

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
ADVANCED COLLEGE OF ENGINEERING AND MANAGEMENT
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
KALANKI, KATHMANDU



A Minor Project Final Defense Report on
Mobile Application for Commercial Farming

Project Code: CT654

Submitted By:

Ankit Khanal 30115

Ayush Aryal 30121

Ayush Marhatta 30122

Supervisor:

Er. Bikash Acharya

Er. Sameep Dhakal

A Minor Project Final report submitted to the Department of Electronics and
Computer Engineering in the partial fulfillment of the requirements for Degree of
Bachelor of Engineering in Computer Engineering Kathmandu, Nepal

29 April, 2023

ADVANCED COLLEGE OF ENGINEERING AND
MANAGEMENT

DEPARTMENT OF COMPUTER AND ELECTRONICS ENGINEERING

APPROVAL LETTER

The undersigned certify that they have read and recommended to the Institute of Engineering for acceptance, a project report entitled “Mobile Application for Commercial Farming”

submitted by:

Ankit Khanal 30115

Ayush Aryal 30121

Ayush Marhatta 30122

In partial fulfillment for the degree of Bachelor in Computer Engineering.

.....
Project Supervisor

.....
Project Supervisor

.....
External Examiner

.....
Er. Laxmi Prasad Bhatt
Academic Project coordinator
Department of Computer and Electronics Engineering

Date: 29 April, 2023

ACKNOWLEDGEMENT

We take this opportunity to express our deepest and sincere gratitude to our Project Supervisor **Er. Bikash Acharya** and **Er. Sameep Dhakal** for their insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project and also for his/her constant encouragement and advice throughout our Bachelor's programme.

We express our deep gratitude to **Er. Ajaya Shrestha**, Head of Department of Electronics and Computer Engineering, **Er. Bikash Acharya**, Deputy Head, Department of Electronics and Computer Engineering, **Er. Laxmi Prasad Bhatt**, Academic Project Coordinator, Department of Electronics and Computer Engineering for their regular support, co-operation, and coordination.

The in-time facilities provided by the department throughout the Bachelors program are also equally acknowledgeable.

We would like to convey our thanks to the teaching and non-teaching staff of the Department of Electronics & Communication and Computer Engineering, ACEM for their invaluable help and support till the date. We are also grateful to all our classmates for their help, encouragement and invaluable suggestions.

Finally, yet more importantly, I would like to express our deep appreciation to my grandparents, parents, siblings for their perpetual support and encouragement throughout the Bachelor's degree period.

Ankit Khanal 30115

Ayush Aryal 30121

Ayush Marhatta 30122

ABSTRACT

This project aims to develop an application(app) for farmers to sell their agricultural products with minimal involvement of the middleman(brokers). The app has been developed with the focus on collaborative filtering algorithm. Agile Model Methodology was used for various stages of our application development. Items are recommended to the end users based on the ratings they provide. Cosine similarity determines the similarities among every user which enables a recommendation system. A user is recommended items based on how they rated other items. The estimated rating of an item is calculated by aggregating the ratings of other users for that item. This application is based on Business-to Consumer(B2C) approach.

Keywords: *Mobile Application, Collaborative filtering, Flutter, Agile Model, B2C*

Table of Contents

ACKNOWLEDGEMENT.....	iii
ABSTRACT.....	iv
List of Figures.....	vii
List of Abbreviations/Acronyms.....	ix
CHAPTER 1	
INTRODUCTION.....	1
1.1 Background.....	1
1.2 Motivation.....	2
1.3 Statement of the Problem.....	2
1.4 Project objective.....	3
1.5 Significance of the study.....	3
CHAPTER 2	
LITERATURE REVIEW	4
2.1 Different Mobile Applications for Farmers	4
2.1.1 Different Nepali Agricultural Applications	4
2.1.2 Some of the agricultural applications based on India	5
2.2 Collaborative Filtering	7
CHAPTER 3	
FEASIBILITY STUDY	8
3.1 Technical Feasibility	8
3.2 Operational Feasibility	8
3.3 Economical Feasibility.....	8
3.4 Schedule Feasibility	8
CHAPTER 4	
REQUIREMENT ANALYSIS.....	9
4.1 Functional requirements:.....	9
4.2 Non-Functional requirements:	9
4.3 Hardware Requirements.....	9
4.4 Software Requirements	10
CHAPTER 5	
SYSTEM DESIGN AND ARCHITECTURE.....	11

5.1 System Block Diagram	11
5.2 Use Case Diagram.....	12
5.3 Data Flow Diagram (DFD)	13
5.4 Entity-Relation (ER) Diagram	14
CHAPTER 6	
METHODOLOGY	15
6.1 Development Process.....	15
6.2 Data Collection	17
6.3 Data Analysis and Recommendations system building	20
6.3.1 Cosine Similarity	20
CHAPTER 7	
RESULT AND ANALYSIS.....	25
7.1 Results.....	25
7.2 Analysis.....	37
CHAPTER 8	
CONCLUSION, LIMITATION AND FUTURE ENHANCEMENT	41
8.1 Conclusion	41
8.2 Limitations	41
8.2.1 Limitations to the app	41
8.2.2 Limitations in recommendation system	41
8.3 Future Enhancement	42
REFERENCES.....	43

List of Figures

Title	Page
Fig 5.1: System Block Diagram	11
Fig 5.2: Use Case Diagram.....	12
Fig 5.3(a): DFD Level 0.....	13
Fig 5.3(b): DFD Level 1.....	13
Fig 5.4: Generated E-R Diagram.....	14
Fig 6.1: Agile Methodology.....	17
Fig 6.2(a): Table Creation in Django.....	17
Fig 6.2(b): Database Schema.....	18
Fig 6.2(c): Database Schema Expanded.....	18
Fig 6.2(d): Populating Database.....	19
Fig 6.3.1: Collaborative filtering using Cosine Similarity of a test system.....	21
Fig 6.3: Item recommendation response for a particular user.....	22
Fig 6.4: Server Response Log.....	23
Fig 6.5: Testing for a Server Interaction.....	24
Fig 7.1(a): Admin Dashboard Login Page.....	25
Fig 7.1(b): Admin Dashboard Landing Page.....	26
Fig 7.1(c): System User Page.....	27
Fig 7.1(d): User Change Page.....	27
Fig 7.1(e): Item Page.....	28
Fig 7.1(f): Rating Page.....	28
Fig 7.1(g): Orders Page.....	29
Fig 7.1(h): Cart Page.....	29
Fig 7.1(i): Delete Page.....	30
Fig 7.1(j): API Login Page.....	30
Fig 7.1(k): API Landing Page.....	31
Fig 7.1(l): API Base Landing Page.....	31
Fig 7.1(m): JSON Response.....	31

Fig 7.1(n): App Login Page.....	32
Fig 7.1(o): Home Page (Recommended Product).....	33
Fig 7.1(p): Home Page (Browse Products).....	34
Fig 7.1(q): Product Page.....	35
Fig 7.1(r): Cart Page.....	36

List of Abbreviations/Acronyms

GDP: Gross Domestic Product

App: Application

B2C: Business-to-Consumer

API: Application Program Interface

CF: Collaborative Filtering

DFD: Data Flow Diagram

ER: Entity Relationship

CHAPTER 1

INTRODUCTION

1.1 Background

Nepal is an Agricultural Country. The agriculture sector engages around 66% of the total population in Nepal [1]. It contributes one-third (36%) of the nation's Gross Domestic Product (GDP) significantly to the national economy [1]. And, the farmers engaged in agriculture who make all of us have our food on time so that we are living are not getting the correct value of what they have been doing to date. Middle Man (Broker) plays a vital role in the country's agricultural demand and supply chain. They have set up a syndicate through which they are filling their stomach overnight with farmers' hard work and profit, due to which farmers couldn't even collect the amount they spent producing agricultural products. Brokers are meant to ensure that the transaction can run smoothly and that each party has the necessary information, but here in Nepal, brokers are different from how they should operate [2]. For example: "In a personal interview with Anjil Upreti, one of the students who went to his hometown Dhading on holiday bought 1kg of Tomato for Rs.50, but after returning from his hometown when he bought the same Tomato of 1kg in Kathmandu the price was ghastly which was Rs. 200". The reason for these drastic changes in the price of the same agricultural product is the middleman's syndicate. This is the cause of the massive suicide rate of farmers increasing daily. There is a need for such a solution so that this problem can be solved and farmers can get the entire profit from their hard work so that middleman (broker) syndicate can be eradicated. Something needs to catch up, so this is where our project idea emerged. An application(app) can be developed to bridge the gap between farmers and market or consumers. This can ensure farmers get the right amount of their product for which they have been deprived for so long. Developing a Mobile Application that connects farmers to all the markets and farmers engaging in trading agricultural products can be a simple yet reliable way to bring back the farmer's correct product value. This project aims to develop an application(app) for farmers and consumers to get optimum benefits from their agricultural products by eradicating the alliance of the middlemen(brokers).

1.2 Motivation

When we were going through a newspaper one day, we came up with the shocking news that, recently, Sugarcane farmer Narayan Raya Yadav of Ramnagar, Sarlahi, died of a heart attack while protesting in Kathmandu to get his dues back, which had been pending for a long time [3]. The primary motivation for our project is to address these types of incidents due to which farmers have been suffering for a long time and to break the middleman's syndicate through which all of the farmer's profit is solely consumed by them. And to break the middleman's alliance, we need to develop an application through which farmers can directly sell their products in the market without the involvement of brokers. This application will enable the farmers to operate a clean Business-to-Consumer (B2C) approach beaking traditional Business-to-Middleman-to-Consumer. This project will be a noble cause not only from the farmer's point of view but also from other angles, such as this will address Sustainable Development Goals (SDG): 1- No Poverty, 2- Zero Hunger, 3-Good Health and Well-being and 4-Life on Land.[4]

1.3 Statement of the Problem

Farmers still have to face the unbearable compulsion to depend on the middleman's syndicate. This conventional structure of our society has to be changed because the actual deserving person of the profit are the farmers which has been ingested by the brokers. This causes unwanted human loss (the suicide of farmers) and poverty. There is a platform lagging that can completely transpose the miserable life of those farmers who are dreaming to live a quality life continuing the farming without any burden of the middleman's syndicate and to connect farmers more effectively to market.

1.4 Project objective

- To Develop a Model Mobile Application for Farmers to Sell their Product.

1.5 Significance of the study

This project will enable farmers to sell their agricultural product independently i.e, without the involvement of middleman. Various other aspects such as life standard upliftment of farmers, no unwanted human loss, education for farmers and sustainable development goals (zero hunger, no poverty, good health and well-being and life on land) can be achieved. Farmers will get the right value of money of their products which has been lagging.

CHAPTER 2

LITERATURE REVIEW

2.1 Different Mobile Applications for Farmers

The Internet of Things (IoT) has several uses in the field of digital agriculture, including crop growth monitoring, fertilizer selection, irrigation decision support systems, etc. In this study, an IoT device is utilized to collect data about agriculture, which is then saved in a cloud database. Big data analysis powered by the cloud is utilized to assess data such as fertilizer needs, crop analysis, market needs, and stock needs for the crop. Then, a forecast is made using data mining techniques, and the farmer receives the information via a mobile app. Our ultimate goal is to use this projected knowledge to boost crop productivity and control the cost of agriculture for the goods.[5]

According to a study, disruptive technologies have recently attracted a lot of attention from a variety of industries, including agricultural. The main focus of the contribution is the conceptualization and understanding of the AI-driven AgriTech environment pertinent to agricultural operations for smart, efficient, and sustainable farming. For further research into the operational environment of AgriTech, the study offers a single normative reference for the definition, context, and future directions of the area.[6]

2.1.1 Different Nepali Agricultural Applications

Kheti

Kheti, an Agri-Tech platform has been connecting the Farmers and Households providing transparency to buy and sell Fresh vegetables & fruits. It's a B2C app that is based on monthly subscription. Generally bulk orders are taken from farmers and also from the consumers and the bidding process is way more transparent and efficient.[7]

Smart Krishi

The app offers details on livestock, farming, and agricultural insurance, as well as its application process. One of the greatest agricultural applications in Nepal, Smart Krishi, contains a wide range of information from the production of cereal crops to the cultivation of vegetables, cash crops, fruit crops, and non-timber forest products. Additionally, there is detailed information on commercial goat farming, boar goat farming, buffalo and cow rearing, and fish production systems. This application's platform for connecting farmers and specialists is one of its most intriguing aspects.

Through "Krishak Sanjal," seasoned farmers may communicate with one another and exchange concerns, wants, and experiences. Farmers may access this online network to learn whatever they need to know about training programs that are going to be held by some organization, agriculture credit information, marketing and so on.[7]

Hamro Krishi

The app has a section titled "FAQs" that provides answers to some frequently asked questions regarding raising livestock (including pigs and poultry, cows and buffalo, and ostrich fish), growing cereal crops (including rice, wheat, maize, and finger millet), and growing vegetables, fruits, and spices. For 26 districts in three geographic regions, Hamro Krishi provides a weekly weather-based agriculture advisory that may also be applicable to neighboring districts with a similar terrain and geographic region. The newsletter includes potential agricultural operations according to the crop and the weather. Success stories, a market price report, and details on farm insurance are other highlights.[7]

Geo Krishi

Cooperatives, agribusiness, and farmer-based enterprises will have access to a powerful, adaptable, and personalized digital agricultural platform and its web-based system. Farmers will be able to give timely, contextualized information and advice services through mobile applications. Based on the particular requirements and needs of each registered farmer, this system will offer real-time advises and propose ideal agricultural techniques. A number of ICT-based advising services are included in the packaged solutions. Geokrishi Farm, Geokrishi Ext, Geokrishi Enterprise, Samuhik Bazar are the sister versions of this app.[8]

2.1.2 Some of the agricultural applications based on India

Kisan Suvidha

An all-encompassing smartphone app called Kisan Suvidha was created to assist farmers by giving them instant access to pertinent information. They may get information about dealers, market pricing, agro warnings, plant protection, IPM practices, and more with the press of a button. To best enable farmers, special features

have been incorporated, such as extreme weather alerts, market prices of commodities in the closest area, and the highest price in the state as well as India.[9]

Agri-Market

The Agri-Market mobile app may be used to find out the market price for crops at marketplaces located 50 kilometers away from the device. With the use of mobile GPS, this app locates users automatically and retrieves the current crop price in any marketplaces within a 50-kilometer radius. If a person does not want to utilize GPS location, there is another way to obtain the price of any market and any crop.[9]

Kheti Badi

Indian farmers and agricultural dealers can use this app. Now, they can access the costs online. The majority of agricultural products from all Indian agriculture markets, market yards, and mandis are covered by the app. App currently displays prices from the previous day's close. With the aid of this app, farmers may switch from chemical farming to organic farming, make better financial decisions, and interact with customers to increase their share of the market.[9]

Kisaan Market

With the help of Kisaan Market, Indian farmers can communicate directly with consumers and cut costs on brokerage and shipping. It finally increases their advantages and insights and provides them with the most recent information on Mandi pricing, Krushi Tips, local weather, and news in their preferred language. Features include the most recent Mandi prices (market yard rates), a weather prediction specific to the user's region, Krushi tips, government programs and subsidies for farmers, sale items, buy items, and more.[9]

Mandi Prices

The foundation of every Product's marketing is its information systems. On their smartphones, Indian farmers may now access wholesale pricing for farm products at multiple Mandis. On their smartphones, farmers may submit information about the products they have produced. Every day, the price information is updated many times. The Data is loaded from government servers.[9]

Bhuvan Hailstrom App

A smartphone app has been created to record agricultural losses brought on by hailstorms. The mobile app will be put into the tablet or phone that the agriculture officer will take into the field. The following information may be recorded by this smartphone app: a picture of the field with the latitude and longitude, the name of the crop, the date of sowing, the date of expected harvesting, and the source of irrigation. This data will be automatically displayed on the Bhuvan Portal, making analysis simple.[9]

2.2 Collaborative Filtering

Huge amounts of user data are generated and gathered daily as a result of the mobile Internet's quick development and adoption. The key element of a recommender system is now how to fully utilize these pervasive data. It has been extensively explored and applied to predict mobile users' interests and generate appropriate recommendations through collaborative filtering (CF). One of the studies initially present a framework for the CF recommender system, which is based on a variety of user data, such as user ratings and user behaviors. We explain the main characteristics of these two types of data. Also, a number of common CF algorithms are divided into memory-based and model-based techniques and contrasted. In an effort to support the suggested framework, two case studies are offered.[10]

Memory-based CF algorithms use historical data volumes directly to estimate ratings on target items and offer suggestions to the user who is currently active. On the other hand, model-based CF can make use of certain data mining techniques to build a prediction model based on the available data.[10]

User-based CF first determines how similar the active user and other users are. Pure cosine, adjusted cosine, and Pearson correlation coefficient are typical CF similarity measurements. The active user's neighbors are chosen based on their great similarity. The cosine value of the angle between two vectors is measured via pure cosine similarity. Users are represented by the rows of the rating matrix in the user-based CF, with the missing values set to 0.[10]

CHAPTER 3

FEASIBILITY STUDY

3.1 Technical Feasibility

This study defines if the project can be feasible in terms of technical requirements or not. All the necessary technologies are well-established and widely used, with a large developer community and resources available online. All the modern system that has access to the internet will support our app and will run smoothly. Overall, the technical feasibility of the project is high.

3.2 Operational Feasibility

A project needs to be applicable in the real world. If it exists only in the theoretical approach, it cannot be considered beneficial. The app has the potential to provide a more efficient and streamlined shopping experience for customers, and can help farmers reach a wider audience. No user training is required. Any failure in the system can be maintained. Therefore, this project is feasible operationally.

3.3 Economical Feasibility

The project should be economically beneficial to every person in general. Our project requires only the maintenance cost of server and for hosting with zero production cost. Additionally, the project can help farmers sell their products at a fair price, while also providing customers with affordable and fresh products. Hence it can be said that the project is economically feasible.

3.4 Schedule Feasibility

The development of the app will require time and resources. The project can be divided into several stages such as requirements gathering, design, development, testing, and deployment. The timeline for each stage will depend on the size and complexity of the project. However, the use of well-established technologies and frameworks can help speed up the development process. Therefore, the schedule feasibility of the app is moderate to high.

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional requirements:

The functional requirements of the system are:

1. User registration and login system.
2. Search and filter functionality for products.
3. Product description and image display.
4. Shopping cart system to add and remove products.
5. Checkout order.
6. User profile and order history management.
7. Product rating system.
8. Product Recommendation based on rating.

4.2 Non-Functional requirements:

The non-functional requirements of the system are:

1. User-friendly interface with easy navigation.
2. Fast and responsive app loading speed to ensure a smooth user experience.
3. Compatibility with different devices, browsers, and operating systems.
4. Scalability to handle a large number of users and products.
5. Reliable backup and recovery system to prevent data loss in case of technical failures.
6. Compliance with data protection laws and regulations.

4.3 Hardware Requirements

- Processor- Intel core i5/AMD Ryzen 5 and above.

- RAM 4GB DDR and above.
- Secondary storage 2GB

4.4 Software Requirements

- Operating System: Windows/ Linux/ IOS/ Android
- Programming language: Python, dart, java script
- Numpy Version: 1.3.0 and above
- Django and Django Rest Framework
- Flutter: cross platform application development framework
- Code editor: VS Code

CHAPTER 5

SYSTEM DESIGN AND ARCHITECTURE

5.1 System Block Diagram

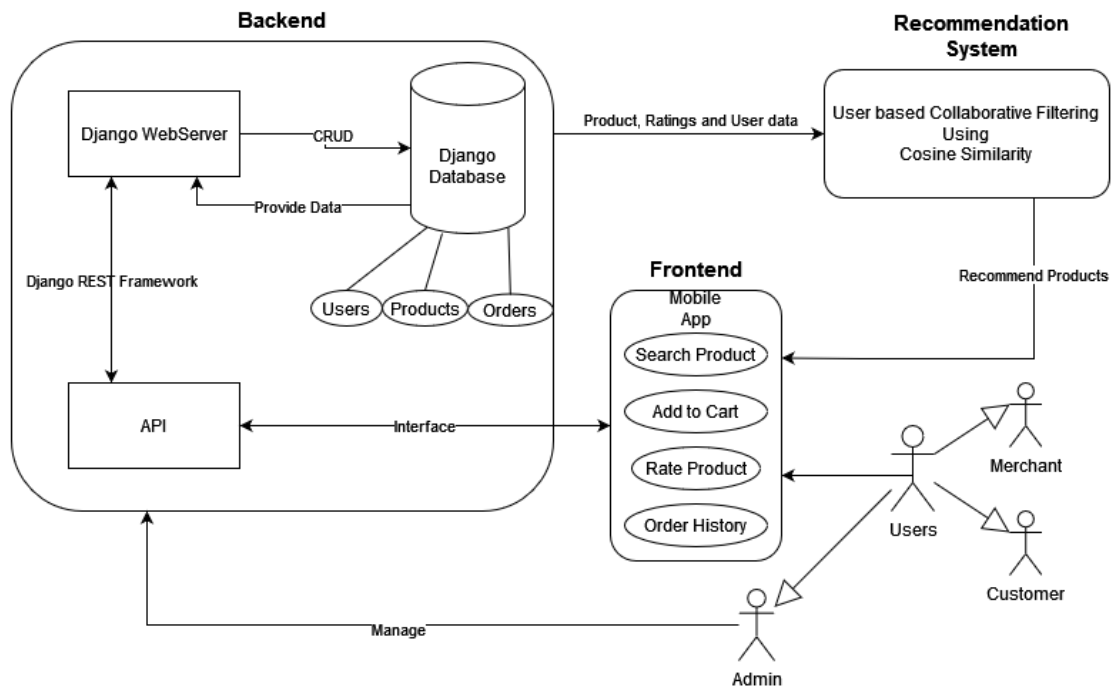


Figure 5.1: System Block Diagram

5.2 Use Case Diagram

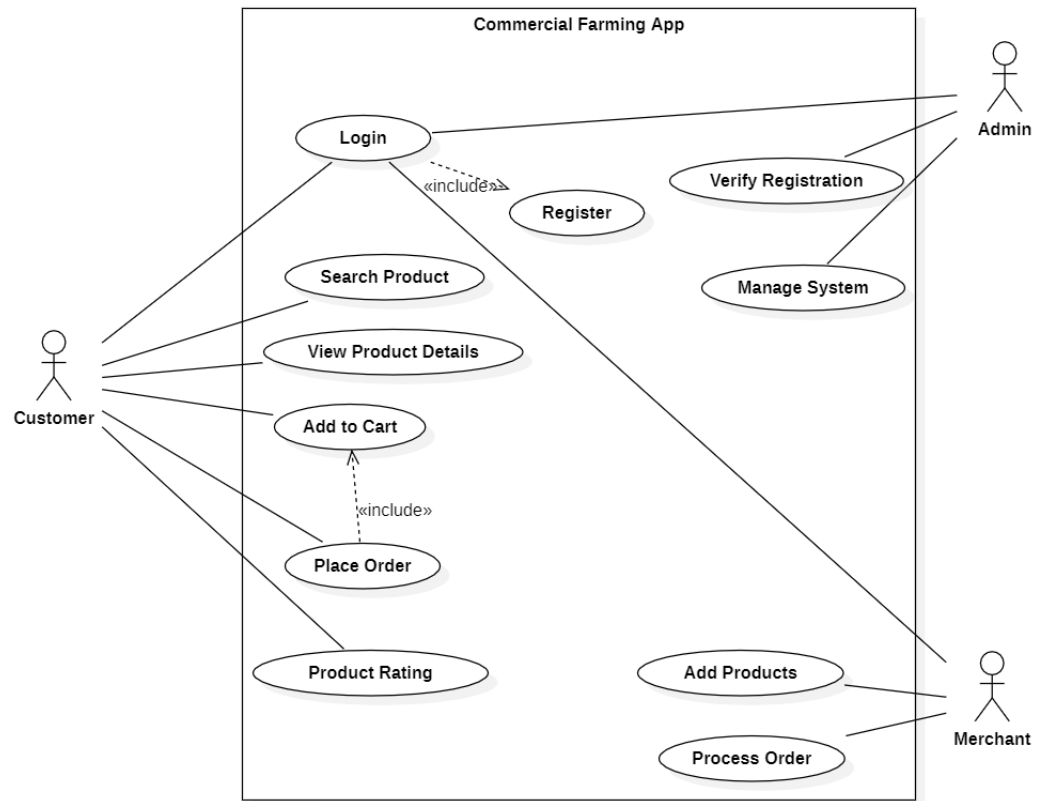


Fig 5.2: Use Case Diagram

5.3 Data Flow Diagram (DFD)

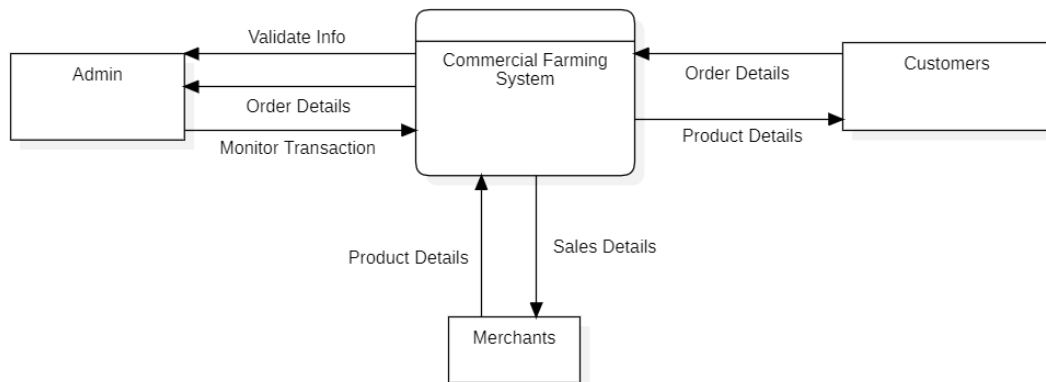


Fig 5.3(a): DFD Level 0

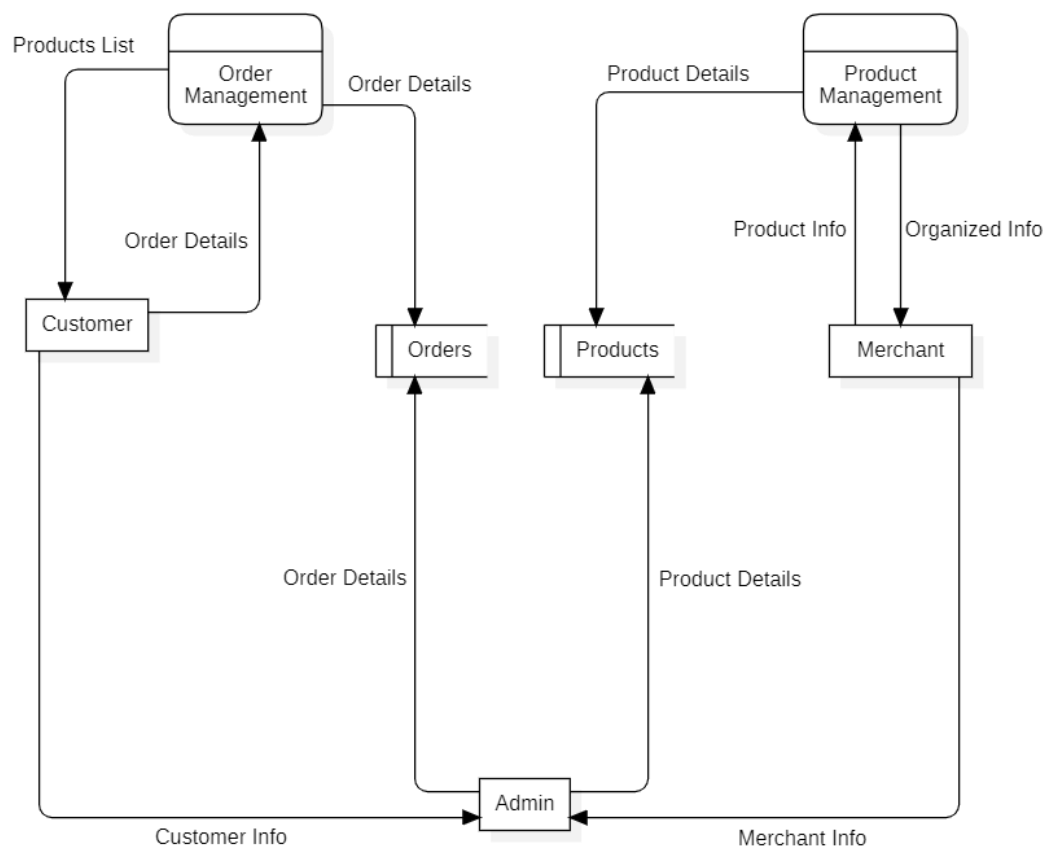


Fig 5.3(b): DFD Level 1

5.4 Entity-Relation (ER) Diagram

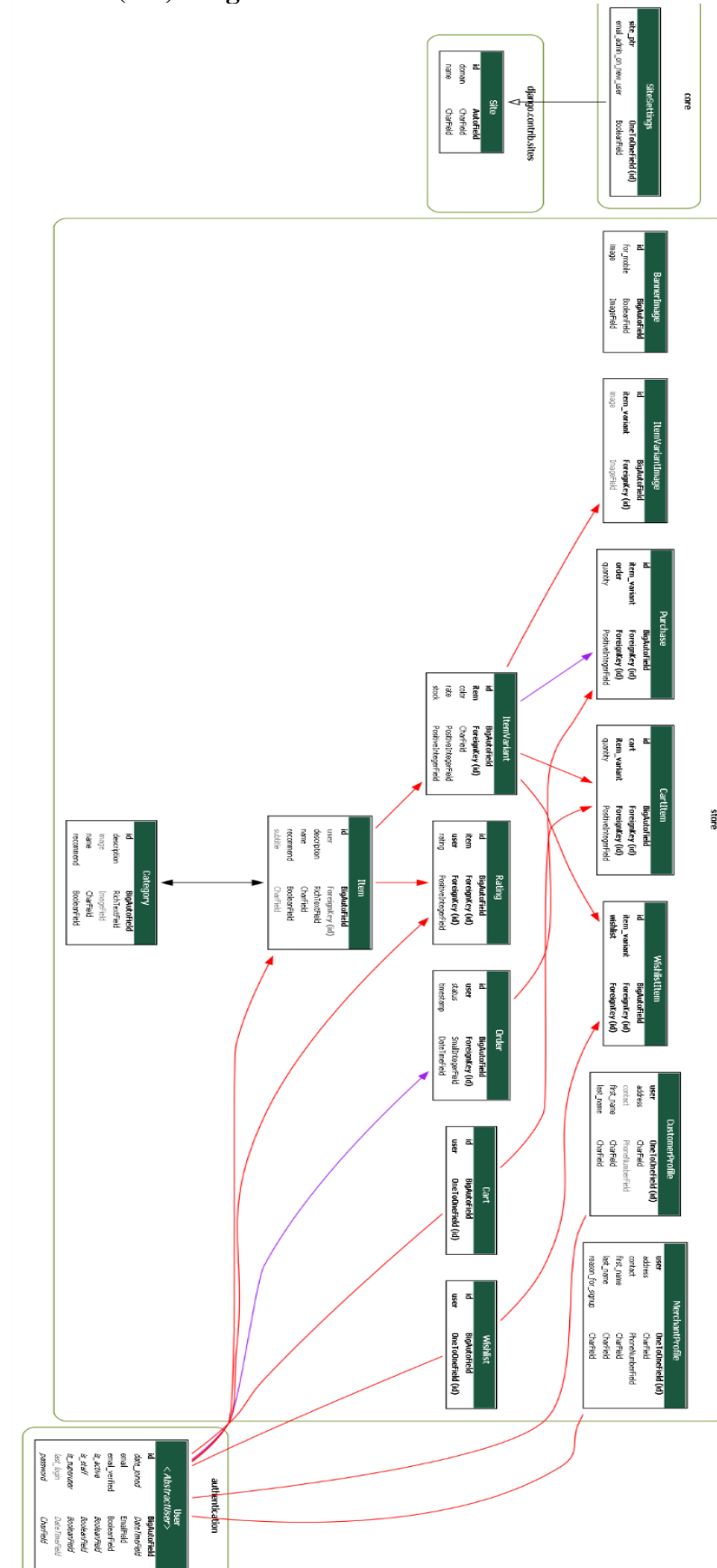


Fig 5.4: Generated E-R Diagram

CHAPTER 6

METHODOLOGY

This chapter includes the methodology adopted in the completion of the project. It covers data collection, analysis of the data (Cosine Similarities) and recommendation system building, backend interaction and system testing.

6.1 Development Process

The mobile software engineering process may be classified into the following four primary phases after reading and examining the survey data. Mobile specialists have further approved these phases as a tried-and-true method for developing effective mobile applications from conception to release.[11]

6.1.1 Phase 1: Envision

Planning and analysis are part of this step. Participants stressed that determining the mobile initiatives, problem, purpose, objective, and audience for whom the application will be built is the first step in creating an excellent mobile application. Planning comes after analysis, and its primary purpose is to ensure that team members are well-informed about the objectives for product creation, mobile technology, and inventive design needs. It is advised that this goal should start with the layout of a mobile product, which involves planning for estimates, strategies, and user experiences, and then identifying the precise solution.[11]

6.1.2 Phase 2: Solution

This phase is categorized into two significant parts design and development. The design state creates a blueprint or visual representation of the application using specific tools, such as UI/UX, that give the app's client a certain feel. The actual development of an app should start when the client approves the design, with iterations according to the project plan.[11]

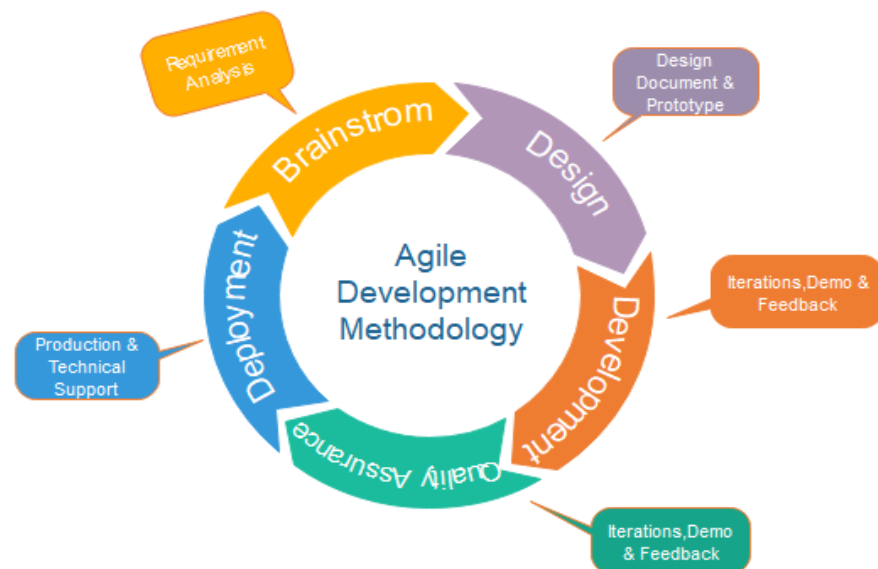
6.1.3 Phase 3: Quality Assurance

After phase two, the application should be tested using the project plan, requirements, specifications, wireframes, and drawings developed in the previous phases after the

design and development stage. According to professional mobile tester participants, the testing phase should include the definition of test cases, automated testing, and testing on emulators. The test cases should be defined under the integrated system and module standalone test standards. User Acceptance Testing (UAT) should be followed by final issue repairs, user-approved improvements, and user testing before receiving client approval.[11]

6.1.4 Phase 4: Product Release

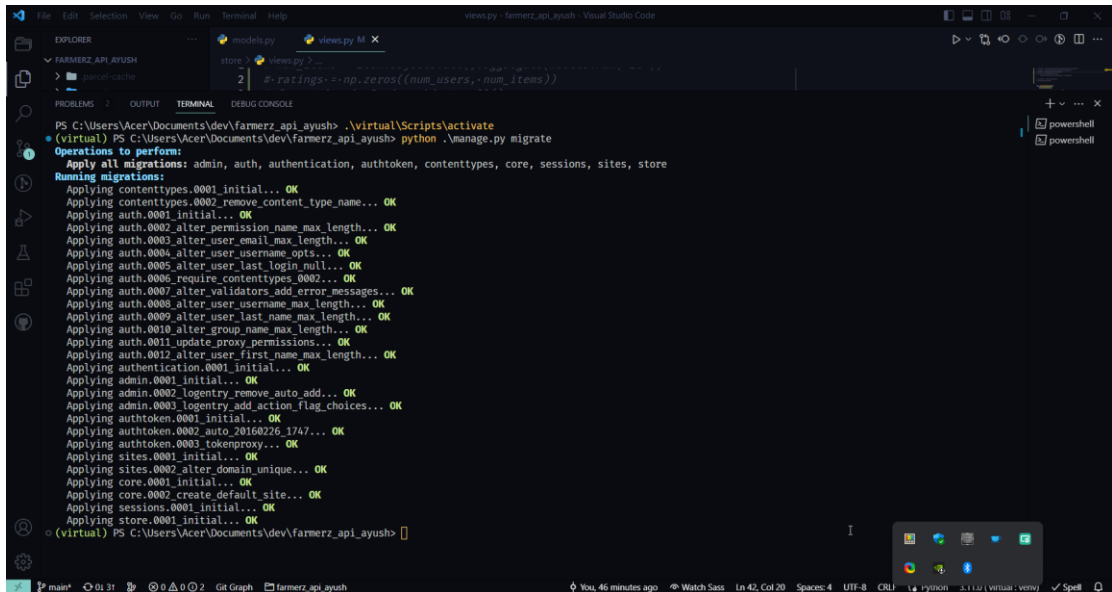
Deployment, Support and Maintenance are the critical factors considered in this phase. After receiving final customer feedback and approval, the mobile app development team members began preparing for deployment by installing the produced mobile app on the server and submitting it to the app store. Most survey respondents strongly underlined the need to identify faults based on user and market feedback through memory optimization, automated crash reporting, and user modification requests. They strongly suggested that bug patches be included in routine iterative releases with either app store or corporate deployment for product maintenance (support) and product enhancement (upgrade); with platform upgrades, new features, and functionality, this support should enhance the app. [11]



[11, Fig.6.1: Agile Methodology]

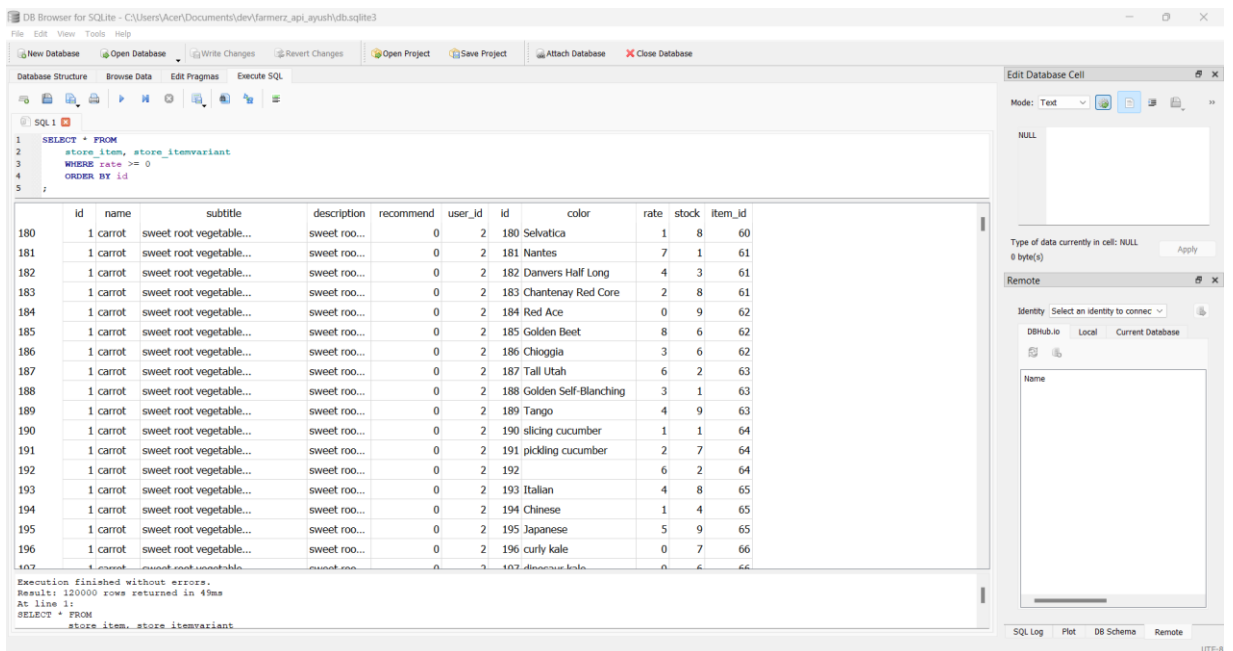
6.2 Data Collection

Sample data of varous vegetable were taken from the internet and a database with radomized user data were kept.



```
PS C:\Users\Acer\Documents\dev\farmerz_api_ayush> .\virtual\Scripts\activate
(virtual) PS C:\Users\Acer\Documents\dev\farmerz_api_ayush> python .\manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, authentication, authtoken, contenttypes, core, sessions, sites, store
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying authentication.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying authtoken.0001_initial... OK
  Applying authtoken.0002_auto_20160226_1747... OK
  Applying authtoken.0003_tokenproxy... OK
  Applying sites.0001_initial... OK
  Applying sites.0002_alter_domain_unique... OK
  Applying core.0001_initial... OK
  Applying core.0002_create_default_site... OK
  Applying sessions.0001_initial... OK
  Applying store.0001_initial... OK
(virtual) PS C:\Users\Acer\Documents\dev\farmerz_api_ayush>
```

Fig 6.2(a): Table creation in Django



	id	name	subtitle	description	recommend	user_id	id	color	rate	stock	item_id
180	1	carrot	sweet root vegetable...	sweet roo...	0	2	180 Selvatica		1	8	60
181	1	carrot	sweet root vegetable...	sweet roo...	0	2	181 Nantes		7	1	61
182	1	carrot	sweet root vegetable...	sweet roo...	0	2	182 Danvers Half Long		4	3	61
183	1	carrot	sweet root vegetable...	sweet roo...	0	2	183 Chantenay Red Core		2	8	61
184	1	carrot	sweet root vegetable...	sweet roo...	0	2	184 Red Ace		0	9	62
185	1	carrot	sweet root vegetable...	sweet roo...	0	2	185 Golden Beet		8	6	62
186	1	carrot	sweet root vegetable...	sweet roo...	0	2	186 Chioggia		3	6	62
187	1	carrot	sweet root vegetable...	sweet roo...	0	2	187 Tall Utah		6	2	63
188	1	carrot	sweet root vegetable...	sweet roo...	0	2	188 Golden Self-Blanching		3	1	63
189	1	carrot	sweet root vegetable...	sweet roo...	0	2	189 Tango		4	9	63
190	1	carrot	sweet root vegetable...	sweet roo...	0	2	190 slicing cucumber		1	1	64
191	1	carrot	sweet root vegetable...	sweet roo...	0	2	191 pickling cucumber		2	7	64
192	1	carrot	sweet root vegetable...	sweet roo...	0	2	192		6	2	64
193	1	carrot	sweet root vegetable...	sweet roo...	0	2	193 Italian		4	8	65
194	1	carrot	sweet root vegetable...	sweet roo...	0	2	194 Chinese		1	4	65
195	1	carrot	sweet root vegetable...	sweet roo...	0	2	195 Japanese		5	9	65
196	1	carrot	sweet root vegetable...	sweet roo...	0	2	196 curly kale		0	7	66

Fig 6.2(b): Database Schema

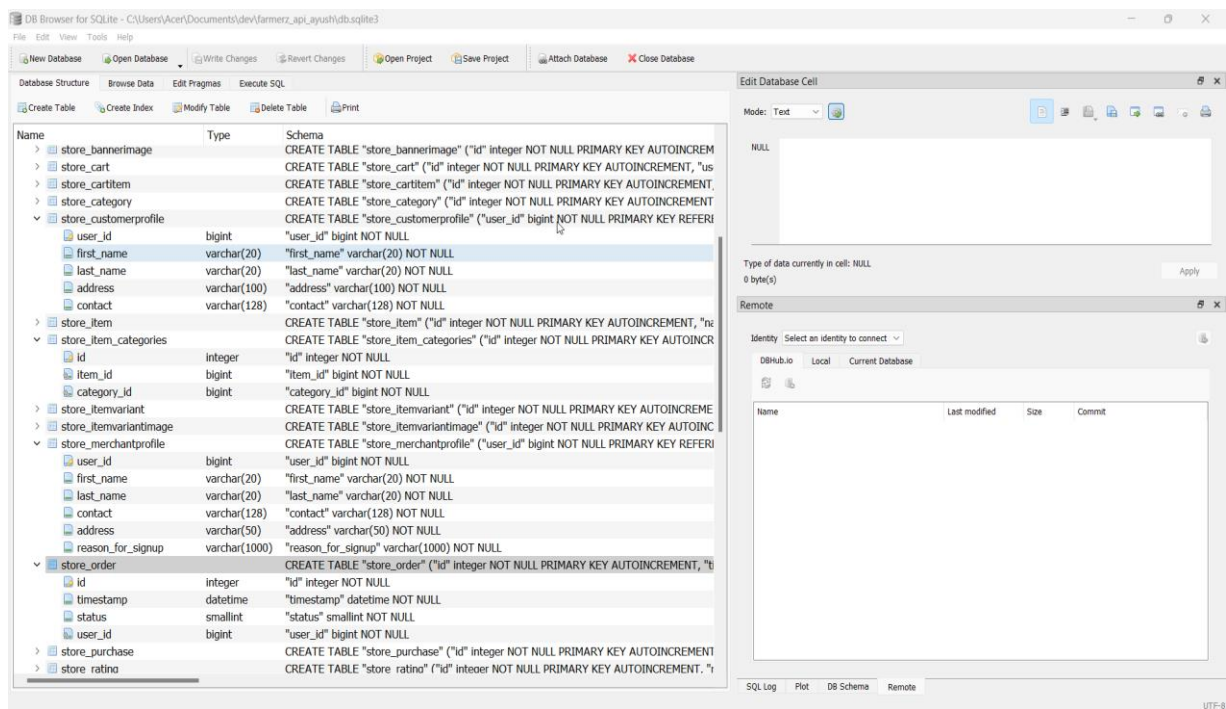


Fig 6.2(c): Database Schema Expanded

```
Added zucchini
Added watercress
Added mustard greens
Added collard greens
Added arugula
Created customer 'customer@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer1@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer2@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer3@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer4@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer5@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer6@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer7@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer8@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
Created customer 'customer9@example.com', with password 'shark@123'
Created 5 random orders
Filled cart with 5 items
```

Fig 6.2(d): Populating Database

6.3 Data Analysis and Recommendations system building

6.3.1 Cosine Similarity

Cosine similarity is a measure of similarity between two non-zero vectors defined in an inner product space. Cosine similarity is the cosine of the angle between the vectors; that is, it is the dot product of the vectors divided by the product of their lengths. It follows that the cosine similarity does not depend on the magnitudes of the vectors, but only on their angle. The cosine similarity always belongs to the interval $[-1,1]$. For example, two proportional vectors have a cosine similarity of 1, two orthogonal vectors have a similarity of 0, and two opposite vectors have a similarity of -1. In some contexts, the component values of the vectors cannot be negative, in which case the cosine similarity is bounded in $[0,1]$.

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$

A similarity matrix is constructed. In a similarity matrix a single element is the measure of similarity between two users. Thus, a similarity matrix consists of the similarity of all the combinations of two vectors in the dataset.

```

Rating matrix:
[[2 1 3 2 1 6 1 5 5 3]
 [2 3 6 3 2 3 0 5 0 4]
 [1 3 0 0 4 6 3 1 4 0]
 [4 5 3 1 3 6 1 2 2 4]
 [4 3 5 3 6 2 2 1 2 6]
 [5 5 6 6 2 5 0 2 0 4]
 [4 6 4 0 4 4 5 0 5 1]
 [3 1 6 6 5 2 6 5 4 3]
 [5 1 3 1 6 2 0 3 3 4]
 [3 6 2 1 2 4 1 3 6 6]]

Similarity Matrix:
[[1.      0.881 0.807 0.814 0.661 0.802 0.763 0.757 0.732 0.832]
 [0.881 1.      0.741 0.825 0.835 0.896 0.834 0.91  0.78  0.828]
 [0.807 0.741 1.      0.875 0.793 0.83  0.898 0.762 0.724 0.84 ]
 [0.814 0.825 0.875 1.      0.841 0.888 0.854 0.674 0.809 0.9 ]
 [0.661 0.835 0.793 0.841 1.      0.859 0.787 0.831 0.919 0.784]
 [0.802 0.896 0.83  0.888 0.859 1.      0.922 0.851 0.746 0.834]
 [0.763 0.834 0.898 0.854 0.787 0.922 1.      0.858 0.793 0.826]
 [0.757 0.91  0.762 0.674 0.831 0.851 0.858 1.      0.849 0.659]
 [0.732 0.78  0.724 0.809 0.919 0.746 0.793 0.849 1.      0.768]
 [0.832 0.828 0.84  0.9  0.784 0.834 0.826 0.659 0.768 1.    ]]

Estimated ratings for 2:
[2.694 2.885 3.384 2.276 2.876 3.397 2.31  2.383 3.265 3.12 ]

Recommended item ids:
(5, 2, 8, 9, 1, 4, 0, 7, 6, 3)

```

Fig 6.3.1: Collaborative filtering using Cosine Similarity of a test system.

GET /item/recommend/

Get items similar to this

```
GET /api/item/recommend
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
```

```
[
  141,
  99,
  190,
  171,
  81,
  156,
  29,
  145,
  178,
  66
]
```

Fig 6.3: Item recommendation response for a particular user.

6.4 Backend Interaction

Django Rest API was used to create model of the database and create a restful API to interact with the server.

```
[14/Mar/2023 21:41:10] "GET /admin/authentication/user/ HTTP/1.1" 200 22134
[14/Mar/2023 21:41:10] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:12] "GET /admin/authentication/user/1/change/ HTTP/1.1" 200 24474
[14/Mar/2023 21:41:12] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:14] "POST /admin/authentication/user/1/change/ HTTP/1.1" 302 0
[14/Mar/2023 21:41:14] "GET /admin/authentication/user/ HTTP/1.1" 200 22134
[14/Mar/2023 21:41:14] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:16] "GET /admin/authentication/user/1/change/ HTTP/1.1" 200 24482
[14/Mar/2023 21:41:16] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:21] "POST /admin/authentication/user/1/change/ HTTP/1.1" 302 0
[14/Mar/2023 21:41:21] "GET /admin/authentication/user/ HTTP/1.1" 200 22134
[14/Mar/2023 21:41:21] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:22] "GET /admin/authentication/user/1/change/ HTTP/1.1" 200 24474
[14/Mar/2023 21:41:22] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:25] "POST /admin/authentication/user/1/change/ HTTP/1.1" 302 0
[14/Mar/2023 21:41:25] "GET /admin/authentication/user/ HTTP/1.1" 200 22134
[14/Mar/2023 21:41:25] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:27] "GET /admin/authentication/user/1/change/ HTTP/1.1" 200 24482
[14/Mar/2023 21:41:27] "GET /admin/jsi18n/ HTTP/1.1" 200 3343
[14/Mar/2023 21:41:29] "GET /admin/authentication/user/1/history/ HTTP/1.1" 200 10499
Method Not Allowed: /api/item/
[14/Mar/2023 21:46:46] "POST /api/item/?number=100&query=a&metaInformation=true HTTP/1.1" 405 41
[14/Mar/2023 21:46:51] "GET /api/item/?number=100&query=a&metaInformation=true HTTP/1.1" 200 21303
[14/Mar/2023 21:47:14] "GET /api/item/200 HTTP/1.1" 200 9620
[14/Mar/2023 21:47:14] "GET /static/rest_framework/css/bootstrap-tweaks.css HTTP/1.1" 200 863
[14/Mar/2023 21:47:14] "GET /static/rest_framework/css/default.css HTTP/1.1" 200 1152
[14/Mar/2023 21:47:14] "GET /static/rest_framework/css/prettify.css HTTP/1.1" 200 817
[14/Mar/2023 21:47:14] "GET /static/rest_framework/js/csrf.js HTTP/1.1" 200 1719
[14/Mar/2023 21:47:14] "GET /static/rest_framework/js/ajax-form.js HTTP/1.1" 200 3597
[14/Mar/2023 21:47:14] "GET /static/rest_framework/css/bootstrap.min.css HTTP/1.1" 200 121457
[14/Mar/2023 21:47:14] "GET /static/rest_framework/js/bootstrap.min.js HTTP/1.1" 200 39680
[14/Mar/2023 21:47:14] "GET /static/rest_framework/js/jquery-3.5.1.min.js HTTP/1.1" 200 89476
[14/Mar/2023 21:47:14] "GET /static/shared/logo.png HTTP/1.1" 200 6291
[14/Mar/2023 21:47:14] "GET /static/rest_framework/js/prettify-min.js HTTP/1.1" 200 13632
[14/Mar/2023 21:47:14] "GET /static/rest_framework/js/default.js HTTP/1.1" 200 1268
[14/Mar/2023 21:47:14] "GET /favicon.ico HTTP/1.1" 301 0
[14/Mar/2023 21:47:14] "GET /static/favicon.ico HTTP/1.1" 200 15086
[14/Mar/2023 21:47:27] "GET /api/item/200 HTTP/1.1" 200 9620
Method Not Allowed: /api/item/
[14/Mar/2023 21:47:49] "POST /api/item/?number=100&query=a&metaInformation=true HTTP/1.1" 405 41
```

Fig 6.4: Server Response Log

6.5 System Testing

Insomnia, a API testing platform, was used to interact and test API end points.

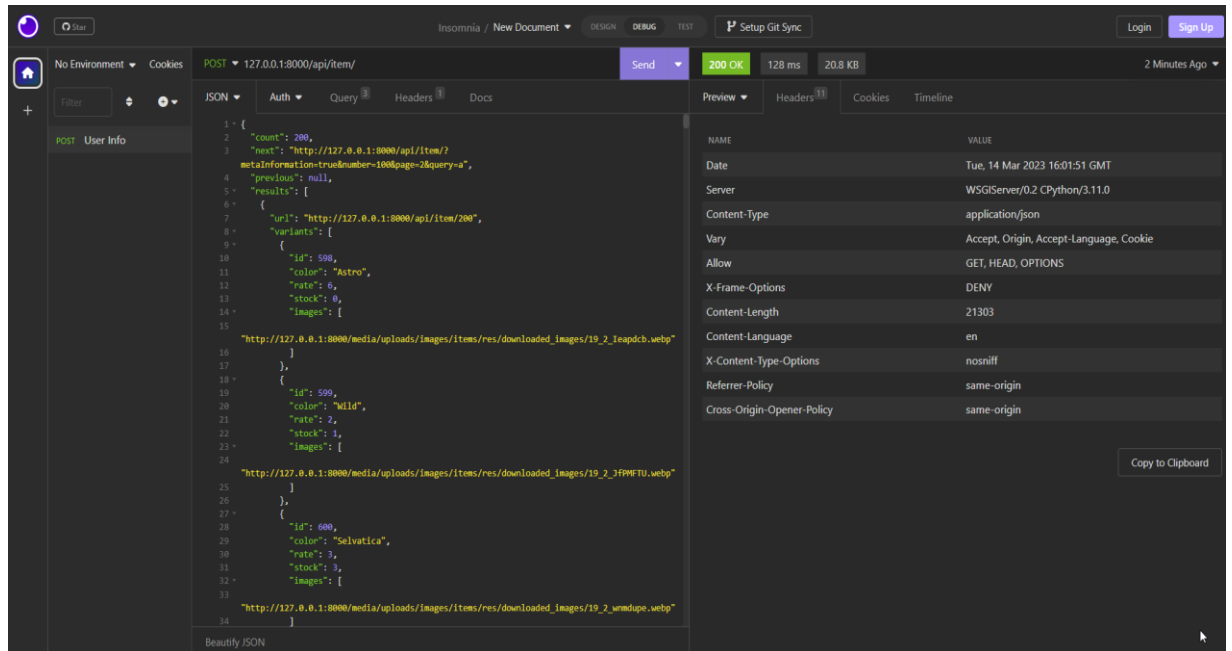


Fig 6.5: Testing for Server Interaction

CHAPTER 7

RESULT AND ANALYSIS

7.1 Results

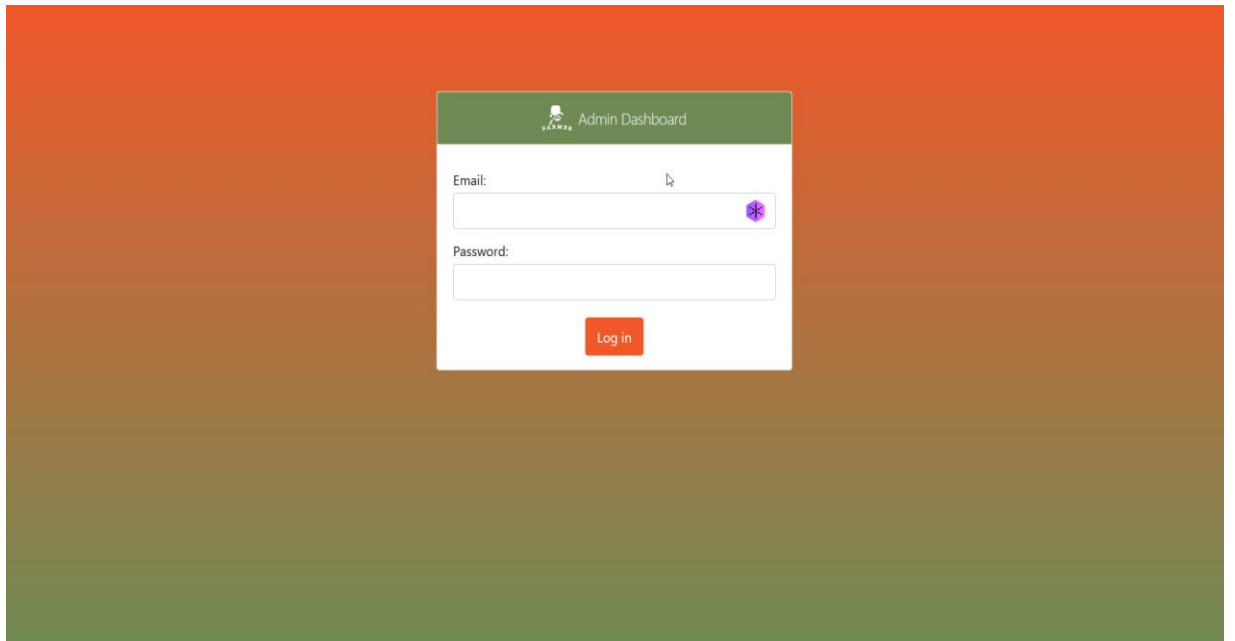


Fig7.1(a): Admin Dashboard Login Page

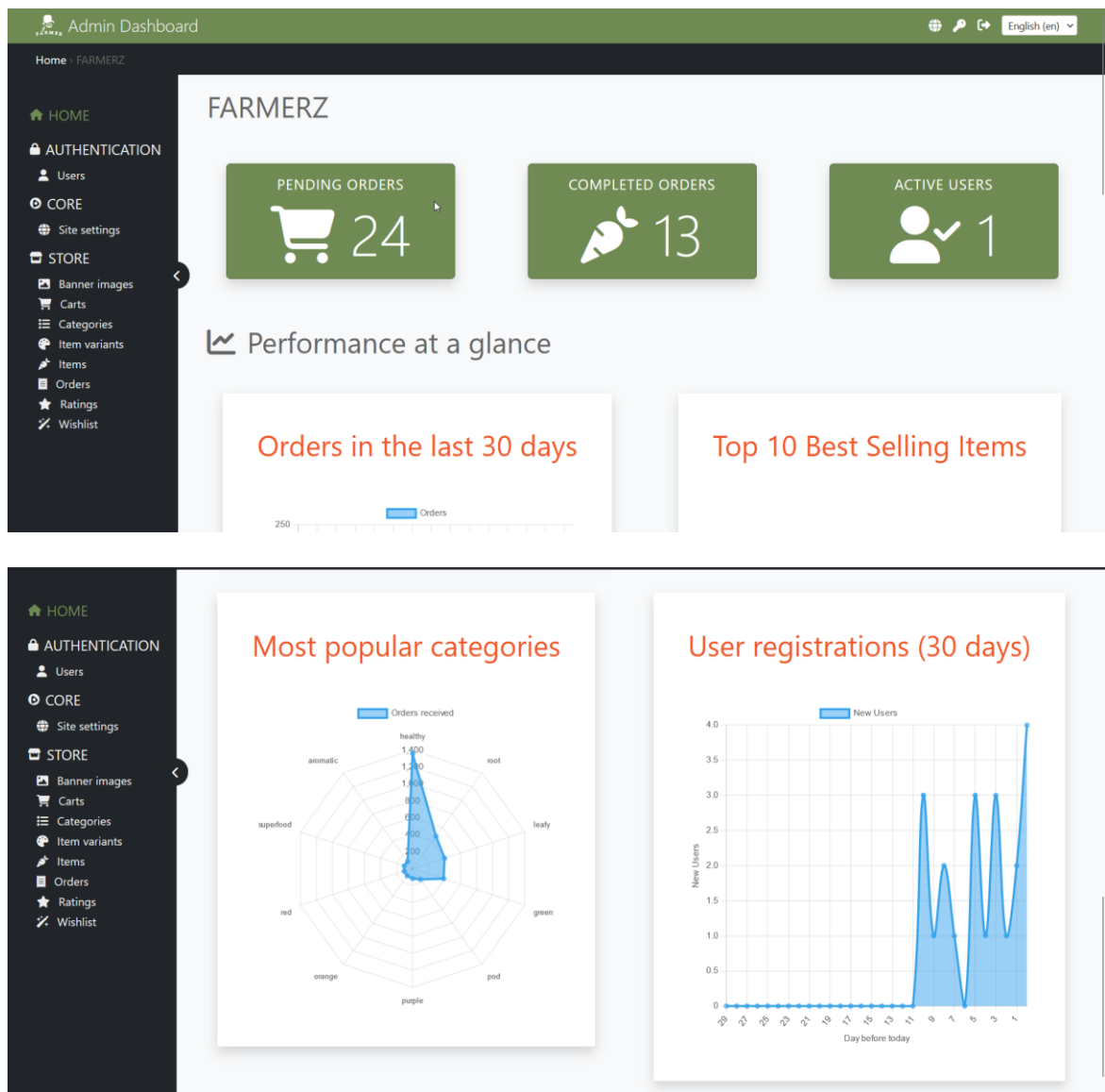


Fig7.1(b): Admin Dashboard Landing Page

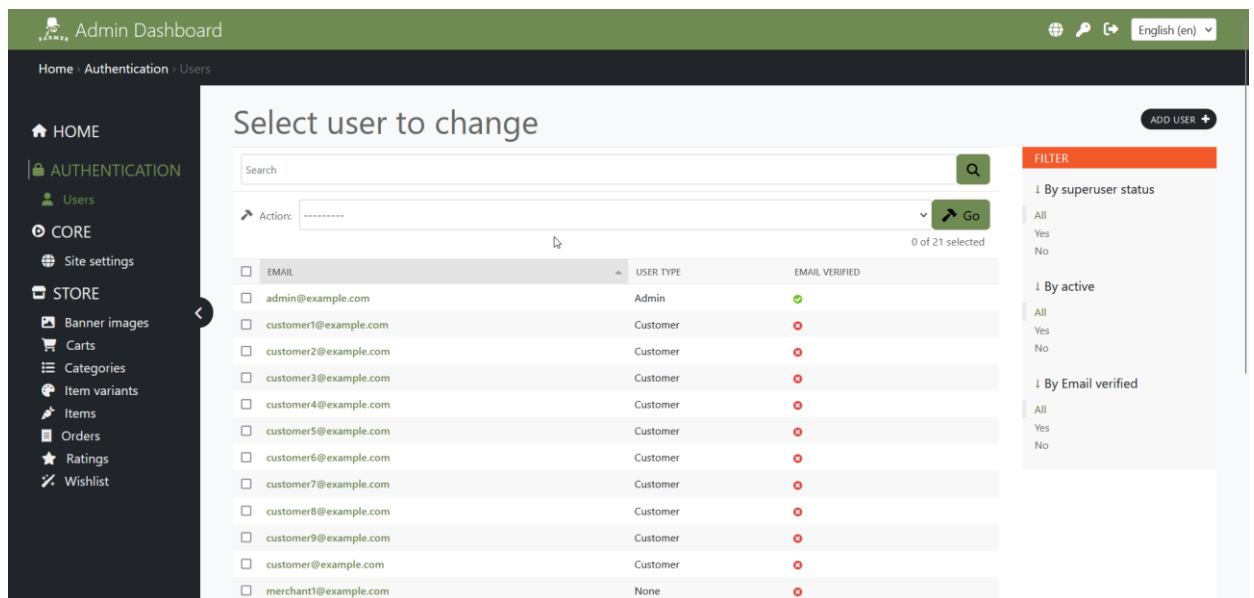


Fig 7.1(c): System User Page

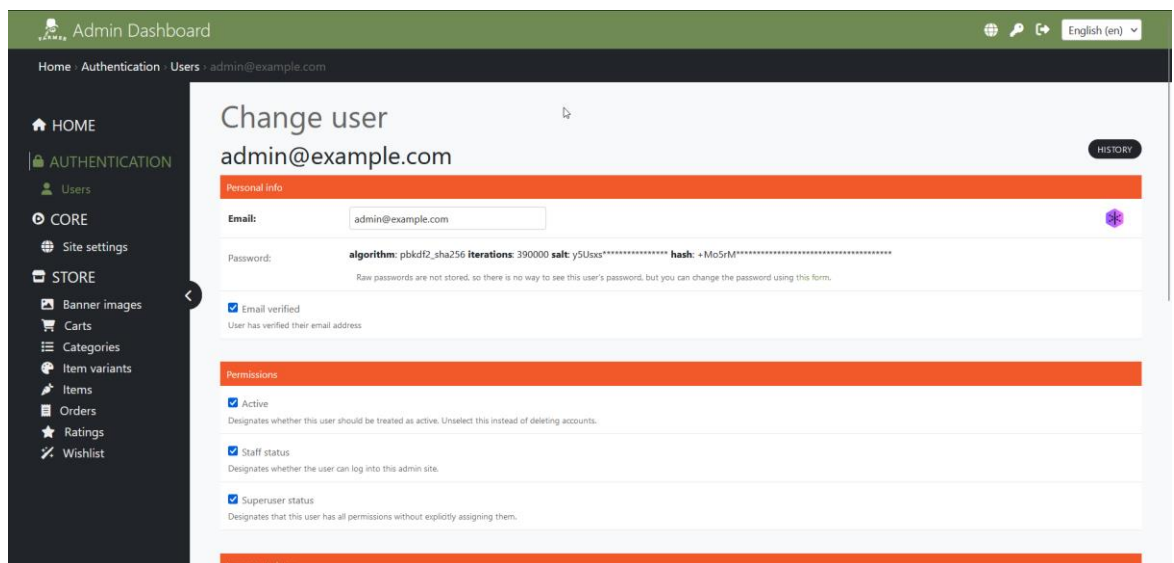


Fig 7.1(d): User Change Page

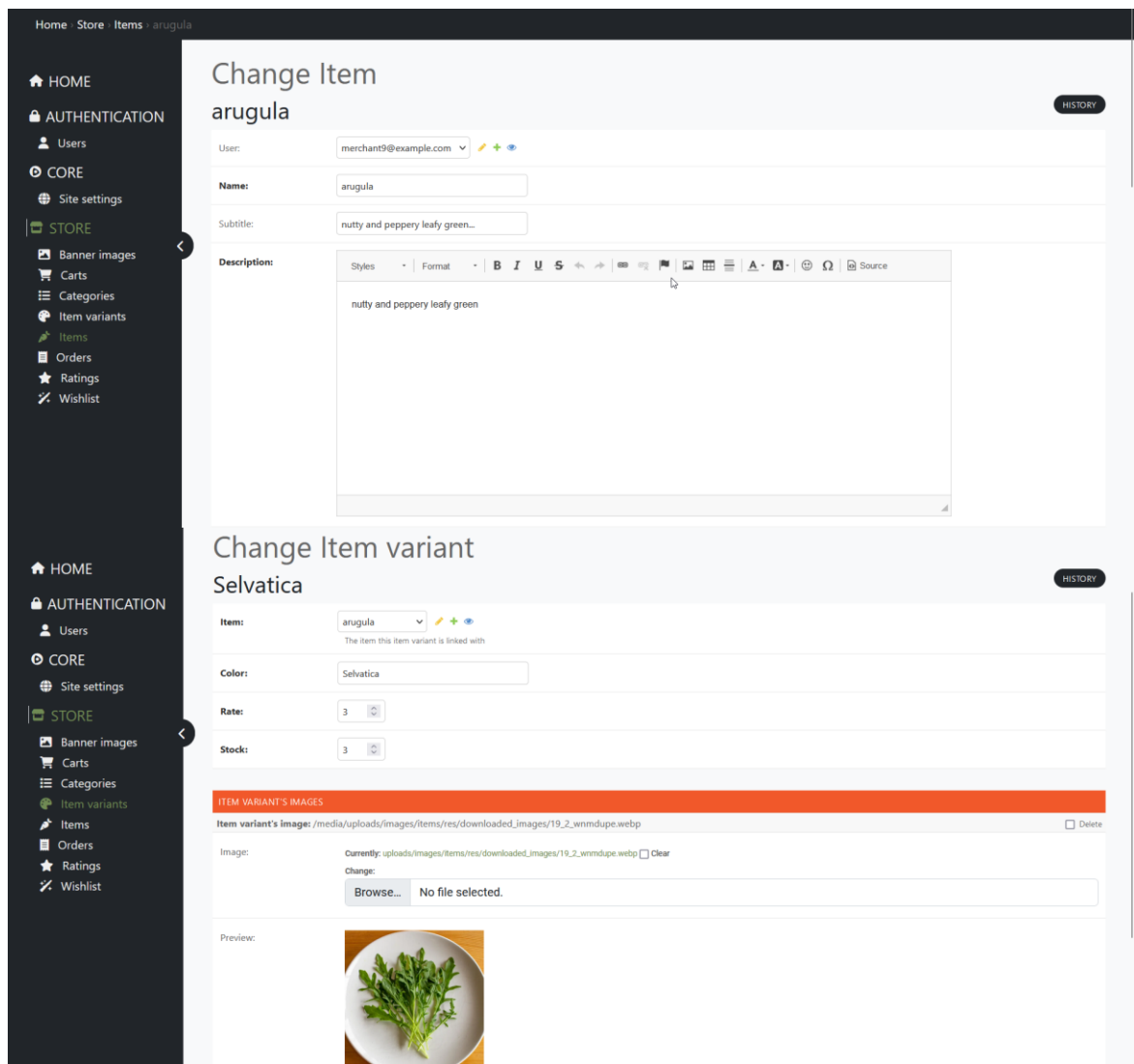


Fig 7.1(e): Item Page

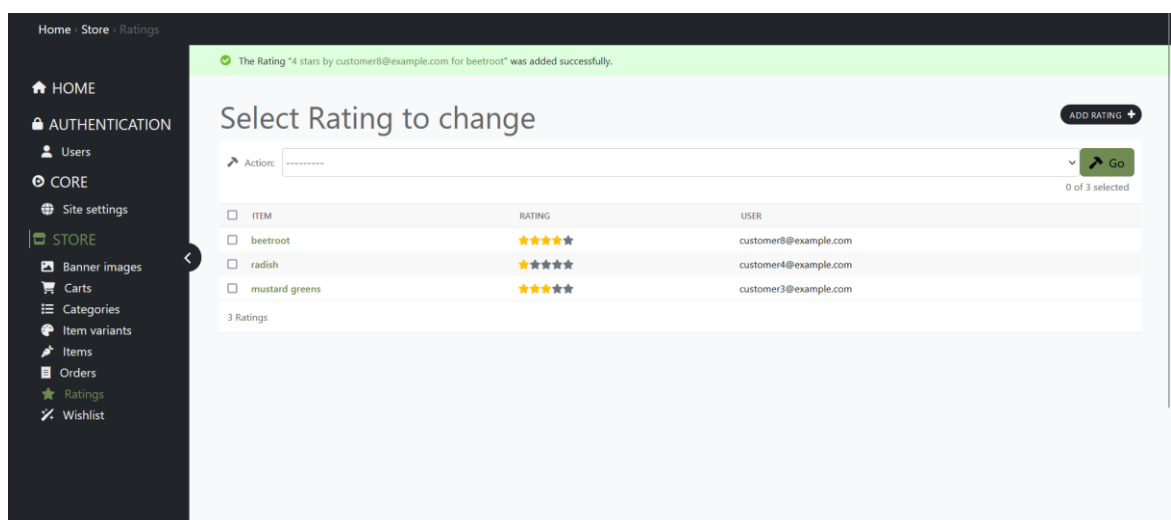


Fig 7.1(f): Rating Page

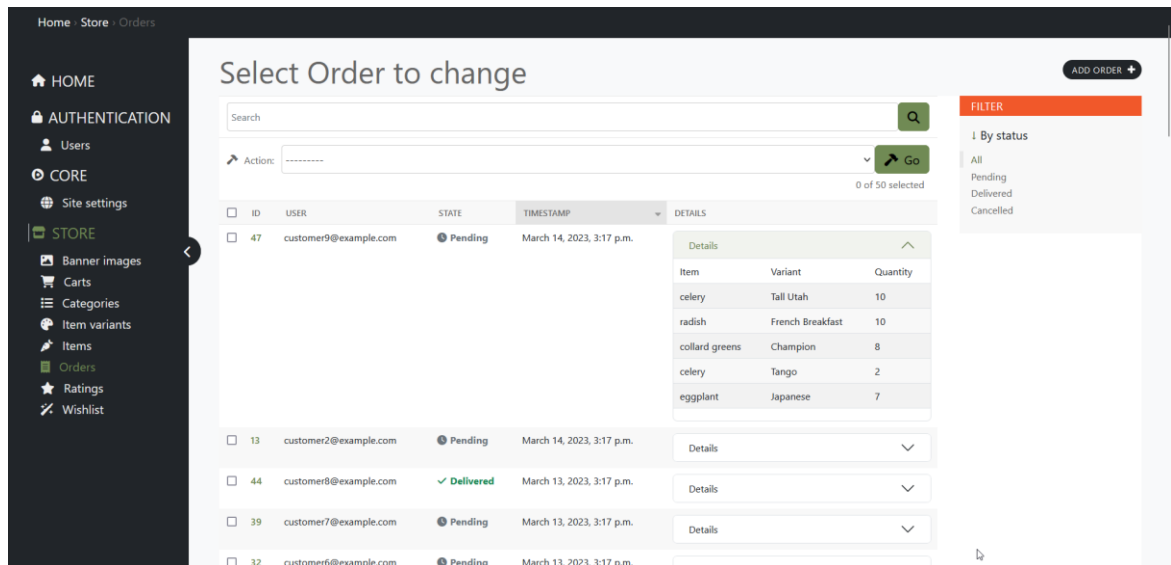


Fig 7.1(g): Orders Page

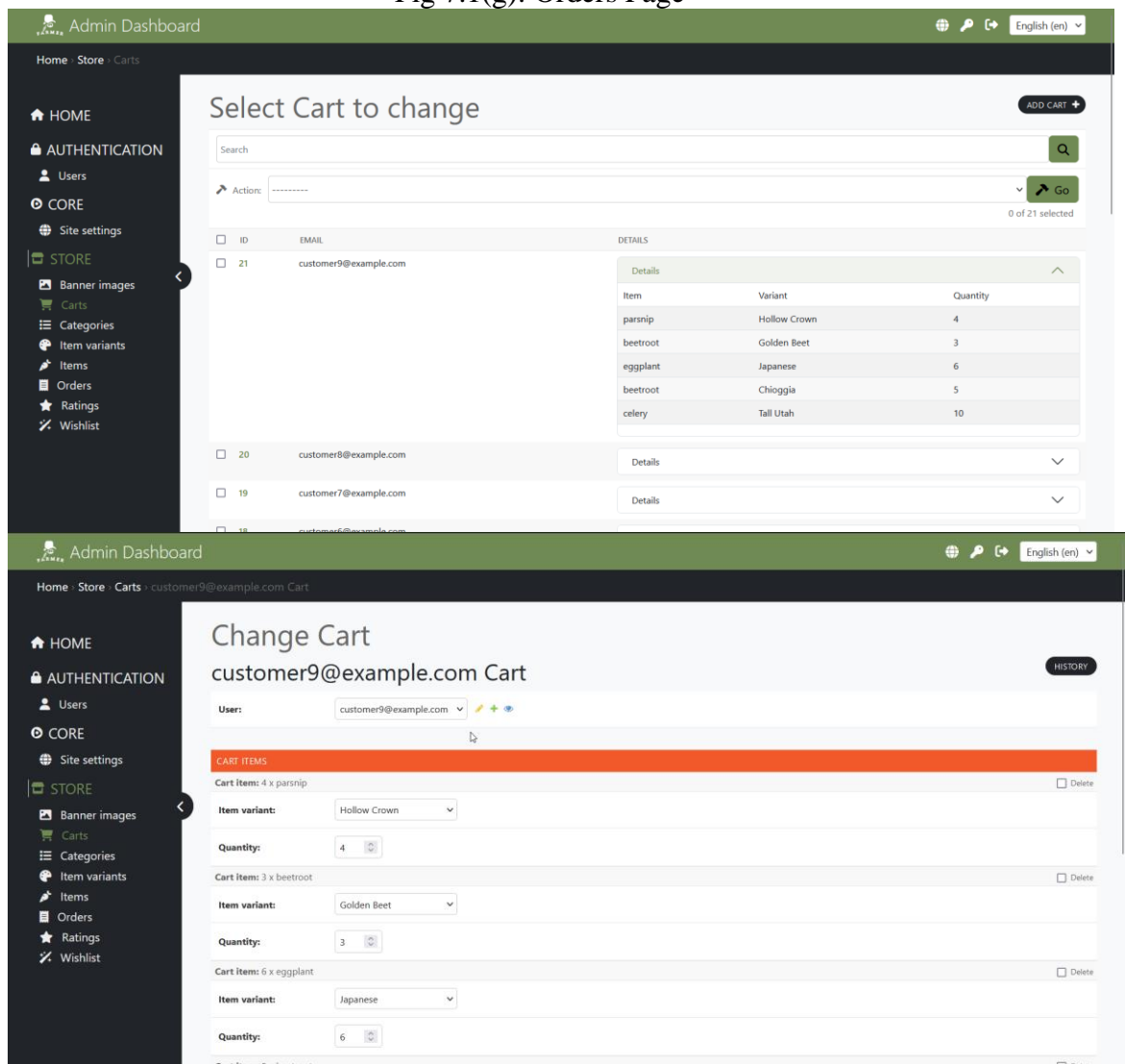


Fig 7.1(h): Carts Page

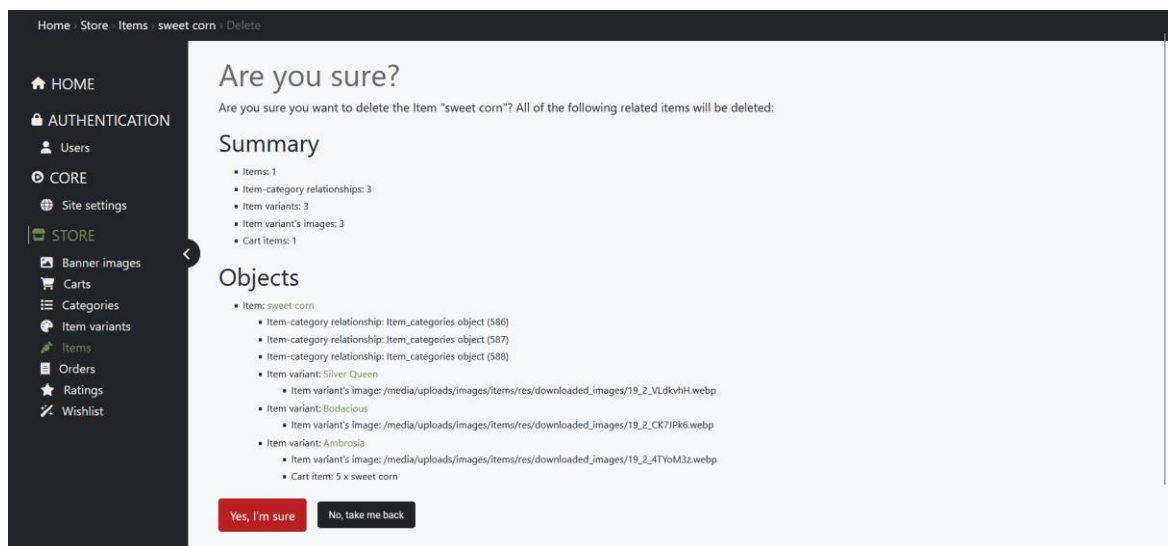


Fig 7.1(i): Delete Page

Email:

Password:

Log in

Fig 7.1(j): API Login Page

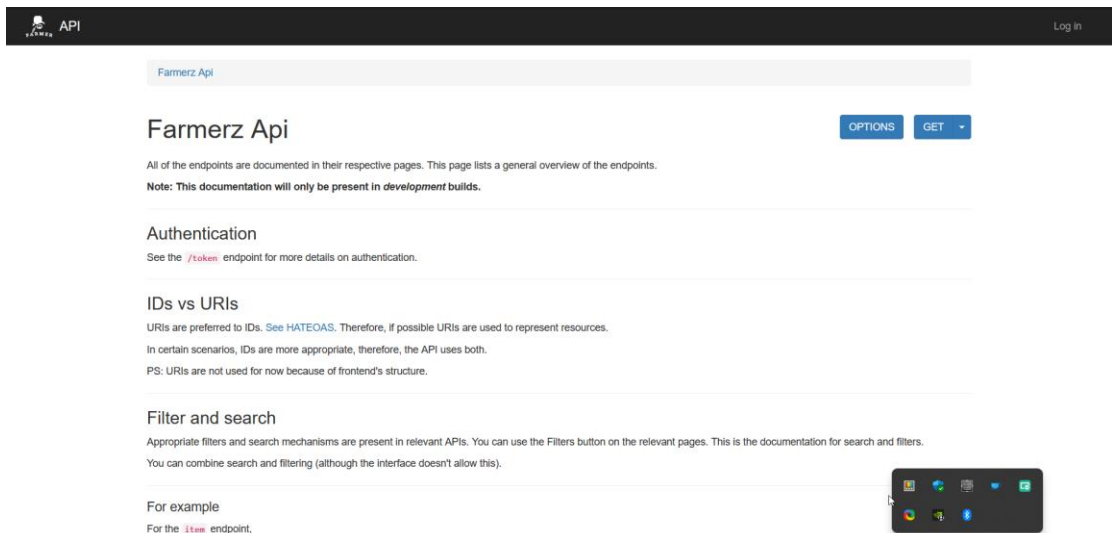


Fig 7.1(k): API Landing Page

API is rate limited to unauthenticated users for now. More complex rate limits can be added later.

About errors and validation

We follow the convention from [DRF docs. Read here.] (<https://www.django-rest-framework.org/api-guide/exceptions>)

All 400 series errors return a JSON containing a "detail" field which explains the error.

The exception is validation errors for form like structures. In those cases, the JSON response contains:

- field name as keys and list of error messages as value. Example: `{ 'password': ['must contain 8 characters', 'common password'] }`
- Non field errors are present as `non_field_errors`. Example: `{ 'non_field_errors': ['user has been banned by admin'] }`
- nested fields work as explained in DRF docs.

Authentication vs Authorization

- 401 - Unauthorized is returned for authentication failure.
- 403 - Forbidden is returned for authorization failure.

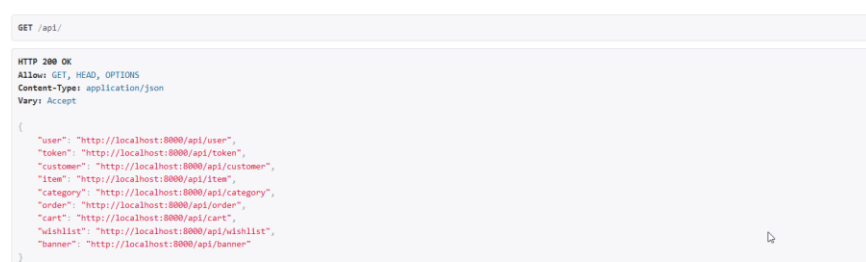


Fig 7.1(l): API Base Endpoints

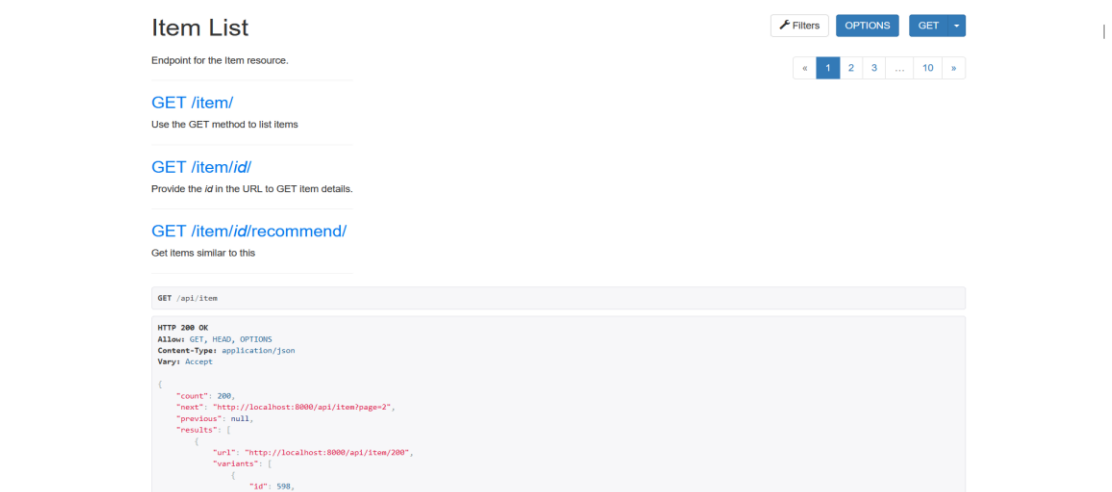


Fig 7.1(m): JSON Response

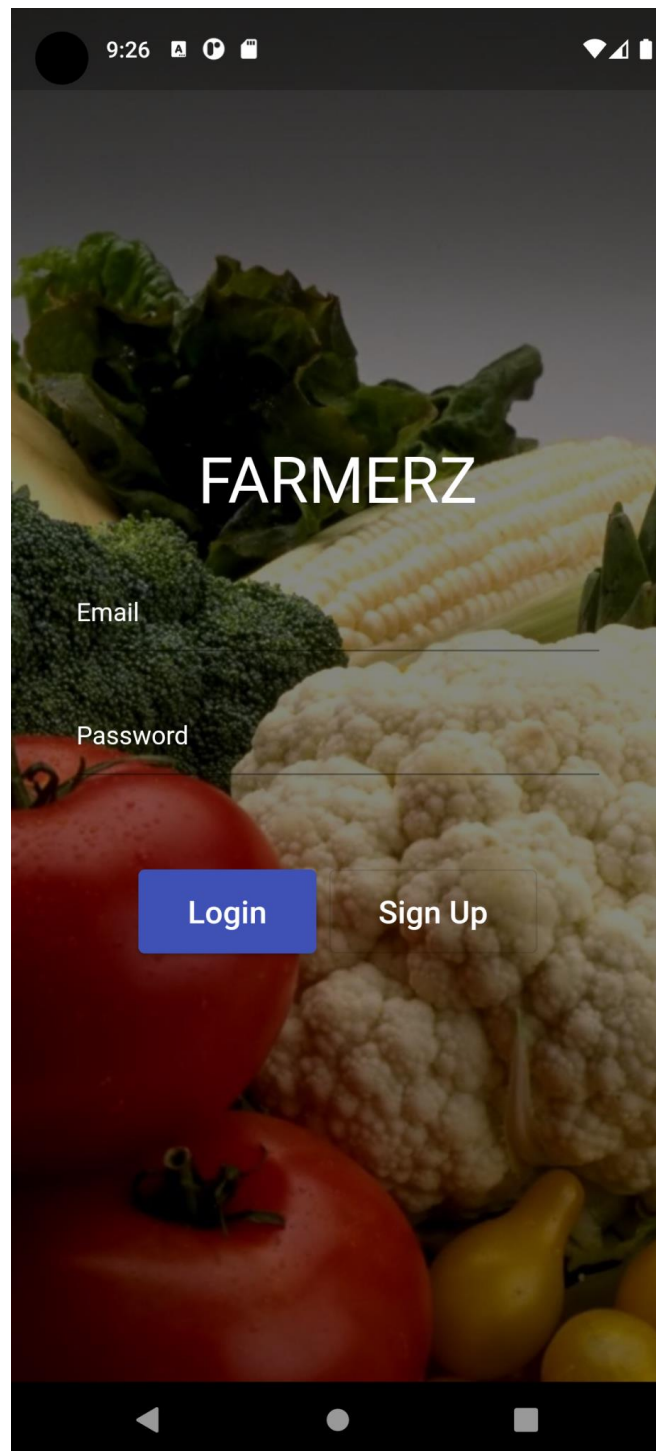


Fig 7.1(n): App Login Page

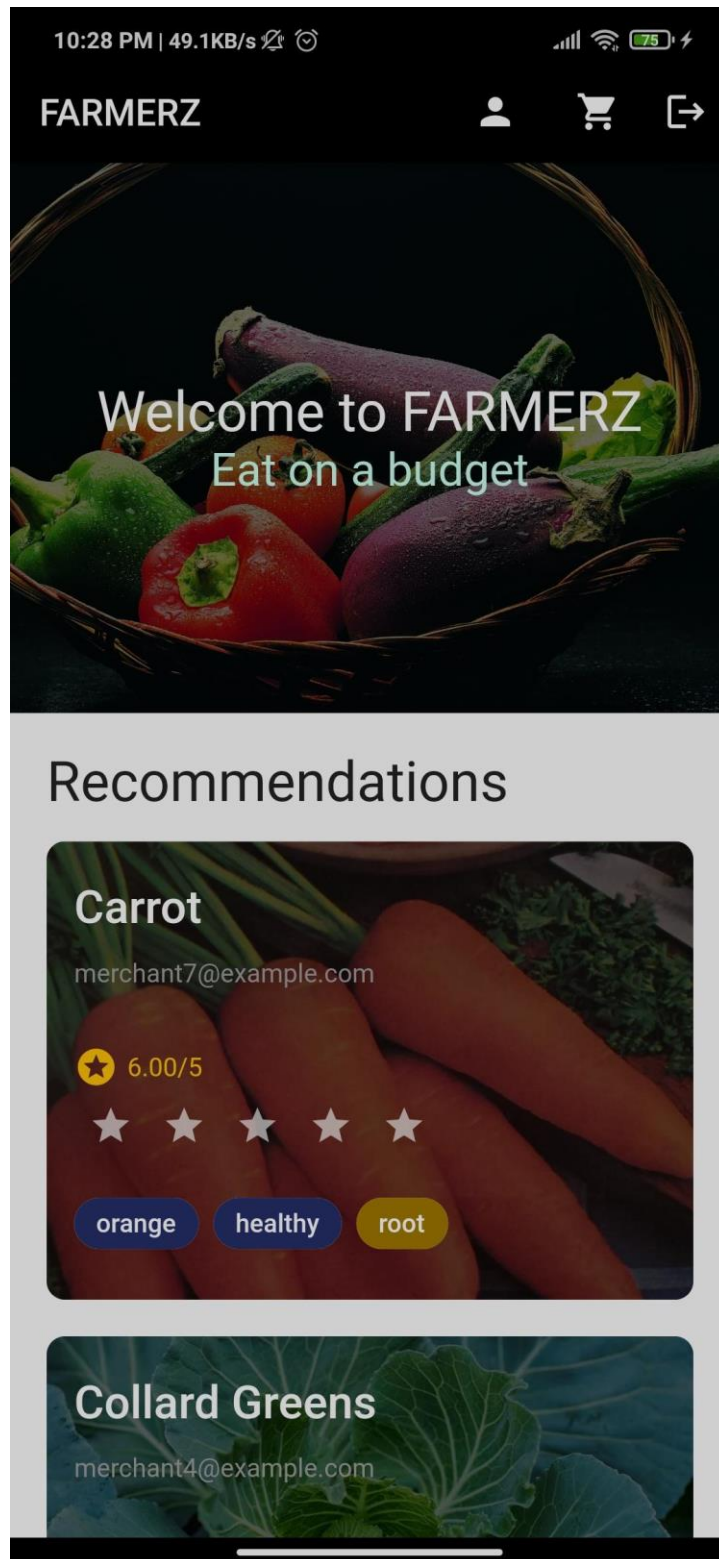


Fig 7.1(o): Home Page (Recommended Products)

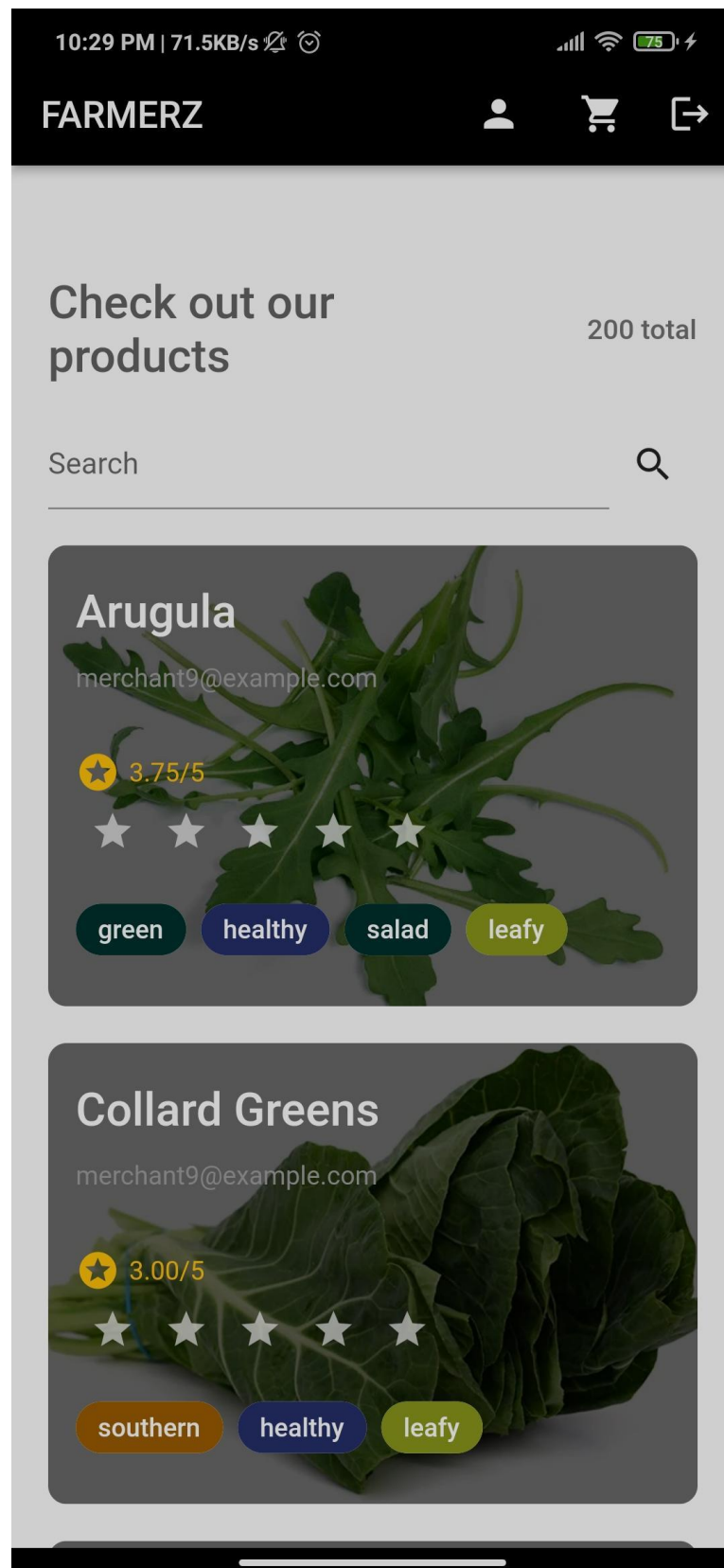
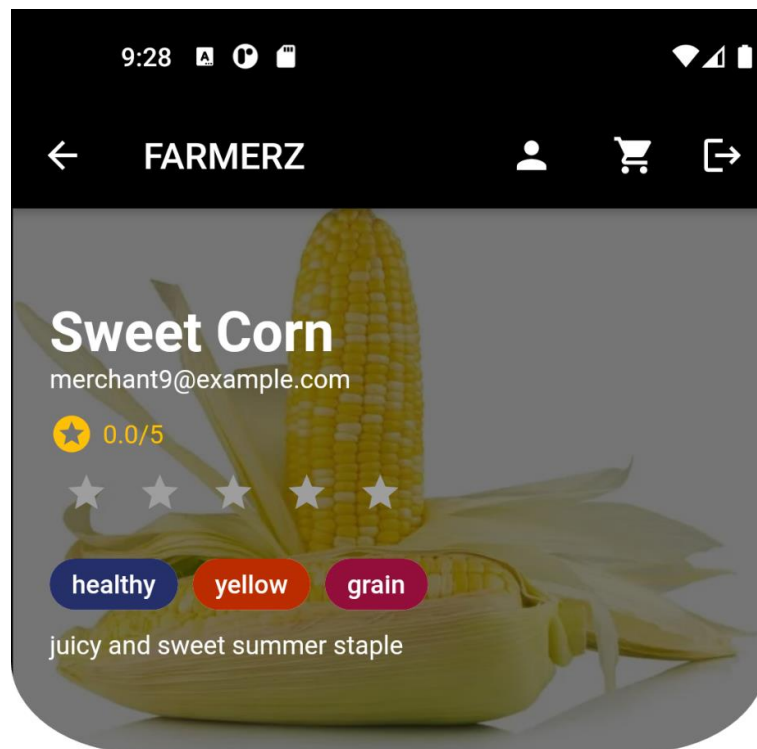


Fig 7.1(p): Home Page (Browse Products)



Variants by vendor

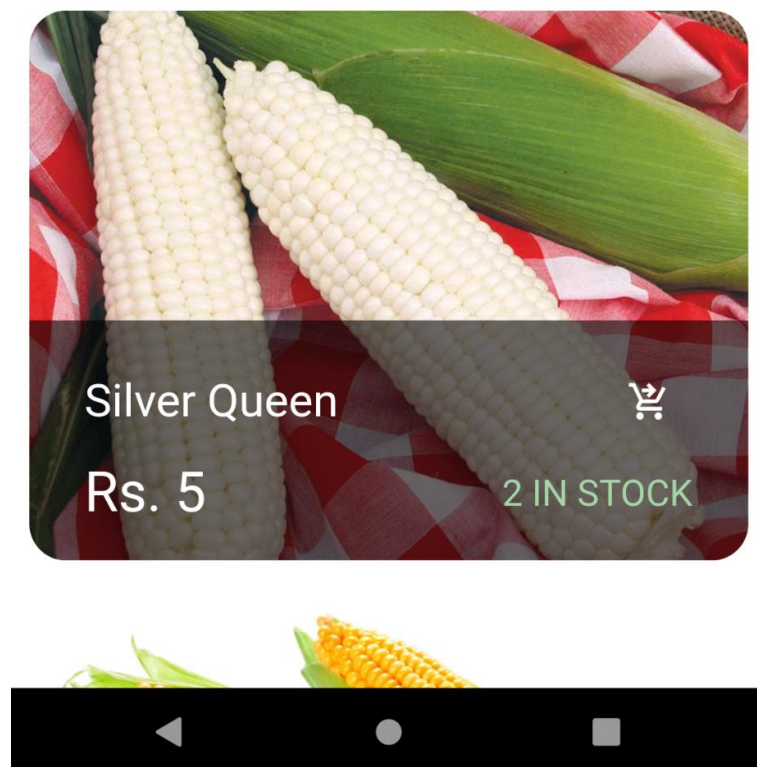


Fig 7.1(q): Product Page

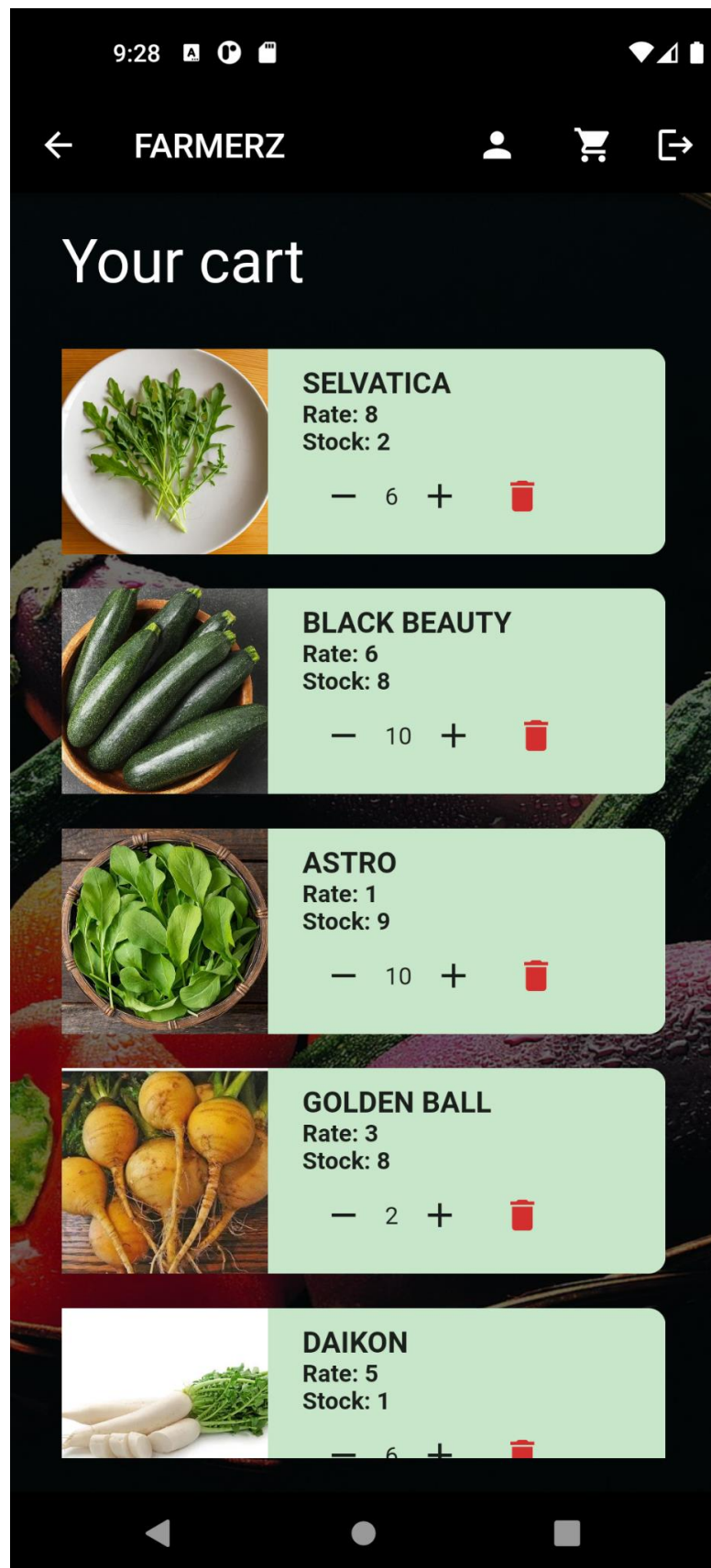


Fig 7.1(r): Cart Page

7.2 Analysis

User-based collaborative filtering is a popular recommendation algorithm that works by finding similar users based on their past behavior or preferences, and then recommending items to a target user based on the items that those similar users have liked or purchased in the past.

Here is a step-by-step explanation of how user-based collaborative filtering algorithm works:

1. **Data Collection:** The algorithm begins by collecting user behavior data, such as purchases, ratings, or views. This data is typically stored in a matrix where each row represents a user, and each column represents an item.
2. **Similarity calculation:** The algorithm then calculates the similarity between users based on their past behavior or preferences. Common similarity metrics used in collaborative filtering include cosine similarity, Pearson correlation, and Jaccard similarity.
3. **Finding similar users:** Once the similarity between users has been calculated, the algorithm identifies the most similar users to the target user, based on their similarity scores.
4. **Generating recommendations:** The algorithm generates a list of recommended items for the target user based on the items that the similar users have liked or purchased in the past. The recommendations are typically ranked by their relevance or predicted rating.
5. **Refining the recommendations:** The algorithm can further refine the recommendations by applying various techniques, such as user-based normalization, to adjust for the differences in the rating scales or behavior patterns of different users.

Overall, user-based collaborative filtering can be a powerful recommendation algorithm for e-commerce sites and other applications, as it relies on the collective behavior and preferences of similar users to generate personalized recommendations for individual users.

For example, in user based approaches, the value of ratings user u gives to item i is calculated as an aggregation of some similar users' rating of the item:

$$r_{u,i} = \text{aggr}_{u' \in U} r_{u',i}$$

Where U denotes the set of top N users that are most similar to user u who rated item i . Some examples of the aggregation function include:

$$r_{u,i} = \frac{1}{N} \sum_{u' \in U} r_{u',i}$$

$$r_{u,i} = k \sum_{u' \in U} \text{simil}(u, u') r_{u',i}$$

The cosine-based approach defines the cosine-similarity between two users x and y as:

$$\text{simil}(x, y) = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \times \|\vec{y}\|} = \frac{\sum_{i \in I_{xy}} r_{x,i} r_{y,i}}{\sqrt{\sum_{i \in I_x} r_{x,i}^2} \sqrt{\sum_{i \in I_y} r_{y,i}^2}}$$

A series of item is recommended to a user based on collaborative filtering algorithm. Here, cosine similarity is used to determine the similarities amongst other users.

Drawbacks:

While collaborative filtering is a powerful and widely used, there are some potential drawbacks to using it in an e-commerce app:

1. **Sparsity:** The user-item matrix used in collaborative filtering can be very sparse, especially in the case of e-commerce apps where users may only purchase a small fraction of the available items. Cosine similarity can be sensitive to sparsity, which means that it may not be an accurate measure of similarity when there are few overlapping items between users.
2. **Cold start problem:** Collaborative filtering suffers from the cold start problem, where new items or users have no or few ratings, making it difficult to generate accurate recommendations. Cosine similarity is no exception to this problem, and additional techniques such as content-based filtering or hybrid filtering may be necessary to overcome it.
3. **Scalability:** Calculating cosine similarity between all pairs of users or items can be computationally expensive, especially as the number of users or items grows. This can be a significant issue in e-commerce apps with large product catalogs and user bases.
4. **Limited user context:** Collaborative filtering algorithms only consider the ratings of items by users, without taking into account any other contextual information such as user demographics, purchase history, or browsing behavior. This can limit the ability of the algorithm to generate personalized and relevant recommendations for individual users.
5. **Popularity bias:** Collaborative filtering algorithms tend to recommend popular items, as they are more likely to be rated by a large number of users and have high cosine similarity scores. This can result in a bias towards popular and mainstream items, which may not be suitable for all users.

Overall, while collaborative filtering is powerful and flexible in recommendation systems, it is not without its limitations and potential drawbacks in the context of e-commerce apps. Careful consideration and additional techniques may be required to mitigate these issues and provide accurate and relevant recommendations for users.

There are several ways to minimize the potential drawbacks of using cosine similarity in an e-commerce app:

1. Sparsity: One way to overcome sparsity is to use matrix factorization techniques, such as singular value decomposition (SVD) or non-negative matrix factorization (NMF), to reduce the dimensionality of the user-item matrix and capture latent factors that explain the rating patterns. These techniques can help to fill in missing ratings and improve the accuracy of cosine similarity.
2. Cold start problem: To overcome the cold start problem, additional techniques such as content-based filtering or hybrid filtering can be used in conjunction with collaborative filtering. Content-based filtering can provide recommendations based on the attributes of items, while hybrid filtering combines multiple recommendation techniques to provide more accurate and diverse recommendations.
3. Scalability: To address scalability issues, parallel processing and distributed computing techniques can be used to speed up the calculation of cosine similarity.
4. Limited user context: To incorporate additional contextual information about users, techniques such as user profiling, demographic segmentation, and preference elicitation can be used to capture user preferences and interests. This information can be combined with collaborative filtering to provide more personalized and relevant recommendations.
5. Popularity bias: To reduce the influence of popularity bias, techniques such as item weighting or novelty filtering can be used to give more weight to less popular or niche items.

CHAPTER 8

CONCLUSION, LIMITATION AND FUTURE ENHANCEMENT

8.1 Conclusion

The purpose of this project was to develop an application that would allow farmers to create a catalog of their agricultural products and sell them via an online platform. Also, consumers should be recommended products based on their rating to the other products.

A user “M” is recommended items based on how they rated other items. The estimated rating of an item “I” is calculated by aggregating the ratings of other users for that item. The aggregation function is a simple weighted average of the ratings. A rating by a user “A” is given a weight in the range of $[0, 1]$. This weight for the rating is calculated by the similarity between user's “M” and “A”. The similarity measure used is the cosine similarity.

8.2 Limitations

8.2.1 Limitations to the app

Features such as sign up, catalog management via app, order viewing etc is not currently possible. That being said, this can be achieved via the admin dashboard panel, and the API.

8.2.2 Limitations in recommendation system

1. Lack of personalization: Cosine similarity does not take into account the individual preferences or behavior of customers. For example, two customers may have similar purchase histories, but one may prefer to buy eco-friendly products while the other may prefer to buy products based on price or brand.
2. Scalability: As the number of products and users increases, the computational complexity of calculating similarity matrix can become very high, making it difficult to scale the system.
3. Cold start problem: Cosine similarity is less effective when there is little or no

data available for new products or customers. This can be a problem in e-commerce sites where new products are frequently added to the catalog or new customers sign up.

8.3 Future Enhancement

1. Content based filtering along with present system.
2. Frontend app completion.
3. Better UI/UX.
4. Integrate Payment System.

REFERENCES

1. Food and Agriculture Organization of the United Nations. (2022). *Nepal at a Glance*, fao.org.[Online].Available at: <https://www.fao.org/nepal/fao-in-nepal/nepal-at-a-glance/en/#:~:text=Agriculture%20sector%20engages%20around%2066,farming%20to%20small%20scale%20enterprises>. [Accessed: 4 December 2022].
2. McKayla Girardin(2022, August. 4). *What Is a Broker?* Forage.[Online]. Available at:<https://www.theforage.com/blog/careers/what-is-a-broker#:~:text=A%20broker%20is%20a%20person,estate%2C%20finance%2C%20and%20trade>. [Accessed: 4 December 2022].
3. Himalayan News Service, “Agitating sugarcane farmer dies,” *Himalayan Times*, Dec.30,2020.Accessed:Dec.05,2022.[Online].Available: <https://thehimalayantimes.com/nepal/agitating-sugarcane-farmer-dies>
4. United Nations, “The 17 Goals,” *United Nation*. <https://sdgs.un.org/goals> (accessed Dec. 06, 2022).
5. S. Rajeswari, K. Suthendran and K. Rajakumar, "A smart agricultural model by integrating IoT, mobile and cloud-based big data analytics," 2017 International Conference on Intelligent Computing and Control (I2C2), 2017, pp. 1-5, doi: 10.1109/I2C2.2017.8321902.
6. K. Spanaki, U. Sivarajah, M. Fakhimi, S. Despoudi, and Z. Irani, “Disruptive technologies in agricultural operations: a systematic review of AI-driven AgriTech research,” *Annals of Operations Research*, Jan. 2021, doi: 10.1007/s10479-020-03922-z.
7. “Ashar 15: Best Nepali Agriculture Apps for Successful Farming in 2020,” *TechSathi*, Jun. 29, 2020. <https://techsathi.com/best-nepali-agriculture-apps>
8. “Geokrishi-Digitalizing Agriculture in Nepal,” *geokrishi.farm*. <https://geokrishi.farm/solution/> (accessed Dec. 11, 2022).
9. R. L. M. Meena B. Jirli, M. Kanwat and N.K., “Mobile Applications for Agriculture and Allied Sector,”*www.ijcmas.com*. <https://www.ijcmas.com/abstractview.php?ID=6659&vol=7-2-2018&SNo=281> (accessed Dec. 11, 2022).

10. Z. Yang, B. Wu, K. Zheng, X. Wang and L. Lei, "A Survey of Collaborative Filtering-Based Recommender Systems for Mobile Internet Applications," in *IEEE Access*, vol. 4, pp. 3273-3287, 2016, doi: 10.1109/ACCESS.2016.2573314.
11. H. K. Flora, X. Wang, and S. V.Chande, "An Investigation into Mobile Application Development Processes: Challenges and Best Practices," *International Journal of Modern Education and Computer Science*, vol. 6, no. 6, pp. 1–9, Jun. 2014, doi: 10.5815/ijmecs.2014.06.01.