce: Integrating Personalized Recommendations with Demand Forecasting.

Final bachelor project

Tatenda Mawango

Author of the project

junior assist. Arnas Nakrošis

Supervisor

Kaunas, 2024

Kaunas University of Technology

Faculty of Informatics

Developing a Comprehensive Machine Learning System for E-Commerce: Integrating Personalized Recommendations with Demand Forecasting.

Final bachelor project

Informatics (6121BX010)

Tatenda Mawango

Author of the project

junior assist. Arnas Nakrošis

Supervisor

assoc. prof. Darius Naujokaitis

Reviewer

Kaunas, 2024

Kaunas University of Technology

Faculty of Informatics

Tatenda Mawango

Developing a comprehensive machine learning system for E-commerce: Integrating personalized recommendations with demand forecasting.

Declaration of Academic Integrity

I confirm the following:

1. I have prepared the final degree project independently and honestly without any violations of the copyrights or other rights of others, following the provisions of the Law on Copyrights and Related Rights of the Republic of Lithuania, the Regulations on the Management and Transfer of Intellectual Property of Kaunas University of Technology (hereinafter – University) and the ethical requirements stipulated by the Code of Academic Ethics of the University.

2. All the data and research results provided in the final degree project are correct and obtained legally; none of the parts of this project are plagiarized from any printed or electronic sources; all the quotations and references provided in the text of the final degree project are indicated in the list of references.

3. I have not paid anyone any monetary funds for the final degree project or the parts thereof unless required by the law.

4. I understand that in the case of any discovery of the fact of dishonesty or violation of any rights of others, the academic penalties will be imposed on me under the procedure applied at the University; I will be expelled from the University and my final degree project can be submitted to the Office of the Ombudsperson for Academic Ethics and Procedures in the examination of a possible violation of academic ethics.

Tatenda Mawango

Confirmed electronically

Kaunas University of Technology

Faculty of Informatics

Task of the bachelor's final degree project

1. The topic of the Bachelor's final project:

Developing a comprehensive machine learning system for E-commerce: Integrating personalized recommendations with demand forecasting.

2. The aim of the Bachelor's final project:

The aim of this project is to create a machine learning based system that enhances e-commerce by delivering personalized recommendations and precise demand forecasting, improving user engagement and inventory management.

3. The objectives of the Bachelor's final project:

1. To conduct a comprehensive market analysis of existing e-commerce systems and solutions, assessing their approaches to personalization and demand forecasting, as well as identifying gaps and opportunities for innovation in the market.
2. To perform a detailed evaluation of datasets relevant to e-commerce applications, such as user interactions, purchase histories, and product details, to determine their suitability for supporting advanced machine learning models for personalized recommendations and demand forecasting.
3. To design and develop an integrated e-commerce system that not only incorporates machine learning models for personalized product recommendations and demand forecasting but also addresses the needs of both users and administrators, including product management and user experience considerations.
4. To ensure the seamless integration of the recommendation and forecasting models within the broader system architecture, facilitating a consistent user experience that balances personalization with efficient inventory management.
5. To evaluate the overall system's performance in terms of user satisfaction, operational efficiency, and market competitiveness, including the accuracy of machine learning predictions and the system's adaptability to evolving market and user needs.

4. The deadlines for submitting the Bachelor's final project:

To the supervisor (8 workdays before the defence at the Department): 2024 May 16

To the reviewer (6 workdays before the defence at the Department): 2024 May 23

To the Committee (3 workdays before the defence at the Department): 2024 June 4

5. Functional requirements for the object or system to be developed:

1. Personalized Recommendation System
   - **User Account Management:** The system shall enable users to create and manage their accounts, including personal information.
   - **Personalized User Experience:** The system shall utilize machine learning to analyze user data (purchase history, browsing behavior) to generate personalized product recommendations.
   - **Interactive Recommendation Features:** Users shall be able to add recommended items directly to their shopping basket from the recommendation interface.
2. Shopping and Basket Management
   - **Basket Functionality:** Users shall be able to add, remove, and edit items in their shopping basket, including adjustments to item quantities.
   - **Checkout Process:** The system shall facilitate a seamless checkout process.
   - **Order History:** Users shall have access to their order history.
3. Demand Forecasting Model
   - **Data-Driven Inventory Management:** The system shall analyze sales data and external factors to forecast demand, aiding in inventory decision-making.
   - **Visualization and Reporting Tools:** Inventory managers shall receive visual forecasts and reports to support strategic planning.
4. Administrative Functions
   - **Comprehensive Product Management:** Administrators shall be able to add, update, manage product listings, and categorize products according to various attributes.
   - **System Monitoring and User Insights:** Tools shall be provided for administrators to monitor overall system performance and gain insights into user behavior and preferences.

5. Non-functional requirements for the object or system to be developed:

1. **Performance and Responsiveness:** The system shall ensure that all page's load within 3 seconds under normal operating conditions to provide a smooth user experience.
2. **System Availability:** The system shall aim for an uptime of 99%, ensuring it is available around the clock to accommodate users in different time zones and peak traffic periods.
3. **Error Handling and User Notification:** The system shall implement robust error handling mechanisms to identify issues proactively and notify users, accordingly, ensuring transparency and maintaining trust.

4. **Scalability:** The infrastructure shall be designed to handle a 20% annual increase in user traffic without compromising performance or user experience.
5. **Security and Data Privacy:** The system shall use security measures to protect user data to adhere to GDPR(General Data Protection Regulation) and other relevant data protection regulations.
6. **User Interface and Experience:** The system shall be designed with a user-friendly interface, ensuring ease of navigation and accessibility for all user demographics.

Tatenda Mawango. Developing a comprehensive machine learning system for E-commerce: Integrating personalized recommendations with demand forecasting. Bachelor's final project / supervisor junior assistant Arnas Nakrošis; Kaunas University of Technology, Faculty of Informatics.

Field of study (group of fields of study): Informatics (Computing).

Key words: Python, Machine Learning, Collaborative filtering, E-commerce.

Kaunas, 2024. 74 pages.

Summary

This project focuses on developing a machine learning-based system to enhance e-commerce platforms by integrating personalized recommendations and demand forecasting. It aims to improve user engagement and optimize inventory management by addressing challenges like the cold-start problem (recommending new products). The project explores hybrid recommendation systems that combine content-based and collaborative filtering techniques to enhance the accuracy and variety of recommendations. It also explores demand forecasting models to predict product demand, helping inventory decision-making. The project's goal is to create a comprehensive system that improves user satisfaction and operational functions in the e-commerce website.

Tatenda Mawango. Išsamios mašininio mokymosi sistemos, skirtos el. prekybai, kūrimas: suasmenintų rekomendacijų integravimas su paklausos prognozavimu. Bakalauro baigiamasis projektas vadovas jaunesnysis asistentas Arnas Nakrošis; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatika (Informatikos mokslai)

Reikšminiai žodžiai: Python, mašininis mokymasis, panašių paieškų filtravimas, el. Prekyba.

Kaunas, 2024. 73 puslapiu.

Santrauka

Šis projektas skirtas mašininiu mokymusi pagrįstos sistemos kūrimui, siekiant pagerinti elektroninės prekybos platformas integruojant asmenines rekomendacijas ir paklausos prognozavimą. Šiuo būdu siekiama pagerinti vartotojų įsitraukimą ir optimizuoti atsargų valdymą sprendžiant tokias problemas kaip naujo paleidimo problema (rekomenduoti naujus produktus). Šiame projekte nagrinėjamos hibridinės rekomendacijų sistemos, kuriose derinami turiniu pagrįsti ir bendradarbiaujantys filtravimo metodai, siekiant padidinti rekomendacijų tikslumą ir įvairovę. Taip pat tiriami paklausos prognozavimo modeliai, siekiant numatyti produktų paklausą, padedant priimti sprendimus dėl atsargų. Projekto tikslas – sukurti išsamią sistemą, gerinančią vartotojų pasitenkinimą ir veiklos funkcijas elektroninės prekybos svetainėje.

Contents

List of tables

List of figures

Introduction

The rapid growth of e-commerce platforms like Big Basket has both transformed the retail industry and brought new challenges in navigating massive product inventories. To aid customer decision-making, recommender systems are essential [1]. These systems analyze user behavior, product details, and broader trends to deliver personalized suggestions.

Traditional recommendation approaches include content-based filtering and collaborative filtering. Content-based filtering leverages product descriptions and metadata to recommend items like a user's past purchases [2]. Collaborative filtering identifies users with similar tastes and recommends items that those users have liked. However, these methods can struggle with dynamic e-commerce challenges like introducing new products (the cold-start problem) and recommending from a vast, diverse inventory (the long-tail problem).

To address these issues, hybrid recommendation systems have emerged as a powerful solution. By combining content-based and collaborative filtering techniques, these systems seek to improve the accuracy, diversity, and serendipity of recommendations. This analysis section will delve into the landscape of recommendation systems, emphasizing hybrid strategies for e-commerce. We will draw on established techniques and innovative research to analyze trends, address implementation hurdles, and highlight promising avenues for further innovation to platforms like Big Basket.

Aim and objectives.

The aim of this project is to create a machine learning based system that enhances e-commerce by delivering personalized recommendations and precise demand forecasting, improving user engagement and inventory management efficiency.

Objectives:

1. To conduct a comprehensive market analysis of existing e-commerce systems and solutions, assessing their approaches to personalization and demand forecasting, as well as identifying gaps and opportunities for innovation in the market.
2. To perform a detailed evaluation of datasets relevant to e-commerce applications, such as user interactions, purchase histories, and product details, to determine their suitability for supporting advanced machine learning models for personalized recommendations and demand forecasting.
3. To design and develop an integrated e-commerce system that not only incorporates machine learning models for personalized product recommendations and demand forecasting but also addresses the needs of both users and administrators, including product management and user experience considerations.
4. To ensure the seamless integration of the recommendation and forecasting models within the broader system architecture, facilitating a consistent user experience that balances personalization with efficient inventory management.
5. To evaluate the overall system's performance in terms of user satisfaction, operational efficiency, and market competitiveness, including the accuracy of machine learning predictions and the system's adaptability to evolving market and user needs.

Structure of the work

The outline of the document will be as follows:

- Introduction:
  - Sets the context and outlines the main objectives of the project.
- Analysis:
  - Explores the features of e-commerce systems, focusing on personalized recommendations and demand forecasting.
- Project:
  - Details the design and development process of the integrated e-commerce system, describing the methodologies and technologies used.
- Implementation and Testing:
  - Provides a comprehensive overview of the system's development and validation, including the performance of the machine learning models.
- Documentation for the User:
  - Offers guidance on how to use the system, ensuring that both customers and administrators can effectively navigate and utilize its features.

1. Analysis

1.1. Introduction

Recommendation systems have become a trend in the modern digital landscape, shaping how users interact with vast aspects of information, products, and content. From suggesting what movie to watch next to providing tailored news feeds, recommenders intelligently filter options and enhance the user experience. These systems leverage sophisticated algorithms and data analysis to understand individual preferences and deliver personalized suggestions. This section delves into the key concepts, techniques, and challenges that underpin recommender systems.

1.2. Types of Recommendation Systems

Collaborative filtering (CF): Collaborative filtering is a mechanism that utilizes the previous interactions of users with items, such as ratings, purchases, views, and so on, to find patterns and correlations. It infers based on the discovered relations.

User-based CF: Predicts the items a user would like by finding similar users and looking at what other users like.

Item-based CF: Suggests items to the user that have shown similarity to those with which the user has interacted in the past.

Literature Review: Collaborative Filtering

Collaborative Filtering (CF): Techniques and Applications – This is a review paper on collaborative filtering, with a particular focus on being one of the front-running techniques in the domain of recommender systems. It considers the predictive power of CF algorithms, derived from historical user-item interactions, and discusses issues such as data sparsity, scalability, and the cold-start problem. The paper generally classifies methods for collaborative filtering into two categories: memory-based methods and model-based methods, with different mechanisms for generating recommendations [3].

Review of Collaborative Filtering Recommendation System – The process of collaborative filtering is further elaborated on by distinguishing between user-based and item-based CF. It is further broken down into memory-based and model-based collaborative filtering techniques, with their features focusing on the role of similarity measures, such as cosine similarity and Pearson correlation, in enhancing the accuracy of recommendations. On the contrary, the paper discusses the cold-start problem and presents hybrid systems as a solution, showing the adaptability of CF to different contexts of recommendation [4].

Content-Based Filtering: Here, recommendations are based on the attributes and characteristics of items themselves. Analysis of features such as product descriptors, movie genres, or music tags enable matching to user preferences.

Literature Review: Content-Based Filtering

This section reviews significant contributions to content-based filtering, highlighting the importance of item features and the integration of contextual information for personalized recommendations. It

discusses semantic web technologies and ontologies' role in enhancing recommendation systems' accuracy by enabling sophisticated user profile matching and item suggestions. Recent studies emphasize the potential of semantic-based approaches and ontology representation languages in resolving ambiguities and improving the quality of recommendations by leveraging controlled vocabularies and their semantic relationships [5].

Hybrid Filtering: These systems blend the strengths of CF and content-based methods, often overcoming their respective weaknesses. Hybrids enable more nuanced recommendations and can address issues like the cold-start problem.

Literature Review: Hybrid Approaches

Hybrid Recommender Systems: A Systematic Literature Review - This paper provides a comprehensive overview of the state of the art in hybrid recommender systems over the past decade. It follows a systematic methodology to analyze hybrid recommender systems, addressing research questions related to the challenges these systems aim to solve, the combination of data mining and machine learning techniques used, hybridization classes, application domains, evaluation strategies, and future research directions. The review highlights the importance of hybrid systems in enhancing recommendation quality by combining multiple recommendation strategies to mitigate their individual limitations [6].

A Systematic Literature Review on the Hybrid Approaches for Recommender Systems - This research focuses on analyzing the progress and identifying opportunities for further research in hybrid recommender systems. It outlines the latest trends, challenges, methodologies, datasets, application domains, and evaluation metrics associated with hybrid approaches. The study underscores the significance of hybrid systems in addressing the limitations of singular recommendation strategies through the integration of various techniques and algorithms, thereby improving the overall recommendation quality [7].

1.2.1. Key Concepts and Techniques

User Profiles: Recommender systems model users to capture their interests and behaviors. Profiles may include explicit data (ratings, preferences) and implicit information derived from actions (browsing history, time spent on items).

Item Representation: Like user profiles, items are represented using attributes, metadata, or learned feature vectors. This representation underpins similarity calculations and content-based matching.

Similarity Metrics: Measuring similarity between users or items is central to recommenders. Common metrics include Cosine Similarity, Pearson Correlation, and Jaccard Similarity, chosen based on data characteristics and system goals.

Literature Review: Similarity Metrics

In recommender systems, similarity metrics are crucial for both memory-based and model-based collaborative filtering approaches, as well as hybrid filtering methods. User-based filtering calculates new item ratings based on similar users' ratings, while item-based filtering predicts ratings using similar items' ratings. Model-based systems develop predictive models from data features, efficiently

addressing sparsity and scalability. Hybrid filtering combines techniques to improve accuracy and performance [8].

Matrix Factorization: This powerful technique reduces the dimensionality of user-item matrices, revealing latent dimensions linked to preferences. Matrix factorization improves sparsity, scalability, and prediction accuracy.

Literature Review: Matrix Factorization

Matrix factorization techniques in recommender systems are pivotal for enhancing recommendation quality by addressing the sparsity and scalability challenges. These techniques decompose the user-item interaction matrix into lower-dimensional latent factor spaces, capturing hidden preferences and item characteristics. This approach not only improves prediction accuracy but also allows for the incorporation of additional information such as temporal dynamics and social relationships, offering a comprehensive framework for personalized recommendations [9].

Neighborhood-Based Methods: CF uses neighborhoods (similar users/items). Techniques vary in forming neighborhoods (k-nearest neighbors etc.) and weighting neighbors for accurate recommendations.

Literature Review: Neighborhood-Based Methods

Neighborhood-based methods in recommender systems are celebrated for their simplicity, justifiability, efficiency, and stability. These methods rely on local associations to make recommendations, which can be especially useful for suggesting items outside a user's typical preferences, aiding in the discovery of new genres or favorites. Despite their advantages, these methods may face challenges like limited coverage and sensitivity to sparse ratings. Advanced techniques have been developed to address these issues, enhancing their application in various recommender systems [10].

Deep Learning: Neural networks bring sophistication to recommenders, learning complex patterns from rich data including text, images, and sequential interactions.

Literature Review: Deep Learning

Deep learning has significantly impacted recommender systems by addressing challenges like accuracy, scalability, and cold-start problems. It excels in capturing complex non-linear relationships between users and items, offering substantial improvements over traditional methods. This evolution is detailed in a systematic review, providing insights into the integration of deep learning techniques within recommender systems, emphasizing its effectiveness across various domains [11].

1.2.2. Evaluation

Offline Evaluation: Accuracy metrics (Precision, Recall, F1-score, MAP, NDCG) assess recommendations against withheld (historical) data.

Online Evaluation: Live A/B testing and user studies gauge recommendation impact on user satisfaction, engagement, and overall experience.

Literature Review: Evaluating recommender systems involves balancing offline and online metrics, with offline evaluation focusing on accuracy metrics like Precision, Recall, F1-score, MAP, and NDCG. However, these metrics have limitations, notably in capturing the user's qualitative experience such as serendipity, surprise, and diversity. Online evaluations through A/B testing and user studies offer insights into user satisfaction and engagement, providing a more holistic view of a system's impact [12].

1.2.3. Conclusion

Recommendation systems represent a continuously evolving field. Advancements in machine learning, increasing data availability, and focus on responsible implementation propel further progress. This constant refinement holds the key to unlocking greater personalization, user engagement, and enhanced decision-making within the ever-expanding world of online interaction.

1.3. Hybrid Approaches – Motivations and Strategies

Hybrid recommendation systems combine the strengths of collaborative filtering and content-based techniques, aiming to create a more powerful and flexible recommendation engine. These approaches often lead to a performance boost and enhanced ability to tailor recommendations to diverse user needs.

1.3.1. Motivations for Employing Hybrid Approaches

1. Addressing the Cold-Start Problem: When new users or items enter a system, it can be challenging to generate meaningful recommendations due to a lack of historical interaction data. Hybrid recommenders tackle this by incorporating content-based features that enable recommendations even without ample user-item interaction history.
2. Overcoming Sparsity: Recommender systems frequently deal with user-item matrices where most entries are empty (unrated items). Hybrid approaches mitigate this sparsity issue by using both historical ratings and item features to fill in the blanks.
3. Improving Recommendation Accuracy: Combining the complementary views gleaned from collaborative and content-based filtering often leads to superior accuracy compared to employing either technique in isolation.
4. Increasing Serendipity and Diversity: Over-reliance on purely collaborative approaches can lead to a "filter bubble," where users receive recommendations that mirror their existing preferences too closely. Hybrid systems inject surprise and diversity by relying on a broader range of signals derived from both content and interactions.
5. Adapting to Varied User Profiles and Needs: Within a large user base, some individuals may have richer interaction histories than others. Hybrid systems adapt recommendations to different levels of user data availability, allowing for flexible personalization.

1.3.2. Strategies for Building Hybrid Recommenders

1. Hybrid recommendation approaches differ in the manner they fuse collaborative and content-based elements. Here are some common strategies:
2. Weighted Hybrids: The simplest approach. Here, scores from individual recommenders (CF and content-based) are calculated and then combined using a weighted average.
3. Switching Hybrids: These systems rely on decision logic to choose between CF or content-based recommendations based on factors like the availability of user data or the nature of the item being recommended.

4. Mixed Hybrids: Recommendations emerge by presenting results from several recommender systems simultaneously, enabling the user to have broader exposure to items they may find interesting.
5. Feature Combination Hybrids: In this approach, features used in the content-based component are augmented with data derived from collaborative interactions, providing a richer basis for recommendations.
6. Feature Augmentation Hybrids: Output from one recommendation technique is used as an additional input feature for another recommender system.
7. Cascade Hybrids: This multistage strategy entails refining recommendations in successive steps. Early stages may focus on using broader signals for a quick first pass; later stages might focus on fine-tuning the recommendation set through a different technique.
8. Meta-level Hybrids: This approach builds a learned model that predicts which base recommender algorithm will generate the best result for a particular user-item pair.

1.3.3. Real-world situations

Scenario 1: Movie Recommendation

- Challenge: Movie recommender systems face sparsity challenges (many users do not rate many movies) and the cold-start problem for new releases.
    - Hybrid Approach: A hybrid approach could leverage:
    - Collaborative Filtering: Identifies similar users or movies based on available ratings.
    - Content-Based Filtering: Uses metadata like genre, cast, director, and plot keywords to discover undiscovered gems.
    - Strategy: A feature combination hybrid might include collaborative data (e.g., which genres this user typically enjoys) as inputs to the content-based engine, improving personalization.

Scenario 2: E-commerce Product Recommendation

- Challenge: Recommending items to new users (no purchase history) and tackling extremely vast product catalogs with potential niche items.
    - Hybrid Approach: A combination of:
    - Collaborative Filtering: Looks at purchase histories of similar shoppers.
    - Content-Based Filtering: Analyzes product descriptions, specifications, and categories to match user browsing behavior with product attributes.
    - Strategy: A cascade hybrid could work effectively here. An initial content-based stage narrows down the massive catalog for potential matches. A collaborative-filtering refinement follows within the subset for highly targeted recommendations.

Scenario 3: Music Streaming Recommendation

- Challenges: Recommending less popular and new artists can be difficult while also managing vast amounts of musical content.
- Hybrid Approach: A system can exploit:
    - Collaborative Filtering: Discovers patterns in shared listening histories.
    - Content-Based Filtering: Uses audio analysis (tempo, instruments, etc.), musical metadata (genre, tags), or song lyrics to find similarities.

- Strategy: Feature augmentation here works well. Collaborative data about a user's typical mood/activity listening preferences can enhance content-based recommendations for new discoveries.

1.4. Challenges and Applications for E-commerce

E-commerce has transformed the way products and services are bought and sold, offering convenience, a vast selection, and the ability to tap into a global marketplace. However, the burgeoning world of e-commerce also presents specific challenges for businesses and necessitates strategies designed to thrive in a highly competitive digital environment.

1.4.1. Key Challenges

Challenge 1: Building Trust and Credibility

- Impact: A lack of trust can lead to higher bounce rates, lower conversion rates, and reputation damage. Customers need reassurance before handing over money and sensitive information.
- Solutions:
  - High-Quality Visuals and Content: Invest in professional product photography, videos, and 360-degree views. Provide rich, accurate descriptions (materials, dimensions, uses, etc.).
  - Social Proof: Encourage reviews and ratings. Display badges for recognized industry certifications, trust seals, or endorsements. Feature testimonials from satisfied customers.
  - Security: Protect every page with HTTPS (SSL/TLS certificates). Visually communicate safe transactions using security logos. Display a clear privacy policy and terms of use.

Challenge 2: Intense Competition

- Impact: In a crowded online marketplace, getting noticed and differentiating yourself from similar sellers is vital to drive sales and retain customers.
- Solutions:
  - Niche-Down: Find a focus area (product type, audience, specific values) for strong branding. Avoid trying to be everything to everyone.
  - Customer Service Excellence: Initiative-taking support, quick resolution of issues, and personalized interaction build loyalty that beats just competing on price.
  - Targeted Marketing: Employ paid ads, retargeting, email newsletters, and social media to reach your ideal audience within the noise.
  - SEO and Content Creation: Blog posts, product guides, and videos improve your visibility on search results and position you as an expert in your field.

Challenge 3: Shopping Cart Abandonment

- Impact: Users leaving mid-purchase lost revenue. Each abandoned cart represents potential sales unfulfilled. Identifying friction points is key to reducing this.
- Solutions:
  - Seamless Checkout: Minimize steps, allow guest checkout, and avoid unexpected extra costs (like surprise shipping fees) at the last stage.
  - Progress Indicators: Show the checkout process visually to reduce anxieties about how many steps are left.

- Save Cart Functionality: Let users easily save items for later if they are not ready to decide immediately.
- Abandoned Cart Remarketing: Automated emails or messages reminding users of items left behind help win back a percentage of lost sales.

1.4.2. Applications of E-commerce

E-commerce goes far beyond traditional retail. Here is a look at specific applications highlighting its broader impact:

1. Business-to-Consumer (B2C)

   – Description: The most familiar form of e-commerce, where businesses sell products or services directly to individual consumers. Examples: Amazon, fashion e-tailers, online bookstores.
   – Challenges:
      - Intense competition for attention and market share.
      - Consumer-centric marketing, exceptional website UX.
      - Handling returns and individual consumer issues.

2. Business-to-Business (B2B)

   – Description: Transaction of goods or services between businesses. This includes wholesalers, supply chain procurement, business software platforms, specialized industry marketplaces.
   – Challenges:
      - Often entails complex, higher-value sales processes.
      - May prioritize functionality and efficiency over design-heavy consumer interfaces.
      - Focus on relationship management and long-term contracts.
      - Integration with existing procurement systems may be necessary.

3. Consumer-to-Consumer (C2C)

   – Description: Platforms facilitate transactions between individuals. This includes online marketplaces (eBay, Etsy) and peer-to-peer services (task marketplaces, ride-sharing platforms).
   – Challenges:
      - Maintaining trust and safety mechanisms on the platform.
      - Dispute resolution systems and buyer/seller protection policies are crucial.
      - Network effects – a critical mass of buyers and sellers is needed for success.

1.5. Conclusion

This analysis has looked at various types of recommendation systems, including collaborative filtering, content-based filtering, and hybrid approaches. Collaborative filtering techniques, such as user-based and item-based Collaborative filtering, use historical user-item interactions to generate recommendations. Content-based filtering, on the other hand, relies on item attributes and

characteristics to match user preferences. Hybrid systems combine the strengths of both approaches, often addressing limitations such as the cold-start problem and data sparsity.

Key concepts and techniques in recommendation systems include user profiles, item representation, similarity metrics, matrix factorization, and deep learning. User profiles capture user interests and behaviors, while item representation enables content-based matching. Similarity metrics measure the likeness between users or items, and matrix factorization techniques address sparsity and scalability issues. Deep learning brings sophistication to recommendation systems by learning complex patterns from rich data sources.

The evaluation of recommendation systems is necessary for assessing their performance. Offline evaluation focuses on accuracy metrics, while online evaluation measures user satisfaction and engagement through testing and user studies. Hybrid approaches, which combine collaborative filtering and content-based techniques, are motivated by the need to address challenges such as the cold-start problem, data sparsity, and improving recommendation accuracy. These strategies often lead to enhanced personalization, user engagement, and decision-making in the ever-expanding world of online interaction.

2. Project

2.1. Requirements specification

2.1.1. Restrictions on the project

1. Solution Constraints: The system must efficiently manage a large dataset within the processing limits of Streamlit and SQLite, ensuring quick response times and high availability.
2. Deployment Environment: The solution will be developed for a web-based interface, deployed using Streamlit, and accessible through standard web browsers. SQLite will be used for database management, which should be configured to manage concurrent access and secure data storage effectively.
3. Time frame: Development is structured to be completed within the academic timeline, aiming for a prototype ready for testing at the end of the first year of study.
4. Budget: As a thesis project, there is no allocated budget for development expenses; however, resource allocation for hosting and maintenance will be minimal, utilizing free tiers of cloud services where possible.

2.1.2. Functional requirements

1. User Account Management: The system shall enable users to create and manage their accounts, including personal information.
2. Personalized User Experience: The system shall utilize machine learning to analyze user data (purchase history) to generate personalized product recommendations.
3. Interactive Recommendation Features: Users shall be able to add recommended items directly to their shopping basket from the recommendation interface.
4. Basket Functionality: Users shall be able to add, remove, and edit items in their shopping basket, including adjustments to item quantities.
5. Checkout Process: The system shall facilitate a seamless checkout process.
6. Order History: Users shall have access to their order history.
7. Data-Driven Inventory Management: The system shall analyze sales data and external factors to forecast demand, aiding in inventory decision-making.
8. Visualization and Reporting Tools: Inventory managers shall receive visual forecasts and reports to support strategic planning.
9. Comprehensive Product Management: Administrators shall be able to add, update, manage product listings, and categorize products according to various attributes.

2.1.3. Non-functional requirements

1. Look and Feel: The user interface should be clean, modern, and visually appealing, ensuring an engaging user experience.
2. Usability: The system should be intuitive and user-friendly, accommodating users with varying levels of technical skills. Minimal training should be required to use the system, with clear documentation and helpful tips integrated directly into the user interface.
3. Performance: All interactions within the system, including data retrieval, processing, and rendering, should be optimized for speed. Response times for loading pages and executing database queries with SQLite should not exceed 3 seconds under normal operational conditions.
4. Operational Conditions: Designed to be robust and dependable, the system must ensure uptime close to 90%, with streamlined error handling and recovery processes to minimize downtime.

5. Maintainability and Portability: The codebase should be well-organized, modular, and commented, facilitating easy updates and maintenance. The application should be platform-independent, capable of running in any standard web environment without requiring specific configurations.
6. Security: Adequate measures must be implemented to protect data integrity and privacy.
7. Scalability: Although initially developed for a limited user base, the system should be capable of scaling up to accommodate more users and complex data interactions. The architecture should support scalability both in terms of database (SQLite) management and user load handling in Streamlit.

2.1.4. Technical specification

Hardware Requirements Server Specifications (for hosting the application):

- Processor: Intel i5 or equivalent (minimum eight cores)
- Memory: 16 GB RAM
- Storage: 500 GB SSD (Solid State Drive)
- Network: 1 Gbps Ethernet

Client Specifications:

- Processor: Minimum Intel Celeron.
- Memory: 4 GB RAM
- Storage: 50 GB HDD (Hard Disk Drive) or SSD
- Network: Stable internet connection with a minimum speed of 10 Mbps
- Web Browser: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge, etc.

2.2. System design

2.2.1. Logical architecture of the system

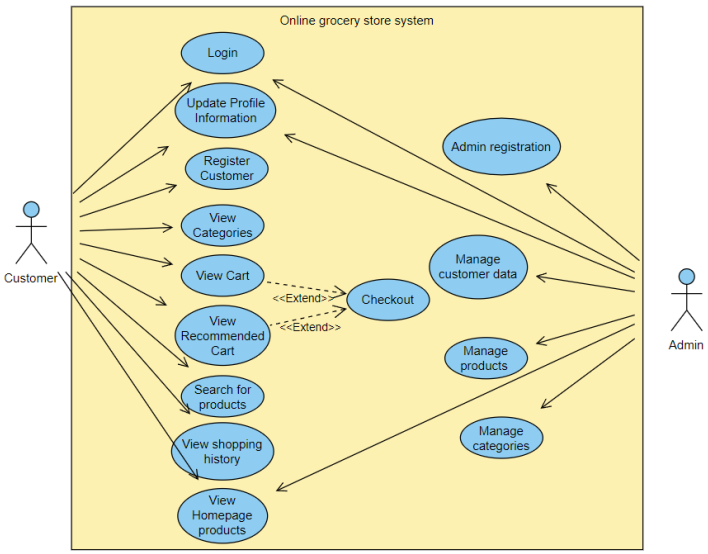

Figure 1: System use case.

Use Case Diagram for Online Grocery Store System(see Figure 1)

Actors:

1. Customer:

- Represents the end-users who interact with the online grocery store to perform various shopping-related activities.

2. Admin:

- Represents the administrative users who manage the system's backend operations, including product, customer, and category management.

Customer Use Cases:
- Login: Allows customers to log into their accounts using their credentials.
- Update Profile Information: Enables customers to update their personal information, such as name, email, and password.
- Register Customer: Allows new users to create an account and register as a customer.
- View Categories: Provides customers with the ability to browse different product categories.
- View Cart: Displays the current items in the customer's shopping cart.
- Checkout (extends View Cart): Facilitates the process of purchasing items in the cart and completing the transaction.
- View Recommended Cart: Shows personalized product recommendations based on the customer's past purchases and preferences.
- Search for Products: Enables customers to search for specific products using keywords.
- View Shopping History: Allows customers to view their past orders and purchase history.
- View Homepage Products: Displays featured or popular products on the homepage for the customer.

Admin Use Cases:
- Admin Registration: Allows new administrators to register and create an admin account.
- Manage Customer Data: Enables administrators to view, update, and manage customer information.
- Manage Products: Provides functionality for adding, updating, and deleting products in the inventory.
- Manage Categories: Allows administrators to organize products into categories and manage these categories.

Extends Relationship:
- The Checkout use case extends the View Cart use case, indicating that checkout is an optional but related extension of viewing the cart.
- The View Recommended Cart use case extends the View Cart use case, showing that recommended products can be viewed as part of the cart experience.

2.2.2. Operational logic of the system

2.2.2.1. Activity diagrams

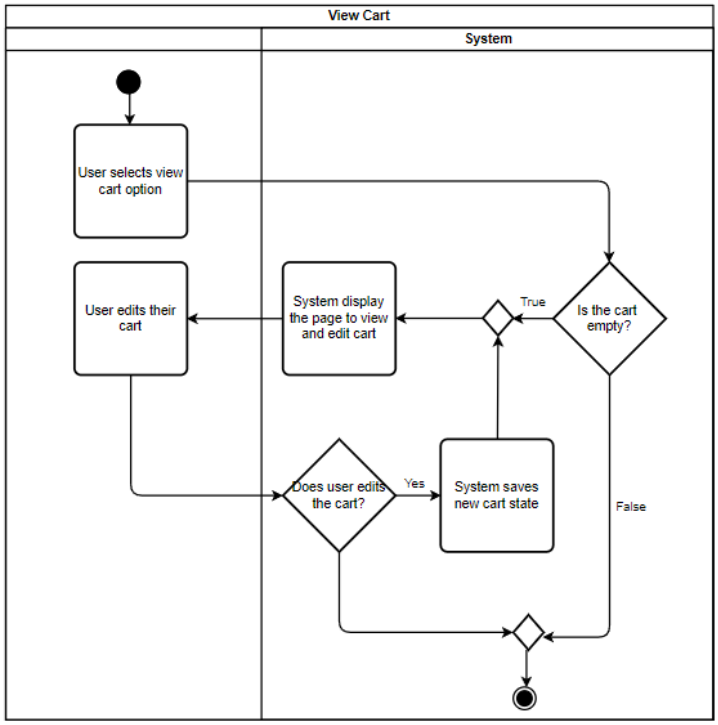

Figure 2: View cart activity diagram.

View cart activity process(see Figure 2):

- Actor: Customer
- Description: Shows products a user has added.
- Pre-condition: The user is logged into the system and has navigated to the shopping cart section.
- Trigger Condition: View cart button is clicked.
- Post-condition: The user's shopping cart is updated according to their edits, and the system accurately reflects the current state of the cart.

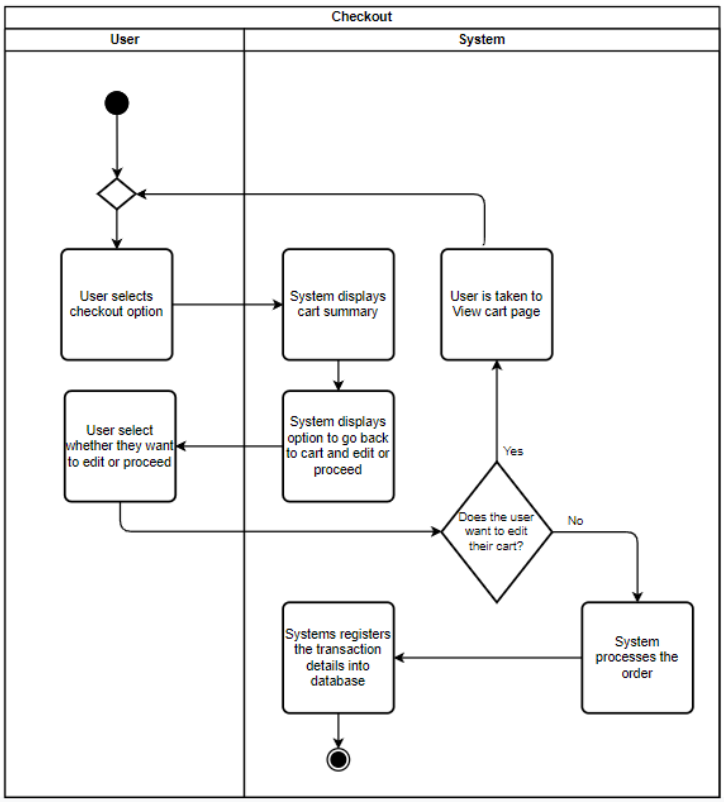

Figure 3: Checkout activity diagram.

Checkout activity process(see Figure 3):

- Actor: Customer
- Description: The process where customers are satisfied with the products in their cart and are done shopping.
- Pre-condition: The user has items in their cart and initiates the checkout process.
- Trigger Condition: Checkout button is clicked.
- Post-condition: The user has successfully checked out and the transaction details are stored in the system.

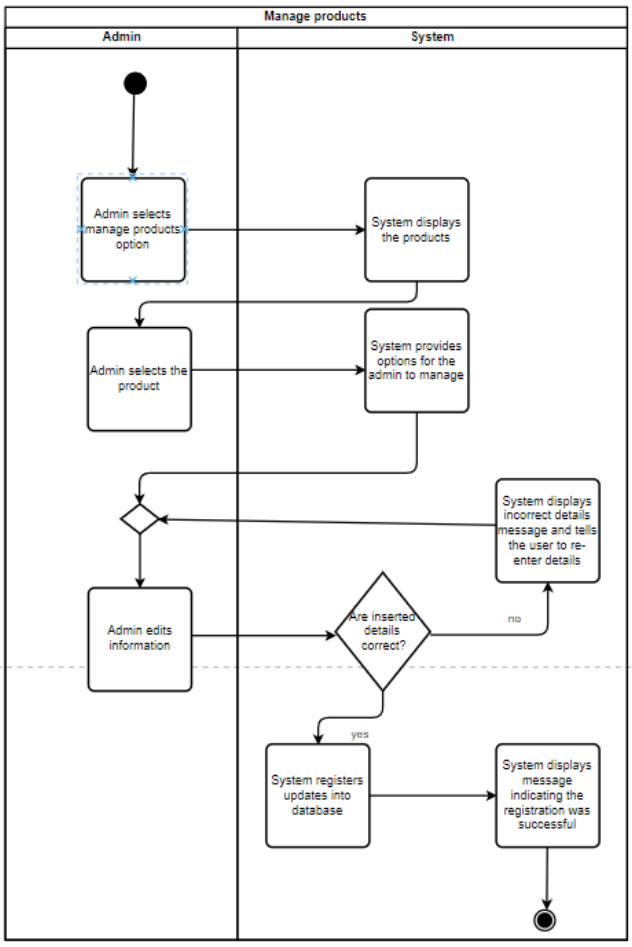

Figure 4: Manage products activity diagram.

Manage products activity process(see Figure 4):

- Actor: Admin
- Description: The admins can change product details.
- Pre-condition: The admin is logged into the system .
- Trigger Condition: Edit button is clicked.
- Post-condition: The product information is updated in the system's database, and the admin receives confirmation of the successful update.

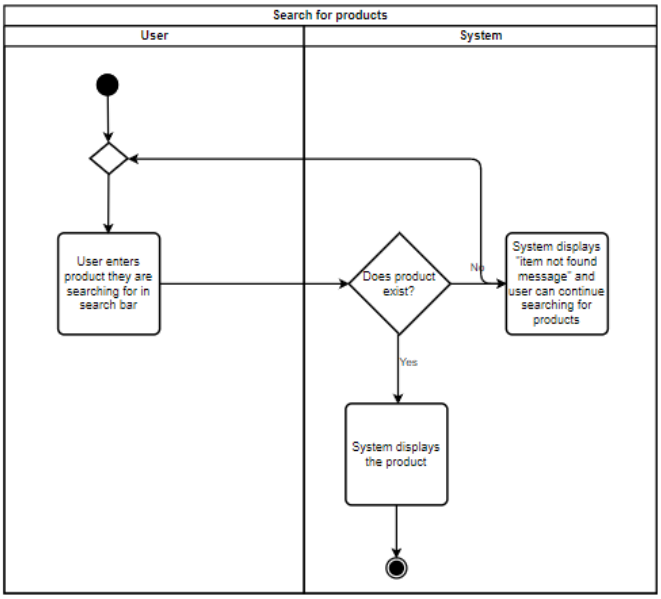

Figure 5: Search for products activity diagram.

Search for products activity process(see Figure 5):

- Actor: Customer and Admin
- Description: Browse products from the search bar.
- Pre-condition: The actor is on the e-commerce platform.
- Trigger Condition: The use case is initiated when the actor inputs a search term into the search bar and submits the search request.
- Post-condition: The actor is either shown the product they searched for or receives a message that the product could not be found.

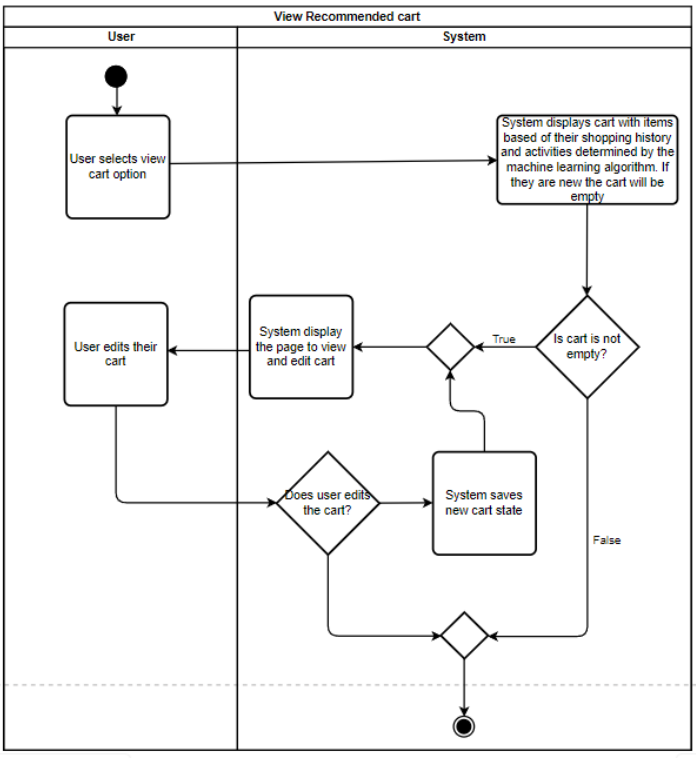

Figure 6: View recommended cart activity diagram.

View recommended cart activity process(see Figure 6):

- Actor: Customer
- Description: Shows products the system added using machine learning.
- Pre-condition: The user is logged into the system and has navigated to the recommended cart button.
- Trigger Condition: View recommended cart button is clicked.
- Post-condition: The user's recommended cart is updated according to their edits, and the system reflects the current state of the cart.

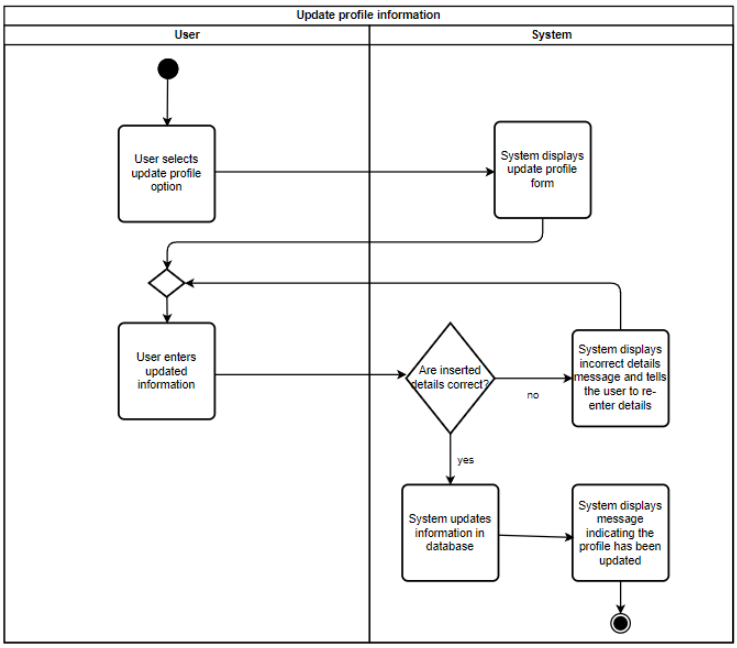

Figure 7: Update profile information activity diagram.

Update profile information activity process(see Figure 7):

- Actor: Customer and Admin
- Description: The actor can change their profile details.
- Pre-condition: The user is logged into the system.
- Trigger Condition: Update profile button is clicked.
- Post-condition: The user's profile information is updated in the database, and the user receives confirmation of the changes.

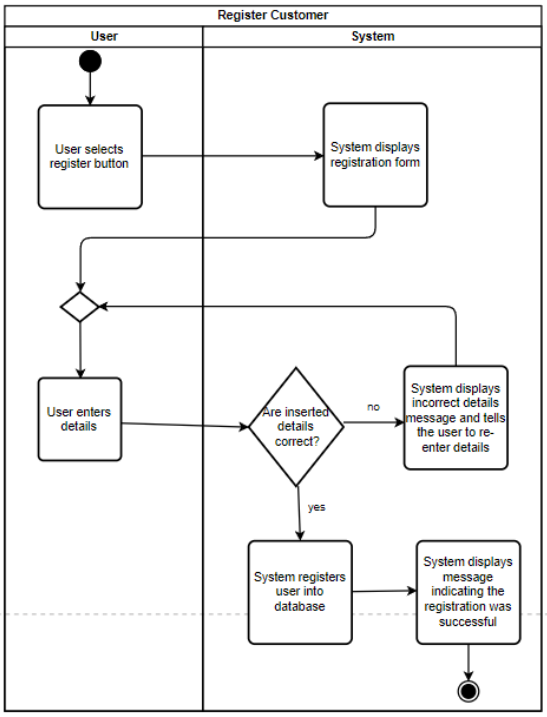

Figure 8: Register customer activity diagram.

Register customer activity process(see Figure 8):

- Actor: Customer
- Description: A new customer wants to create an account.
- Pre-condition: The user is on the registration page of the system.
- Trigger Condition: The user selects the registration button on the login page.
- Post-condition: The user's account is created in the system, and they can now log in with their credentials.

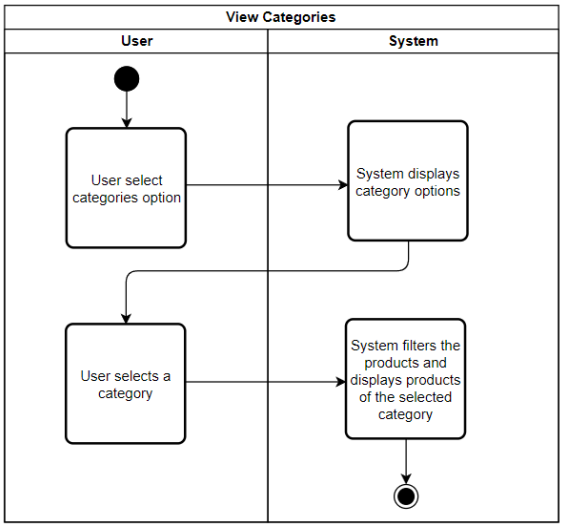

Figure 9: View categories activity diagram.

View categories activity process(see Figure 9):

- Actor: Customer

– Description: Browse products in a specified category.
– Pre-condition: The user is on the homepage.
– Trigger Condition: The user selects the categories dropdown on the homepage.
– Post-condition: The system displays the products they belong to the category selected.

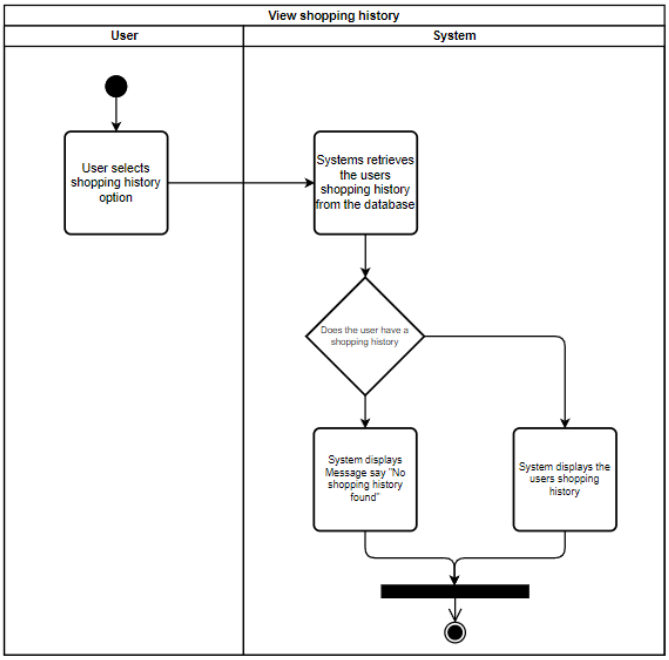

Figure 10: View shopping history activity diagram.

View shopping history activity process(see Figure 10):

– Actor: Customer
– Description: User wants to see their shopping history.
– Pre-condition: The user is logged in.
– Trigger Condition: User selects the shopping history button.
– Post-condition: System displays the users shopping history receipts .

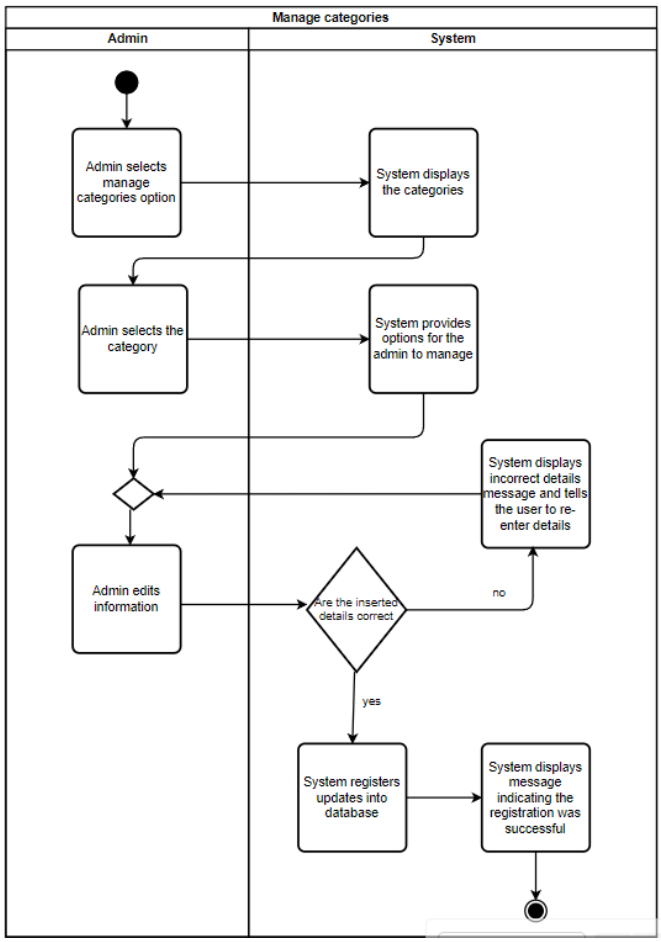

Figure 11: Manage categories activity diagram.

Manage categories activity process(see Figure 11):

- Actor: Admin
- Description: The manager wants to edit something about a category.
- Pre-condition: The admin is logged in.
- Trigger Condition: Manage categories button is clicked.
- Post-condition: The product information is updated in the system's database, and the admin receives confirmation of the successful update.

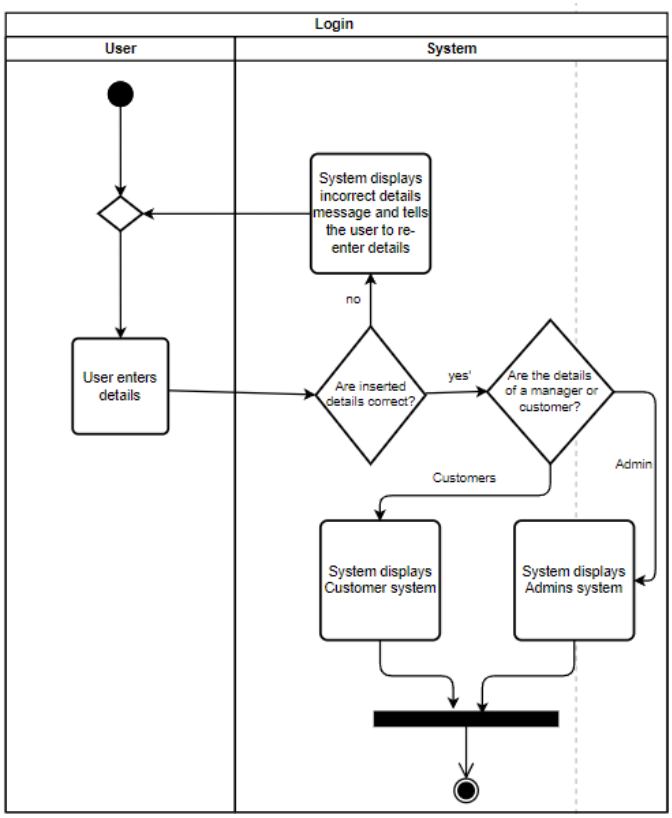

Figure 12: Login activity diagram.

Login activity process(see Figure 12):

- Actor: Customer and Admin
- Description: The actor wants to login into the system.
- Pre-condition: The actor is on the website.
- Trigger Condition: Login button is clicked.
- Post-condition: The actor is now logged in to the website with access to more website features.

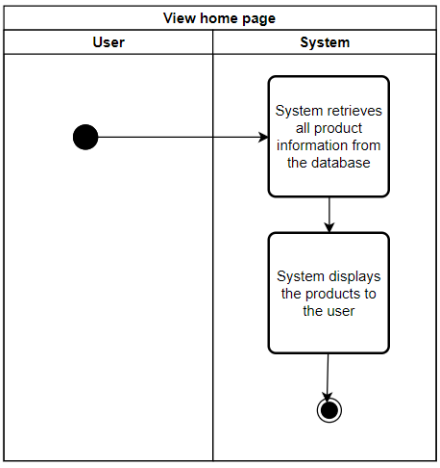

Figure 13: View Homepage activity diagram.

View Homepage activity process(see Figure 13):

- Actor: Customer and Admin
- Description: Browse products and receive personalized recommendations.
- Pre-condition: User is not yet on the website or on another page.

- Trigger Condition: Homepage button is clicked.
- Post-condition: The website displays the homepage products and features to the actor.

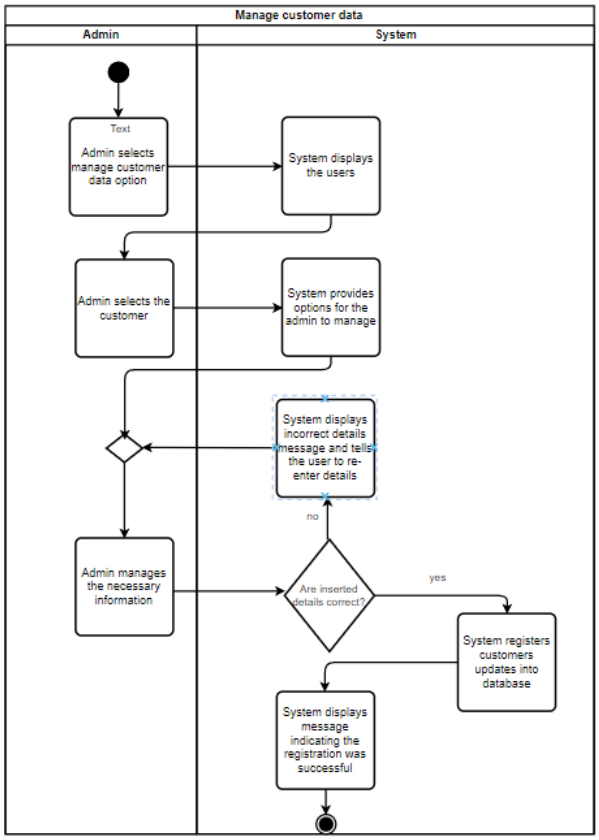

Figure 14: Manage customer data activity diagram.

Manage customer data activity process(see Figure 14):

- Actor: Admin
- Description: The manager wants to edit something about a customer.
- Pre-condition: User is logged into their account.
- Trigger Condition: Manage customers button is clicked.
- Post-condition: The customer information is updated in the system's database, and the admin and customer receive confirmation of the successful update.

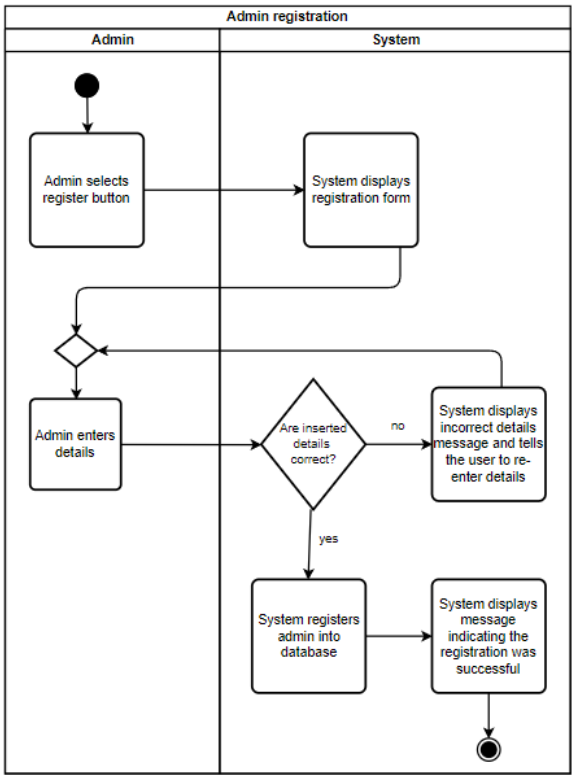

Figure 15: Admin registration activity diagram.

Admin registration activity process(see Figure 15):

- Actor: Admin
- Description: Admin wanted to add another admin to the system.
- Pre-condition: User is logged into their account.
- Trigger Condition: Register admin button is clicked.
- Post-condition: The admin receives the success indicator about the new admin.

2.2.2.2. Sequence diagrams.

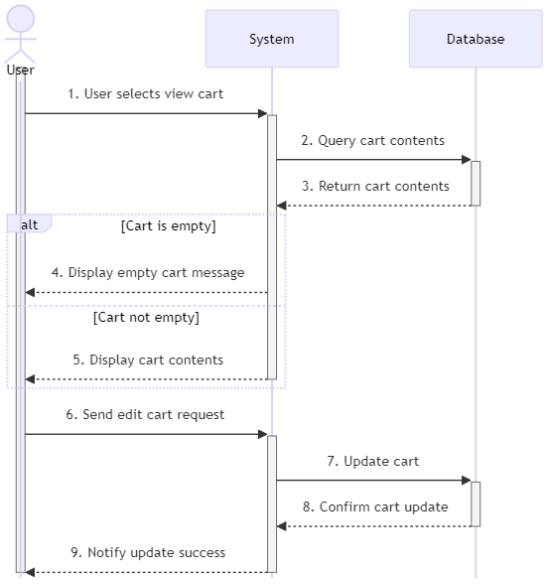

Figure 16: View cart sequence diagram.

Actors - User, System, Database

Flow Description(see Figure 16):

- User: User selects "view cart".
- System: Queries cart contents from the Database.
- Database: Returns cart contents to the System.

Alternate Flow:

- If cart is empty:
  - The system displays an “empty cart" message.
- If cart is not empty:
  - System displays cart contents.
- User: Sends an "edit cart" request.
- System: Updates the cart in the Database.
- Database: Confirms cart update to the System.
- System: Notifies the user of the update success.

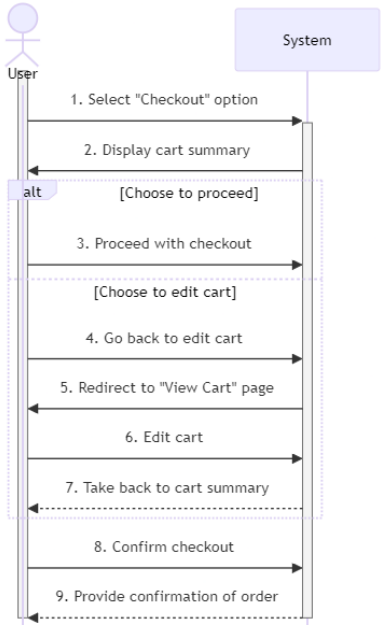

Figure 17: Checkout sequence diagram.

Actors - User, System

Flow Description(see Figure 17):

- User: Selects "Checkout" option.
- System: Displays cart summary.

Alternate Flow:

- If user chooses to proceed:
  - User proceeds with checkout.
- If user chooses to edit cart:
  - User goes back to edit cart.

- System redirects to the “View Cart" page.
- User edits cart.
- System takes user back to cart summary.

– System: Confirms checkout.
– System: Provides confirmation of order.

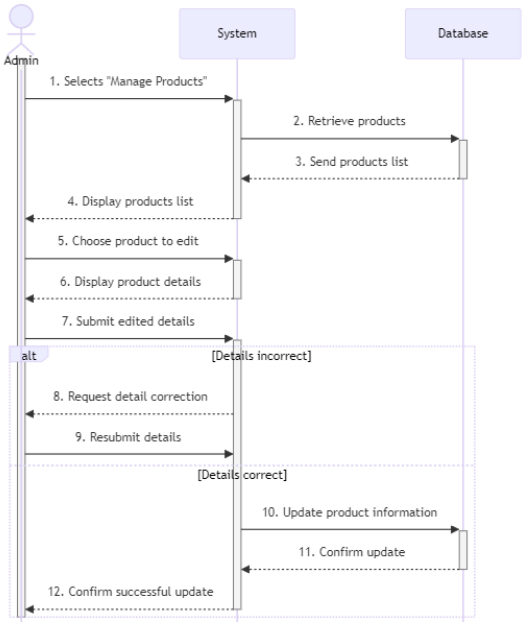

Figure 18: Manage products sequence diagram.

Actors – Admin, System, Database

Flow Description(see Figure 18):

– Admin Action: Selects "Manage Products".
– System: Retrieves products from the Database.
– Database: Sends products list to the System.
– System: Displays products list.
– Admin Action: Chooses a product to edit.
– System: Displays product details.
– Admin Action: Submits edited details.

Alternate Flow:

– If details are incorrect:
  - System requests detailed correction.
  - Admin resubmits details.
– If details are correct:
  - System updates product information in the Database.
– Database confirms the update.
– System: Confirms successful update to the Admin.

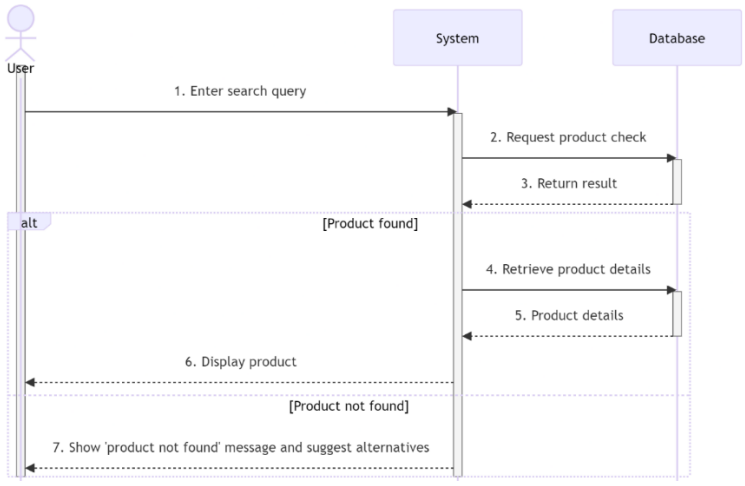

Figure 19: Search query sequence diagram.

Actors - User, System, Database

Flow Description(see Figure 19):

- User: Enters a search query.
- System: Requests product check from the Database.
- Database: Returns the result to the System.

Alternate Flow:

- If product is found:
  - System retrieves product details from the Database.
  - Database provides product details to the System.
  - System displays the product.
- If product is not found:
  - System shows "product not found" message and suggests alternatives.

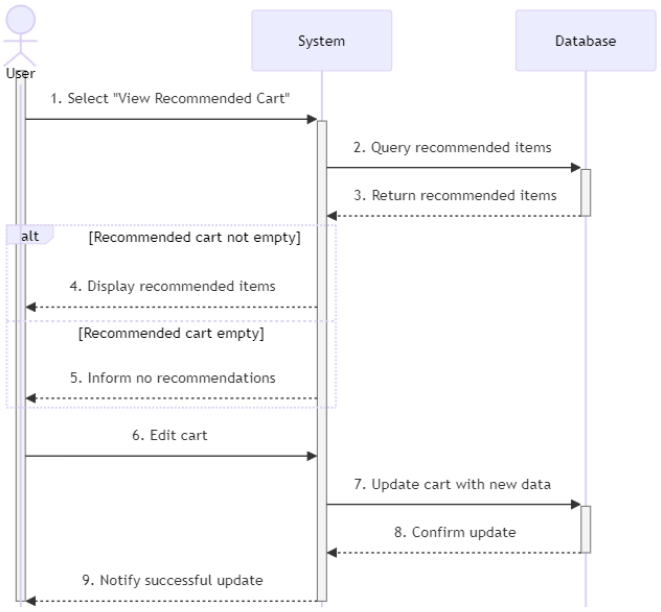

Figure 20: View Recommended cart sequence diagram.

Actors - User, System, Database

Flow Description(see Figure 20):

- User: Select "View Recommended Cart".
- System: Queries recommended items from the Database.
- Database: Returns recommended items to the System.

Alternate Flow:

- If recommended cart is not empty:
  - The system displays recommended items.
- If recommended cart is empty:
  - System informs the user that there are no recommendations.
- User: Edits the cart.
- System: Updates cart with new data in the Database.
- Database: Confirms the update to the System.
- System: Notifies the user of the successful update.

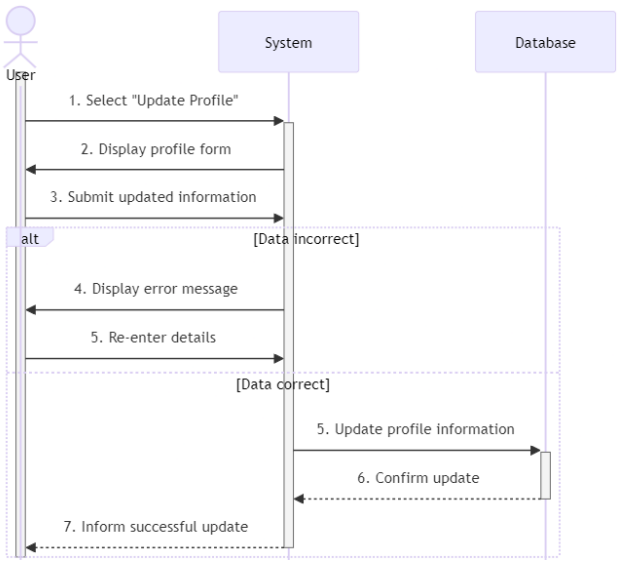

Figure 21: Update profile sequence diagram.

Actors – User, System, Database

Flow Description(see Figure 21):

- User: Selects "Update Profile".
- System: Displays profile form.
- User: Submits updated information.

Alternate Flow:

- If data is incorrect:
  - System displays error message.
  - User re-enters details.
- If data is correct:
  - System updates profile information in the Database.
  - Database confirms the update.

– System: Informs the user of the successful update.

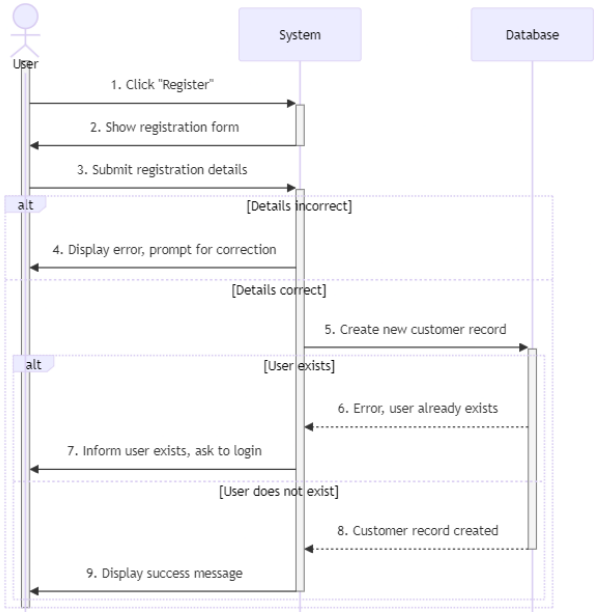

Figure 22: Register sequence diagram.

Actors - User, System, Database

Flow Description(see Figure 22):

– User: Click "Register".
– System: Shows registration form.
– User: Submits registration details.

Alternate Flow:

– If details are incorrect:
  - System displays errors and prompts for correction.
– If details are correct:
  - System attempts to create a new customer record in the Database.

Alternate Flow:

– If user already exists:
  - Database returns error indicating user already exists.
  - System informs user of existing account and asks to login.
– If user does not exist:
  - Database creates customer records.
  - System displays success message.

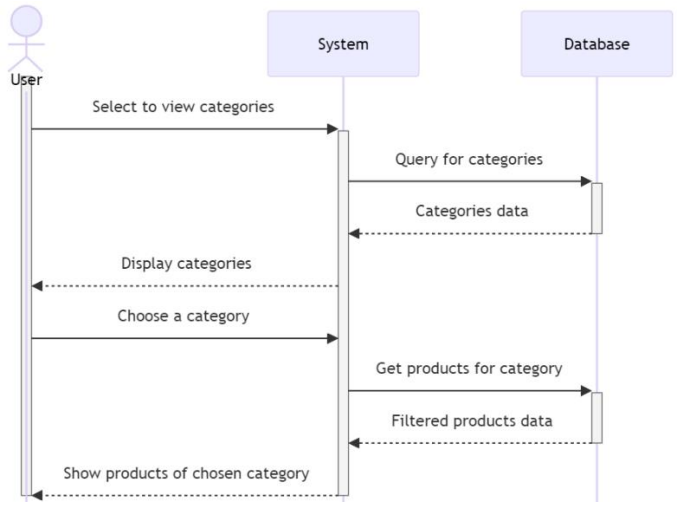

Figure 23: View categories sequence diagram.

Actors – User, System, Database

Flow Description(see Figure 23):

- User: Selects to view categories.
- System: Queries the Database for categories.
- Database: Returns categories data to the System.
- System: Displays categories to the User.
- User: Chooses a category.
- System: Gets products for the chosen category from the Database.
- Database: Returns filtered products data to the System.
- System: Shows products of the chosen category to the User.

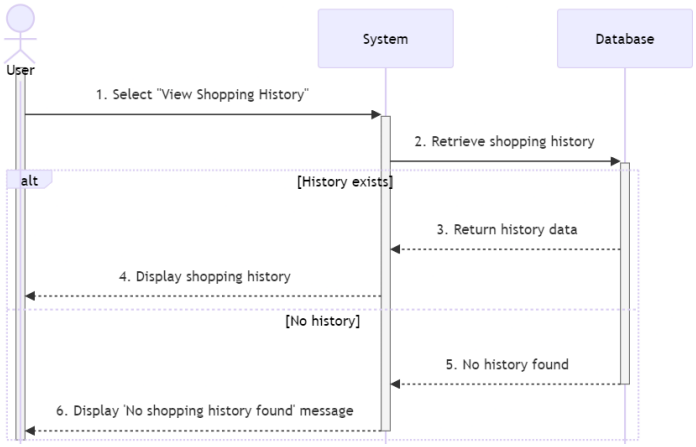

Figure 24: View shopping history sequence diagram.

Actors – User, System, Database

Flow Description(see Figure 24):

- User: Selects "View Shopping History".
- System: Retrieves shopping history from the Database.

Alternate Flow:

- If history exists:
  - Database returns history data.
  - System displays shopping history.
- If no history exists:
  - Database indicates no history found.
  - The system displays the “No shopping history found" message.

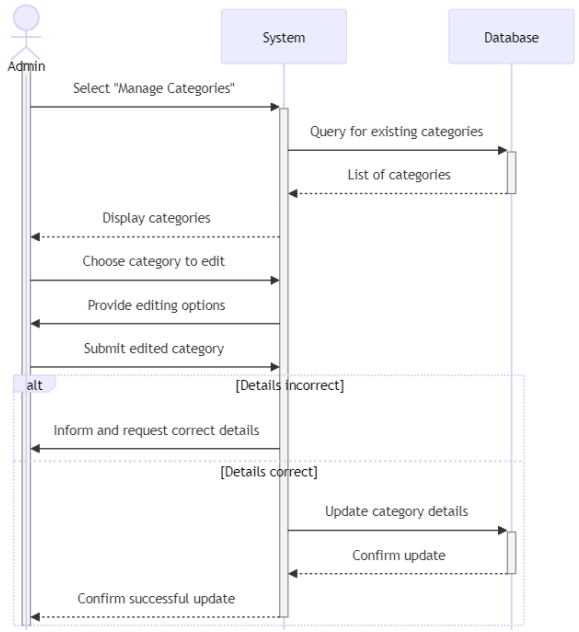

Figure 25: Manage categories sequence diagram.

Actors – Admin, System, Database

Flow Description(see Figure 25):

- Admin Action: Selects "Manage Categories".
- System: Queries for existing categories from the Database.
- Database: Returns list of categories to the System.
- System: Displays categories to the Admin.
- Admin Action: Chooses a category to edit.
- System: Provides editing options.
- Admin Action: Submits edited category.

Alternate Flow:

- If details are incorrect:
  - System informs and requests correct details.
- If details are correct:
  - System updates category details in the Database.
  - Database confirms the update.
- System: Confirms successful update to the Admin.

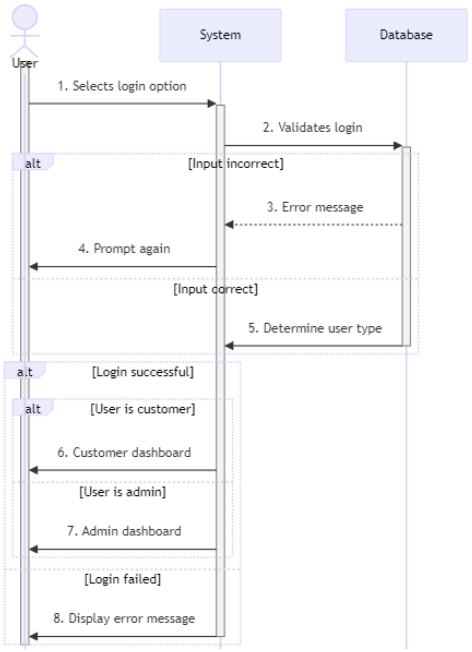

Figure 26: Login sequence diagram.

- Actors – User, System, Database
- Flow Description(see Figure 26):
- User: Selects login option.
- System: Validates login with the Database.

Alternate Flow:

- If input is incorrect:
    - Database returns error message.
    - System displays error message.
    - System prompts User to try again.
- If input is correct:
    - Database returns user details.
    - System determines user type.

Alternate Flow:

- If login is successful:
    - If user is a customer:
        - System displays the customer dashboard.
    - If user is an admin:
        - System displays the admin dashboard.
- If login fails:
    - System displays error message.

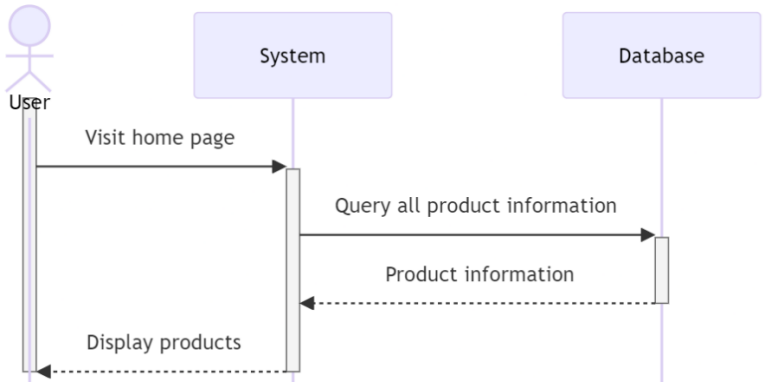

Figure 27: View homepage sequence diagram.

Actors – User, System, Database

Flow Description(see Figure 27):

- User: Visits the home page.
- System: Queries the Database for all product information.
- Database: Returns product information to the System.
- System: Displays products to the User.

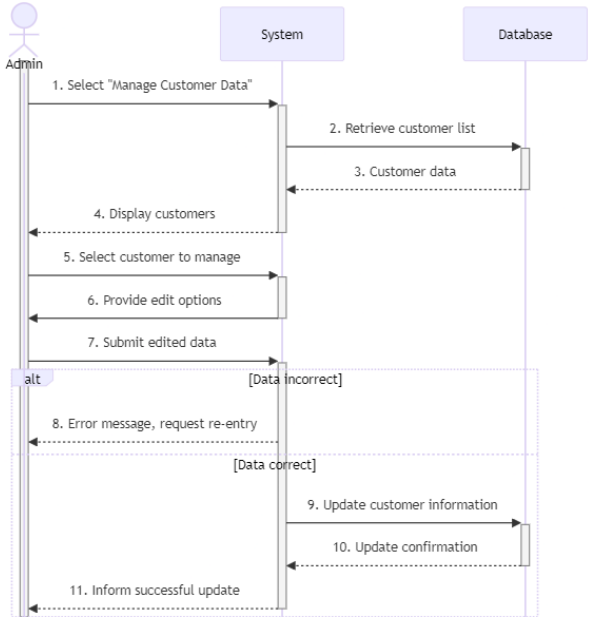

Figure 28: Manage customer data sequence diagram.

Actors – Admin, System, Database

Flow Description(see Figure 28):

- Admin Action: Selects "Manage Customer Data".
- System: Retrieves customer list from the Database.
- Database: Returns customer data to the System.
- System: Displays customers to the Admin.
- Admin Action: Selects a customer to manage.
- System: Provides editing options.
- Admin Action: Submits edited data.

Alternate Flow:

- If data is incorrect:
  - System displays an error message and requests re-entry.
- If data is correct:
  - System updates customer information in the Database.
  - Database confirms the update.
- System: Informs the Admin of the successful update.

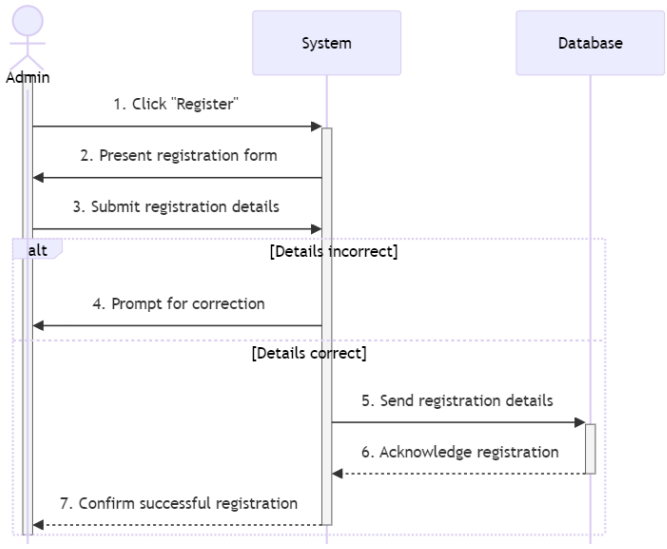

Figure 29: Register sequence diagram.

Actors – Admin, System, Database

Flow Description(see Figure 29):

- Admin Action: Clicks "Register".
- System: Presents registration form.
- Admin Action: Submits registration details.

Alternate Flow:

- If details are incorrect:
  - System prompts for correction.
- If details are correct:
  - System sends registration details to the Database.
  - Database acknowledges the registration.
- System: Confirms successful registration to the Admin.

2.2.3. Description of the artificial intelligence models to be used.

2.2.3.1. Demand Forecasting using Random Forest Regressor

Overview**:** The demand forecasting solution utilizes a Random Forest Regressor to predict daily product demand. This ensemble learning method combines multiple decision trees to enhance prediction accuracy and reduce overfitting.

Core Concepts:

- Random Forest Fundamentals:
- Constructs a multitude of decision trees from bootstrapped subsets of the dataset.
- Each tree operates on a random subset of features to ensure diverse predictions.
- Final output is the average of individual tree predictions, ensuring robustness.

Methodology:

1. Data Preparation:

   - Dataset Integration: Merged and cleaned the order and product datasets.
   - Date Processing: Converted OrderDate to datetime format to facilitate temporal analysis.
   - Aggregation: Grouped data by ProductName and OrderDate to compute daily demand totals.

2. Feature Engineering:

   - Created relevant features such as ProductName and Quantity.
   - Ensured no missing values to maintain dataset integrity.

3. Model Training:

   - Parameter Selection: Utilized GridSearchCV to optimize hyperparameters:
     - n_estimators = 100
     - max_depth = 10
     - min_samples_split = 2
     - min_samples_leaf = 1
   - Training: Split the data into training and testing sets and trained the model on X_train and y_train.

4. Evaluation:

   - Measured performance using Mean Squared Error (MSE) and R-squared to validate model accuracy.
   - Objective**:** To provide accurate daily demand forecasts for various products, aiding inventory, and supply chain management.

2.2.3.2. Product Recommendation System Using XGBoost

Overview**:** The product recommendation system employs XGBoost, an efficient gradient boosting algorithm, to predict the quantities customers are likely to purchase, enhancing customer satisfaction and sales efficiency.

Core Concepts:

- XGBoost Essentials:
  - Builds a sequence of models, each correcting the errors of its predecessor.
  - Uses decision trees as base learners, optimized through gradient boosting.
  - Incorporates regularization to prevent overfitting.

Methodology:

1. Data Handling:

- Data Cleaning: Removed duplicates and encoded categorical variables.
- Feature Creation: Developed features like TotalOrders, AvgQuantity, MostBoughtProduct, DayOfWeek, and Month.

2. Model Configuration:

   - Hyperparameter Optimization: Leveraged GridSearchCV for tuning:
     - n_estimators = 200
     - max_depth = 3
     - learning_rate = 0.01
     - objective = 'reg:squarederror'
   - Training Execution: Model trained on selected features to predict product quantities.

3. Prediction Process:

   - Generated predictions for all products for a specific customer.
   - Sorted products by predicted quantity to identify top recommendations.

4. Evaluation:

   - Evaluated using Mean Squared Error (MSE) to ensure model reliability.
   - Analyzed feature importance to understand model behavior.
   - Objective**:** To recommend products effectively by predicting purchase quantities, leveraging customer order history and product attributes.

Feature Details:
- ProductID: Unique identifier for products.
- CustomerID: Unique identifier for customers.
- DayOfWeek, Month: Temporal features to capture purchasing trends.
- TotalOrders, AvgQuantity, MostBoughtProduct: Customer behavior metrics.

2.2.4. Data model specification

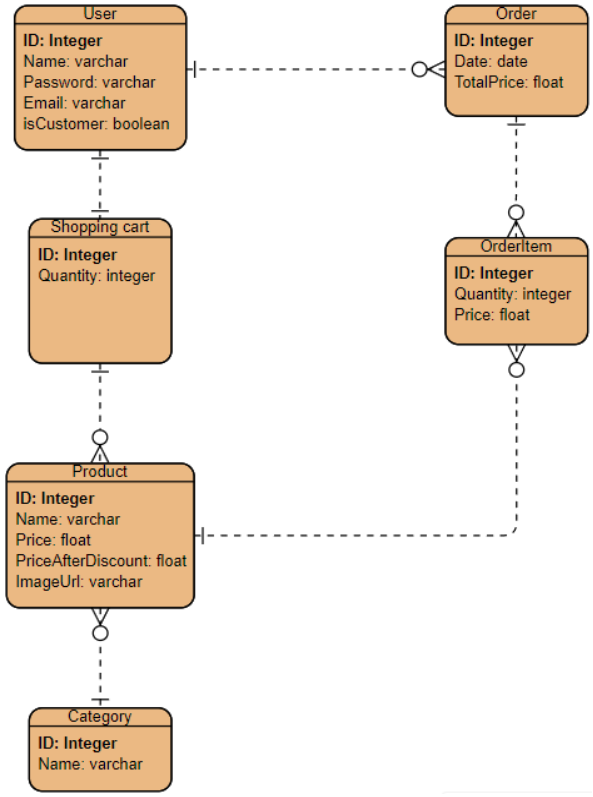

Figure 30: Entity relationship diagram.

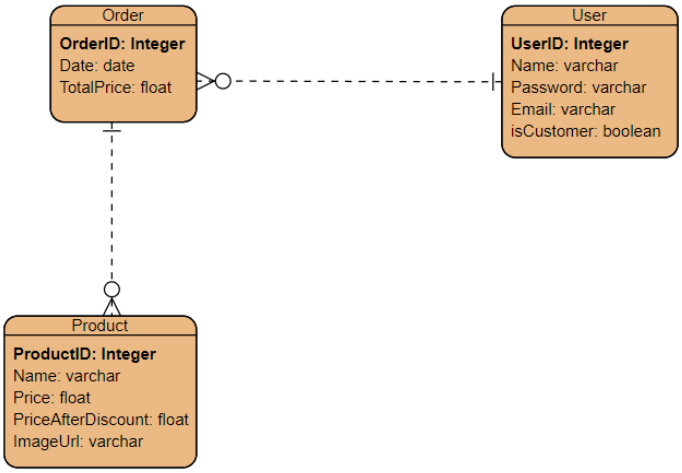

Figure 31: Entity diagram.

Database Schemas

The system uses a relational database to manage data, with the following key entities and relationships (see Figure 30 and Figure 31):

1. User: This table stores information about the users of the system.

   – ID (Integer): Unique identifier for each user.
   – Name (varchar): Name of the user.
   – Password (varchar): Password for user authentication.
   – Email (varchar): Email address of the user.

- isCustomer (boolean): Indicates whether the user is a customer.

2. Order: This table stores information about orders placed by users.
   - ID (Integer): Unique identifier for each order.
   - Date (date): Date when the order was placed.
   - TotalPrice (float): Total price of the order.
3. Order Item: This table stores information about individual items within an order.
   - ID (Integer): Unique identifier for each order item.
   - Quantity (integer): Quantity of the product ordered.
   - Price (float): Price of the product.
4. Product: This table stores information about the products available in the system.
   - ID (Integer): Unique identifier for each product.
   - Name (varchar): Name of the product.
   - Price (float): Price of the product.
   - PriceAfterDiscount (float): Price of the product after discount.
   - ImageUrl (varchar): URL of the product image.
5. Category: This table stores information about product categories.
   - ID (Integer): Unique identifier for each category.
   - Name (varchar): Name of the category.
6. Shopping Cart: This table stores information about items in a user's shopping cart.
   - ID (Integer): Unique identifier for each cart item.
   - Quantity (integer): Quantity of the product added to the cart.

2.2.5. User interface model

The user navigation system(see Figure 32) for the website begins with a login page where users can enter their credentials and register as a customer if they don't have an account. Once logged in, users can navigate through the website, filter products by searching for specific items or categories, view their selected products, and edit them. They can return to the homepage to add more products or proceed to the checkout page. Customers can also view a recommended shopping card generated based on a trained model of other customers, which they can remove, increase, or decrease quantities before adding the recommended items to the main cart. They can also edit their profile and view their shopping history. Admins have the same view as customers but can edit and delete items instead of adding products to the cart. They can add products to the website, register admins, manage customers, and access a demand forecasting page to see the top ten products on demand based on the customer's shopping history. Overall, the user navigation system provides a user-friendly and efficient shopping experience.

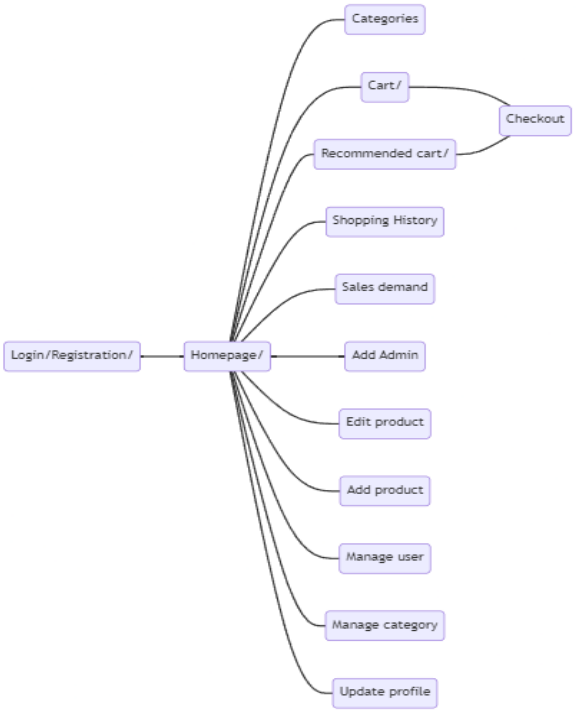

Figure 32: System User Interface Model.

2.3. Conclusions of the design part

Product recommendation and demand forecasting systems have been successful in their implementation and testing, but they have identified several shortcomings and limitations. Key issues include the cold start problem, which affects recommendations for new users or products without historical data, and the scalability of the system with large historical data and transaction volumes. The performance of the model heavily depends on the quality and completeness of historical purchase data. To improve the system's performance, it is recommended to increase the number of data sources, such as user history, product reviews, and feedback, to enrich the feature set. Hybrid models combining collaborative, content-based, and other filtering techniques could be used to improve the quality of recommendations. Explainable AI techniques could also be used to add efficiency and transparency to the recommendation process. For forecasting demand systems, the quality and coverage of input data determine the quality of the forecast. The current XGBoost model is complex and requires hyperparameter tuning and computational resources. Proper feature engineering can build in enhanced seasonality, trends, promotions, and other effects that impact demand better. Detection mechanisms for anomalies and automated hyperparameter tuning techniques are essential for handling large datasets. Real-time forecasting capabilities should be supported by an infrastructure that can be scaled on the cloud.

3. Implementation and testing

3.1. System implementation model

3.1.1. Product Recommendation

System Implementation Model

The product recommendation system consists of multiple components, each serving a specific role. Below is a component diagram outlining the main components and their interactions:

1. Data Preprocessing Component: Loads and cleans the data, manages missing values, and encodes categorical features.
2. Feature Engineering Component: Generates additional features from the raw data such as temporal features, aggregated metrics, and other derived variables.
3. Model Training Component: Trains various machine learning models including XGBoost, RandomForest, Linear Regression, LightGBM, CatBoost and XGBoost.
4. Model Evaluation Component: Evaluates the performance of each trained model using appropriate metrics such as Mean Squared Error (MSE) and R-squared (R^2).
5. Prediction Component: Uses the trained models to recommend products to customers.
6. Visualization Component: Produces visualizations for understanding model predictions and for reporting purposes.

Component Interfaces and Descriptions

7. Data Preprocessing Component
   - Interface: load_data(), clean_data()
   - Description: Loads datasets, manages missing values, and encodes categorical features.
8. Feature Engineering Component
   - Interface: generate_features()
   - Description: Creates new features from the existing data to improve model performance.
9. Model Training Component
   - Interface: train_model(model_type)
   - Description: Trains different machine learning models based on the specified model_type.
10. Model Evaluation Component
    - Interface: evaluate_model()
    - Description: Evaluates models using performance metrics.
11. Prediction Component
    - Interface: predict_recommendations()
    - Description: Uses trained models to recommend products for customers.
12. Visualization Component
    - Interface: plot_results()

– Description: Generates plots and visualizations of the recommendation results.

Key Data Entry Forms

The system utilizes data files for input rather than interactive forms. The key datasets include:
– SimulatedOrders.csv (Orders in database table): Contains historical order data.
– ProductsOnWebsite.csv (ProductsOnWebsite in database table): Contains product details.

3.1.2. Demand Forecasting

System Implementation Model

The demand forecasting tool consists of multiple components, each serving a specific role in the system. Below is a component diagram that outlines the main components and their interactions:

1. Data Preprocessing Component: Responsible for loading and cleaning the data, managing missing values, and encoding categorical features.
2. Feature Engineering Component: Generates additional features from the raw data such as temporal features, aggregated metrics, and other derived variables.
3. Model Training Component: Trains various machine learning models including XGBoost, RandomForest, and LightGBM.
4. Model Evaluation Component: Evaluates the performance of each trained model using appropriate metrics such as Mean Squared Error (MSE) and R-squared (R^2).
5. Prediction Component: Uses the trained models to predict future demand and generates output for analysis.
6. Visualization Component: Produces visualizations for understanding model predictions and for reporting purposes.

Component Interfaces and Descriptions

7. Data Preprocessing Component
   – Interface: load_data(), clean_data()
   – Description: Loads datasets, manages missing values, and encodes categorical features.
8. Feature Engineering Component
   – Interface: generate_features()
   – Description: Creates new features from the existing data to improve model performance.
9. Model Training Component
   – Interface: train_model(model_type)
   – Description: Trains different machine learning models based on the specified model_type.
10. Model Evaluation Component
    – Interface: evaluate_model()
    – Description: Evaluates models using performance metrics.
11. Prediction Component
    – Interface: predict_demand()

– Description: Uses trained models to forecast demand for specified periods.

12. Visualization Component

– Interface: plot_results()
– Description: Generates plots and visualizations of the forecast results.

Key Data Entry Forms

The system utilizes data files for input rather than interactive forms. The key datasets include:
– SimulatedOrders.csv (Orders in database table): Contains historical order data.
– ProductsOnWebsite.csv (ProductsOnWebsite in database table): Contains product details.

3.2. Analysis of an AI model

3.2.1. Product Recommendation

Strategy for Analyzing the AI Model

The Machine learning solution for product recommendation involves comparing multiple models to determine the most effective one. The strategy includes:
1. Data Preparation: Cleaning and preprocessing the data, generating features.
2. Model Training: Training various models with the prepared data.
3. Model Evaluation: Comparing the models using Mean Squared Error (MSE) and other relevant metrics (see Table 1).
4. Visualization: Interpreting the results graphically to understand model performance.

Training of AI Models

The following models are trained:
– Linear Regression
– Random Forest
– LightGBM
– XGBoost
– CatBoost

Comparison of Different AI Models

Table 1: The models are evaluated based on MSE for Product recommendation.

Model	Mean Squared Error (MSE)	R-squared(R2)
Linear Regression	1.9994034490868362	0.0008864053305761566
Random Forest	2.204189713966017	-0.10153169350713709
LightGBM	2.0001580137755313	0.00043202713629875156
XGBoost	1.9992487927808198	0.0008091166917633119
CatBoost	1.999634241563028	0.0006937794204125991
Neural Network	2.0010119336667263	5.285364523266445e-06

Graphical Interpretation of Results

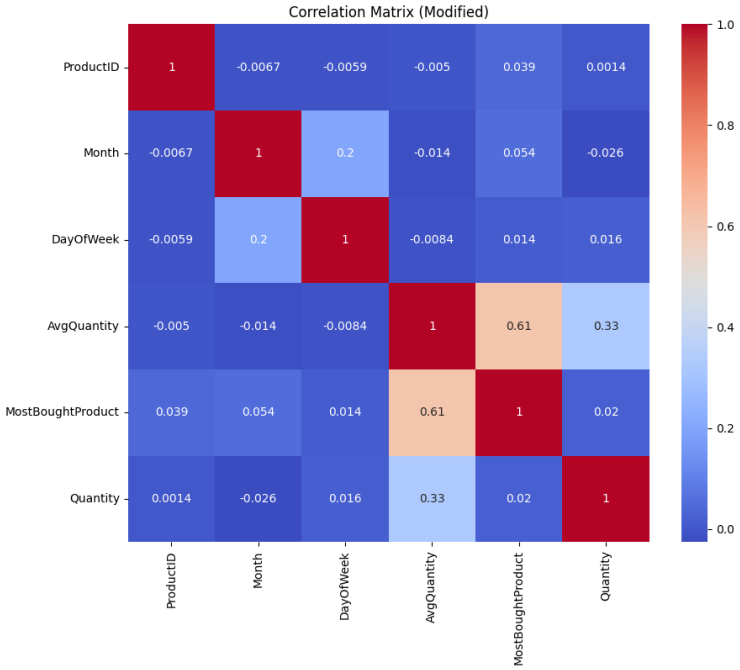

Figure 33: Correlation matrix for product recommendation.

Correlation matrix

The correlation matrix provides a visual representation of the relationships between different features and the target variable (Quantity) (see Figure 33).

Insights:

- ProductID and Quantity have an extremely low positive correlation (0.0014), indicating that the product ID has a minimal direct effect on the quantity ordered.
- Month and Quantity have a negative correlation (-0.026), suggesting that the time of the year might slightly impact the quantity ordered.
- AvgQuantity has a moderate positive correlation (0.33) with Quantity, indicating that average past quantities can be a useful predictor for future quantities.
- MostBoughtProduct has a strong correlation with AvgQuantity (0.61), suggesting that the most frequently bought products are also purchased in larger quantities.

The correlation matrix helps in understanding the relationships between features, guiding the feature selection process for model building.

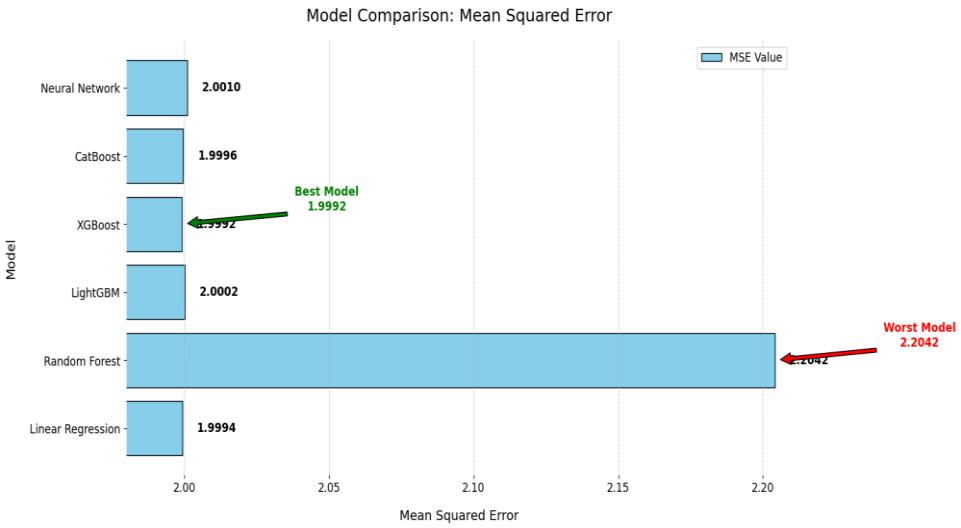

Figure 34: Bar plot for MSE comparison.

Model Comparison: Mean Squared Error

The bar plot compares the Mean Squared Error (MSE) of different models used for product recommendation(see Figure 34). Lower MSE values indicate better model performance.

Insights:

- XGBoost model has the lowest MSE (1.9992), making it the best performing model among those evaluated.
- Random Forest model has the highest MSE (2.2042), indicating it is the worst performing model in this comparison.
- Other models like CatBoost (1.9996), LightGBM (2.0002), Neural Network (2.0010), and Linear Regression (1.9994) also show competitive performance for the XGBoost model.

The XGBoost model is the most accurate model for product recommendation based on the MSE metric. The score is with the fine-tuned MSE values for each model.

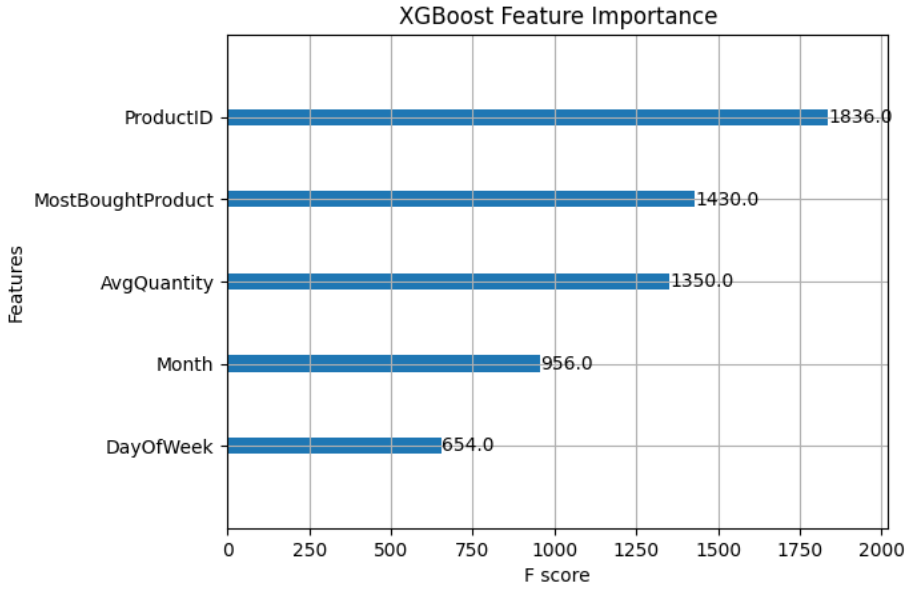

Figure 35: Bar plot for F score values of features.

XGBoost Feature Importance

The feature importance plot displays the significance of each feature used in the XGBoost model. The importance is measured by the F score(see Figure 35), which reflects the number of times a feature is used in a decision tree split.

Insights:

- ProductID is the most important feature, with an F score of 1836.0, highlighting its significance in predicting the quantity ordered.
- MostBoughtProduct and AvgQuantity also show high importance scores (1430.0 and 1350.0 respectively), indicating they are crucial for the model's predictions.
- Month (956.0) and DayOfWeek (654.0) have lower importance scores, suggesting they have a lesser impact on the model's predictions compared to other features.

Understanding feature importance helps in refining the model by focusing on significant features and potentially removing less important ones to enhance model efficiency and interpretability. The high importance of ProductID, MostBoughtProduct, and AvgQuantity suggests that customer purchasing patterns and product popularity play key roles in predicting future quantities.

3.2.2. Demand Forecasting

Strategy for Analyzing the AI Model

The AI solution for demand forecasting involves comparing multiple models to determine the most effective one. The strategy includes:

1. Data Preparation: Cleaning and preprocessing the data, generating features.
2. Model Training: Training various models with the prepared data.
3. Model Evaluation: Comparing the models using Mean Squared Error (MSE) and other relevant metrics(see Table 2).
4. Visualization: Interpreting the results graphically to understand model performance.

Training of AI Models

The following models are trained:

- Random Forest
- LightGBM
- XGBoost

Comparison of Different AI Models

Table 2: The models are evaluated based on MSE for Demand forecasting.

Model	***Mean Squared Error (MSE)***	***R Squared(R^2)***
Random Forest	3.715208200498192	0.08215642824105174
LightGBM	3.8045362208255464	0.060087907498547755
XGBoost	3.761866133614211	0.07062956846066182

Graphical Interpretation of Results

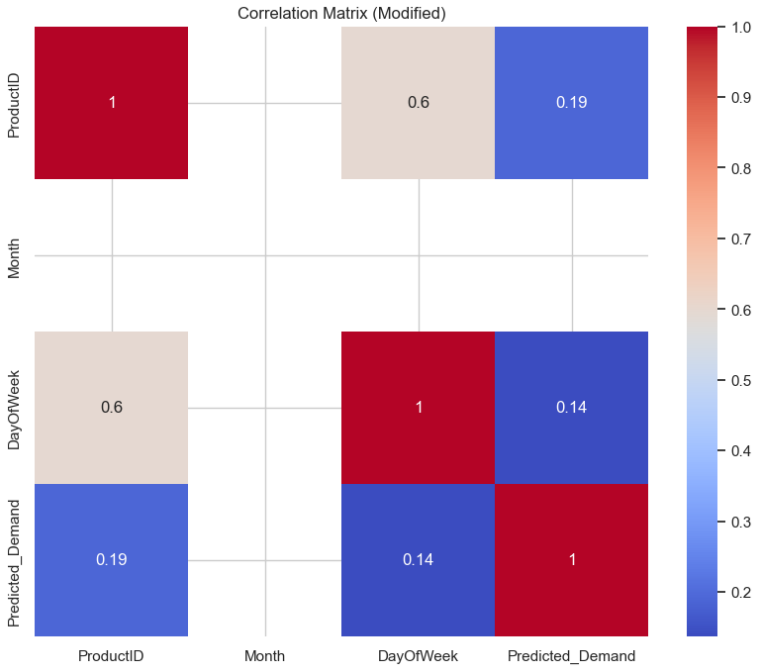

Figure 36: Correlation matrix for demand forecasting.

Correlation Matrix

The correlation matrix provides a visual representation of the relationships between different features and the target variable (Predicted_Demand)(see Figure 36).

Insights:

- ProductID and DayOfWeek have a moderate positive correlation (0.6), indicating that the product ID and the day of the week are related.
- DayOfWeek and Predicted_Demand has a low positive correlation (0.14), suggesting that the day of the week has a slight impact on predicted demand.
- ProductID and Predicted_Demand also shows a low positive correlation (0.19), indicating that the type of product has some influence on the predicted demand.

The correlation matrix helps in understanding the relationships between features, guiding the feature selection process for model building.

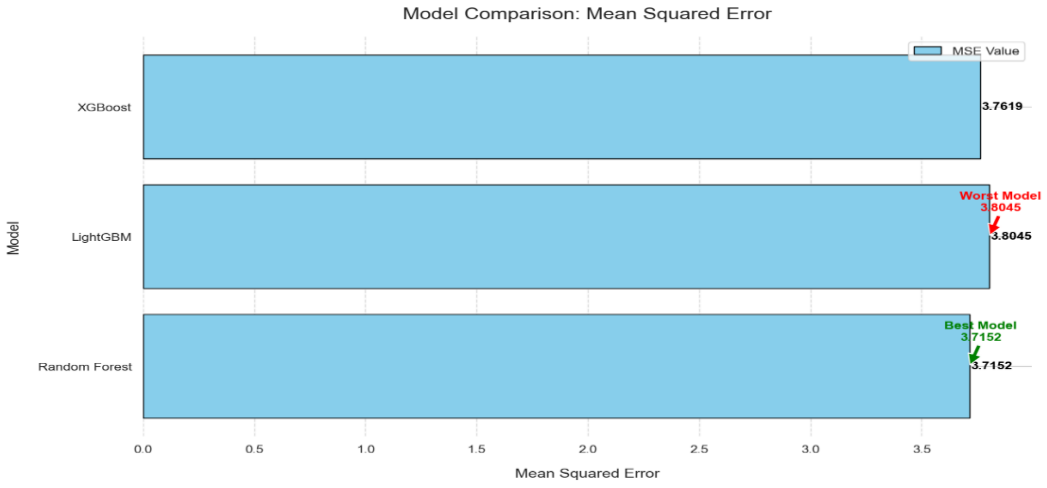

Figure 37: Bar plot for MSE comparison.

Model Comparison: Mean Squared Error

The bar plot compares the Mean Squared Error (MSE) of different models used for demand forecasting(see Figure 37). Lower MSE values indicate better model performance.

Insights:

- Random Forest model has the lowest MSE (3.7152), making it the best performing model among those evaluated.
- LightGBM model has the highest MSE (3.8045), indicating it is the worst performing model in this comparison.
- XGBoost shows a competitive performance with an MSE of 3.7619.

The Random Forest model is identified as the slightly more accurate model for demand forecasting based on the MSE metric.

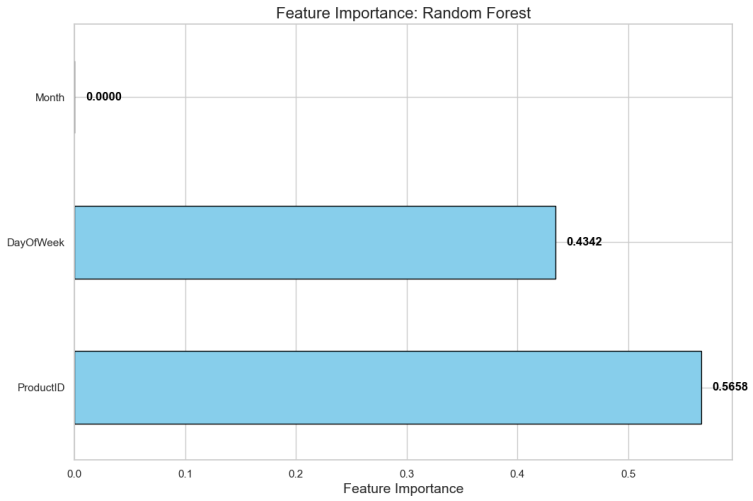

Figure 38: Bar plot for F score value of features.

Feature Importance: Random Forest

The feature importance plot displays the significance of each feature used in the Random Forest model (see Figure 38). The importance is measured by the F score, which reflects the number of times a feature is used in a decision tree split.

Insights:

- ProductID is the most important feature, with an important score of 0.5658, highlighting its significance in predicting demand.
- DayOfWeek also shows a significant importance score (0.4342), indicating it is crucial for the model's predictions.
- Month has a negligible importance score (0.0000), suggesting it has little to no impact on the model's predictions.

Understanding feature importance helps in refining the model by focusing on significant features and potentially removing less important ones to enhance model efficiency and interpretability. The high importance of ProductID and DayOfWeek suggests that these factors play key roles in predicting future demand.

3.3. System and model testing

Explanation of Results:

Unit Testing: The data preprocessing components were evaluated to ensure they correctly cleaned, transformed, and prepared the data for model training.

Integration Testing: The integration of feature engineering with model training was verified to ensure that features were correctly generated and provided to the models.

Performance Testing: Performance testing was conducted using large datasets to evaluate the models' scalability and amount of time for execution. The models maintained their accuracy and performed at the same amount of time even as the data volume increased.

3.3.1. Product Recommendation

Criteria for Testing

The system is evaluated based on:

- Accuracy of predictions (measured by Mean Squared Error, MSE).
- Robustness to different datasets.
- Scalability with increasing data volume.

Evaluate Plan (see Table 3)

1. Unit Testing: Verify individual components such as data preprocessing, feature engineering. And model training.
2. Integration Testing: Ensure components like feature engineering, model training, and prediction generation work together as expected.
3. Performance Testing: Evaluate the performance and scalability of the model with large datasets.

Evaluate Data Sets

- Training Data: Historical order data from 1 January 2022 up to 31 December 2022.
- Evaluate Data: Data from 1 January 2023 to 31 March 2023 was used for validating the models.

Table 3: Test Results for Product recommendations.

Evaluate Type	Description	Result
Unit Testing	Verify data preprocessing components	Data preprocessing components verified successfully.
Integration Testing	Ensure feature engineering integrates with model training	Feature engineering and model training integrated well.
Performance Testing	Evaluate model scalability	Models performed well with increased data volume, maintaining accuracy and the same amount of time.

3.3.2. Demand Forecasting

Criteria for Testing

The system is evaluated based on:

- Accuracy of predictions (measured by Mean Squared Error, MSE).

- Robustness to different datasets.
- Scalability with increasing data volume.

Evaluate Plan(see Table 4)

1. Unit Testing: Verify individual components such as data preprocessing, feature engineering. And model training.
2. Integration Testing: Ensure components like feature engineering, model training, and prediction generation work together as expected.
3. Performance Testing: Evaluate the performance and scalability of the model with large datasets.

Evaluate Data Sets

- Training Data: Historical order data from 1 January 2022 up to 31 December 2022.
- Evaluate Data: Data from 1 January 2023 to 31 March 2023 was used for validating the models.

Table 4: Test Results for Demand forecasting.

Evaluate Type	**Description**	**Result**
Unit Testing	Verify data preprocessing components	Data preprocessing components verified successfully.
Integration Testing	Ensure feature engineering integrates with model training	Feature engineering and model training integrated well.
Performance Testing	Evaluate model scalability	Models performed well with increased data volume, maintaining accuracy and the same amount of time.

3.3.3. System testing

System testing was conducted to ensure that the developed e-commerce system operates correctly under various conditions and meets the specified requirements. The testing included unit testing, integration testing, performance testing, security testing, usability testing, and compatibility testing(see Table 5). The following table summarizes the results of these tests:

Table 5: System testing results.

Test Type	**Description**	**Result**
Unit Testing	Verified individual system modules, such as data preprocessing, feature engineering, and model training.	All individual components operated correctly, with data preprocessing, feature engineering, and model training verified successfully.
Integration Testing	Ensured that different system modules (example, feature engineering, model training, prediction generation) work together seamlessly.	Integration of components was smooth, with no major issues detected. Feature engineering and model training integrated well, and prediction generation worked as expected.
Performance Testing	Evaluated system performance and scalability under varying loads.	The system-maintained accuracy and performed well with increased data volume. The models execution times remained consistent.
Security Testing	Conducted tests to identify and fix potential security issues.	No significant vulnerabilities were found. Data protection measures

		were effective, ensuring privacy and security.
Usability Testing	Assessed the user interface for ease of use and intuitiveness.	The interface was found to be user-friendly and intuitive.
Compatibility Testing	Verified that the system works across different devices and browsers.	The system was compatible with various browsers.

Summary of Test Results:

The testing process ensured that the system operates as intended under various conditions, providing a reliable e-commerce platform. Future improvements could include further optimization for real-time processing and enhanced hybrid models to address issues such as the cold start problem and data integration from diverse sources.

3.4. Shortcomings, limitations, and opportunities for further development of the system

3.4.1. Product Recommendation

3.4.1.1. Shortcomings and Limitations

1. Cold Start Problem:
   - The model struggles with recommending products for new users or products with little historical data. This is a common issue in recommendation systems, known as the cold start problem.
2. Scalability:
   - While performance testing indicated good scalability, the system may still face challenges with extremely large datasets or extremely high transaction volumes, requiring further optimization.
3. Feature Dependence:
   - The model heavily relies on historical purchase data and might not perform well if this data is sparse or incomplete.
4. Real-Time Processing:
   - The current implementation may not support real-time recommendations efficiently due to the time required for feature engineering and model prediction.

3.4.1.2. Opportunities for Further Development

5. Incorporating More Data Sources:
   - Integrate additional data sources such as user browsing history, product reviews, and social media interactions to enrich the feature set and improve recommendation accuracy.
6. Hybrid Recommendation Systems:
   - Develop hybrid models that combine collaborative filtering, content-based filtering, and other techniques to address the cold start problem and improve overall recommendation quality.

7. Real-Time Recommendation Engine:
   - Optimize the system for real-time processing to provide instant recommendations based on the latest user interactions and data.
8. Model Interpretability:
   - Implement explainable AI techniques to make the recommendation process more transparent and understandable for users.

3.4.2. Demand Forecasting

3.4.2.1. Shortcomings and Limitations

1. Data Quality and Availability:
   - The accuracy of the demand forecasting model is highly dependent on the quality and completeness of the input data. Missing or inaccurate data can significantly affect predictions.
2. Model Complexity:
   - More complex models like XGBoost and LightGBM require extensive hyperparameter tuning and computational resources, which can be a limitation for smaller organizations.
3. Generalization:
   - The model might not generalize well across different product categories or markets without extensive retraining and customization.

3.4.2.2. Opportunities for Further Development

1. Enhanced Feature Engineering:
   - Develop more sophisticated features that capture seasonality, trends, promotions, and other factors affecting demand. This can be achieved through advanced time-series analysis techniques.
2. Anomaly Detection:
   - Integrate anomaly detection mechanisms to identify and manage outliers or unusual patterns in the data, improving forecast robustness.
3. Incorporating External Data:
   - Use external data sources such as economic indicators, weather data, and social media trends to improve the accuracy of demand forecasts.
4. Automated Hyperparameter Tuning:
   - Implement automated hyperparameter tuning techniques (e.g., Bayesian optimization) to simplify model optimization and improve performance without extensive manual effort.
5. Scalable Infrastructure:

– Invest in scalable cloud-based infrastructure to manage large datasets and support real-time forecasting capabilities.

4. Documentation for the user

4.1. User Guide

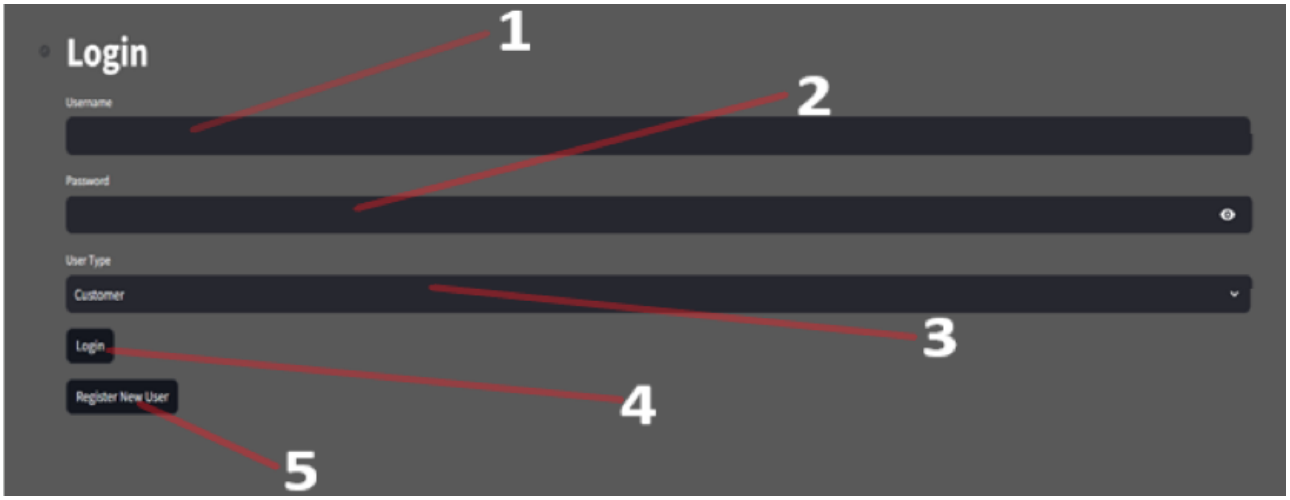

Figure 39: Login page.

Login page (see Figure 39)

1. Users enter their username here.
2. The user enters their password.
3. The user selects their user type.
4. The user clicks the “login” button to login after entering their credentials.
5. The user can choose to create an account by clicking the “Register New User” button.

Figure 40: Registration page.

Registration page (see Figure 40)

1. User enters their desired username that will be unique.
2. User enters their password.
3. User confirms their password.
4. User clicks the “Register” button to be registered and become a customer.
5. User clicks the “Already have an account? Login” button to login if they are already registered.

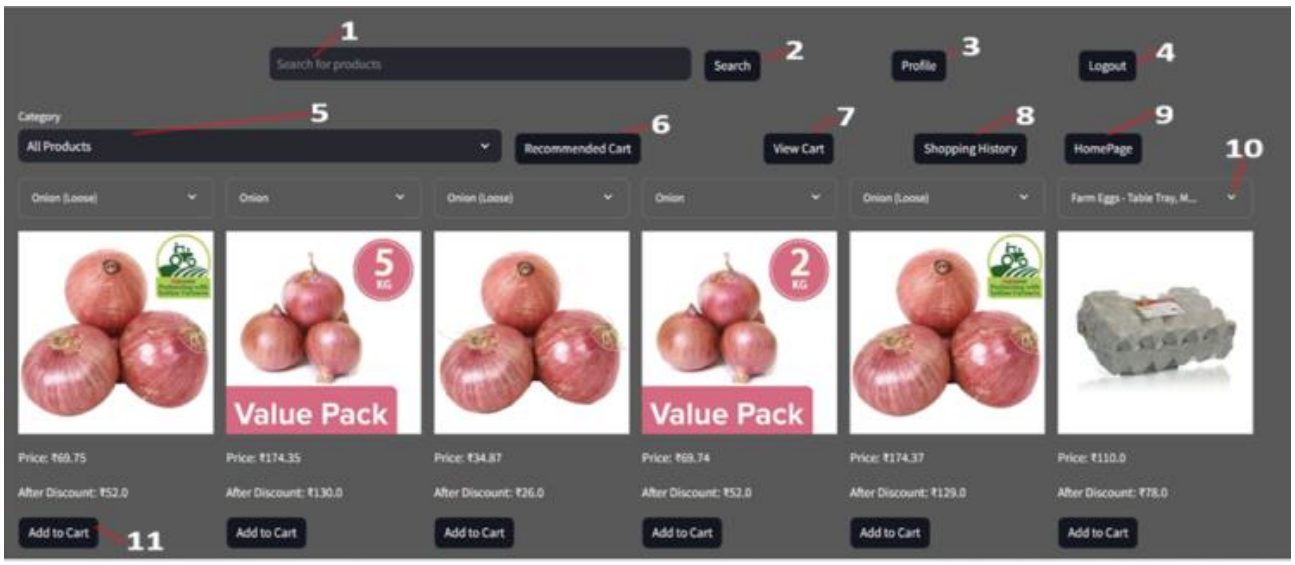

Figure 41: Home page.

Home page (see Figure 41)

1. The user can enter the product they are looking for.
2. The user clicks “Search” button to filter the products based on what they are looking for.
3. The user clicks “Profile” button to view and edit their profile.
4. The user clicks “Logout” button to logout of the website.
5. The user clicks or enters the category they want and click enter to show filter data of the specific query.
6. The user clicks “Recommended cart” button to see recommended items page.
7. The user clicks “View cart” button to see cart page.
8. The user clicks “Shopping history” button to view their past orders.
9. The user clicks “Homepage” button to go to the homepage.
10. The user clicks “dropdown” icon to view full product name.
11. The user clicks “Add to cart” button to add the product to their shopping cart.

Figure 42: Recommended cart page part 1.

Figure 43: Recommended cart page part 2.

Recommended cart pages (see Figure 42 and Figure 43)

1. The user clicks the `-` icon to decrease the quantity of that specific product.
2. The user clicks the `+` icon to increase the quantity of that specific product.
3. The user clicks the “Remove” button to remove the product from their shopping cart.
4. The user clicks the “Add to Main Cart” button to move the product to the main cart.

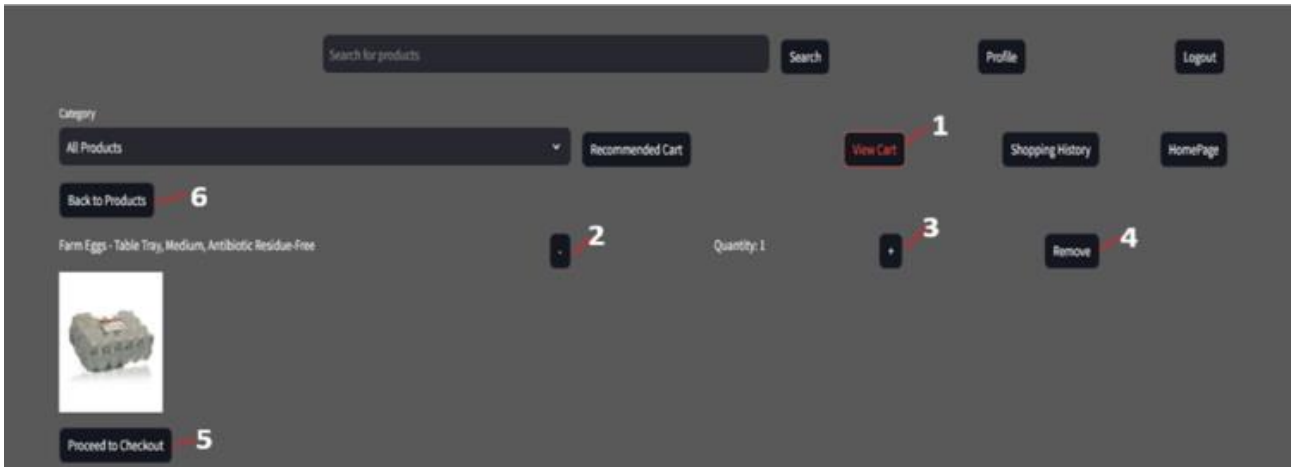

Figure 44: Cart page.

Cart page (see Figure 44)

1. The user clicks the "View Cart" button to view their shopping cart.
2. The user clicks the `-` icon to decrease the quantity of that specific product.
3. The user clicks the `+` icon to increase the quantity of that specific product.
4. The user clicks the “Remove” button to remove the product from their shopping cart.
5. The user clicks the “Proceed to checkout” button to proceed to the checkout page.
6. The user clicks the “Back to products” button to go back to the homepage.

Figure 45: Checkout page.

Checkout page (see Figure 45)

1. The user clicks the “Confirm Order” button to finish their shopping experience.
2. The user clicks the “Back to Cart” button to go back to the Cart page.

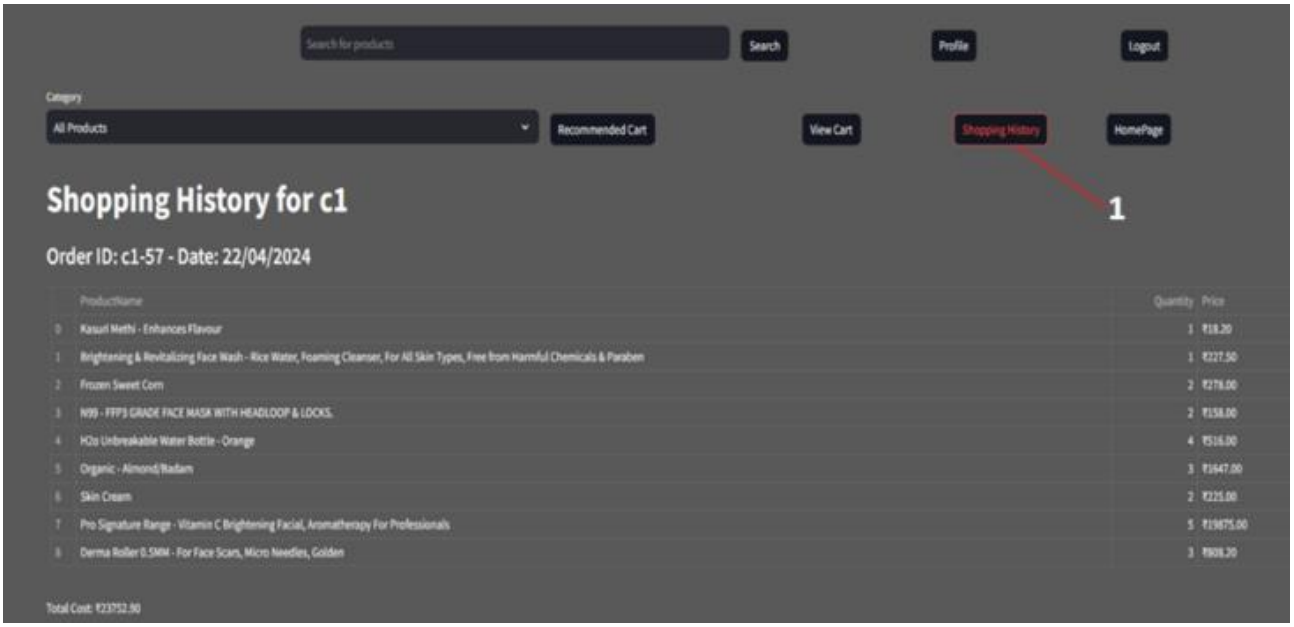

Figure 46: Shopping history page.

Shopping history page (see Figure 46)

1. The user clicks the "Shopping History" button to view their shopping history.

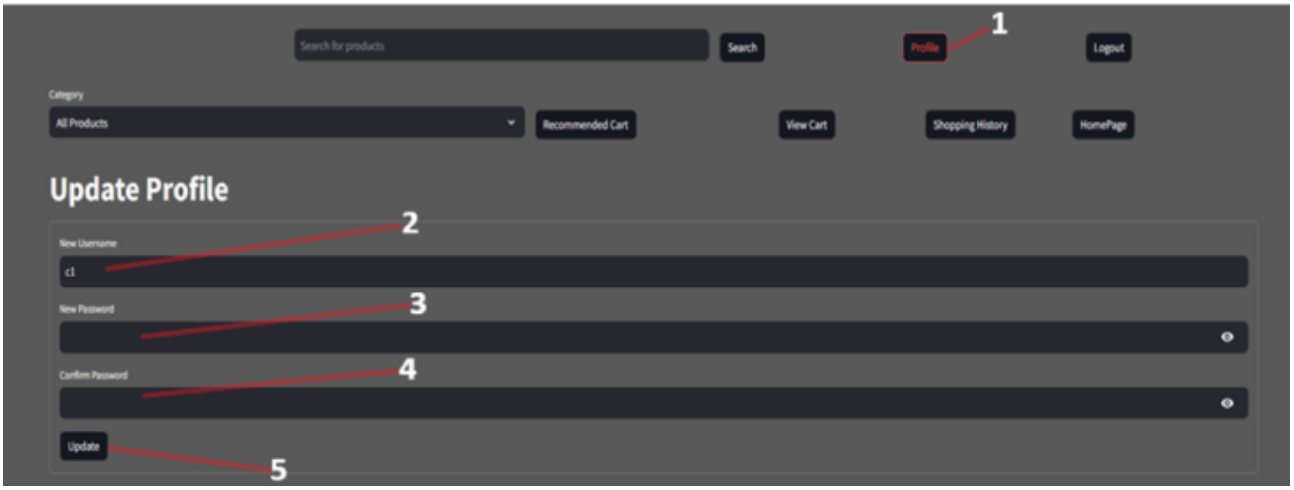

Figure 47: Update profile page.

Update profile page (see Figure 47)

1. The user clicks the “Profile” button to view their Profile page for editing.
2. User enters their new desired username that will be unique.
3. User enters their new password.
4. User confirms their new password.
5. User clicks the “Update” button to change their details.

4.2. System Administrator's Guide

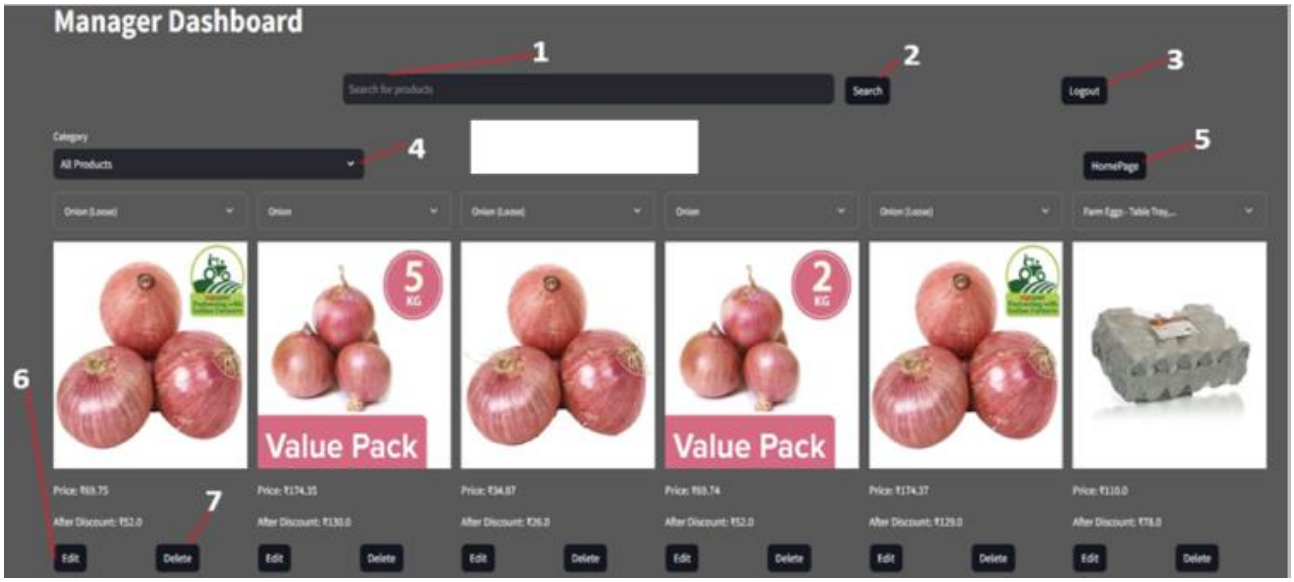

Figure 48: Admins home page.

Admins home page (see Figure 48)

1. Admin can enter the product they are looking for.
2. The admin clicks “Search” button to filter the products based on what they are looking for.
3. The admin clicks “Logout” button to log out of the website.
4. The admin clicks or enters the category they want and click enter to show filter data of the specific query.
5. The admin clicks “Homepage” button to go to the homepage.
6. The admin clicks “Edit” button to open the form to edit the product on the website.
7. The admin clicks “Delete” button to delete the product from the website.

Figure 49: Admins sidebar.

Admins sidebar (see Figure 49)

1. The admin clicks the “Add new Product” button on the sidebar to open the form to enter new product details.

2. The admin clicks the “Admin Registration” button on the sidebar to open the page to add a new admin to the system.
3. The admin clicks the “Manage Customers” button on the sidebar to open the page to edit customer details
4. The admin clicks the “Demand Forecasting Panel” button on the sidebar to open the page to view and analyze product demands.

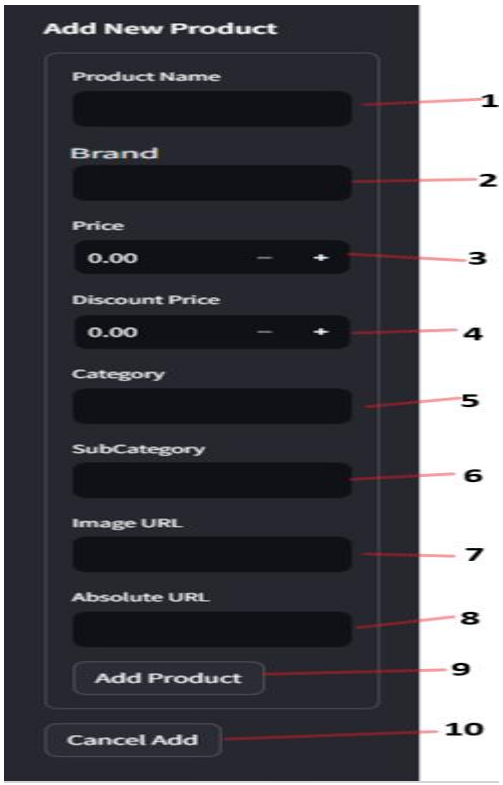

Figure 50: Add new product form.

Add new product form (see Figure 50)

1. The admin enters the name of the new product.
2. The admin enters the brand of the new product.
3. The admin enters the price of the new product.
4. The admin enters the discount price of the new product.
5. The admin enters the category of the new product.
6. The admin enters the subcategory of the new product.
7. The admin enters the image URL of the new product.
8. The admin enters the Absolute URL of the new product.
9. The admin clicks the “Add Product” button to add the new product to the website.
10. The admin clicks the “Cancel Add” button to cancel the process to add new product to the website.

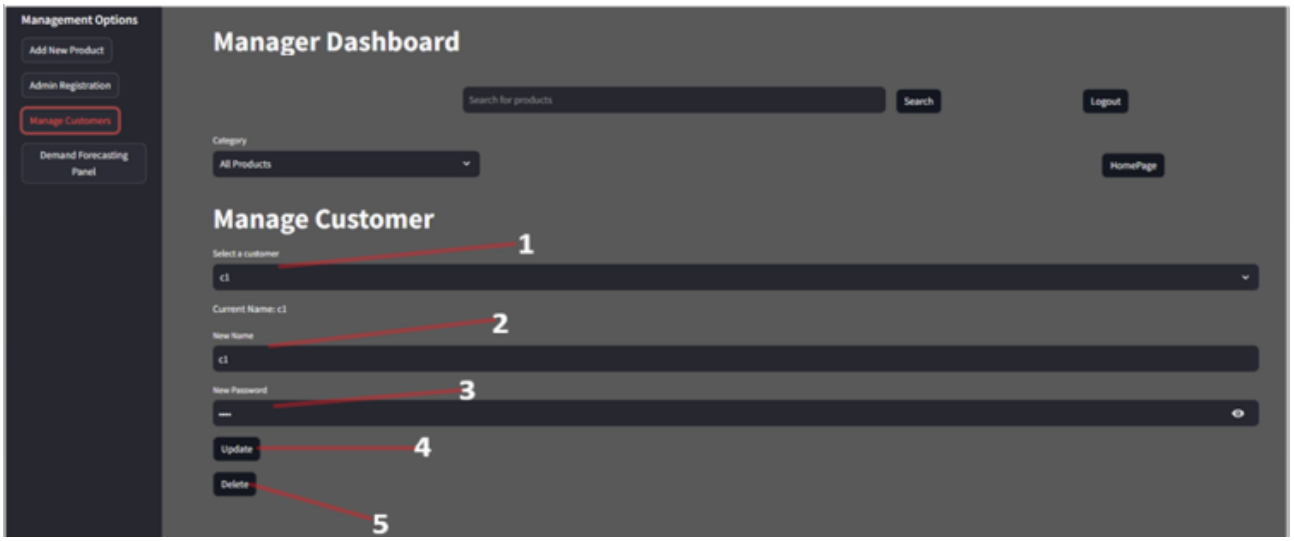

Figure 51: Manage customer page.

Manage customer page (see Figure 51)

1. The admin selects the user to edit details.
2. The admin changes the name or keeps the existing name.
3. The admin changes the password or keeps the existing one.
4. The admin clicks the “Update” button to save the changes.
5. The admin clicks the “Delete” button to delete the user from the system.

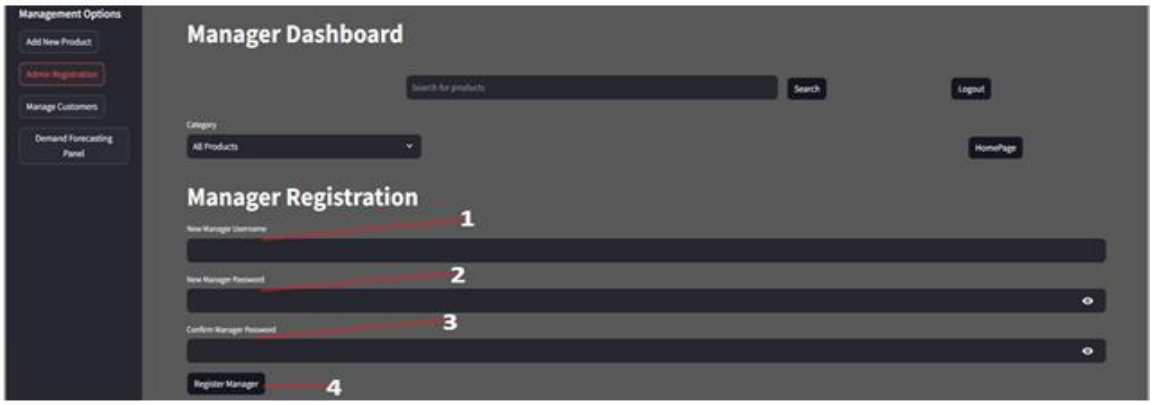

Figure 52: Admin registration page.

Admin registration page (see Figure 52)

1. The admin enters the name of the new admin.
2. The admin enters the password of the new admin.
3. The admin confirms the password of the new admin.
4. The admin clicks the “Register Manager” button to register the new admin.

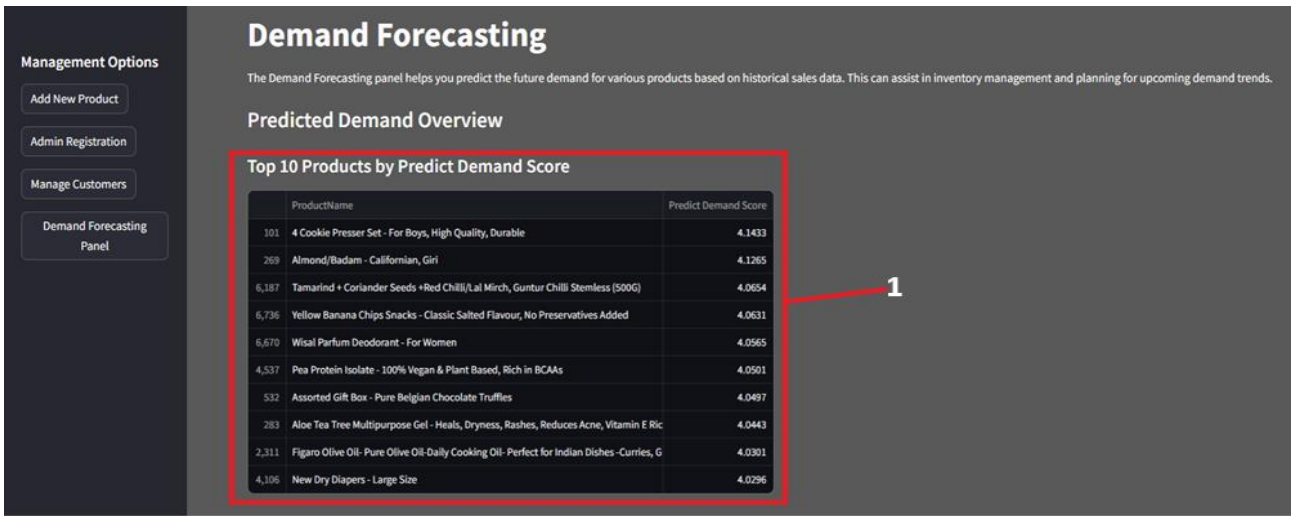

Figure 53: Demand forecasting page part 1.

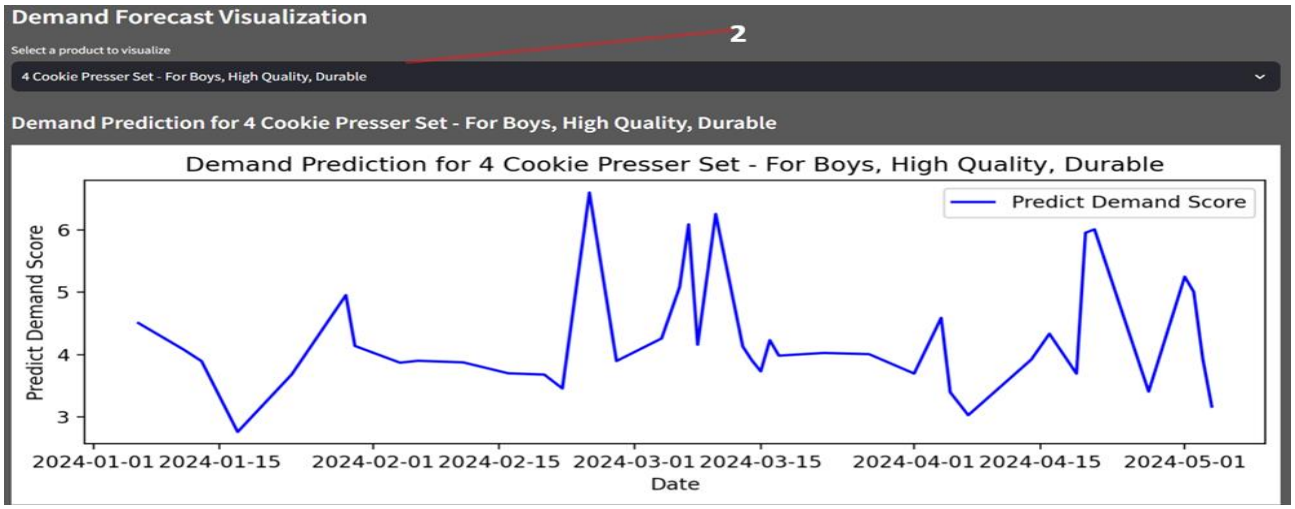

Figure 54: Demand forecasting page part 2.

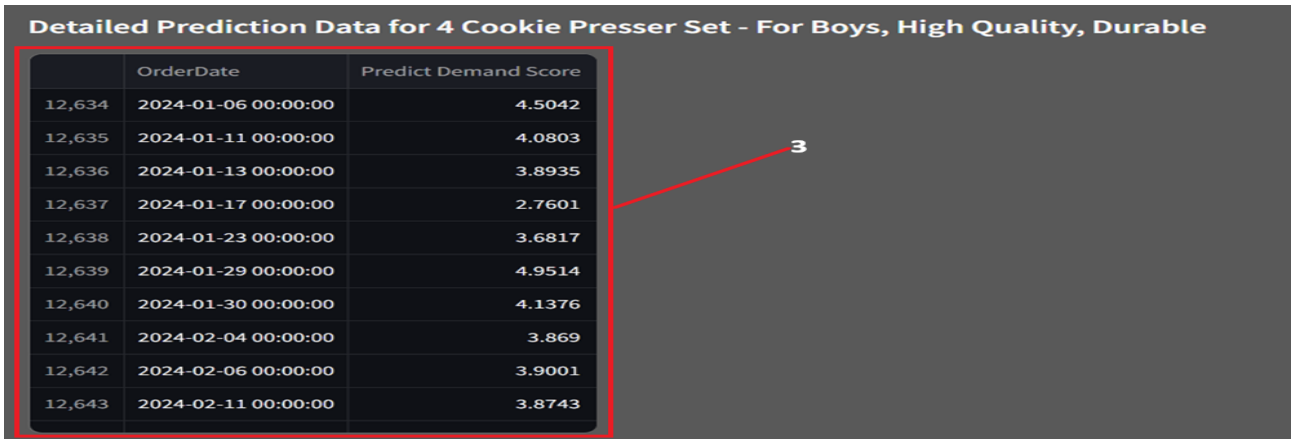

Figure 55: Demand forecasting page part 3.

Demand forecasting page (see Figure 53, Figure 54 and Figure 55)

1. The admin can see the top ten products using demand forecasting.
2. The admin can select a product from the top ten to see its forecast.
3. The admin can see the forecast in more days from the table.

Conclusions

1. In summary, the evaluation shows the improved efficiency in both the operation and customer's satisfaction by the machine learning algorithms used in the Recommendation Engine of Yusp, Beeketing for Shopify, and WooCommerce. The systems, however, lack sophisticated recommendation logic that supports scalability. Yusp uses simple collaborative filtering, WooCommerce relies on imported historical sales datasets, and Beeketing only has fundamental upselling and cross-selling features without the necessity of an algorithm. The gaps highlighted called for additional functionalities, in this case, scalable demand forecasting algorithms, hybrid recommendation models, and real-time data processing. More development in every part of the IoT is necessary because the user base grows at a high rate, and more assured user experience would require further data collection and refinement.
2. The datasets, including purchase histories, and product details, were suitable for developing machine learning models. However, the completeness and quality of historical purchase data are critical. Incorporating external data sources such as location etc., could further enhance the accuracy of demand forecasting models.
3. The integrated e-commerce system successfully utilized machine learning models like XGBoost and Random Forest for personalized recommendations and demand forecasting. These models effectively leveraged historical purchase data to tailor suggestions to individual customer preferences. However, addressing challenges like the cold start problem and integrating various data sources through hybrid recommendation models is necessary for further improvement.
4. The system's architecture supported the seamless integration of recommendation and forecasting models, providing a consistent user experience that balanced personalization with efficient execution time. While scalability and performance testing indicated the system's capability to handle increasing data volumes, further optimizations are needed to enhance real-time processing capabilities.
5. Test results showed improved user satisfaction and operational efficiency. The machine learning techniques improved the accuracy of product recommendations and demand forecasting. To maintain and enhance performance and reliability, it is useful for us to continue fine-tuning the models and combining data sources or use more.

List of references

1. Mydyti, H., Kadriu, A., & Pejic Bach, M. (2023, May 1). Using Data Mining to Improve Decision-Making: Case Study of A Recommendation System Development. Organizacija, 56(2), 138–154. https://doi.org/10.2478/orga-2023-0010
2. Widayanti, R. (2023, September 1). Improving Recommender Systems using Hybrid Techniques of Collaborative Filtering and Content-Based Filtering. Journal of Applied Data Sciences, 4(3), 289–302. https://doi.org/10.47738/jads.v4i3.115
3. Ibrahim, A., Ahmed, A., & Abdullah, A. (2020, May 7). Collaborative Filtering: Techniques and Applications. ResearchGate. https://www.researchgate.net/publication/341216858_Collaborative_Filtering_Techniques_and_Applications
4. Praditya, N. W. P. Y. (2021, June 1). Literature Review Recommendation System Using Hybrid Method (Collaborative Filtering & Content-Based Filtering) by Utilizing Social Media as Marketing. Computer Engineering and Applications Journal, 10(2), 105–113. https://doi.org/10.18495/comengapp.v10i2.368
5. John K. Tarus, Zhendong Niu, & Ghulam Mustafa. (2018). Knowledge-based recommendation: A review of ontology-based recommender systems for e-learning. Retrieved from https://www.researchgate.net/publication/312395451_Knowledge-based_recommendation_a_review_of_ontology-based_recommender_systems_for_e-learning
6. Erion Çano & Maurizio Morisio (2019). An Exploration of Machine Learning Techniques in Recommender Systems. arXiv:1901.03888. Retrieved from https://arxiv.org/abs/1901.03888
7. Victor Giovanni Morales Murillo, David Pinto, Franco Rojas Lopez, & Juan Manuel Gonzalez Calleros. (2022). A Systematic Literature Review on the Hybrid Approaches for Recommender Systems. Retrieved from https://www.researchgate.net/publication/359758635_A_Systematic_Literature_Review_on_the_Hybrid_Approaches_for_Recommender_Systems
8. Chen, X., & Liu, Y. (2021). Challenges and Innovations in Big Data with Recommender Systems. Journal of Big Data, 8(1). https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-
9. Ali, M., & Khan, R. (2020). Matrix Factorization in Recommender Systems: Algorithms, Applications, and Peculiar Challenges. Retrieved from https://www.researchgate.net/publication/335158585_Matrix_Factorization_in_Recommender_Systems_Algorithms_Applications_and_Peculiar_Challenges
10. Wang, T., & Zhang, Q. (2021). Advanced Techniques in Content-Based Recommender System. arXiv:2109.04584. Retrieved from https://arxiv.org/abs/2109.04584
11. Lee, J., & Kim, S. (2021). Deep Learning Recommender Systems: A Survey, Applications and New Dimensions. Retrieved from https://www.researchgate.net/publication/375516337_Deep_learning_recommender_systems_a_survey_applications_and_new_dimensions
12. Sun, Y., & Zhao, L. (2023). New Perspectives on Recommender Systems in Web 3.0. Retrieved from https://ar5iv.org/abs/2310.14379