An-Najah National University

Faculty of Engineering & Information Technology
Department of Computer Engineering

Graduation Project I

JAFFA

Students' names:
Mai Shelbayeh
Dema khalili

Supervisor:
Dr. Emad Natsheh

Presented in partial fulfilment of the requirements for
Bachelor Degree in Computer Engineering

Date: Jan, 2024

Disclaimer

This report, authored student Mai Shelbayeh and Dema Khalili from the Engineering Department at the Faculty of Engineering, An-Najah National University, remains unchanged expect for editorial revisions following assessment. It is important to note that the report may include language and content errors. The opinions presented, along with any conclusions and suggestions are entirely those of the students. An Najah National University disclaims and responsibility or liability for the potential outcomes of utilizing this report for purposes other than originally intended commission

Contents

List of Figures

Abstract

Factories often encounter marketing challenges as consumers tend to make their purchases primarily through supermarkets rather than directly from the factories themselves. Furthermore, marketing products to supermarkets can be a complex endeavor. In response to these challenges, we have developed a mobile application tailored to cater to three distinct user roles, specifically for selling coffee, spices, and various other food packages

The first user assumes the role of an administrator responsible for managing incoming requests. This role encompasses tasks such as accepting or rejecting orders and coordinating with the appropriate delivery personnel to ensure prompt and efficient deliveries. Additionally, administrators have access to a suite of tools for order tracking, profit monitoring, and identifying top-selling products.

The second user category encompasses a diverse group, including restaurant owners, grocery store proprietors, and individual consumers. Our application boasts a robust recommendation system that empowers users to share their product experiences, thereby strengthening our community-driven platform. Users also have the convenience of direct communication with the administrator for any inquiries or assistance they may require

The third user role is that of the delivery personnel, who receive notifications
from the administrator containing order details. Once a delivery is successfully
completed, the administrator is promptly notified of the successful transaction,
ensuring

A fourth user role involves delivery drivers who receive notifications, optimize routes
using the algorithm, provide real-time tracking, verify successful deliveries, and communicate with administrators. This comprehensive system aims to streamline the entire process from order placement to delivery, addressing marketing challenges and enhancing customer experience through efficient logistics management

Chapter 1

Introduction

In response to the prevalent marketing challenges faced by factories, particularly in reaching consumers who predominantly make purchases through supermarkets, we have developed a pioneering mobile application. Tailored to the specific needs of the food industry, our application is designed to facilitate seamless transactions between factories, administrators, diverse buyers, and dedicated delivery personnel. With a focus on coffee, spices, and various food packages, the application introduces a novel approach to address the complexities of marketing products to supermarkets.

At the core of our solution are three distinct user roles, each playing a crucial part in streamlining the entire process from order placement to delivery. The first role is that of the administrator, tasked with efficiently managing incoming requests, coordinating deliveries, and utilizing a suite of tools for order tracking and profit monitoring. The second role caters to a diverse user group, including restaurant owners, grocery store proprietors, and individual consumers, offering them a community-driven platform with a robust recommendation system and direct communication channels with administrators.

The third user role involves dedicated delivery personnel who, upon receiving notifications from administrators, ensure prompt and efficient order deliveries. This role is supported by a fourth user role of delivery drivers who optimize routes, provide real-time tracking, and verify successful deliveries, contributing to an enhanced logistics management system.

By integrating these user roles and functionalities, our comprehensive mobile application not only addresses marketing challenges faced by factories but also aims to elevate the overall customer experience through efficient and transparent logistics management. Through this innovative solution, we strive to bridge the gap between factories and consumers, empowering businesses to thrive in a supermarket-centric market landsca

Chapter 2

Constraints, Standards, and Earlier course work

Our project encountered several key limitations, primarily stemming from the challenges we faced in implementing a recommendation system utilizing technology such as the k-nearest neighbors (KNN) matrix. The lack of readily available information on the intricacies of this implementation posed a significant hurdle that required careful research and experimentation to overcome.

Additionally, we aimed to make our application accessible on both Android and web devices, prompting us to leverage cross-platform tools for development. This decision introduced its own set of complexities, necessitating a strategic approach to ensure optimal performance and user experience across different platforms.

Another constraint we grappled with was the integration of real-time chat functionality. While our application relied on a Mongo database, a non-relational database that lacked inherent support for real-time chat data, we addressed this limitation by seamlessly incorporating Firebase into our system. This strategic integration allowed us to augment our application with robust real-time chat features, enhancing user interaction and engagement.

Drawing from our earlier coursework experiences, we applied methods learned from the Advanced Software course, which proved instrumental in establishing a Node.js server. This server played a pivotal role in facilitating seamless communication between our application and backend processes through HTTP requests. The knowledge gained from this course not only contributed to the core functionality of our project but also played a crucial role in adhering to software engineering standards. It guided us in writing well-organized and clean code, ensuring a standardized approach that aligns with best practices in software development.

In summary, while our project faced challenges related to implementing advanced features, cross-platform compatibility, and real-time functionalities, our strategic problem-solving, research efforts, and the

application of coursework knowledge enabled us to overcome these limitations and deliver a robust and standardized solution.

Chapter 3

Literature Review

The evolving era leads to versatile activities. From the way of thinking to the methods in keeping up the activities that are supported by digitalization. The implementation of digitalization provides impacts on several sectors, especially in business sector. Both electronic information technology and economic globalization have been developed rapidly that put e-commerce's role to predominate the economic activities in many countries [1]. One of the activities of business process is web-based purchasing, by buying the goods are posted by the company, which sell and send the products through the internet [2].

To align the recent technology era that supported by the internet, several innovations are merged to form a method, called as online shopping. It utilizes a platform that specially designed to support business process. Several things that should be considered-in e-commerce, among others is the requirements of development by adopting the effective risk management technique in maintaining certain aspects, such as acritificial

intelligence, financial information, trade secret among companies over information source that its misused regarding to online activities will lead to serious threat [3].

It requires many researches for initiating new online business as well as active data, technology-based integration, financial platform, and volume of products that could be retrieved [4]. Recently, there are numbers of researches in the form of proceeding and journals related to e-commerce implementation. The researches include factors, such as sample, population, sector and area that could be directed to the research gap. The focus towards existing gap could support future researches as well as to maintain the previous research. The research was taken using systematic literature review methods to see the gaps that occurred. This looks at the latest research based on perspective in the global scope. Looking at the perspective compared globally can create an opportunity for an update on business processes with the aim of knowing the achievement of business segmentation and factors that affect the global scope. Therefore, the improvement for problems will be carried out as well as sustainability of research

Chapter 4

Methodology

4.1 Technologies

Developing mobile and web applications involves the use of various tools and programming languages to create and enhance the project. In this chapter, we'll outline the technologies and tools utilized in the development of our project.

4.1.1 Database

For storing and managing application data with an emphasis on flexibility and scalability, we opted for a NoSQL approach using MongoDB as our database solution. MongoDB, a widely-used open-source document-oriented database, aligns well with the dynamic nature of our application data, allowing us to store and retrieve information in a JSON-like format. The decision to choose MongoDB is rooted in its ability to handle large volumes of unstructured data efficiently, making it suitable for our application's evolving data needs. MongoDB's support for dynamic schemas and its capacity to manage relationships between entities provide the necessary flexibility as our application grows and data complexity increases. To streamline the management of our MongoDB database, we leveraged tools like MongoDB Compass for intuitive GUI-based interactions and efficient monitoring. Additionally, we utilized the Mongoose library for Node.js to enhance the interaction with MongoDB by providing a schema-based solution for modeling application data. This combination of MongoDB and supporting tools aligns with our goal of maintaining a responsive and adaptable database infrastructure for our application

4.1.1 Server Side

In managing the server-side of our application, we harnessed the capabilities of Node.js, a

server-side JavaScript runtime, coupled with an online MongoDB server. Node.js proved to

be a fitting choice for our needs, offering a non-blocking, event-driven architecture that

facilitates seamless handling of asynchronous tasks.

4.1.2 API

Within our application, we seamlessly integrated two AI functionalities: a simulated annealing algorithm for optimizing the best paths for drivers and a recommendation system for products. To facilitate communication between our mobile application and the Python-based project, we developed a RESTful Flask API.

An Application Programming Interface (API) is essentially a program designed to interact with other programs. It serves as a bridge, enabling the exchange of data between different components of a system or between different programs.

Choosing Flask for our API was a natural fit, given our reliance on Python for implementing the AI models. Flask, a web framework for Python, offers a range of functions and services specifically tailored for building web applications and handling HTTP requests. Since our AI models were coded in Python, utilizing a Flask API provided a seamless connection between our Python logic and the Flutter application, enabling effective communication and integration of the AI features within the mobile application.

4.1.3 UI Design

In crafting the user interface, our approach was guided by extensive research to adhere to the best standards for developing an appealing and user-friendly e-commerce mobile and web application. To ensure a consistent experience across various platforms, we employed cross-platform mobile development tools, enabling the creation of a unified app for both Android and iOS devices simultaneously.

For the design and implementation of the user interface, we turned to Flutter, an open-source software development kit developed by Google. Flutter excels in building cross-platform applications for mobile, desktop, and the web, all from a single codebase. The programming language utilized by Flutter is Dart.

Flutter distinguishes itself by offering a broad spectrum of fully customizable UI components, contributing to the creation of a visually expressive and adaptable user interface. Moreover, Flutter's widgets seamlessly integrate

essential platform elements such as icons, fonts, scrolling, navigation, and more. A notable efficiency factor is Flutter's compilation of code to native machine code using Dart's native compilers, further enhancing the performance of our application.

4.1.4 Chat and Notifications

A pivotal functionality within our application is the real-time chat, enabling customers to communicate with the shop for inquiries. To bring this feature to life, we seamlessly integrated the Firebase platform with Flutter.

Firebase serves as a Backend-as-a-Service (BaaS) app development platform, offering a suite of hosted backend services, including a real-time database, cloud storage, authentication, and machine learning. Leveraging the compatibility and support for Flutter, we incorporated various Firebase features to enhance the functionality of our application.

In addition to the real-time chat, we implemented a notification system using Firebase Cloud Messaging (FCM). FCM, a versatile cross-platform messaging tool, allows for the reliable and cost-effective delivery of notification messages. This integration further contributes to a seamless and responsive communication experience within our application.

4.2 AI Features

In this part, we are going to explain in detail the AI algorithms we used to build two of the biggest functionalities. The Arabic text sentiment analysis and the products recommendation system.

4.2.1 Recommendation System

The recommendation system plays a pivotal role in JAFFA, enhancing user engagement by offering personalized suggestions for products within our online shopping community.

(A) Definition of a Recommendation System:
A recommendation system predicts user preferences based on their choices, commonly employed in social media and marketing platforms. Two primary approaches guide recommendation systems: Collaborative Filtering (CF) and Content-Based Filtering. JAFFA utilizes a collaborative filtering recommendation system.

(B) Collaborative Filtering:
Collaborative Filtering assumes that users with similar preferences will like similar items. For JAFFA, we implemented a memory-based approach using k-

NN (k-Nearest Neighbors) to calculate predictions based on past user interactions. We filled missing values in a product-customer matrix and used Cosine similarity. The model's efficiency, measured by root mean square error (RMSE), was 23.86%, which is acceptable given the continuously growing dataset.

Limitations include the cold-start problem for new items or users. To address this, we employed a user cold-start strategy by prompting new users to follow at least one topic or genre. An item cold-start is mitigated by suggesting new items based on topics followed by users.

We also implemented a model-based approach using matrix factorization, specifically Singular Value Decomposition (SVD) with TruncatedSVD from sklearn. This method reduces the dimensions of the user-item matrix, providing accurate results even with sparse data in a continuously growing dataset.

(B) Combating the shortcomings of collaborative filtering:
The cold-start problem is addressed through user cold-start and item cold-start strategies. Newly registered users are encouraged to follow topics, aiding the recommendation system in suggesting relevant items. For item cold-start, topics followed by users are utilized to recommend new items. This comprehensive approach ensures effective personalized recommendations while overcoming the challenges of the cold-start problem.

4.2.2 Simulated Annealing Algorithm

The Simulated Annealing algorithm stands as a pivotal element in JAFFA, contributing a strategic layer to the application's intelligence.

(A) Definition of Simulated Annealing Algorithm:
Simulated Annealing is a heuristic optimization algorithm inspired by the annealing process in metallurgy. It is utilized to find a close-to-optimal solution for a complex problem by navigating through a solution space.

Key Features of Simulated Annealing:
• Exploration and Exploitation:
Simulated Annealing balances the exploration of new solutions and the exploitation of existing solutions, allowing for a dynamic approach to problem-solving
(B) Implementation in JAFFA:
For JAFFA, we integrated the Simulated Annealing algorithm to optimize the best paths for drivers within the application. The algorithm considers factors such as distance, traffic conditions, and delivery schedules to provide efficient and context-aware routing.

The Simulated Annealing approach aligns with JAFFA's commitment to enhancing user experience by ensuring that delivery routes are not only time-efficient but also adaptable to real-world conditions. This strategic use of AI contributes to the overall effectiveness and user satisfaction within the application.

4.3 Application Architecture

Our application is organized into multiple components, adhering to the client-server architecture pattern. This section delves into the interaction dynamics among these components.

Client-server architecture is a prominent model for modeling mobile application structures. Within our mobile application, clients maintain constant or partial connections with the server. The client component establishes connections with three distinct servers, each serving a specific purpose. The first is the database server, facilitating data access. The second is the Flask API server, responsible for delivering AI model results. The third server is dedicated to Firebase, managing chat messaging functionality and handling notifications – both sending and receiving. This structured architecture ensures efficient communication and coordination between the client and servers for various application functionalities.

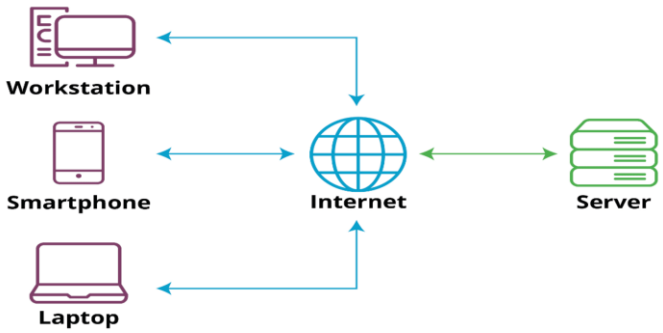

Figure 1 Application Architecture

4.3.1 Database server

At the core of our application's structure is the MongoDB server hosted online, with the server-side logic implemented in Node.js. This integral component manages all incoming requests and responses related to database interactions.

Operational on an online MongoDB server, the Node.js server adeptly handles GET and POST HTTP requests initiated by the client. It efficiently queries the MongoDB database, hosted online, based on the specific nature of each request. Following the execution of the database query, the server meticulously sends back the requested data in JSON format, ensuring seamless communication between the client and the online MongoDB server.

Regarding the database structure, our MongoDB implementation maintains simplicity with sparse relationships between collections. Core entities essential for data representation are encapsulated within these collections. Notably, image files are stored directly in the MongoDB database as Binary Large Objects (BLOBs) or via GridFS for efficient handling of large files. This streamlined approach optimizes the storage and retrieval processes, eliminating the need for a separate PHP server for image file storage.

4.3.2 Firebase

Exclusively leveraging Firebase services, we seamlessly integrated chat messaging, authentication, and app notifications into our application. Firebase, known for its easy and flexible integration with Flutter projects, ensures a real-time connection to Firebase Cloud, enabling the instant reception of new messages and notifications across different client devices.

For authentication purposes, we opted for email/password authentication. This authentication method, along with its associated functionalities, facilitates password resets for users who may forget their credentials. Additionally, the application sends a verification link to the user's email address during the registration process to ensure the validity of the provided email.

In managing chat functionality and notifications, we registered the project on Firebase Cloud and utilized Firestore, a NoSQL database, to

store chat data, including messages and user information. To streamline communication with Firebase Cloud, we incorporated Firebase and Firestore libraries into our Flutter project, allowing for efficient requests and interactions with the Firebase Cloud services. This comprehensive integration enhances the real-time nature of our chat system and ensures seamless communication and authentication processes within the

4.3.3 Flutter application

The core of our application structure revolves around the mobile application, which consistently sends requests to the database, the API, and the Firebase Cloud. All these requests are initiated and handled on the client-side, with minimal processing demands. This design choice ensures that the mobile application remains lightweight and responsive, enabling swift interactions and a seamless user experience.

4.3.4 Flask API

Flask, a Python web framework, excels in managing HTTP requests and serves as an ideal tool for constructing REST APIs.

In our application, we harnessed Flask to create a REST API, offering an interface to facilitate the development of AI models. This API plays a crucial role in obtaining recommended items or determining the best paths for drive orders, delivering the results in JSON format.

The incorporation of an API into our application holds significance for several reasons. Foremost among them is the reduction of computation overhead on the user-side, contributing to a faster and more streamlined application with a smaller footprint. Additionally, utilizing an API for model interaction via HTTP requests minimizes delay times, thanks to the asynchronous nature of these requests. Below is a concise overview of the functionalities encapsulated within our Flask REST

Route	verb	Description
`/get_recommendations`	POST	Get Recommendation items
`/optimize_delivery_route`	POST	Get best path

4.4 Application Design

4.4.1 Admin

(A) Pages (WEB)

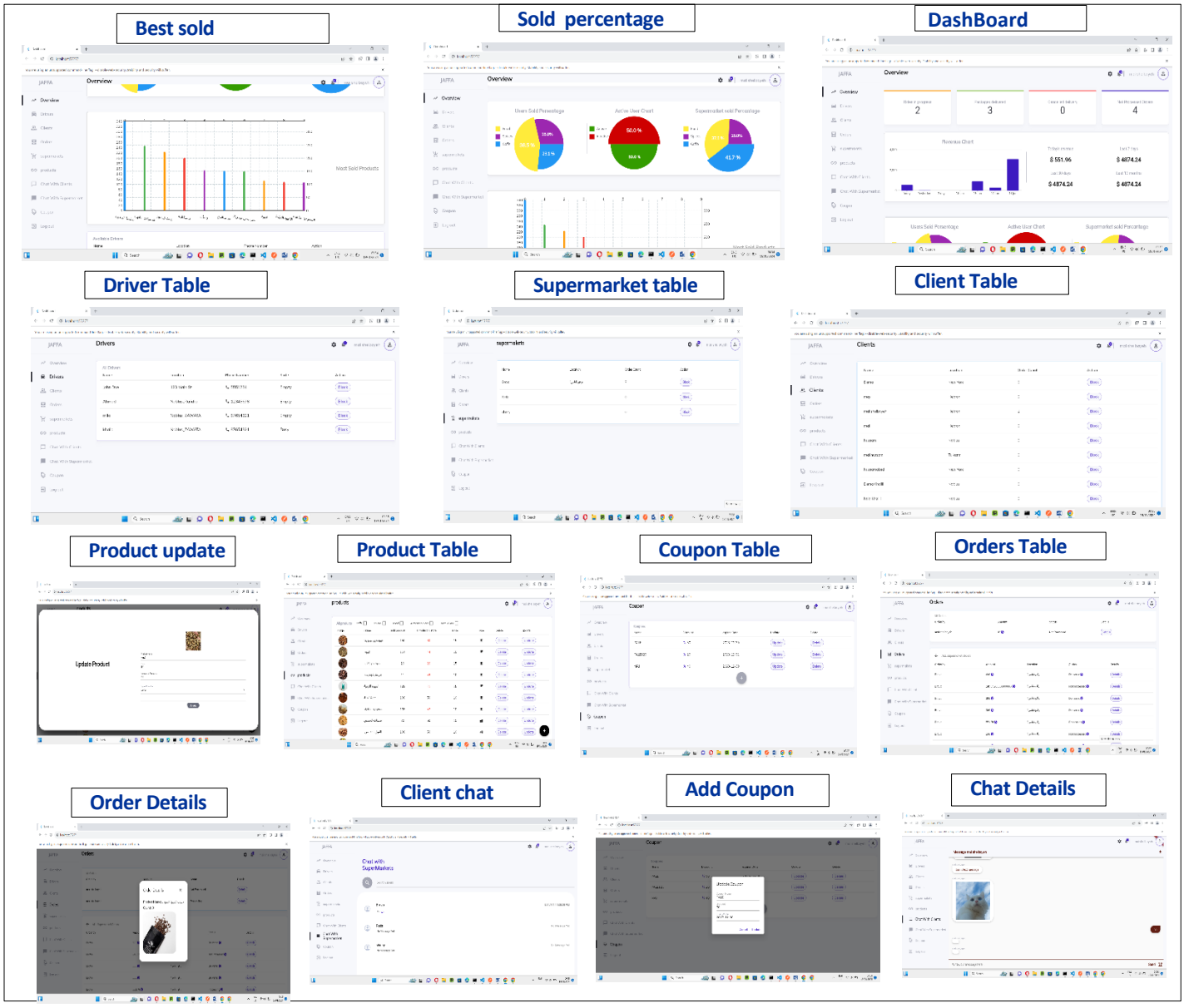

Figure 2 Admin pages

(B) Admin Feature:

In our dynamic web application, the administrator enjoys a comprehensive static analysis tool that provides a clear overview of revenue metrics. This includes detailed insights into daily, weekly, and monthly earnings, enabling effective tracking and informed decision-making.

Visualizing essential data, our application incorporates intuitive pie charts to represent the sold percentage for both supermarkets and client orders. This graphical representation not only simplifies data comprehension but also enhances the user experience.

For user engagement, the platform features a dynamic pie chart highlighting active users within the application. This interactive visualization offers a quick glance at the user base, fostering a sense of community and connection.

Efficient sales management is facilitated through a dedicated chart showcasing the best-selling products. This data-driven approach aids administrators in identifying popular items and optimizing inventory accordingly.

To facilitate seamless communication, our application integrates a chat functionality, enabling clients to interact with supermarkets. This feature enhances user engagement, providing a platform for inquiries, feedback, and personalized interactions.

Real-time notifications keep both clients and supermarkets informed about crucial updates, ensuring a streamlined and responsive user experience. Whether it's order confirmations, promotions, or important announcements, notifications keep users engaged and informed.

Monitoring order dynamics is made simple through a concise summary, showcasing the number of delivered, canceled, and orders in processing. This snapshot enables administrators to stay on top of order management efficiently.

Administrators wield control over various entities, including products, orders, clients, supermarkets, coupons, and drivers. Through an intuitive interface, they can seamlessly add, delete, and update information, ensuring robust control and customization capabilities.

Incorporating these features into our application not only enhances administrative control but also elevates the user experience, making it a powerful tool for efficient business management and user engagement

4.4.2 Supermarket

(A) **Super market Pages**

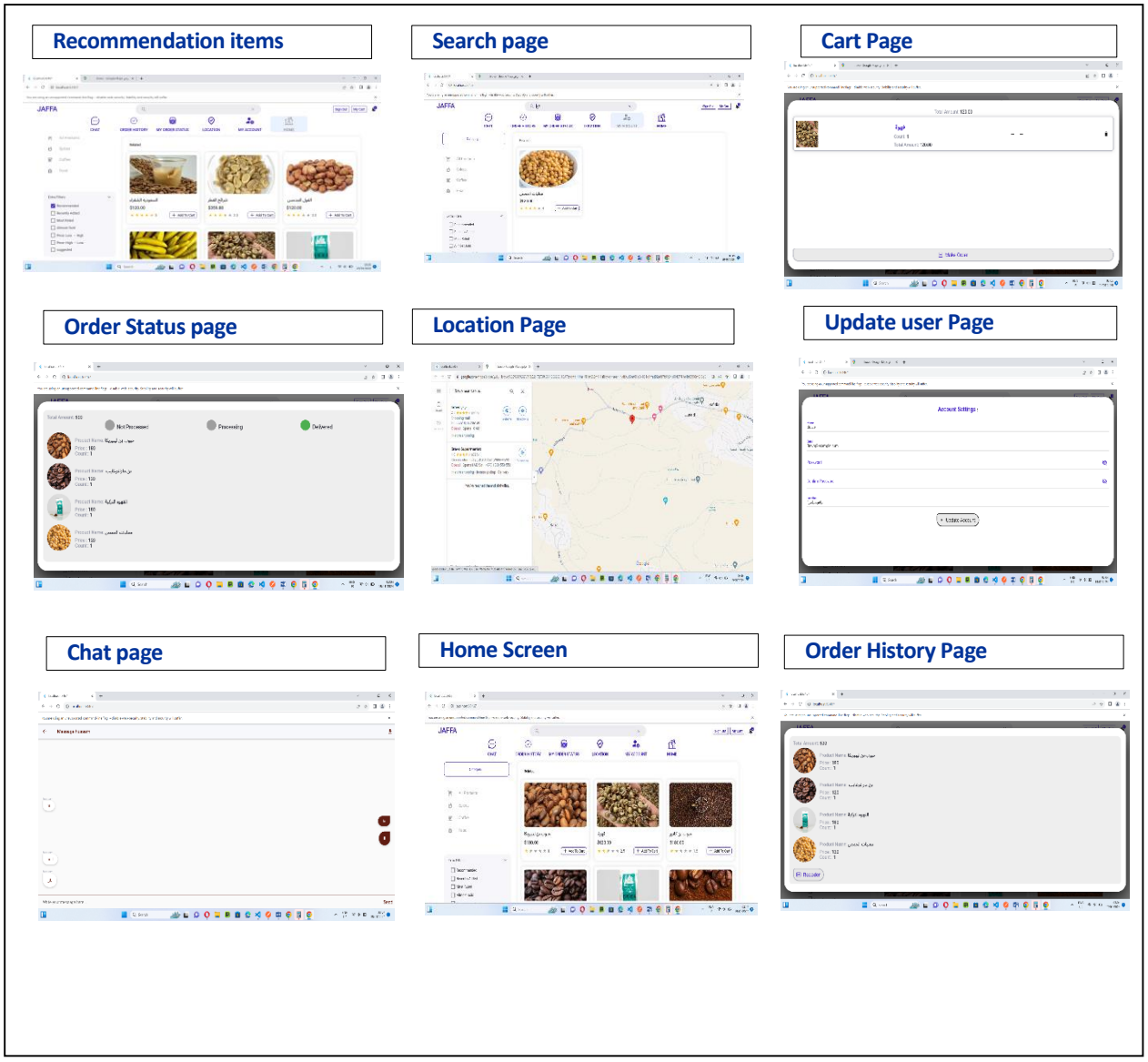

Figure 3 SuperMArket pages

(B) Super market Feature:

On the supermarket side, our application offers a user-friendly interface that empowers clients with a range of convenient features to enhance their shopping experience. Clients can effortlessly browse and filter items using various criteria such as price, best-sold products, and specific categories. This intuitive filtering system ensures a personalized and efficient shopping journey.

To keep clients well-informed, our application provides a comprehensive order history, allowing them to track and review their previous purchases. Additionally, clients can monitor the real-time status of their current orders, ensuring transparency and peace of mind throughout the delivery process.

For added convenience, a dedicated shopping cart allows clients to specify quantities and place orders seamlessly. The cart feature streamlines the ordering process, providing a centralized space for managing selected items.

Clients have the flexibility to customize their experience through account settings, enabling them to manage preferences, update personal information, and tailor the app to their liking. The integration of Google Maps enhances user convenience, allowing clients to pinpoint their location effortlessly.

Our application fosters communication between clients and supermarkets through an integrated chat functionality. This feature enables clients to address concerns, ask questions, or seek assistance, creating a direct line of communication for improved customer service.

Harnessing the power of KNN matrix calculations, our application goes beyond simple product recommendations. Clients receive personalized suggestions based on product ratings from other users, enhancing their exploration of new and highly-rated items.

Keeping clients in the loop, our application ensures they receive timely notifications from both administrators and delivery services. Whether it's updates on promotions, order confirmations, or delivery progress, clients stay informed throughout their entire shopping journey.

By integrating these features, our application transforms the supermarket-client interaction into a seamless and enjoyable experience, fostering user engagement and satisfaction.

4.5 Mobile Application design

4.5.1 User

(A) User pages

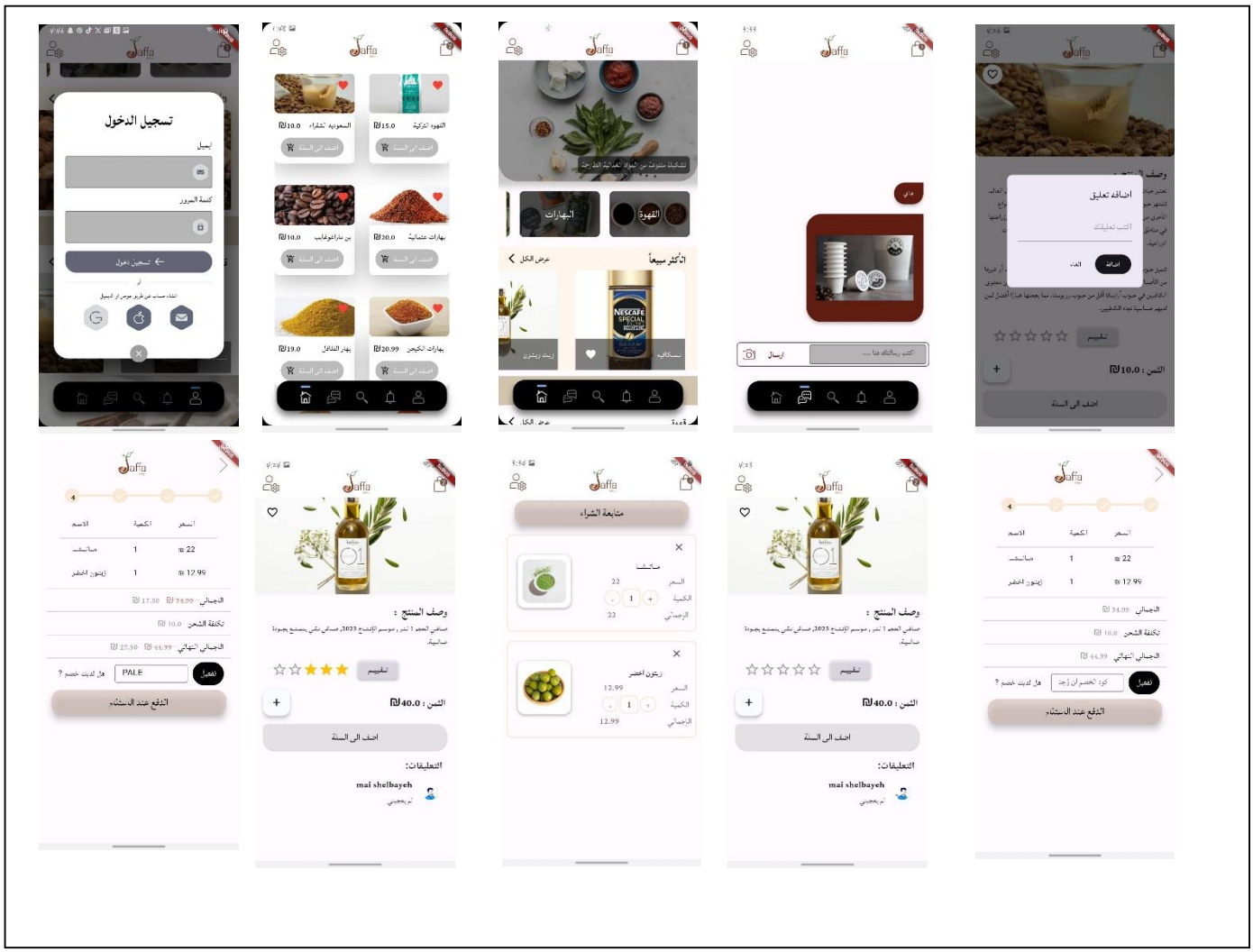

Figure 4 users pages

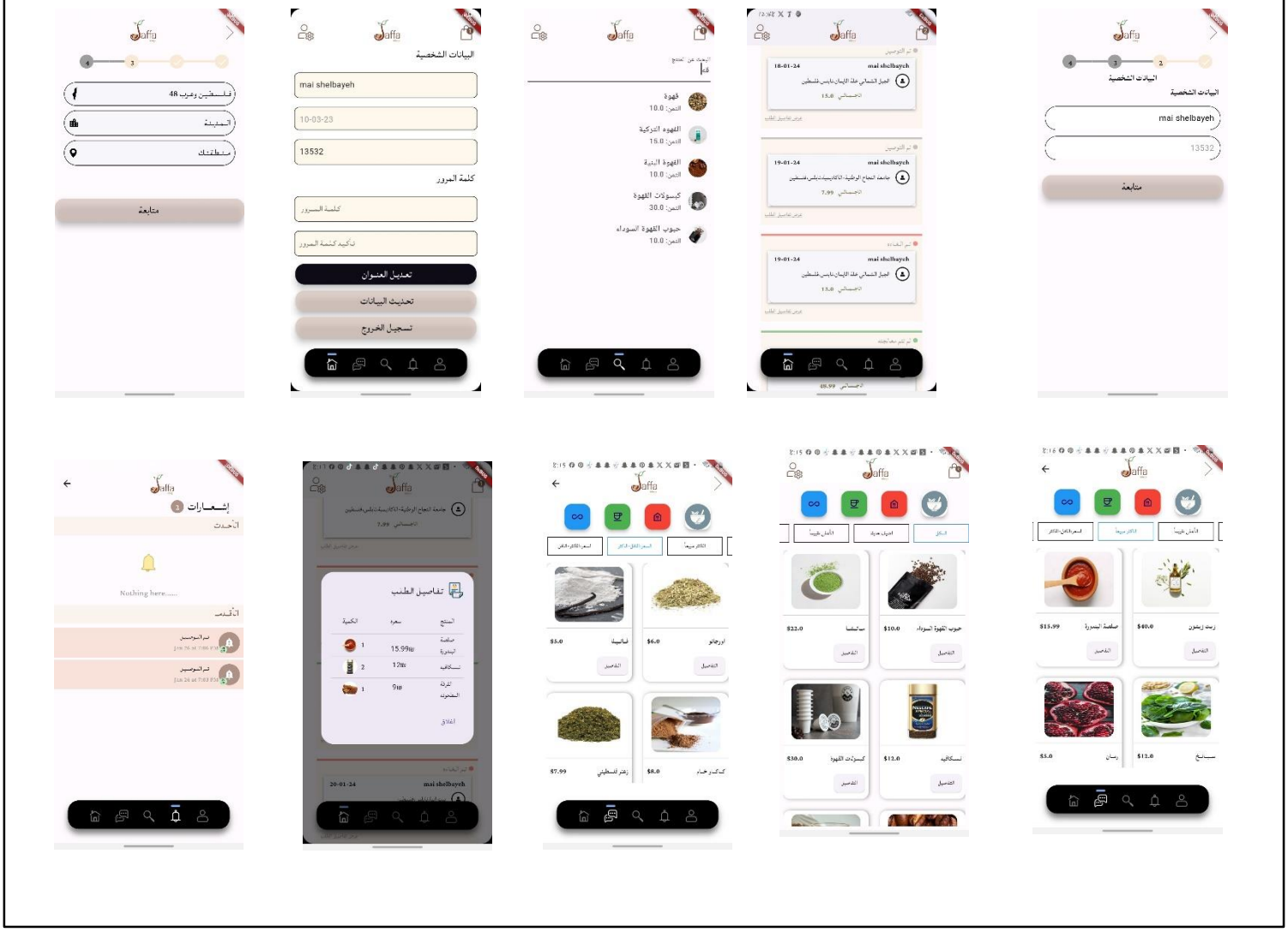

Figure 5 use pages

(B) User Feature

- Home Page:

Showcase categories, best-selling products, and recommended items.
Inspire users with encouraging sentences to enhance their experience.

- Search Page:

Enables users to easily search for specific products.

- Chat Page with Admin:

Provides a direct communication channel for users to address any concerns or problems.

- Product Details Page:

Allows users to view comprehensive information about each product.
Enables users to rate items.
Facilitates the addition of comments, showcasing a community of user opinions.

- Like and Save Products:

Users can express appreciation by 'liking' products.
The application stores liked products and suggests recipes based on these preferences.

- Order Status Page:

Keeps users informed about the current status of their orders.
Coupon Integration:

Users can enter coupons for discounts on selected products.

- Notification Page:

Sends timely updates from the admin, including information about liked products and delivery status.
These features collectively create a user-friendly and engaging environment, offering a seamless shopping experience with personalized recommendations, interactive communication, and convenient order tracking.

4.5.2 Delivery

(A) Delivery Pages

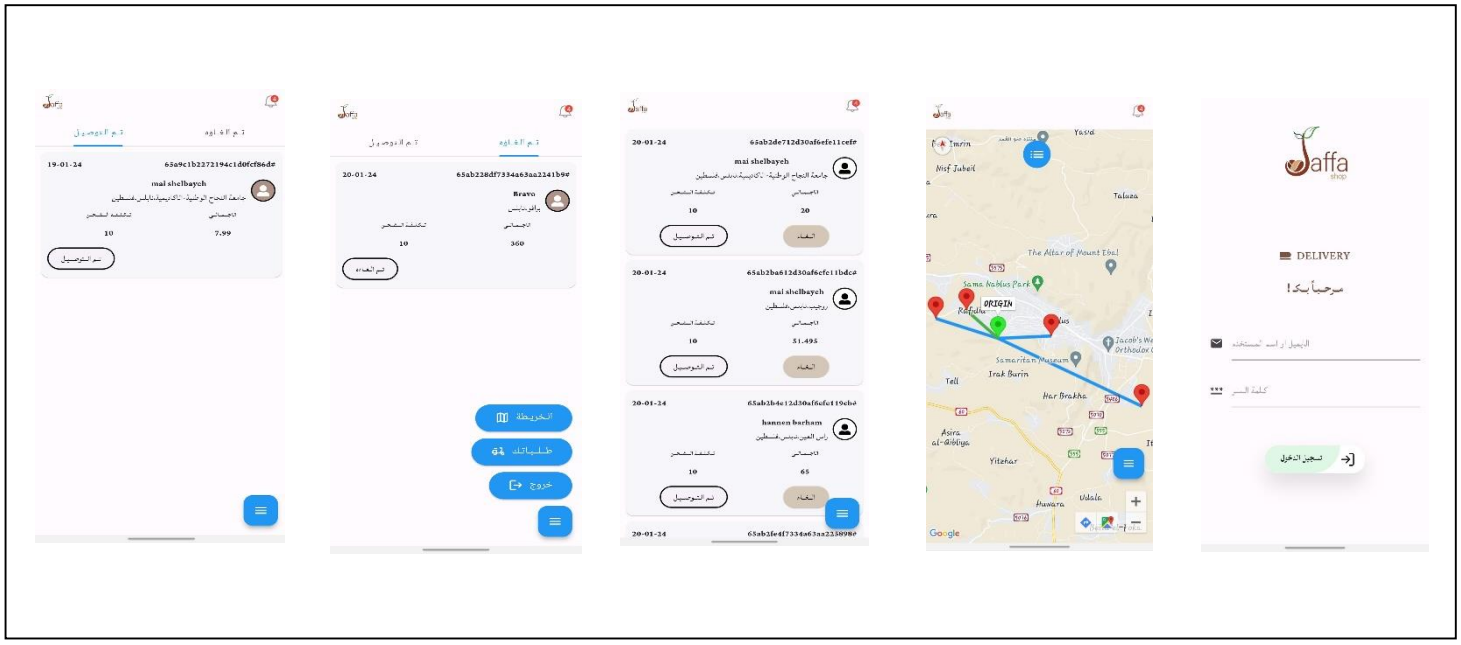

Figure 6 Delivery Pages

(B) Delivery Features

Here's a polished description of the delivery-related features:

- Order Overview:

Display all orders assigned to a specific delivery.
Provide a comprehensive view of the pending deliveries for efficient management.

- Optimized Route Planning:

Utilize simulated annealing for intelligent path optimization, ensuring the most efficient delivery sequence.
Interactive Map Integration:

Visualize the optimized delivery path on Google Maps for real-time navigation.

- Order Status Management:

Empower the delivery personnel to update the status of orders, marking them as 'delivered' or 'canceled' based on the current status.
Order History:

Maintain a complete history of all delivered and canceled orders for reference and analysis.

- Push Notifications:

Implement push notifications for seamless communication between delivery personnel, clients, and the admin.
Keep both clients and admin informed about important updates related to the delivery process.
These features enhance the delivery process, ensuring optimal route planning, real-time tracking, and effective communication for a streamlined and transparent delivery experience.

4.6 application Schema

4.6.1 Web Schema

In our web application, we adopted a color palette featuring deep blue, black, grey, and white to establish a compelling visual contrast throughout the design. This deliberate choice allowed us to highlight specific functionalities, ensuring a

clear and intuitive user experience. The use of gradient blue tones serves a dual purpose, drawing attention to key elements and elegantly filling expansive spaces within our application. This thoughtful color scheme not only enhances the overall aesthetics but also contributes to user clarity and engagement.

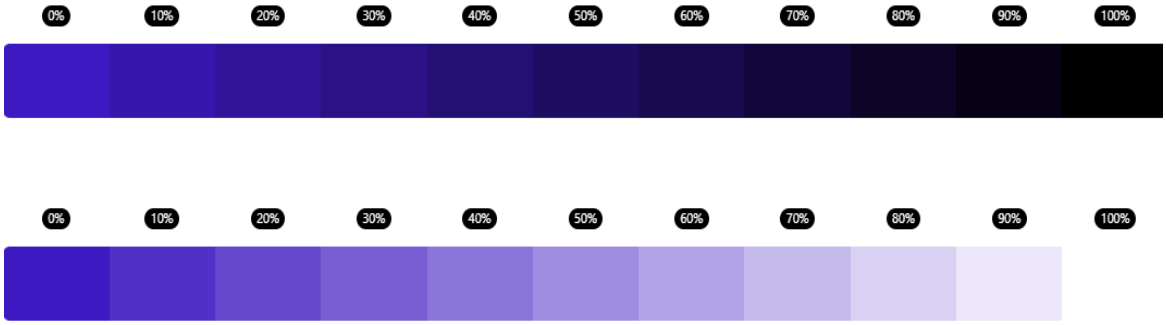

Figure 7 Web color Schema

4.6.2 Mobile Schema

In our mobile application, we have meticulously chosen a color scheme that resonates with the essence of our offerings—coffee and food. The predominant use of brown, accompanied by its various gradations, serves as a visual representation of the warm and rich qualities inherent in our products. Brown, a color associated with earthiness and comfort, is a symbolic nod to the comforting and wholesome experience we aim to provide to our users.

The interplay of white and black within our color palette adds sophistication and contrast, creating a visually appealing and balanced design. White introduces a sense of purity and cleanliness, reflecting our commitment to

quality and freshness in every aspect of our offerings. Meanwhile, black, with its bold and timeless presence, adds a touch of elegance and modernity, aligning with our commitment to delivering a contemporary and stylish mobile experience.

Together, these carefully selected colors not only harmonize with the theme of coffee and food but also contribute to an immersive and aesthetically pleasing visual environment for our users. The combination of brown, white, and black creates a seamless and inviting user interface that enhances the overall experience of exploring and engaging with our mobile application.

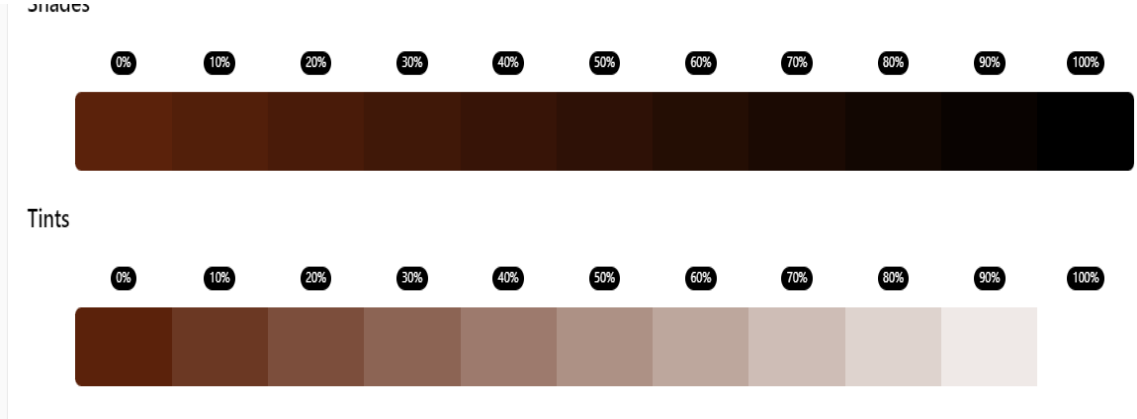

Figure 8 Mobile color schema

4.7 Software Engineering Principles

In our project, prioritizing the development of a robust, scalable application led us to adhere to essential software engineering principles throughout the implementation process.

Minimizing Coupling:
To ensure maintainability and flexibility, we took deliberate steps to minimize code coupling. One notable example is the implementation of the notification panel, which exists as a separate object. When requiring any type of notification, we instantiate an object from the "Notification" class with the necessary parameters. Similarly, we separated the "Order" class from the "ProductItem" class to reduce code dependencies and isolate order-related functionalities. This modular approach extends to elements like text fields in registration and login pages, implemented as independent widgets, and item cards displaying product information and ratings.

Promoting High Cohesion:
We prioritized high cohesion by ensuring that functions within each class are closely related to the primary purpose of that class. For instance, the "TextFieldController" class exclusively handles functionalities related to retrieving text field data and clearing it. The "MyColors" class governs the color scheme used throughout the application, maintaining control over color-related functionalities. Database-related operations, responsible for sending HTTP requests to servers and APIs, were consolidated in the "Database" class, each function dedicated to a specific task in line with the separation of concerns principle.

Information Expert Design:
The server-side took on the role of accessing the database and managing

data exchange with the client. All functionalities and data associated with orders were consolidated within the "Order" class, ensuring a focused and organized structure. Similarly, the "GoogleMap" class retained exclusive access to location-related functionalities. This design approach enhances clarity and promotes efficient management of responsibilities across different components of the application.

Chapter 5

Results and Discussion

The application, structured with a client-server architecture, serves a variety of user roles such as customers, supermarkets, delivery personnel, and administrators. It utilizes MongoDB for a flexible and scalable NoSQL database, Node.js for server-side management, and a Flask API for seamless integration of AI capabilities. The UI, developed using Flutter, ensures a consistent experience across diverse platforms. Firebase enhances user interaction with real-time chat, authentication, and notifications.

AI functionalities, including a recommendation system and Simulated Annealing algorithm, are employed to optimize the user experience. The application's architecture facilitates efficient communication between clients and servers, providing a responsive interface for various functions. User-specific interfaces and controls are customized for customers, supermarkets, delivery personnel, and administrators, ensuring a personalized and enriched experience within the application ecosystem.

The application offers a comprehensive set of features tailored to the unique requirements of each user role. Customers benefit from a user-friendly interface for effortless order placement, order history tracking, and personalized recommendations through collaborative filtering. Supermarkets have access to a robust system for efficient product management, order processing, and direct communication with

customers via real-time chat. Delivery personnel utilize the Simulated Annealing algorithm to optimize delivery routes, ensuring timely and context-aware order fulfillment. Administrators wield extensive controls, overseeing revenue analytics, order statistics, and managing user accounts and system settings. The dynamic and adaptive design, coupled with AI-driven features, enhances the overall user experience across diverse roles.

Chapter 6

Conclusions and Future Work

6.1 Conclusions:

The application, with its client-server architecture and tailored features for various user roles, represents a robust and versatile solution for enhancing the user experience in the domains of customer interactions, supermarket management, delivery logistics, and administrative controls. The integration of MongoDB, Node.js, Flask API, Flutter, and Firebase contributes to a seamless and responsive application ecosystem. The incorporation of AI functionalities, such as recommendation systems and the Simulated Annealing algorithm, optimizes user engagement and satisfaction.

The dynamic and adaptive design ensures a consistent experience across platforms, promoting user-friendly interfaces specific to each role. Real-time chat, authentication, and notifications powered by Firebase enrich communication and interaction within the application.

6.2 Future Work:

Enhanced AI Capabilities: Explore and integrate advanced AI models and algorithms to further enhance recommendation systems, personalization, and decision-making processes.

Scalability and Performance Optimization: Continuously assess

and optimize the application's scalability to accommodate a growing user base and ensure high-performance levels, especially during peak usage periods.

Security Measures: Strengthen security measures, including encryption protocols and authentication mechanisms, to safeguard user data, transactions, and sensitive information.

Localization and Globalization: Implement features to support localization and globalization, ensuring the application can cater to diverse languages, cultures, and market needs.

Analytics and Reporting: Enhance the application's analytical capabilities by integrating comprehensive reporting tools for administrators, providing deeper insights into user behavior, revenue trends, and system performance.

Integration with Emerging Technologies: Explore opportunities to integrate emerging technologies such as augmented reality (AR) or virtual reality (VR) to provide innovative and immersive user experiences.

Accessibility Features: Prioritize the implementation of accessibility features to ensure the application is inclusive and usable by individuals with diverse abilities.

Continuous User Feedback: Establish mechanisms for collecting and analyzing user feedback to identify areas for improvement and prioritize feature enhancements based on user needs and preferences.

By addressing these areas in future development, the application can continue to evolve, staying at the forefront of technological advancements and meeting the ever-changing demands of its users and stakeholders.

Bibliography

[1] *Worldwide e-commerce share of retail sales 2015-2024*.[Online]. Available: https://www.statista.com/statistics/534123/ e-commerce-share-of-retail-sales-worldwide

[2] *Global retail e-commerce sales*. [Online]. Available: https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/

[3] "Navigating the crisis towards a human-centred future of work," *Asia–Pacific Employment and Social Outlook*, pp. 14—-20.

[4] C. Bearu. Business formation statistics. [Online]. Available: https://www.census.gov/econ/bfs/index.html

[5] the future of mobile commerce. [Online]. Available: https://www.shopify.com/enterprise/mobile-commerce-future-trends

[6] *The future of social media in marketing*. [Online]. Available: https://link.springer.com/article/10.1007/s11747-019-00695-1

[7] our m-commerce deep dive. [Online]. Available: https://www.bigcommerce.com/blog/mobile-commerce/

[8] M. Hasan, *The Popularity of Online Shopping is increasing during COVID-19 Pandemic: An Online Study in Khulna City of Bangladesh*. [Online]. Available: https://ideas.repec.org/a/aif/journl/v5y2021i5p88-100.html

[9] A. J. Andrea Villanti, *Social Media Use and Access to Digital Technology*. [Online]. Available: https://www.jmir.org/2017/6/e196

[10] S. P. Kia Dashtipour, *Multilingual Sentiment Analysis: State of the Art and Independent Comparison of Techniques*. [Online]. Available: https://link.springer.com/article/10.1007%2Fs12559-016-9415-7

[11] F. S. Esuli, *Determining the semantic orientation of terms through gloss classification*. [Online]. Available: https://www.jmir.org/2017/6/e196

[12] Lampel and H. Mintzberg, "Customizing customization," *Sloan management review*, vol. 38, no. 1, pp. 21–30, 1996.

[13] J. B. Schafer, J. A. Konstan, and J. Riedl, "E-commerce recommendation applications," *Data mining and knowledge discovery*, vol. 5, no. 1, pp. 115–153, 2001.

[14] S. Wang, Z. Zheng, Z. Wu, M. R. Lyu, and F. Yang, "Reputation measurement and malicious feedback rating prevention in web service recommendation systems," *IEEE Transactions on Services Computing*, vol. 8, no. 5, pp. 755–767, 2014.

[15] W. Conner, A. Iyengar, T. Mikalsen, I. Rouvellou, and K. Nahrstedt, "A trust management framework for service-oriented environments," in *Proceedings of the 18th international conference on World wide web*, 2009, pp. 891–900.

[16] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The adaptive web*. Springer, 2007, pp. 291–324.

[17] R. Van Meteren and M. Van Someren, “Using content-based filtering for recommendation,” in *Proceedings of the machine learning in the new information age: MLnet/ECML2000 workshop*, vol. 30, 2000, pp. 47–56.

[18] GoogleDevelopers. Collaborative filtering advantages & disadvantages. [Online]. Available: https://developers.google.com/machine-learning/recommendation/collaborative/summary

[19] P. Dangeti, *Statistics for Machine Learning*. [Online]. Avail- able: http://carina.fcaglp.unlp.edu.ar/~jgomez/grupo/all/references/books/Book_Statistics_for_ML.pdf

[20] Bealing , Explained for simulated Annealing [Online] Available :https://www.baeldung.com/cs/simulated-annealing

[21] Jason Brownlee , Simulated Annealing From Scratch in Python [Online] Available: https://machinelearningmastery.com/simulated-annealing-from-scratch-in-python/

[22] Developer Documentation , Firebase Chat App Tutorial|Jan 2023 [Online] Available https://deadsimplechat.com/blog/firebase-chat-app-tutorial/