GEBZE TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

PICKER & PACKER MOBILE APP PROJECT

Furkan ÖZEV

Advisor
Assoc. Dr. Habil KALKAN

January, 2022
Gebze, KOCAELI

GEBZE TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

PICKER & PACKER MOBILE APP PROJECT

Furkan ÖZEV

Advisor
Assoc. Dr. Habil KALKAN

January, 2022
Gebze, KOCAELI

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 06/10/2021 by the following jury.

Graduation Project Jury

Advisor	Assoc. Dr. Habil Kalkan	
University	Gebze Technical University	
Faculty	Engineering Faculty	

Member	Dr. Gökhan Kaya	
University	Gebze Techinal University	
Faculty	Engineerin Faculty	

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my advisor Assoc. Prof. Dr. Habil KALKAN who contributed to this project and guided me with his weekly meetings, and to my teacher Dr. Gökhan Kaya who listened to me at every follow-up meeting, and Gebze Technical University for supporting this study.

In addition, I would like to express my respect and love to my family, who supported me in every way during my education, and to all my teachers who gave me information and helped me develop.

January, 2022 **Furkan ÖZEV**

Contents

LIST OF FIGURES

LIST OF ABBREVIATIONS

API : Application Programming Interface
REST : Representational State Transfer
IOS : Iphone Operating System
APK : Android Application Package
MVC : Model-View-Controller
OOP : Object-Oriented Programming
IDE : Integrated Development Enviroment
SHA-256 : Secure Hash Algorithm-256
WEB : World Wide Web
HD : High-Definition Video
QHD : Quad High-Definition Video
APP : Application

SUMMARY

Mobile applications has a very important role in the field of software development and it is constantly evolving. In this project, a mobile application is developed that uses the new technologies in this field. The mobile application provides features to perform the process of the preparation and delivery of online grocery orders. There are two different types of users in the application: Picker, who will perform the operations related to the preparation of the Order, and Packer, who will perform the operations related to the delivery process of the Order. The application has been developed to log in with mail/phone and password. If the user is authorized, he can choose and enter the 'Picker' and 'Packer' pages.

Picker can view his information and photo on his profile page, change his information and photo, change his password, make his status active or inactive. While it can receive new orders in its active state, it will not be able to receive new orders in inactive state. Picker can view active orders on the Orders page, click on the order to view all the details of the order, cancel the order, and scan a cart barcode to get the order to the preparation stage. Picker can view all completed and canceled orders on the Orders History page, click on the order to view all its details, and delete the order from the history. On the Prepare Orders page, Picker can prepare more than one order at the same time in the most optimal way, see the details of the orders, make changes on the order such as adding products, changing products and deleting products, canceling the order preparation process, and completing the order.

Packer can view his information and photo on his profile page, change his information and photo, change his password, make his status active or inactive. While it can receive new orders in its active state, it will not be able to receive new orders in inactive state. Packer can view active orders on the Orders page, view all the details of the order by clicking on the order, cancel the order, scan the specified cart barcode for the order to proceed to the delivery stage. Packer can view all completed and canceled orders on the Orders History page, click on the order to view all its details, and delete the order from the history. On the Delivery Orders page, Packer can deliver

multiple orders at the same time in the most optimal way, see the details of the Orders, view the changes made by Picker on the Order, cancel the order delivery process, and complete the order.

The main and original purpose of this project is to enable both Picker and Packer to perform the order process in the most optimal way, to store the transaction data and to provide data transfer between them. All transactions are carried out by processing on a database in the Cloud environment over the network. In addition, all these functions have been developed to provide both Android mobile devices and IOS mobile devices. Flutter and Android Studio are used as the development tools to provide these features, PostgreSQL as the database system, Heroku/Render cloud-based platform services to serve the Backend APIs, Packer to sort the orders in the most optimal way and to get information such as distance and time. Direction API and Distance Matrix APIs offered by Google Maps Platform were used. All backend APIs are developed with Python Flask. [1][2] [3] [5] [6] [7] [8]

ÖZET

Mobil uygulamalar geliştirici dünyasında çok önemli rol oynamaktadır ve sürekli gelişmektedir. Bu projede bu gelişime ve yeni teknolojilere adapte olmuş olan bir mobil uygulama oluşturmak üzere kurgulandı. Bu bağlamda projedeki mobil uygulama online market siparişlerinin hazırlanması ve teslim sürecini sağlayacak işlevleri içeren şekilde geliştirme yapıldı. Uygulamada, Siparişin hazırlanmasıyla ilgili işlemleri gerçekleştirecek Picker, Siparişin teslim süreciyle ilgili işlemleri gerçekleştirecek Packer olmak üzere iki farklı kullanıcı tipi mevcuttur. Uygulamaya mail/telefon ve şifre ile giriş yapacak şekilde geliştirildi. Kullanıcı yetkisi mevcut ise 'Picker' ve 'Packer' sayfalarına seçim yapıp girebilmektedir.

Picker, profil sayfasından bilgilerini ve fotoğrafını görüntüleyebilir, bilgilerini ve fotoğrafını değiştirebilir, şifresini değiştirebilir, durumunu active veya inaktif yapabilir. Aktif durumunda yeni siparişler alabiliyorken, inaktif durumda yeni sipariş alamayacaktır. Picker, Siparişler sayfasında aktif siparişlerini görüntüleyebilir, siparişe tıklayarak siparişin tüm detaylarını görüntüleyebilir, siparişi iptal edebilir, siparişin hazırlanma aşamasına geçmesi için bir sepet barkodu taratabilir. Picker, Sipariş Geçmişi sayfasında tamamlanan ve iptal edilen tüm siparişleri görüntüleyebilir, siparişe tıklayarak tüm detaylarını görüntüleyebilir, siparişi geçmişten silebilir. Picker, Sipariş Hazırlama sayfasında, aynı anda birden fazla siparişi en optimal şekilde hazırlayabilir, Siparişlerin detaylarını görebilir, Sipariş üzerinde ürün ekleme, ürün değiştirme ve ürün silme gibi değişiklikler yapabilir, sipariş hazırlama sürecini iptal edebilir, siparişi tamamlayabilir.

Packer, profil sayfasından bilgilerini ve fotoğrafını görüntüleyebilir, bilgilerini ve fotoğrafını değiştirebilir, şifresini değiştirebilir, durumunu aktif veya inaktif yapabilir. Aktif durumunda yeni siparişler alabiliyorken, inaktif durumda yeni sipariş alamayacaktır. Packer, Sipariş sayfasında aktif siparişlerini görüntüleyebilir, siparişe tıklayarak siparişin tüm detaylarını görüntüleyebilir, siparişi iptal edebilir, siparişin teslim aşamasına geçmesi için belirtilen sepet barkodunu taratabilir. Packer, Sipariş

Geçmişi sayfasında tamamlanan ve iptal edilen tüm siparişleri görüntüleyebilir, siparişe tıklayarak tüm detaylarını görüntüleyebilir, siparişi geçmişten silebilir. Packer, Sipariş Teslimat sayfasında, aynı anda birden fazla siparişi en optimal şekilde teslim edebilir, Siparişlerin detaylarını görebilir, Sipariş üzerinde Picker tarafından yapılan değişiklikleri görüntüleyebilir, sipariş teslim sürecini iptal edebilir, siparişi tamamlayabilir.

Bu projedeki asıl ve özgün olan amaç hem Picker hem de Packer ın en optimal şekilde sipariş sürecini gerçekleştirebilmesi, işlem verilerinin saklanması, aralarında veri geçişinin sağlanması. Tüm işlemlerin ağ üzerinden Cloud ortamda bir database üzerinde işlenerek gerçekleştirilmesi. Ayrıca tüm bu işlevler hem Android mobil cihazlar hem de IOS mobil cihazlar için sağlayacak şekilde geliştirme yapıldı. Bu bahsedilen özellikleri sağlamak amacıyla geliştirme aracı olarak Flutter ve Android Studio kullanılırken, veri tabanı sistemi olarak PostgreSQL, Backend API lerinin hizmet vermesi için Heroku/Render bulut tabanlı platform servisleri, Packer ın en optimal şekilde siparişleri sıralaması için ve mesafe, süre gibi bilgilerin alınması için Google Maps Platform un sunduğu Direction API ve Distance Matrix API leri kullanıldı. Tüm backend API leri Python Flask ile geliştirildi. [1][2] [3] [5] [6] [7] [8]

1. INTRODUCTION

Today, we do most of our shopping through mobile applications. The increase in this need day by day has led to an increase in companies providing this type of service. Thus, the online grocery shopping sector has become a rapidly growing and competitive sector. Customers can easily select their products through the application, make their payments and complete their orders. Although the process for the customer ends here, the process is just beginning for the market that provides this service in the background. We have two main processes, prepare and deliver the order. The purpose of this application is to enable different types of employees to perform these processes in the most optimal and fast way and to take some necessary actions.

On the verge of all this, the best way to create an application that solves the problem was to do extensive research first. In order to understand what has been done and what can be done in this field, research has been done on various platforms. Researches and analyzes were carried out to determine the necessary functions and what kind of needs should be provided.

After providing sufficient information about the context, researches were carried out to determine the application to be made as a skeleton and applications that perform similar tasks. A new design was made within the knowledge and various information obtained from here.

The main purposes were included in the first design. In other words, a simple, understandable and functional usage scenario has been prepared in order to ensure that the user base of the application is specific and the process progresses quickly. In this scenario, user can view the orders on the main screen and switch to the profile, history and prepare/delivery pages. Each order has its own detail button and functional

buttons. The main design has been updated from the start with various improvements and changes. A modern interface was provided by making use of the latest features provided by Flutter. This design concept was maintained throughout the development process. In this context, it is divided into 4 main parts, namely viewing profile and functions, viewing open orders and functions, viewing past orders and functions, order preparation/delivery and functions. [2] [3]

The report explains in detail all the above-mentioned and briefly informed issues, namely listing the technologies used and for what purpose they are used in the application, explaining the design and interfaces, explaining the database, functions running in the application, API installation and serving in the Cloud environment, third party API used It contains information on the functions of the software, device and user tests, and the conclusion of the success criteria.

2. METHOD AND MATERIALS

First of all, a general plan was determined in this project. First of all, a design to be used roughly in the application was determined and a rough design plan was drawn up for its functions. In line with this design plan, interface studies were made and the project started. Below is the initial design plan and interface of the project. Then this design plan and interface was changed and developed.

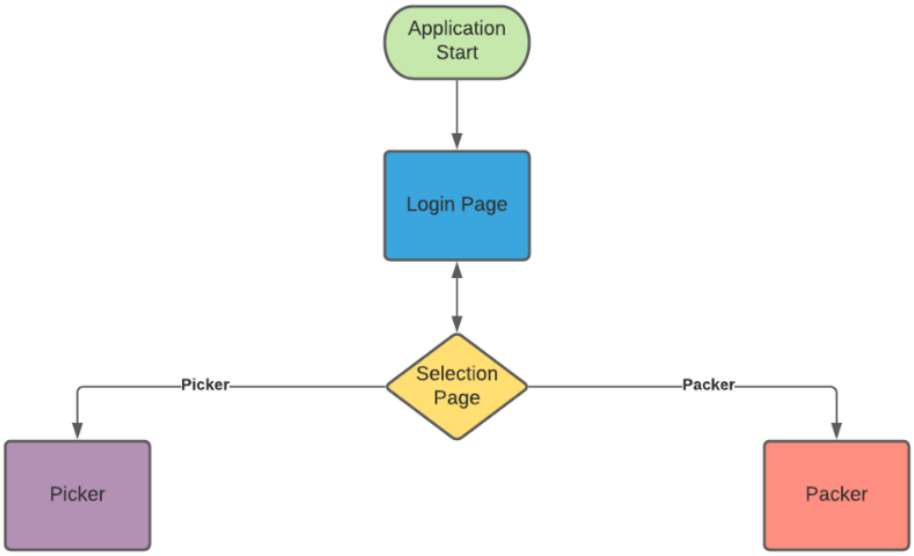

Figure 1 Login Design Plan

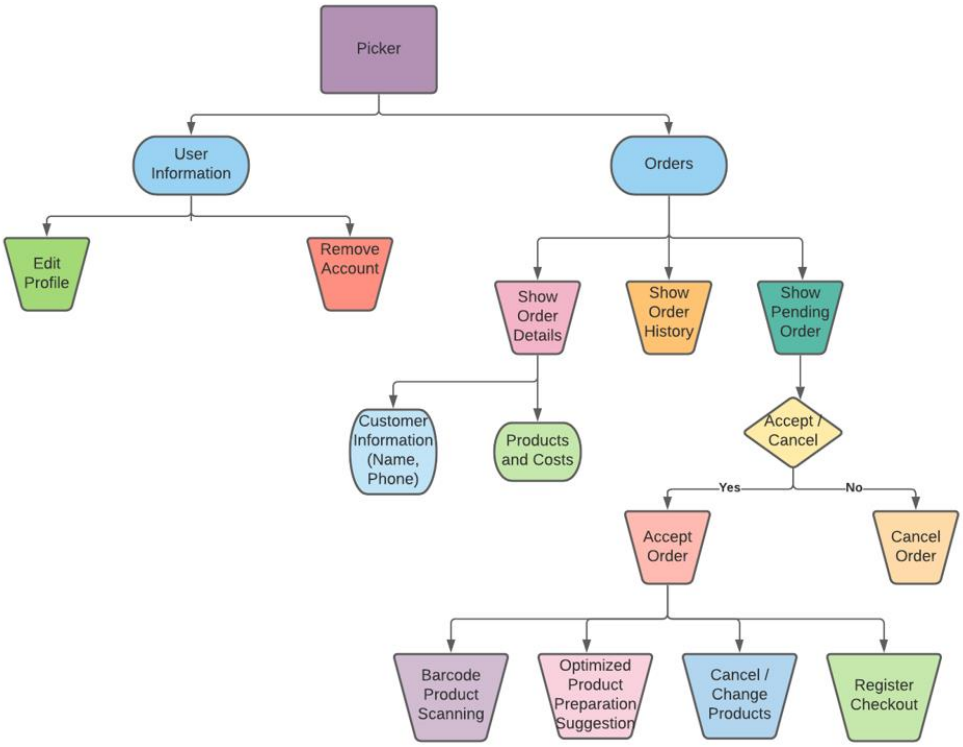

Figure 2 Picker Design Plan

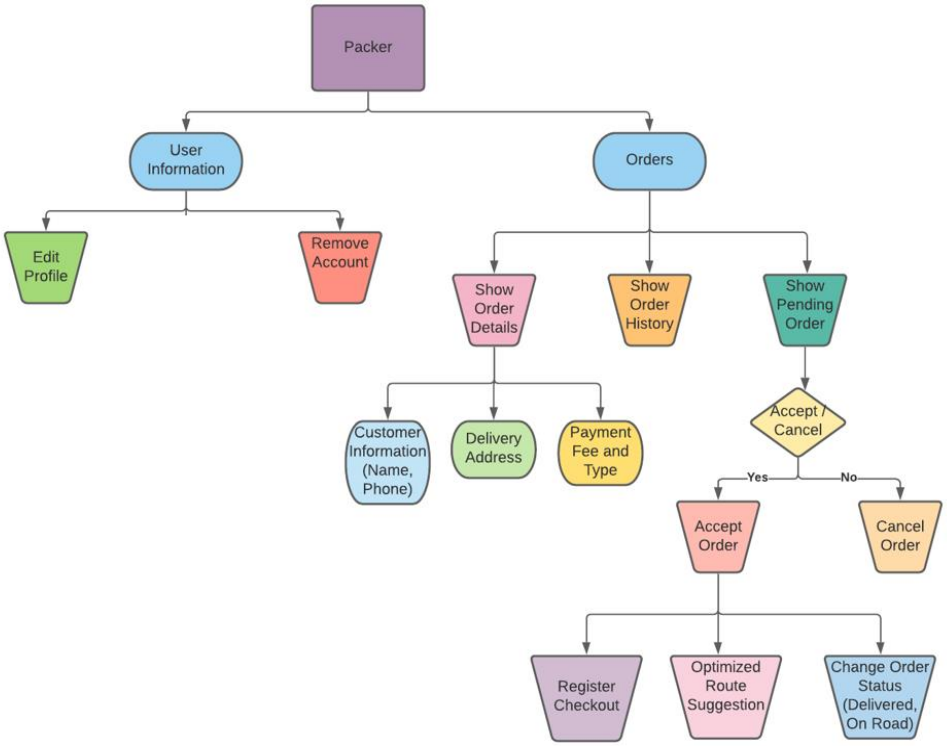

Figure 3 Packer Design Plan

2.1. USED TECHNOLOGIES AND FUNCTIONS

After the design plan in the project was made, which development tools and development features were used for similar applications were investigated.

2.1.1. Android Studio

Android Studio is an integrated development environment, or IDE, built on IntelliJ, developed by Google and JetBrains. In this environment, applications that can run on devices with Android operating system are made in general. It supports Java and Kotlin programming language. In this project, the Android Studio program was used as a development environment to develop the column reading application. The Java programming language was chosen because it was thought to contain wider resources than Java or Kotlin. As the programming principle, OOP (Object Oriented Programming) was used. The MVC model, which is the default model understanding of the Android Studio IDE, was used as the programming model. In this model, the general understandings were managed as follows. [8]

1. Model:

   This component stores application data. Doesn't know about the interface. As a result of the request, the data in the response from the API is kept in the object of the relevant model class and used by the relevant interface widget. In short, it is the bridge between the API and the widget.

2. View

   It is the UI (User Interface) layer that holds the components visible on the screen. It also provides visualization of the data stored in the Model and offers interaction to the user. Page widgets in the application are an example of this part.

3. Controller

   This component establishes the relationship between View and Model. It contains the core application logic and stays aware of the user's behavior and updates the

Model as needed. It ensures that the relevant request is sent to the backend. The example of this part is the Provider files in the application.

2.1.2. Flutter

Flutter is an open source UI development kit created by Google. The most important feature of this development kit is that it can output on many platforms. It can output to platforms such as Android, iOS, Google Fuchsia, Web, Windows, macOS and Linux. It is written in Dart programming language, and unlike the default MVC approach in Android Studio, it acts as both a view, a model, and a controller with a single application.

In this project, the Flutter kit was used to develop the Picker & Packer application on multiple platforms. Used with Dart programming language. As the programming principle, OOP (Object Oriented Programming) was used. [2] [3] [8]

2.1.3. PostgreSQL

PostgreSQL or Postgres is a free and open source, SQL supported relational database management system. Postgres delivers the world's most advanced open source database as a trusted, secure, and scalable service that is optimized for developers. Since there are many relational data and relational operations in our application, a relational database was preferred. At the same time, since our application serves in the cloud environment, its database should serve in the cloud environment. In line with this need and direction, it was preferred because it is easy to use and manageable and can be easily integrated into cloud service platforms such as Heroku and Render. [5] [6]

2.1.4. Flask

Flask is a micro web framework written in Python. It is called a microskeleton because it does not require specific tools or libraries. The database abstraction layer does not have form validation or other components that pre-existing third-party libraries provide common functionality. However, Flask supports extensions that can add app features to itself. Extensions are available for object-relational mappers, forms

validation, upload handling, various open authentication technologies, and various common framework-related tools.

In this project, Flask is used to receive, parse and perform all functions in the application via REST API requests. It is preferred because it can be deployed and developed on both Heroku and Render. Approximately 70 APIs have been developed and most of them are used directly by the application. [1] [5] [6]

2.1.5. Heroku / Render

Heroku is a web cloud platform that enables the dissemination of applications or programs developed by many programming languages. An application distributed by Heroku will have its own unique web address and can be used from that address. In this project, REST APIs written with Heroku Flask were used to work on the cloud, not as a local API. Thus, written APIs and thus the application can be used by multiple users anywhere and at the same time. [1] [5] [6]

Render, on the other hand, was chosen as a platform similar to Heroku, as an alternative.

2.1.6. Adobe XD

Adobe XD (Adobe Experience Design) is an Adobe program used to make UX/UI interface designs for websites or applications. With this application, the preliminary work can be seen in a simpler way while making a mobile application or internet application. In this project, Adobe XD was used to design the template of the application. Thus, it provided the opportunity to visually identify and change the planning and changes before proceeding to the coding stage.

2.1.7. FlutterFlow

FlutterFlow is a technology for visually designing mobile applications, like Adobe XD. Unlike Adobe XD, it can be implemented directly with low code. For this application, it has been benefited very little in terms of making preliminary studies and examining the samples. [14]

3. DESIGN AND INTERFACES

As stated in the method section of the report, first of all, an initial design and initial interface were made. However, this design and interface were further developed over the weeks with feedback. In this section, the final state of the application will be explained and described.

3.1. ANIMATED SPLASH SCREEN

When the application's icon is clicked on the phone and run, it welcomes a moving splash screen. This screen takes about 2 seconds, then the login page opens. Splash screen is designed using Flutter spin_kit module. [2] [3]

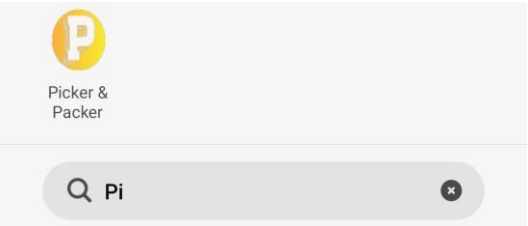

Figure 4 App Launch Icon

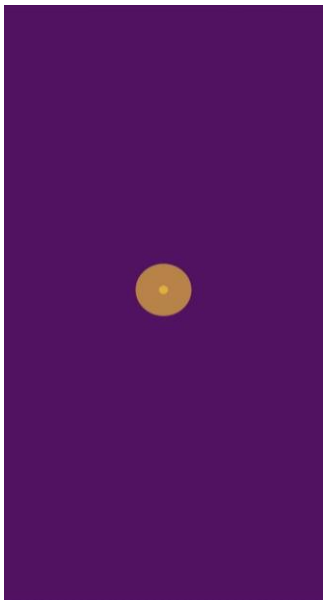

Figure 5 App Splash Screen

3.2. LOGIN PAGE

The user must enter their email or phone number in the first box and their password in the second box. User can change the visibility of the password with the button on the right of the password box. After entering the information, press the Login button. The entered information (email and the SHA256 hash code of the password) is transmitted to the Cloud (Heroku/Render) via the REST API, where controls are made on the database. If the information matches, the credentials information is returned and redirected to the selection page and a notification message appears at the bottom that the login was successful. If the information does not match, an error message appears at the bottom and informs the user.

The Snackbar feature offered by flutter is used for the notification message, the messages appear for a certain period of time and then disappear. [2] [3] [5] [6]

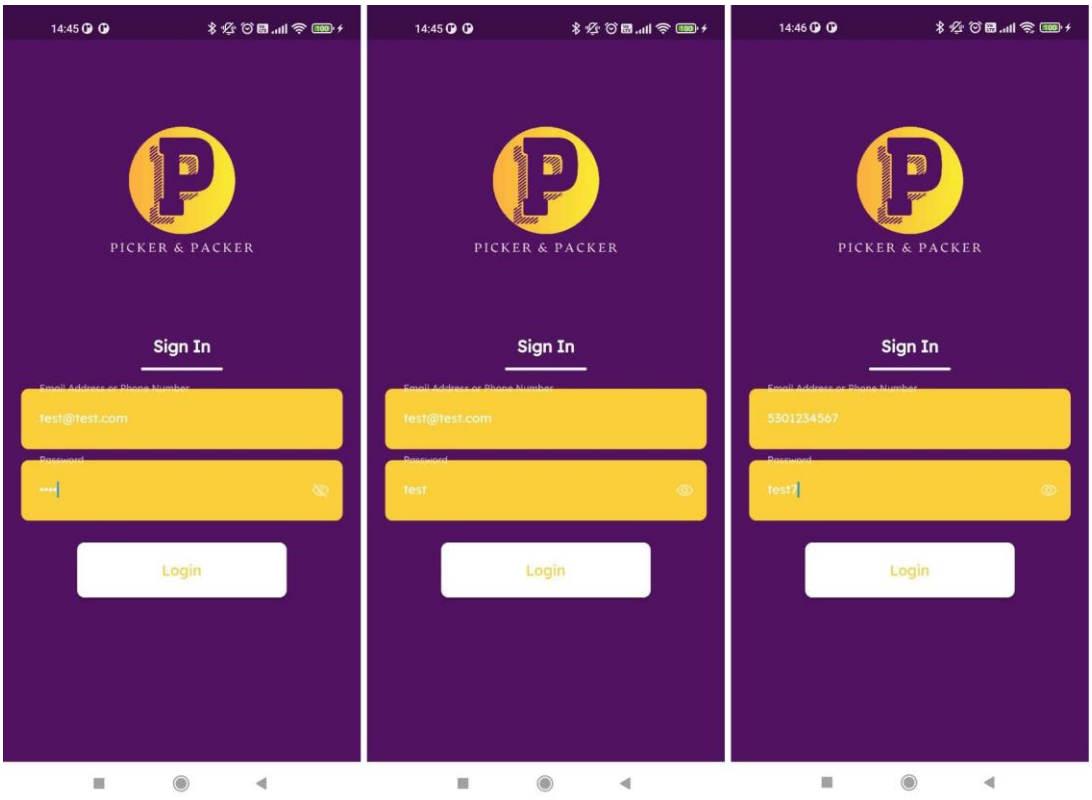

Figure 6 Login Screen

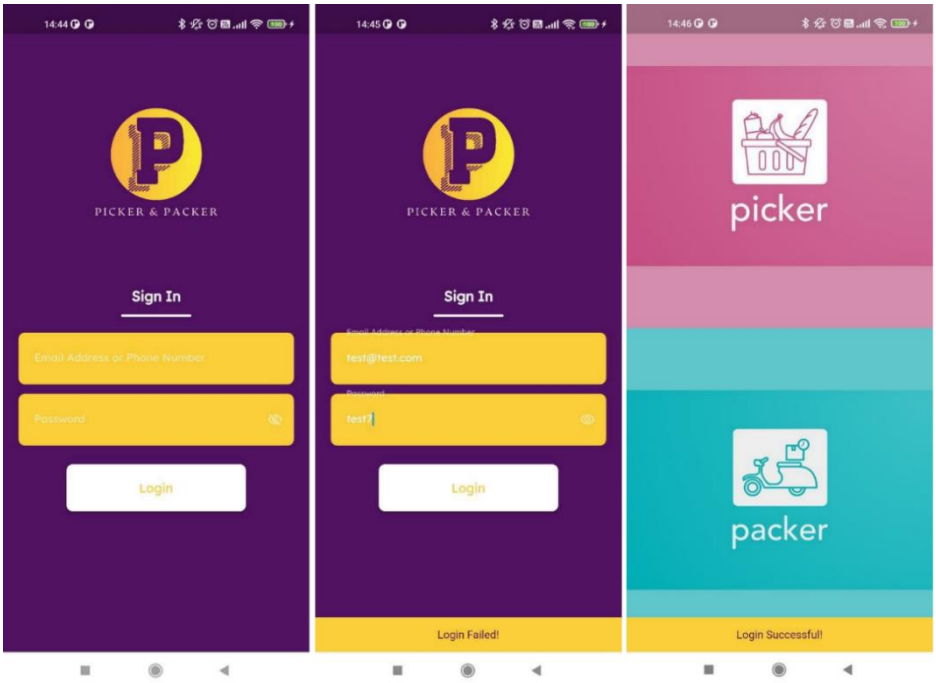

Figure 7 Login Screen-2

3.3. SELECTION PAGE

After the login is successful, the selection page opens, the user can go to the Picker page if user has authorization, and to the Packer page if user has authorization.

Figure 8 Selection Screen

3.4. PICKER PAGE

If the user is a Picker, user can access the Picker section when he chooses Picker. The screen with direct open orders appears. While there are profile, orders and orders history pages in the navigation bar, there is an order preparation button at the top right of the app bar. The user can switch between pages by clicking these buttons. The Navigation Bar feature provided by Flutter is used at the bottom and the App Bar feature is used at the top. [2] [3]

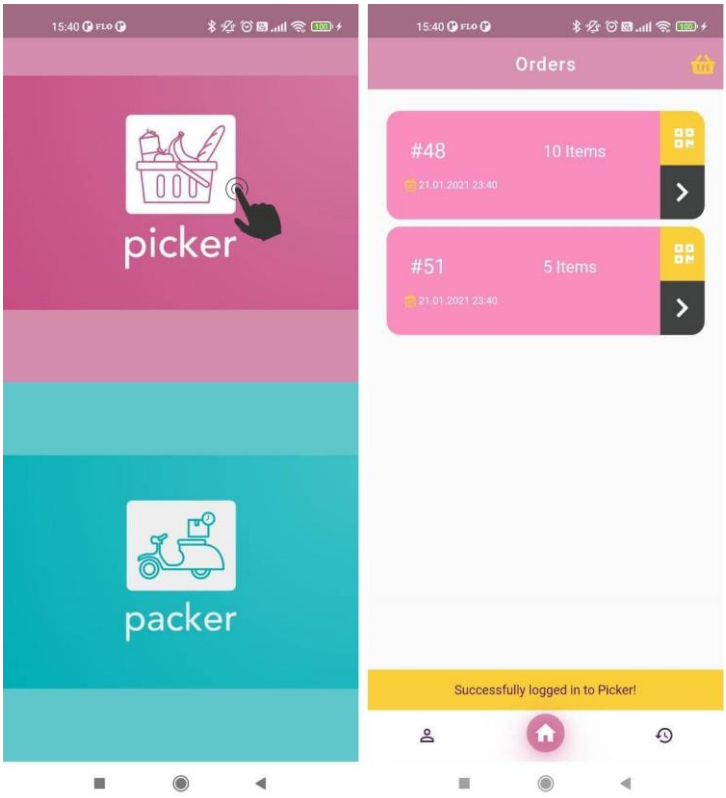

Figure 9 Selection Picker Screen

Figure 10 Navigation Bar

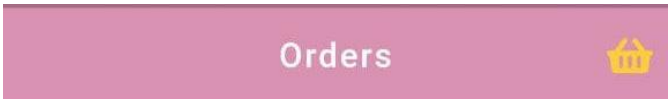

Figure 11 App Bar

3.4.1. Picker Profile Page

User can view user information (name-surname, Picker, Phone number) and photo on Picker Profile page. It can change its active status, while it is active, it can take new orders, while it is inactive, it cannot take new orders. When changing the activity status, the user is warned and asked to confirm. The user logs out by pressing the Logout button and is directed to the login screen again. By pressing the Edit Profile button, he can update his information and photo. User can change user password by pressing the Change Password button.

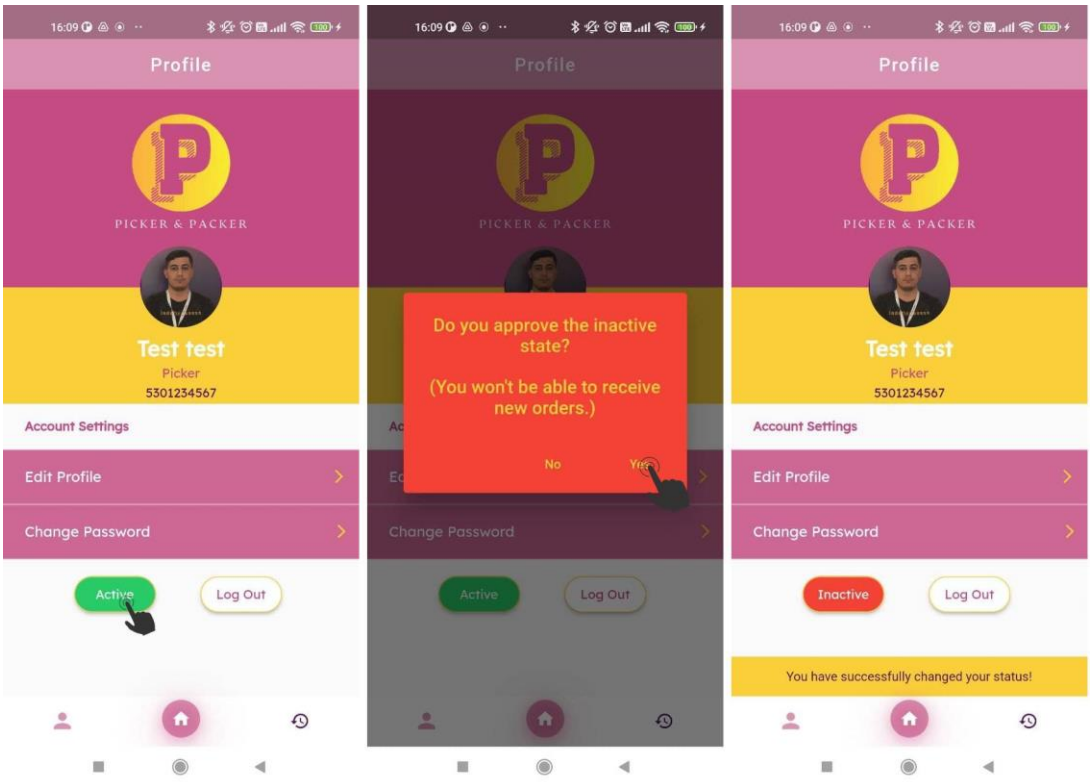

Figure 12 Picker Profile Page

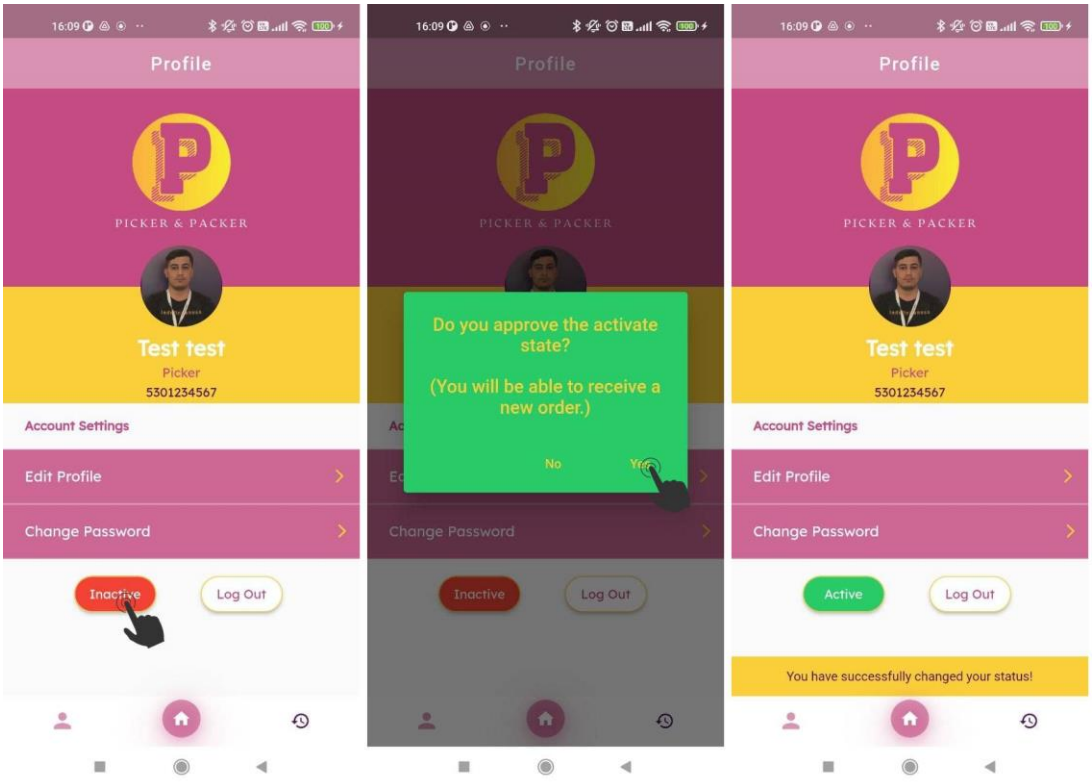

Figure 13 Picker Profile Page-2

3.4.2. Picker Edit Profile Page

User can view and update their information (name-surname, email, age, phone number) and photo. User can take the new photo with the camera, add it from the gallery or remove the photo.

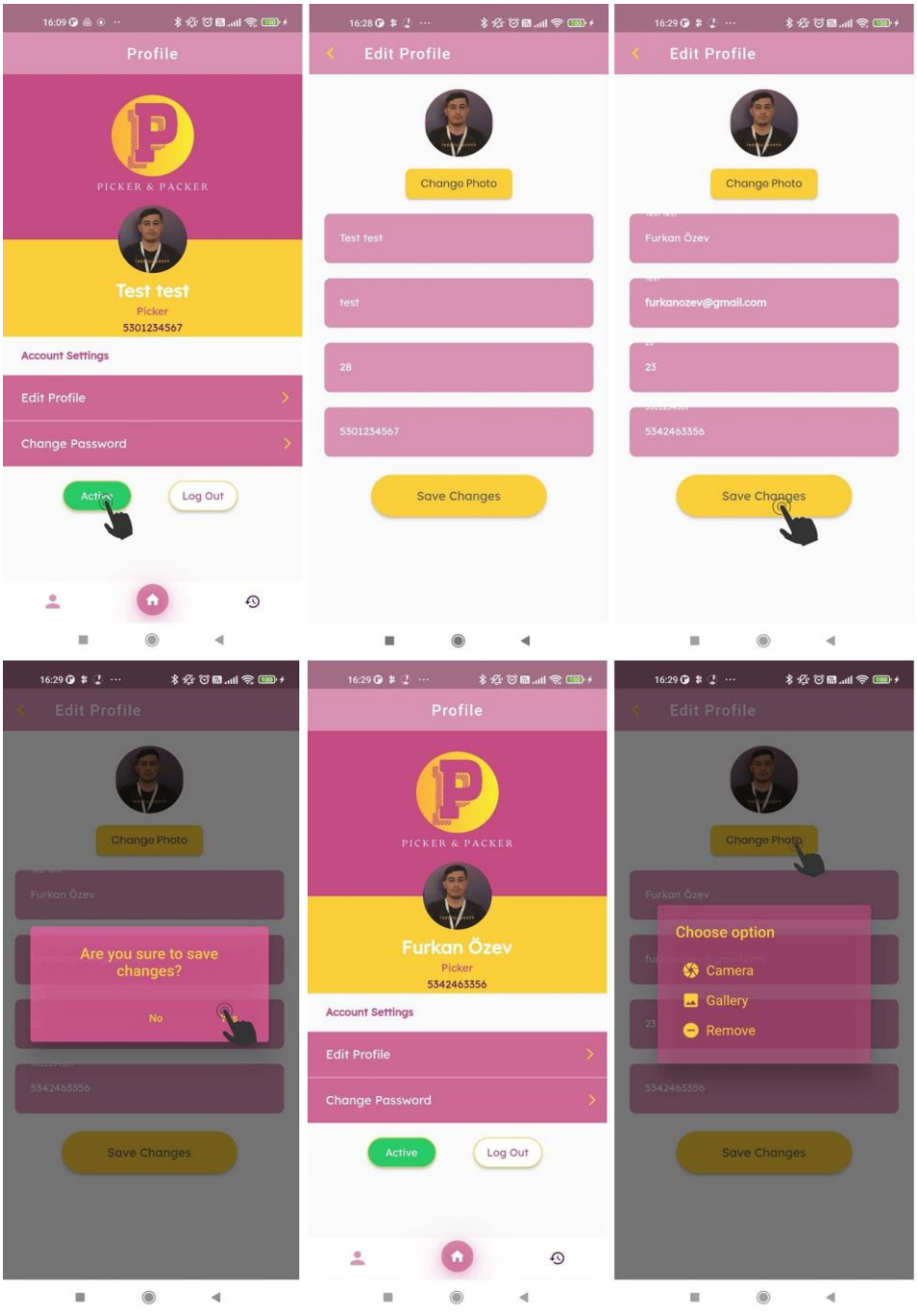

Figure 14 Picker Edit Profile Page

3.4.3. Picker Change Password Page

The user can change his password, he must enter the current password in the first box, the new password in the second box, and the new password in the third box. If any box is empty, an error message is given. If the new password does not match, an error message is given. If the current password is incorrect, an error is given. If there is no error, the password is successfully changed.

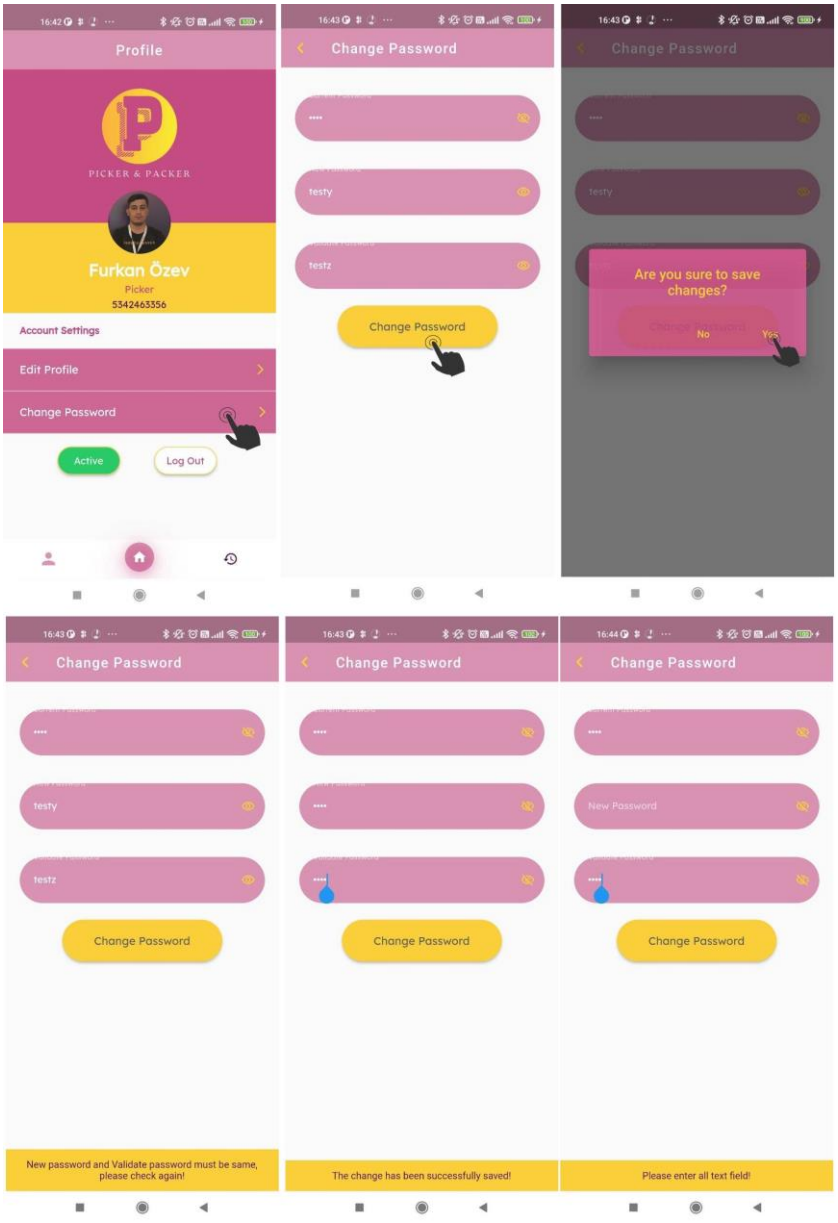

Figure 15 Picker Change Password Page

3.4.4. Picker Orders Page

The user can view open orders, view the details of the relevant order by clicking the detail button, cancel the order, scan a cart barcode for the order and include it in the preparation process. The barcode of the cart it scans must be registered in the system, it cannot scan a random cart and the cart must be available and empty, that is, it must not be used by someone else. Cancel Order button is a slideable button, it needs to slide the order left to view the button. The user can update the order list by dragging the page down. Orders are sorted by delivery date, from the earliest delivered to the latest ordered.

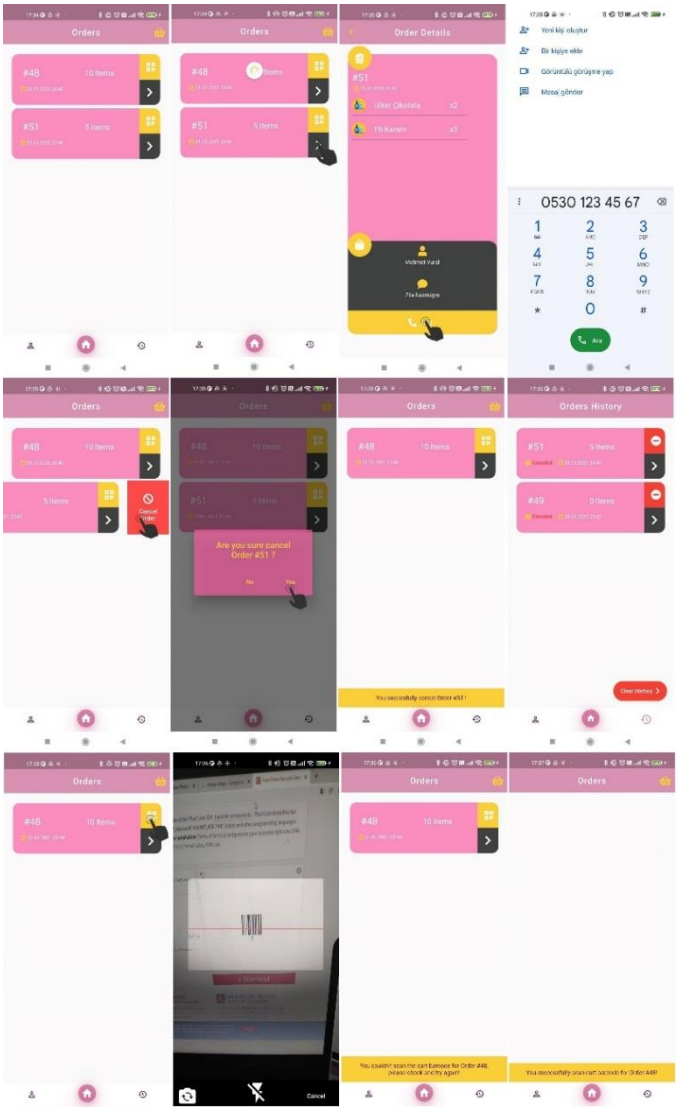

Figure 16 Picker Orders Page

3.4.5. Picker Order Details Page

The user can view all the products in the relevant order together with their quantities. User can view information such as order number, order delivery date, customer name, customer note and call the customer by pressing the Call button.

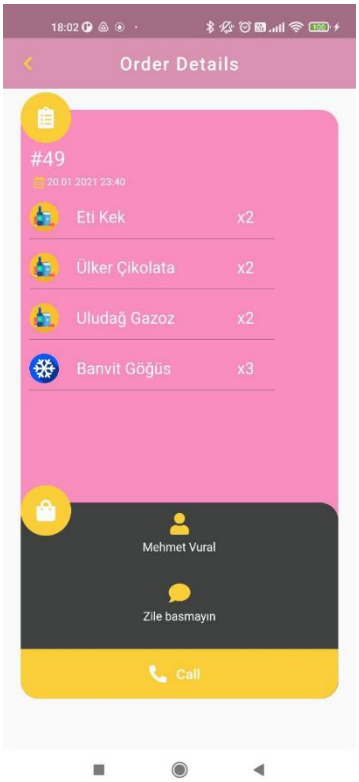

Figure 17 Picker Order Details Page

3.4.6. Picker Orders History Page

The user can view all past (cancelled and completed) orders. It can display changes in completed orders, price difference. User can delete any order from history. User can delete all orders from history. Orders are sorted by last transaction date. With the most recently modified order at the top. So the oldest order will be last. User can view the changes Picker has made on the order, deleted products are shown in red and added products are displayed in green. If the price difference is positive, that amount is refunded, and the negative amount is obtained from the customer.

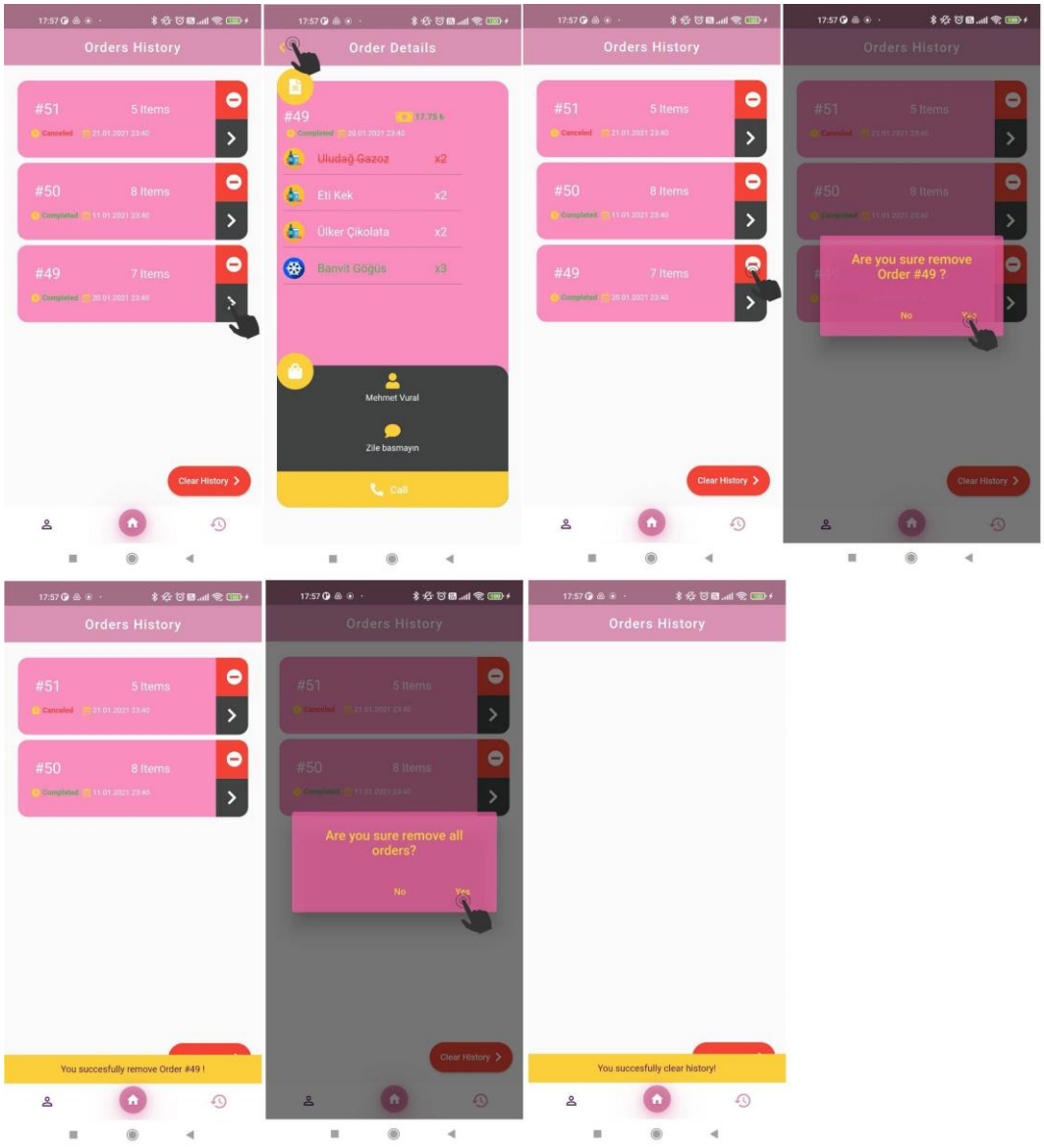

Figure 18 Picker Orders History Page

3.4.7. Picker Order History Details Page

The user can view all the products in the relevant order together with their quantities. Deleted items are shown in red, and later added items are shown in green. If there is a product replaced with an alternative product, it is displayed under the deleted product. The price difference is also noted at the top right. If the number is positive, it indicates the amount to be refunded, if it is negative, it indicates the amount to be received from the customer. User can view information such as order number, order

delivery date, customer name, customer note and call the customer by pressing the Call button.

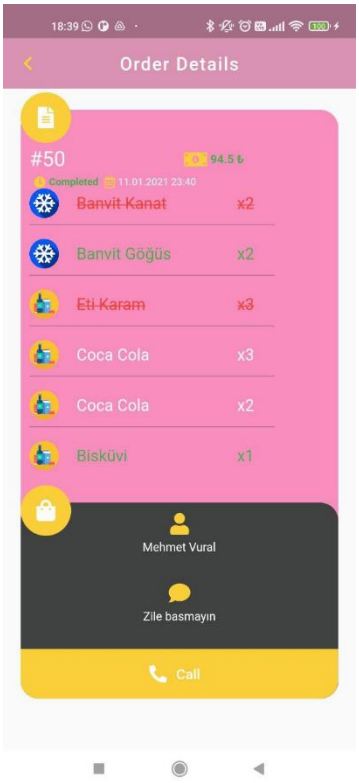

Figure 19 Picker Order History Details Page

3.4.8. Picker Order Prepare Page

The user manages the preparation process of the orders on this page. It enables the preparation of more than one order at the same time. The widget at the top sorts all the products in the carts in the most optimal way according to category, sub-category and order number. Thus, the user can prepare orders with minimum effort, and there is no need for unnecessary wandering in the market. The widget contains the product, category, sub-category, number of products and the barcode of the relevant cart.

In order to add the relevant product to the cart, first click on the Item button shown in yellow. If the correct barcode is not scanned, the user is informed with an error message. If the correct barcode is scanned, the product color turns yellow so that the cart barcode can be scanned. Then, the Cart button shown in green is pressed to scan

the relevant cart barcode. If the correct barcode is not scanned, the user is informed with an error message. If the correct barcode is scanned, the product is deleted if the number of products is one, if the number of products is more than one, the remaining number of products is reduced by one. If the cart barcode is tried to be scanned without scanning the item barcode, the user will encounter an error message. The user can see the products added to the cart in green in the detail section. If there is no product for the relevant cart, it can complete the order.

Displays the orders in the preparation process in the sub-widget, and can access the details of the relevant order by pressing the detail button. Items that have been deleted in the detail section are shown in red, items that are added later are shown with '*', items that are added later with changes are shown with '**', and items in the cart are shown in green. If the user slides the relevant order to the left, they can see the Cancel Prepare and Complete Prepare buttons. These buttons are Slidable buttons, so they can be accessed by swiping left. When Cancel Presses the Prepare button, all changes made to the order are reversed and the order returns to its original state. The user can view the order again in the Orders Page. When he clicks the Complete Order button, the user will encounter an error message if not all products have been added to the cart. If all products have been added to the cart, the order is completed and a packer is assigned an order. The process of assigning an order to Packer is automatic, an order is assigned to the Packer whose status is active and has the least open order. Thus, it is ensured that the order density of Packers is minimized.

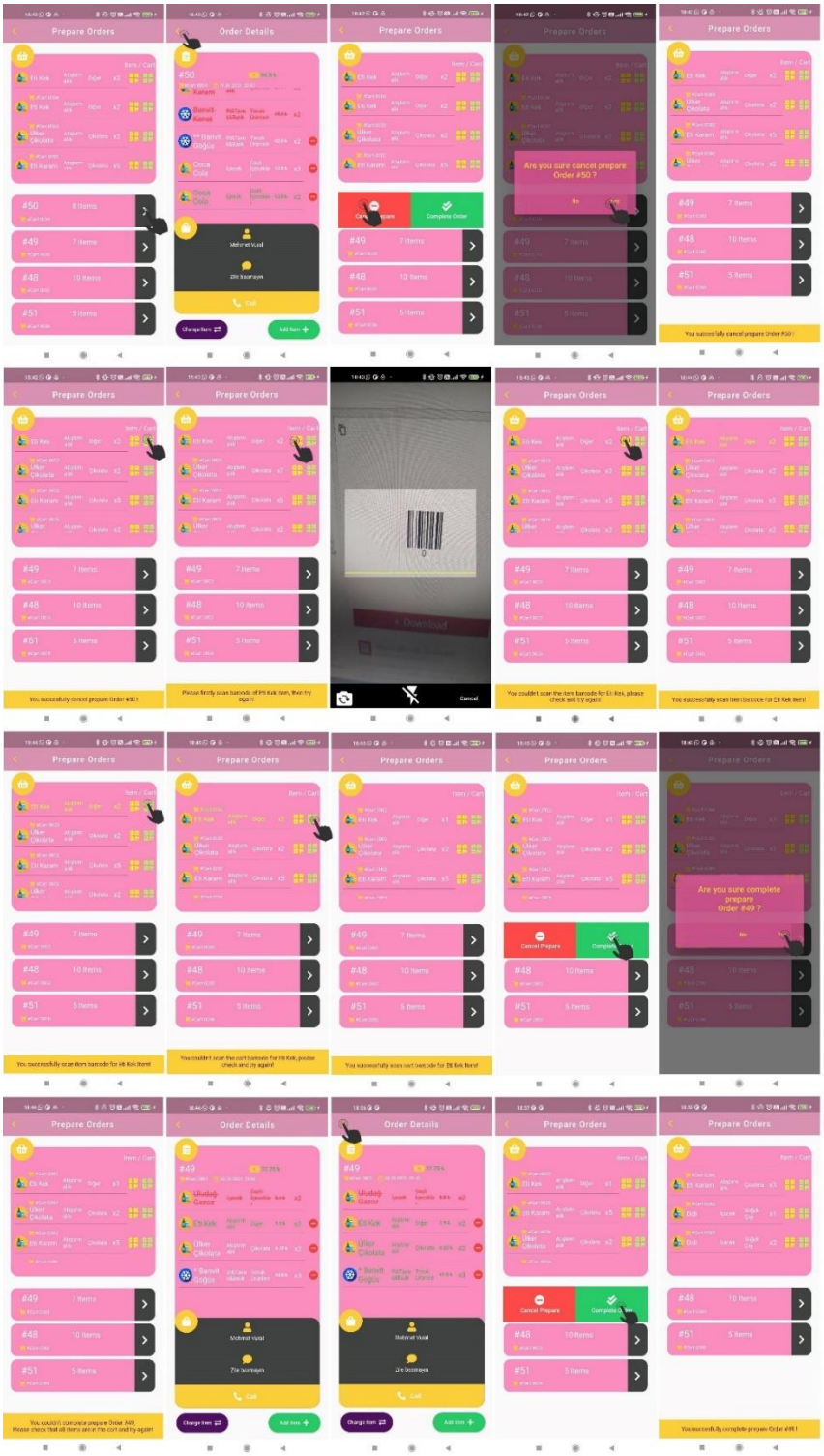

Figure 20 Picker Order Prepare Page

3.4.9. Picker Order Prepare Details Page

The user can view all the products in the relevant order together with their quantities. Items that have been deleted in the detail section are shown in red, items that are added later are shown with '*', items that are added later with changes are shown with '**', If there is a product replaced with an alternative product, it is displayed under the deleted product, and items in the cart are shown in green. The price difference is also noted at the top right. If the number is positive, it indicates the amount to be refunded, if it is negative, it indicates the amount to be received from the customer. User can view information such as order number, order delivery date, customer name, customer note and call the customer by pressing the Call button. User can delete the relevant product by clicking the Delete Item button, the price of the relevant product will be deducted from the order amount. User can add a new product by clicking the Add Item button, the relevant product price is added to the order price. By clicking on the Change Item button, user can change the products in the order, by choosing a product from the product list instead of the selected order, it is changed by the same number, the price difference is reflected in the order amount.

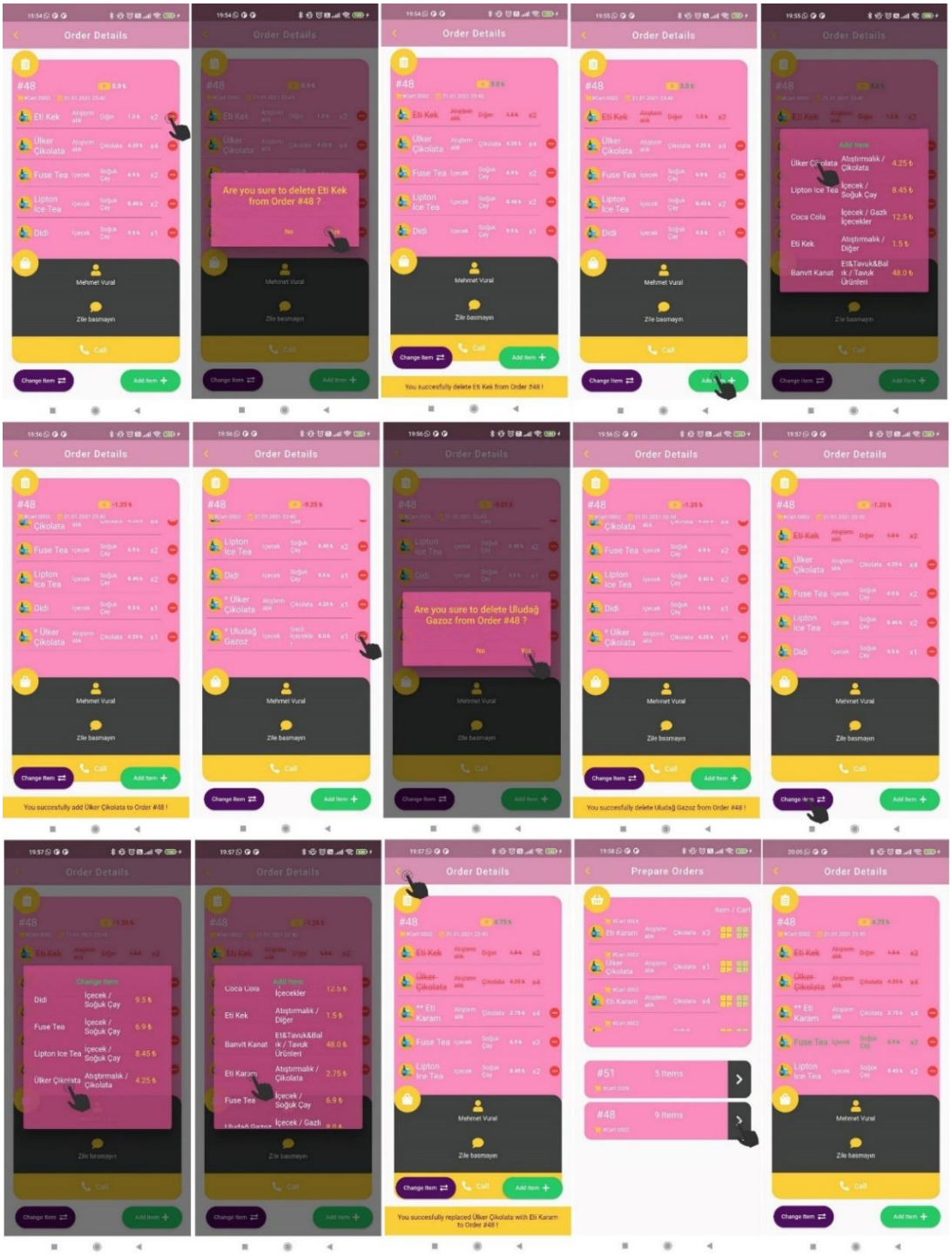

Figure 21 Picker Order Prepare Page

3.5. PACKER PAGE

If the user is a Packer, user can access the Packer section when he chooses Packer. The screen with direct open orders appears. While there are profile, orders and orders history pages in the navigation bar, there is an order delivery button at the top right of the app bar. The user can switch between pages by clicking these buttons. The Navigation Bar feature provided by Flutter is used at the bottom and the App Bar feature is used at the top. [2] [3]

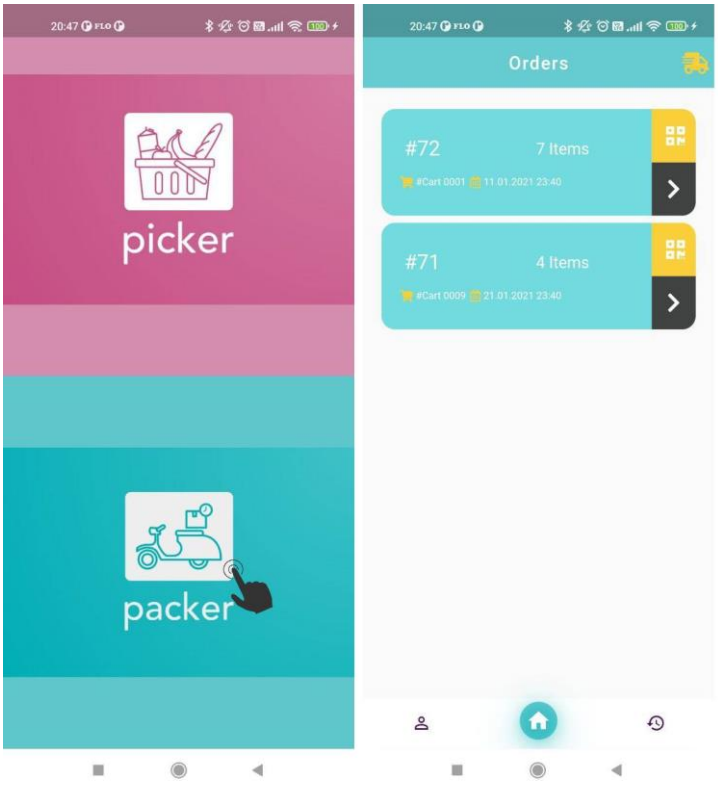

Figure 22 Selection Packer Screen

Figure 23 Navigation Bar

Figure 24 App Bar

3.5.1. Packer Profile Page

User can view user information (name-surname, Packer, Phone number) and photo on Packer Profile page. It can change its active status, while it is active, it can take new orders, while it is inactive, it cannot take new orders. When changing the activity status, the user is warned and asked to confirm. The user logs out by pressing the Logout button and is directed to the login screen again. By pressing the Edit Profile button, he can update his information and photo. User can change user password by pressing the Change Password button.

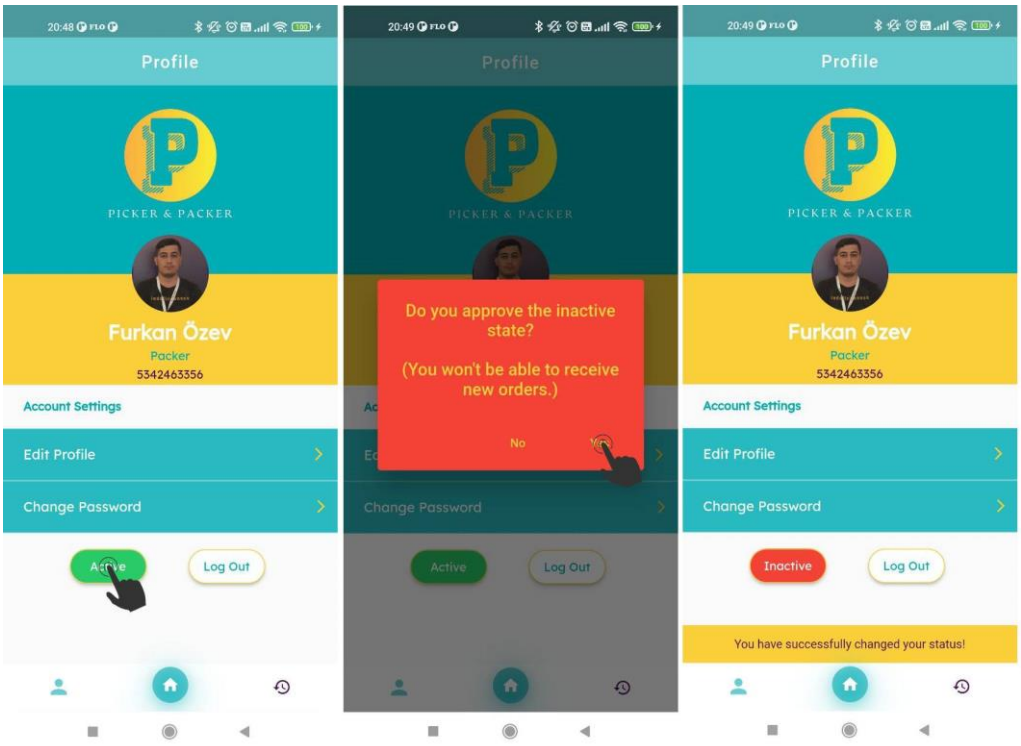

Figure 25 Packer Profile Page

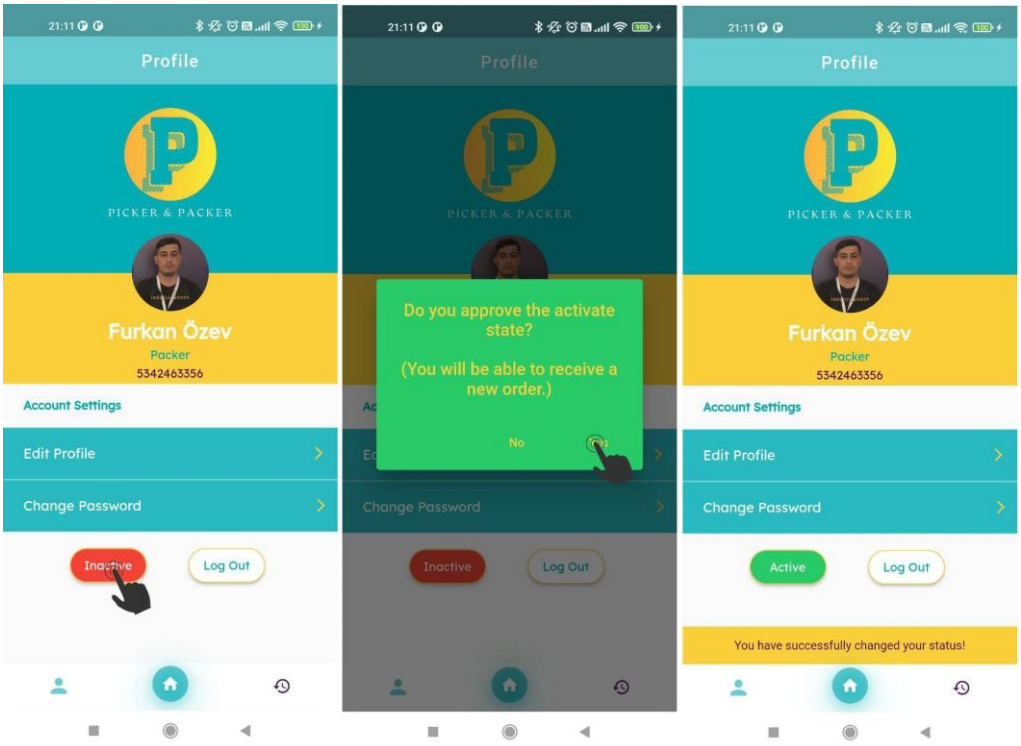

Figure 26 Packer Profile Page-2

3.5.2. Packer Edit Profile Page

User can view and update their information (name-surname, email, age, phone number) and photo. User can take the new photo with the camera, add it from the gallery or remove the photo.

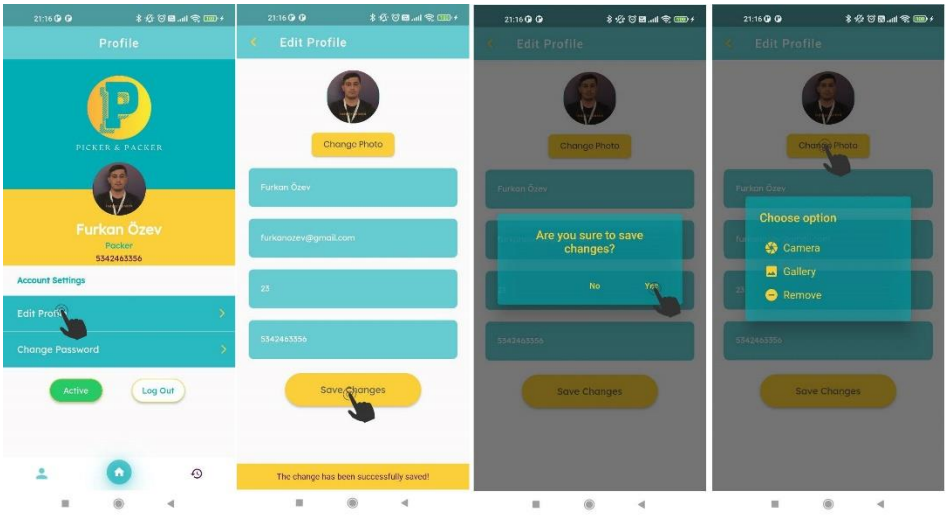

Figure 27 Packer Edit Profile Page

3.5.3. Packer Change Password Page

The user can change his password, he must enter the current password in the first box, the new password in the second box, and the new password in the third box. If any box is empty, an error message is given. If the new password does not match, an error message is given. If the current password is incorrect, an error is given. If there is no error, the password is successfully changed.

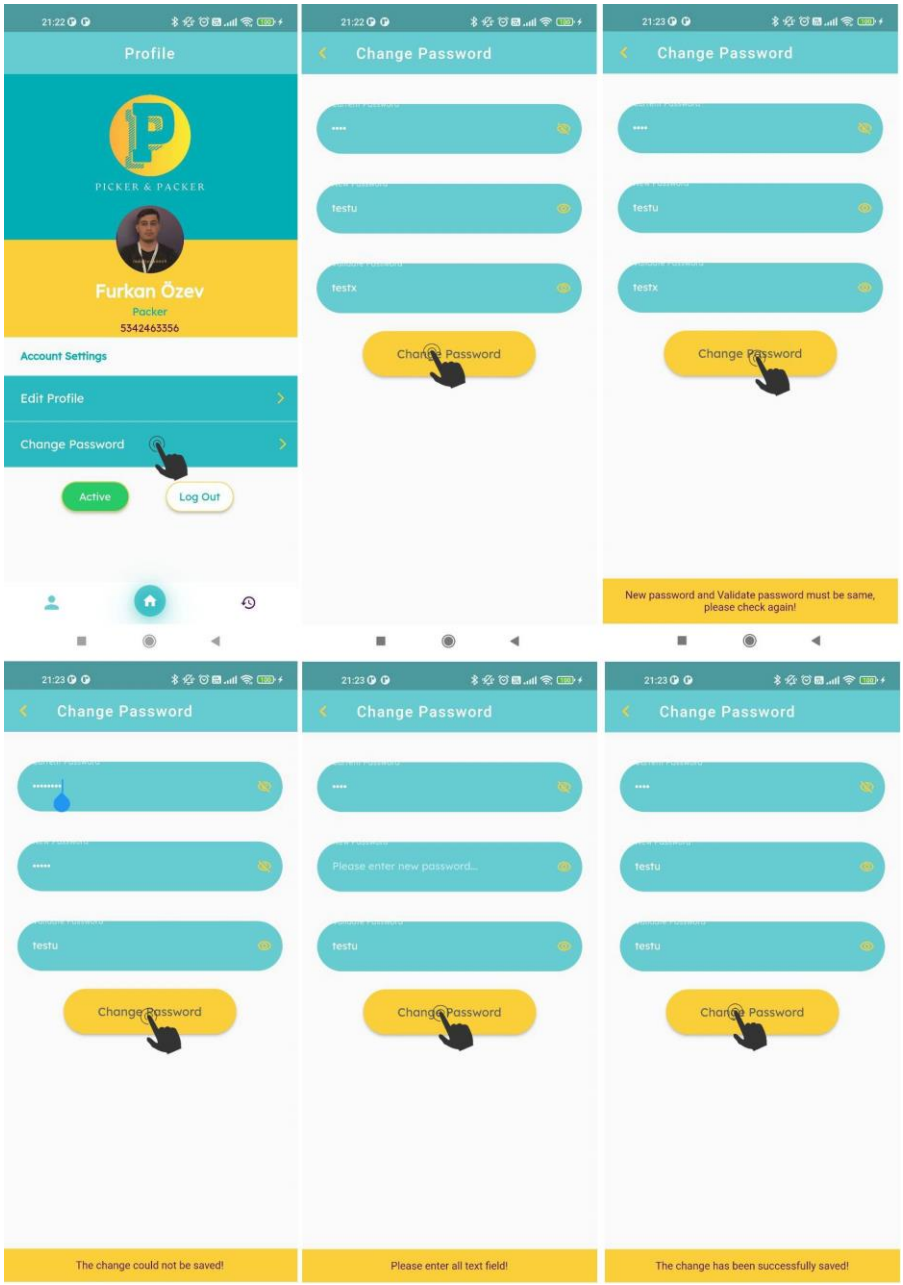

Figure 28 Packer Change Password Page

3.5.4. Packer Orders Page

The user can view open orders, view the details of the relevant order by clicking the detail button, cancel the order, scan a cart barcode to match for the order and include it in the delivery process. The barcode of the cart it scans must be match for order, it cannot scan a random cart, that is, it must be picker completted order. If there is a cold chain product in the cart, it must first confirm the collection of the cold chain product. For this, click the Collect Cold Chain button in the details section. If there is no cold chain product, there is no need to click the button. Cancel Order button is a slideable button, it needs to slide the order left to view the button. The user can update the order list by dragging the page down. Orders are sorted by delivery date, from the earliest delivered to the latest ordered.

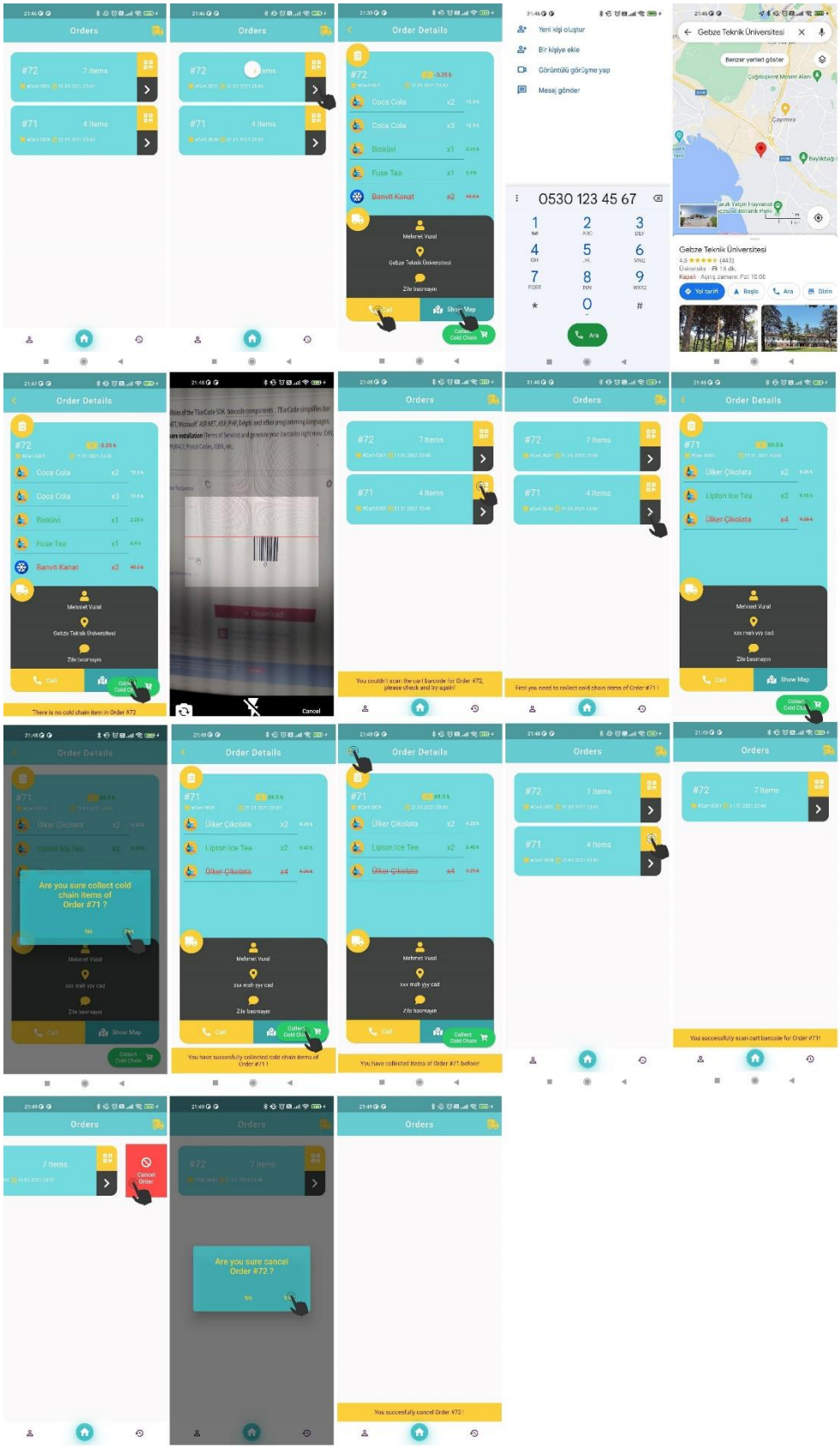

Figure 29 Packer Orders Page

3.5.5. Packer Order Details Page

The user can view all the products in the relevant order together with their quantities. User can view information such as order number, order delivery date, customer name, customer note, delivery address, call the customer by pressing the Call button, show the map by pressing Show Map button.

User can view the changes Picker has made on the order, deleted products are shown in red and added products are displayed in green. If the price difference is positive, that amount is refunded, and the negative amount is obtained from the customer.

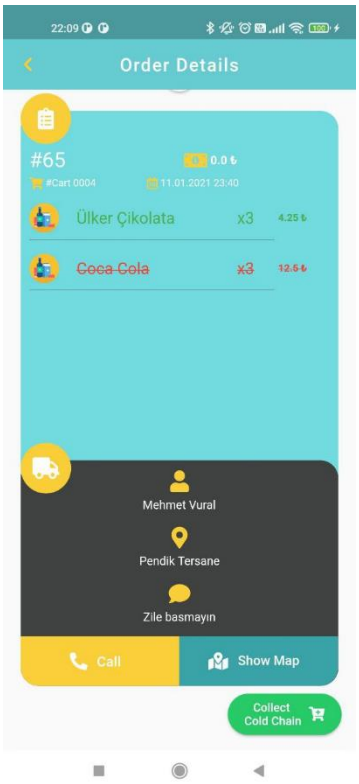

Figure 30 Packer Order Details Page

3.5.6. Packer Orders History Page

The user can view all past (cancelled and completed) orders. It can display changes in completed orders, price difference. User can delete any order from history. User can delete all orders from history. Orders are sorted by last transaction date. With the most recently modified order at the top. So the oldest order will be last. User can view

the changes Picker has made on the order, deleted products are shown in red and added products are displayed in green. If the price difference is positive, that amount is refunded, and the negative amount is obtained from the customer.

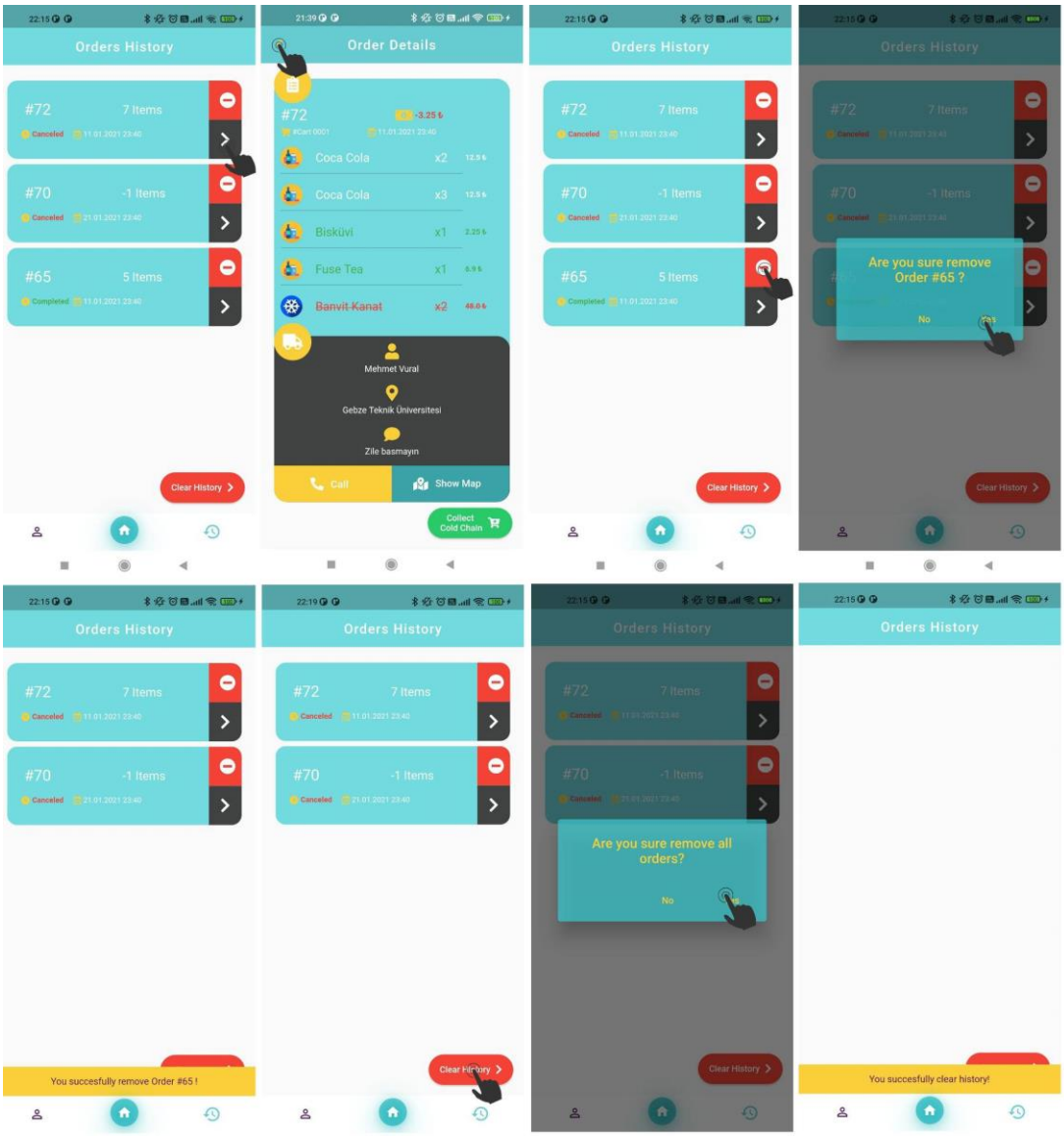

Figure 31 Packer Orders History Page

3.5.7. Packer Order History Details Page

The user can view all the products in the relevant order together with their quantities. Deleted items are shown in red, and later added items are shown in green. If there is a product replaced with an alternative product, it is displayed under the deleted

product. The price difference is also noted at the top right. If the number is positive, it indicates the amount to be refunded, if it is negative, it indicates the amount to be received from the customer. User can view information such as order number, order delivery date, customer name, customer note, order address, call the customer by pressing the Call button and Show map by pressing the Show Map button.

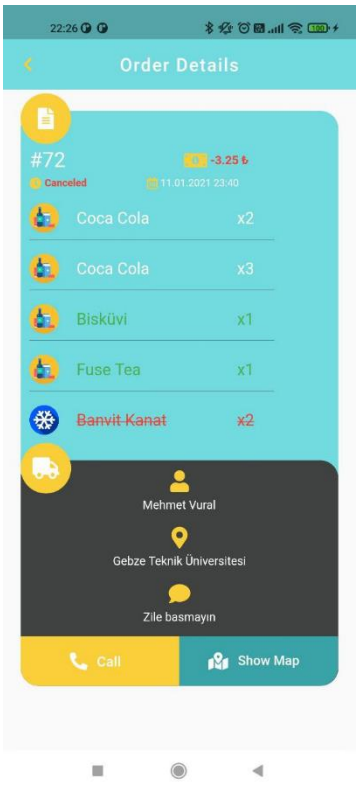

Figure 32 Packer Order History Details Page

3.5.8. Packer Order Prepare Page

The user manages the delivery process of the orders on this page. It enables the delivery of more than one order at the same time. There is a map in the top widget. User can view user's current location and navigate on the map. The location of the selected order is displayed as selected on the map. All orders in the delivery process are ranked in the most optimal way using the Directions API provided by Google Map. While doing this sorting process, it is based on instant traffic data and sorts it based on its current location. If the order process is done as suggested, user can view

the time and distance information between each order in the information section of the relevant orders. This information is also provided based on instant traffic information via Google Map APIs. The user can select the desired order instead of the suggested order and proceed to the delivery of that order. The selected order is marked on the map, when the relevant sign is clicked, it will direct the Google Map application to get a route suggestion. By clicking the Detail button, user can view all the details about the order as well as the required time and distance information if user want to deliver that order directly. When user slide the order to the left, user can see the Slidable Cancel Delivery and Complete Order buttons. When user click the Cancel Delivery button, the order goes from the delivery stage to the open order stage. When user press the Complete Delivery button, the order is delivered. It makes a refresh request by dragging the list down, pulls the current data and updates the list. [7]

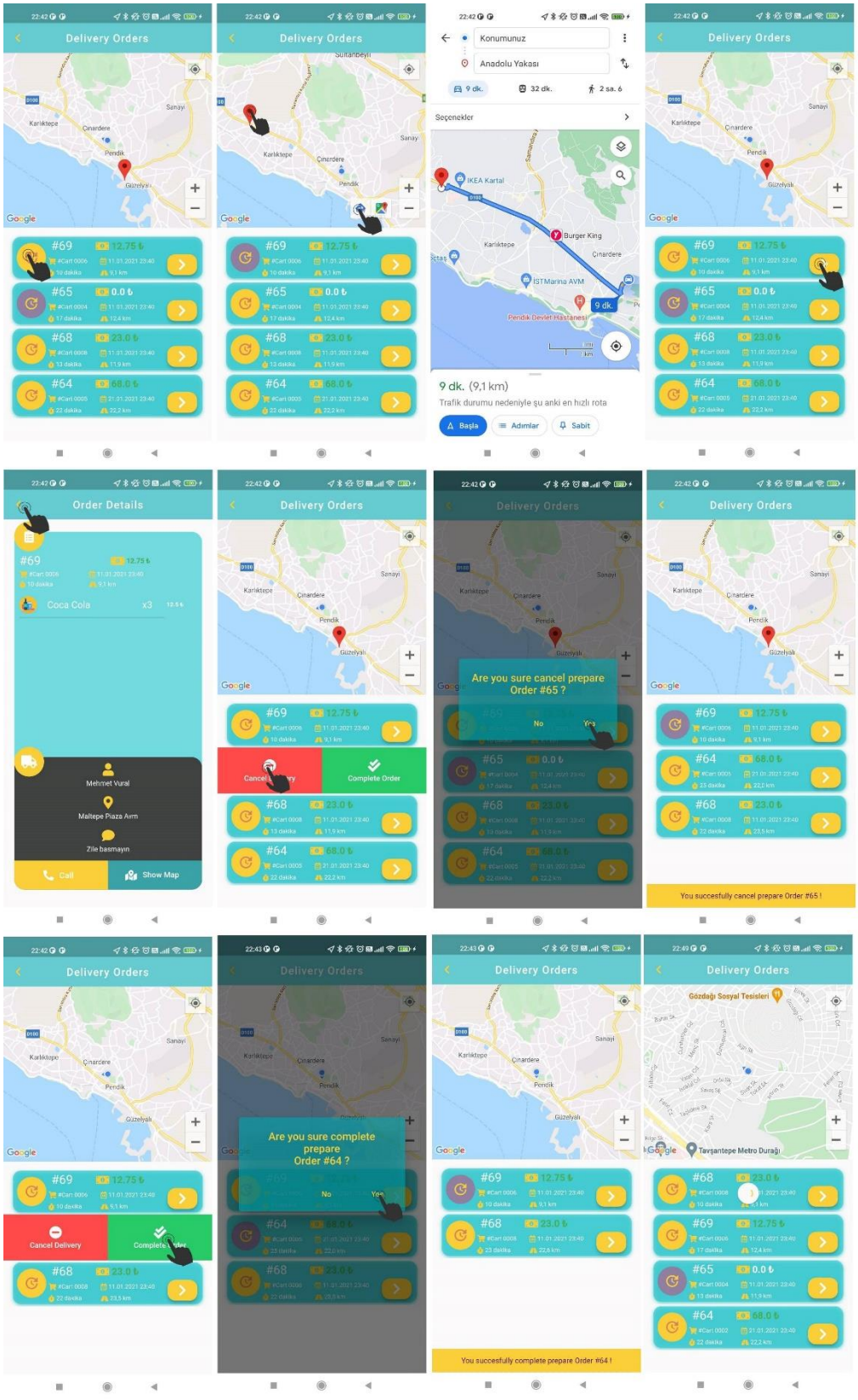

Figure 33 Packer Order Delivery Page

3.5.9. Packer Order Prepare Details Page

The user can view all the products in the relevant order together with their quantities. Deleted items are shown in red, and later added items are shown in green. If there is a product replaced with an alternative product, it is displayed under the deleted product. The price difference is also noted at the top right. If the number is positive, it indicates the amount to be refunded, if it is negative, it indicates the amount to be received from the customer. User can view information such as order number, order delivery date, customer name, customer note, order address, call the customer by pressing the Call button and Show map by pressing the Show Map button.

In addition, if the user wants to take the order directly to this address, it shows the time it will take and the distance according to the current traffic information.

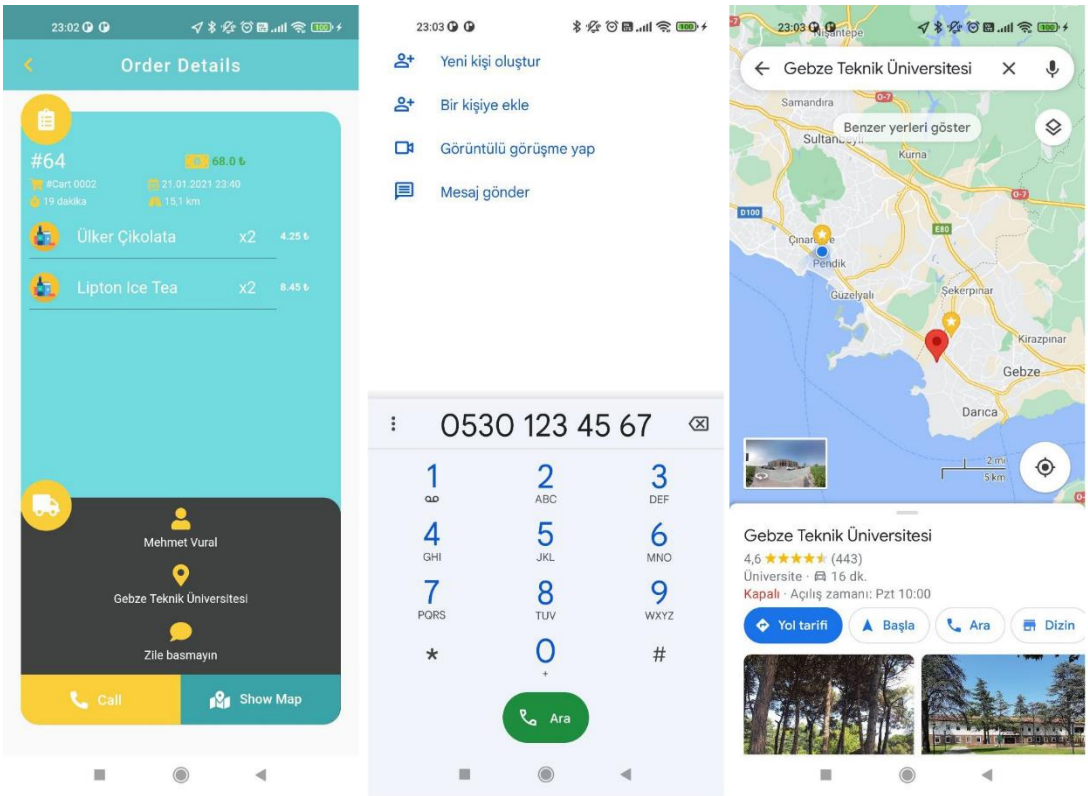

Figure 34 Packer Order Delivery Details Page

4. FLUTTER, API, DATABASE

Build the application One of the 2 components is the interface and the backend services. The features provided by the dart language in Flutter were used in the interface. Various services were provided in the backend, such as providing the functions in the interface, accessing the database and providing some functions, and integrating with 3rd party APIs. [2] [3]

4.1. FLUTTER

When the application's icon is clicked on the phone and run, it welcomes a moving splash screen. This screen takes about 2 seconds, then the login page opens. Splash screen is designed using Flutter spin_kit module. [2] [3]

4.1.1. SnackBar

It is used to briefly inform your users when certain actions occur. It is used to indicate whether the operation was successful, failed, or to issue a warning message. For example, when a user swipes an order on a list, we want to notify them that the order has been deleted. This message disappears after a certain period of time.

4.1.2. AlertDialog

An alert dialog informs the user about situations that require acknowledgement. An alert dialog has an optional title and an optional list of actions. The title is displayed above the content and the actions are displayed below the content.

4.1.3. Email Validator

Email valdiator is used to verify the email address, if the email entered does not match the format, it will not allow it.

4.1.4. AppBar

AppBar is usually the topmost component of the app (or sometimes the bottom-most), it contains the toolbar and some other common action buttons. As all the components in a flutter application is a widget or a combination of widgets. So AppBar is also a built-in class or widget in flutter which gives the functionality of the AppBar out of the box. The AppBar widget is based on Material Design and much of the information is already provided by other classes like MediaQuery, Scaffold as to where the content of the AppBar should be placed. [2] [3]

4.1.5. Navigation Bar

The bottom navigation bar consists of multiple items in the form of text labels, icons, or both, laid out on top of a piece of material. It provides quick navigation between the top-level views of an app. For larger screens, side navigation may be a better fit.

4.1.6. Refresh Indicator

When the child's Scrollable descendant overscrolls, an animated circular progress indicator is faded into view. When the scroll ends, if the indicator has been dragged far enough for it to become completely opaque, the onRefresh callback is called. The callback is expected to update the scrollable's contents and then complete the Future it returns. The refresh indicator disappears after the callback's Future has completed.

When the refresh is done, the request is sent to the backend service, and the interface is updated as a result of the incoming response.

4.1.7. Slidable

Slidable in an application can be used to perform a wide range of tasks with just a simple swipe to either right or left on the tile. It not only makes the UI very user-friendly but also saves a lot of time in doing trivial tasks which if done in other ways can be hectic and redundant to design.

4.1.8. ListView

Listview in flutter is a widget used to display items in a linear manner. For example, list view is used in apps like online shops. Since it is a scrollable widget we can display multiple items on the same screen. If the scroll direction is vertical the children will be arranged one after another from top to bottom. Every child should fill the listview horizontally. When the scroll direction is horizontal the children will be arranged from left to right. Generally, we will use ListTiles as children for a listview but we can use any widget instead. [2] [3]

4.1.9. Map Launcher

Map Launcher is a flutter plugin to find available maps installed on a device and launch them with a marker or show directions. When the user presses the Show Map button, the relevant address is directly accessed via the Google Map application. [2] [3] [7]

4.1.10. Call Launcher

Call Launcher is a flutter plugin to call number installed on a device and launch them with a stated number. When the user presses the Call button, the relevant number is directly accessed call application. [2] [3]

4.1.11. Barcode Scanner

A plugin for Flutter apps that adds barcode scanning support on both Android and iOS. It is used for purposes such as scanning basket barcodes, scanning product barcodes. [2] [3]

4.1.12. Image Picker

A Flutter plugin for iOS and Android for picking images from the image library, and taking new pictures with the camera. It was used so that the user can choose his photo from the camera, gallery. [2] [3]

4.1.13. Page Transition

A design language, such as Material, defines standard behaviors when transitioning between routes (or screens). Sometimes, though, a custom transition between screens can make an app more unique.

4.1.14. Geolocator

A Flutter geolocation plugin which provides easy access to platform specific location services (FusedLocationProviderClient or if not available the LocationManager on Android and CLLocationManager on iOS). Get the current location of the device, Get continuous location updates, Check if location services are enabled on the device. [2] [3]

4.1.15. Flutter SpinKit

It was used to provide animated custom loading. [2] [3]

4.2. BACKEND API

4.2.1. API Setup

As mentioned in the previous sections, REST APIs were deployed in cloud environments so that all functions are stored in the database where the controls are provided and to serve more than one user. All functionality of the app is provided using around 60 APIs. Directions and Distance Matrix APIs provided by Google Map were used in the map-related parts of these APIs. If the data from the application is sending data according to the relevant API, then again in Json format, if the API returns a response, data is sent in Json format. In short, the communication format between the interface and the backend is Json. [7]

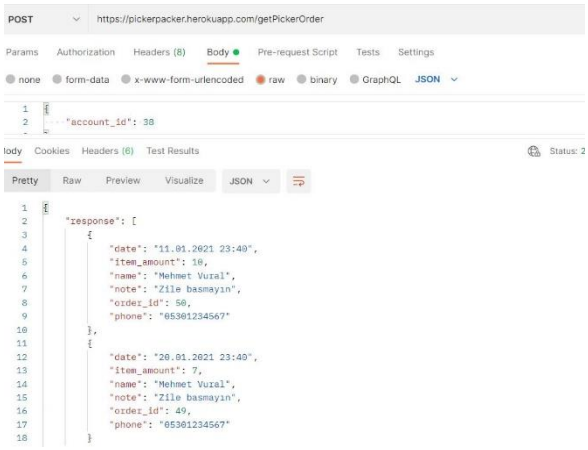

Figure 35 An API Request and Response

4.2.2. Deploy the API on the Web

A cloud platform called Heroku was used to deploy the API on the web. Thanks to this platform, a web URL containing the name of the application is provided, so when a GET type request is sent to this URL, the processed data is returned. Configurations on the Heroku platform were provided by adding Procfile, requirements.txt and runtime.txt next to the Python code. The procfile was a mechanism to report what commands were run on the Heroku platform by API containers. Requirements.txt, on the other hand, was a file that specifies the plugins required for the API to work. Runtime.txt, on the other hand, was a file specifying which Python version it would run with. After all these files were uploaded to Github, the API was activated by deploying and deploying with Heroku. [5]

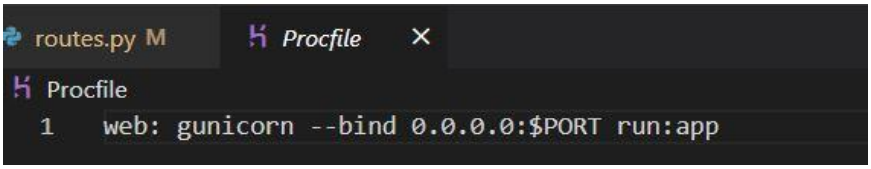

Figure 36 Procfile Content

routes.py M requirements.txt

requirements.txt

```
cachelib==0.4.1
certifi==2021.10.8
charset-normalizer==2.0.10
click==8.0.3
colorama==0.4.4
Flask==2.0.2
Flask-Session==0.4.0
greenlet==1.1.2
gunicorn==20.1.0
idna==3.3
itsdangerous==2.0.1
Jinja2==3.0.3
MarkupSafe==2.0.1
psycopg2==2.9.2
requests==2.27.1
SQLAlchemy==1.4.27
unicode-tr==0.6.1
Unidecode==1.3.2
urllib3==1.26.7
Werkzeug==2.0.2

```

Figure 37 Requirements Content

```
2022-01-10T13:43:32.494316+00:00 heroku[web.1]: Unidling
2022-01-10T13:43:32.502416+00:00 heroku[web.1]: State changed from down to starting
2022-01-10T13:43:35.820189+00:00 heroku[web.1]: Starting process with command `gunicorn --bind 0.0.0.0:35885 run:app`
2022-01-10T13:43:37.197761+00:00 app[web.1]: [2022-01-10 13:43:37 +0000] [4] [INFO] Starting gunicorn 20.1.0
2022-01-10T13:43:37.198169+00:00 app[web.1]: [2022-01-10 13:43:37 +0000] [4] [INFO] Listening at: http://0.0.0.0:35885 (4)
2022-01-10T13:43:37.198211+00:00 app[web.1]: [2022-01-10 13:43:37 +0000] [4] [INFO] Using worker: sync
2022-01-10T13:43:37.201467+00:00 app[web.1]: [2022-01-10 13:43:37 +0000] [9] [INFO] Booting worker with pid: 9
2022-01-10T13:43:37.220898+00:00 app[web.1]: [2022-01-10 13:43:37 +0000] [10] [INFO] Booting worker with pid: 10
2022-01-10T13:43:37.798509+00:00 heroku[web.1]: State changed from starting to up
2022-01-10T13:43:39.685812+00:00 app[web.1]: 10.1.12.103 - - [10/Jan/2022:13:43:39 +0000] "POST /getPickerOrder HTTP/1.1" 200 160 "-" "PostmanRuntime/7.28.4"
2022-01-10T13:43:39.686368+00:00 heroku[router]: at=info method=POST path="/getPickerOrder" host=pickerpacker.herokuapp.com request_id=1747a66a-fe08-4c57-91ab-2b1255
4baaf6 fwd="212.154.66.181" dyno=web.1 connect=0ms service=39ms status=200 bytes=306 protocol=https
2022-01-10T13:44:12.862725+00:00 app[web.1]: 10.1.12.103 - - [10/Jan/2022:13:44:12 +0000] "POST /getPickerOrder HTTP/1.1" 200 290 "-" "PostmanRuntime/7.28.4"
2022-01-10T13:44:12.863330+00:00 heroku[router]: at=info method=POST path="/getPickerOrder" host=pickerpacker.herokuapp.com request_id=418ced6c-bc6f-4444-9090-fa8360
113f37 fwd="212.154.66.181" dyno=web.1 connect=0ms service=3ms status=200 bytes=436 protocol=https
(venv) PS C:\Users\furka\Documents\dev\pickerpacker>
```

Figure 38 Heroku Logs

4.2.3. Google Map API

The Directions API is a web service that uses an HTTP request to return JSON or ML-formatted directions between locations. You can receive directions for several modes of transportation, such as transit, driving, walking, or cycling. Search for directions for several modes of transportation, including transit, driving, walking or cycling. Return multi-part directions using a series of waypoints. Specify origins, destinations, and waypoints as text strings (e.g. "Chicago, IL" or "Darwin, NT, Australia"), as place IDs, or as latitude/longitude coordinates The API returns the most efficient routes when calculating directions. Travel time is the primary factor

optimized, but the API may also take into account other factors such as distance, number of turns and many more when deciding which route is the most efficient.

The Distance Matrix API is a service that provides travel distance and time for a matrix of origins and destinations. The API returns information based on the recommended route between start and end points, as calculated by the Google Maps API, and consists of rows containing duration and distance values for each pair. This service does not return detailed route information. Route information can be obtained by passing the desired single origin and destination to the Directions API. [7]

4.3. DATABASE

4.3.1. ER Diagram

Since the data is relational, a relational database was preferred because there are many different types of data such as data of more than one user at the same time, more than one order data.

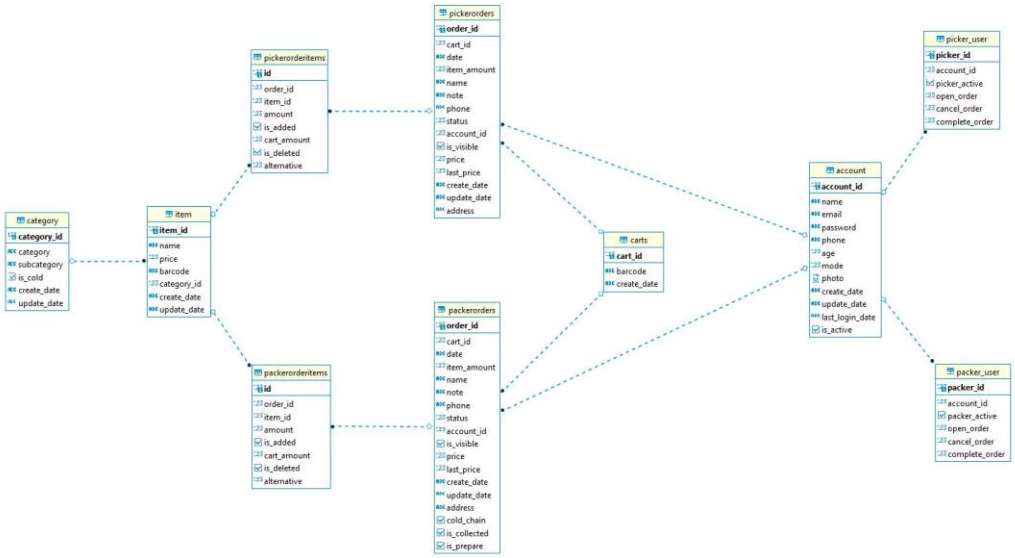

Figure 39 ER Diagram

5. RESULTS & SUCCESS CRITERIAS

This section includes device and user tests of the application. In device tests, it was first tried on Android emulator and iPhone simulators provided by Android Studio and MacOS. In the user test, the application was tested by 10 users and scored according to certain criteria. Feedback received, if any. [8]

5.1. DEVICE TESTS (CRITERIA - 1)

- The application can run on devices running on Android 7.1.1 (Nougat) and above.

5.1.1. Pixel 5 (Emulator)

- Qualcomm Snapdragon 765G.
- 6 inch, 1080 x 2340 (Full HD+) screen.
- 8 GB RAM, 128 GB internal storage.
- Android Nougat 7.1.1.1 (API level 25) x86.

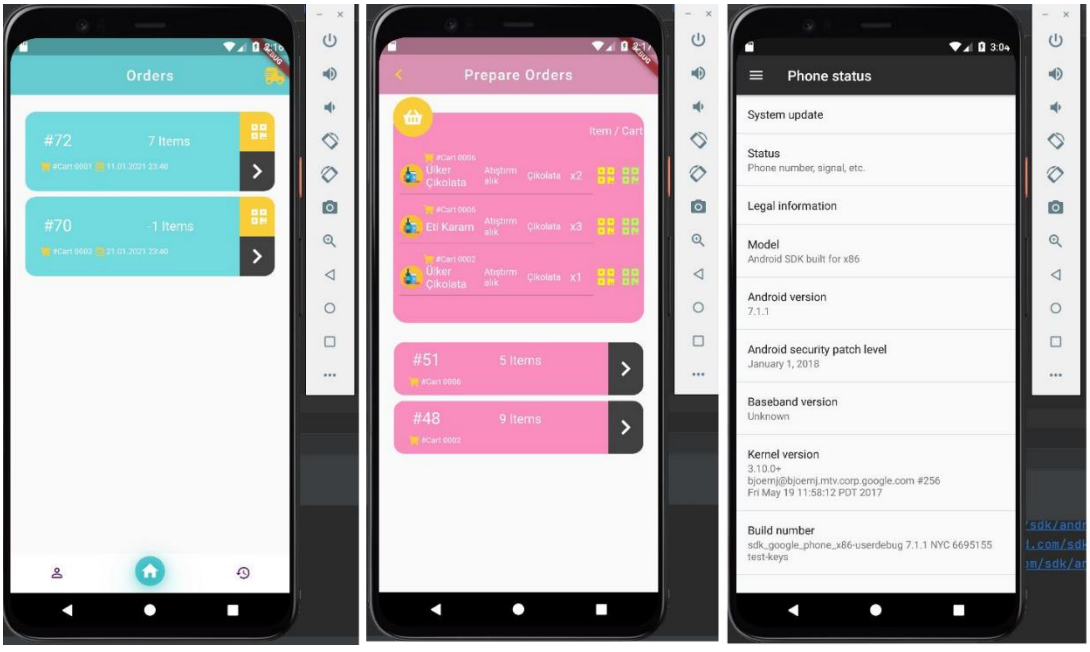

Figure 40 Pixel 5

5.1.2. NEXUS 6P (Emulator)

- Qualcomm Snapdragon 810.
- 5.7 inch, 1440 x 2560 (QHD) screen.
- 3 GB RAM, 32 GB internal storage.
- Android Nougat 9.0 (API level 28) x86.

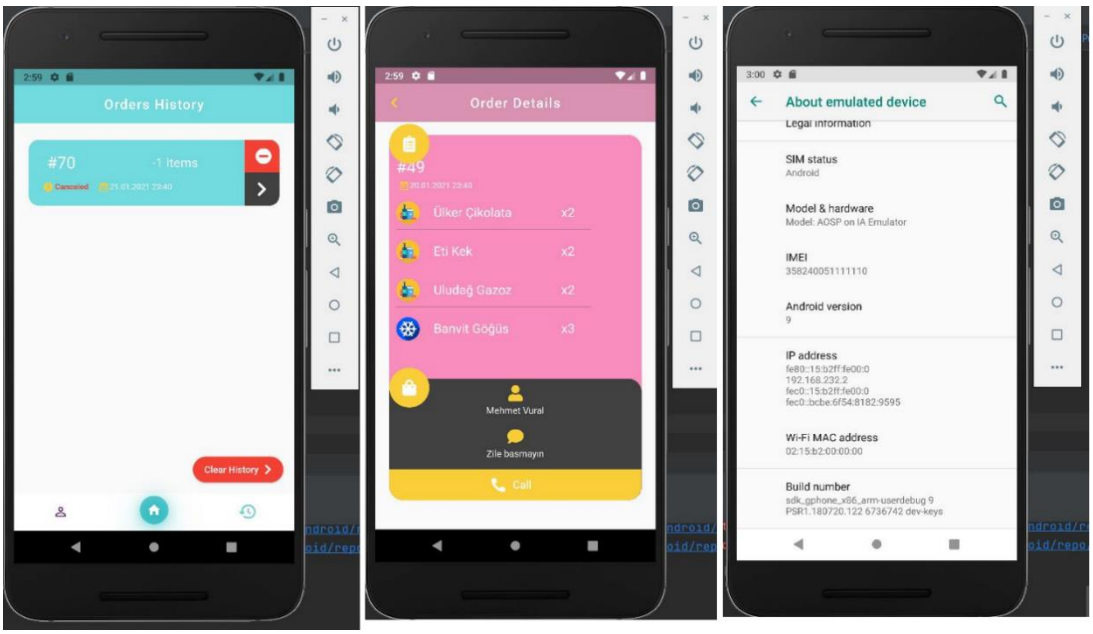

Figure 41 Nexus 6P

5.1.3. XIAOMI MI 11 Lite (Real Device)

- Qualcomm Snapdragon 732.
- 6.55 inch, 1080 x 2400 (QHD) screen.
- 6 GB RAM, 128 GB internal storage.
- Android Nougat 11.0.

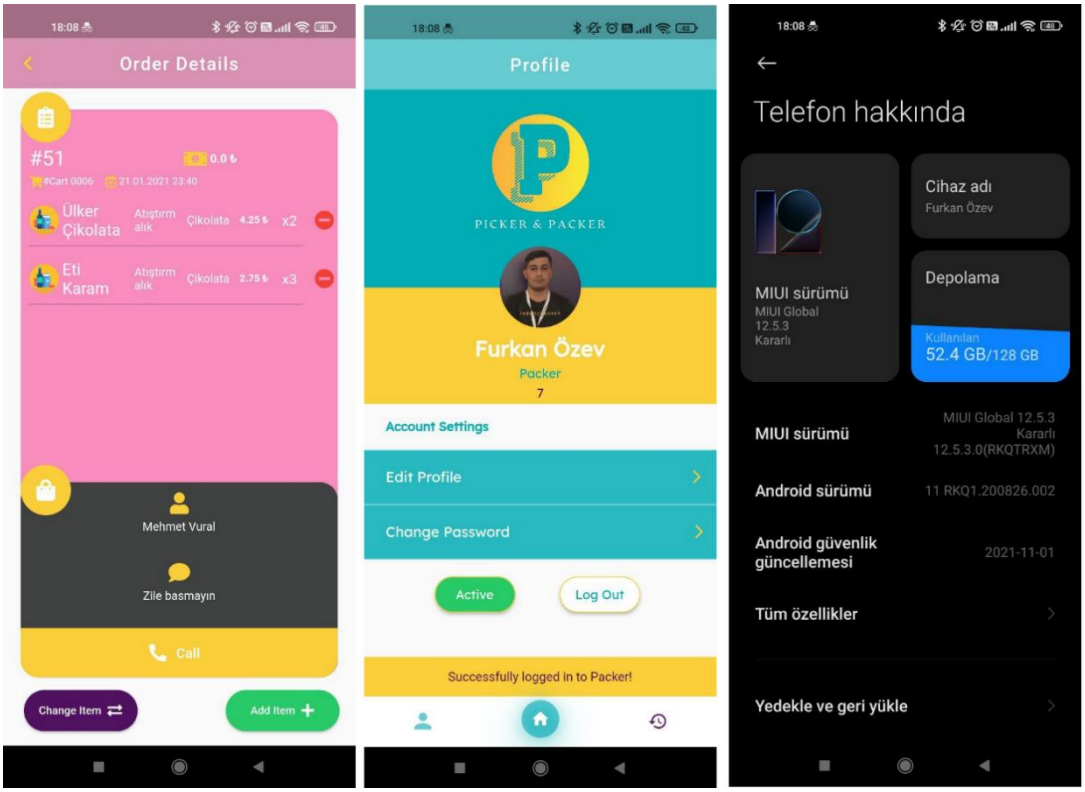

Figure 42 XIAOMI MI 11 Lite

As seen in the results, it has been tested on different types of devices, different android versions, including Android 7, Android 9, Android 11, and success criteria have been met.

5.2. USER TESTS (CRITERIA - 2)

- The application meets 70% validity criteria when tested by 10 different users. (Simplicity, Usability, Functionality, Design).

In this section, 10 users were presented with applications and users were asked to evaluate the application experience according to some criteria. These criteria are generally:

- Simplicity (Comprehensibility)
- Availability
- Functionality
- Design

It consisted of titles. In addition, as a result of all these, the user "Would you use it?" the question was asked. As a result of these evaluations, the following table was obtained.

User	Simplicity	Availability	Functionality	Design
1	9	7	8	6
2	9	9	9	9
3	9	8	10	7
4	10	8	8	7
5	9	9	9	8
6	10	10	10	9
7	9	10	9	9
8	10	9	9	8
9	8	8	7	7
10	10	10	10	8
Average	9,272727273	8,636363636	8,818181818	7,636363636

Figure 43 User Tests

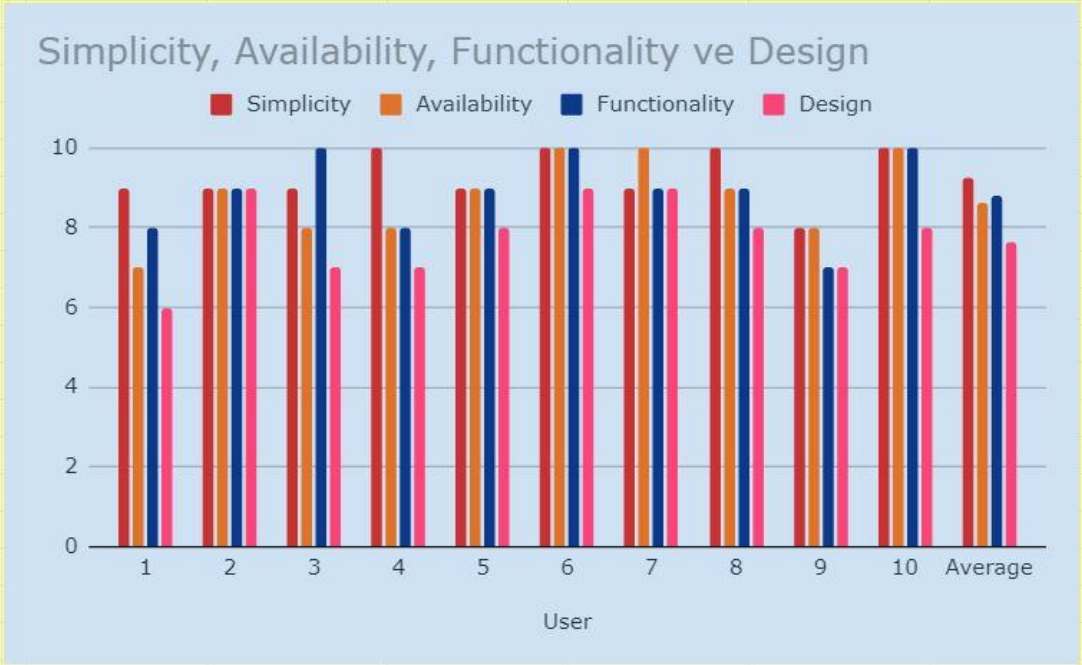

Figure 44 User Tests Graph

As seen in the results, it has received 70% validity in evaluations in terms of simplicity, usability, functionality, and design.

5.3. SCANNING AND MAP INTEGRATION (CRITERIA - 3)

- The application provides product barcode scanning and Map integrations.

5.3.1. Scan Barcode

Most pages of the application have barcode scanning functionality. Both Picker and Packer are used for both basket scanning and product barcode scanning. Therefore, this criterion has been met.

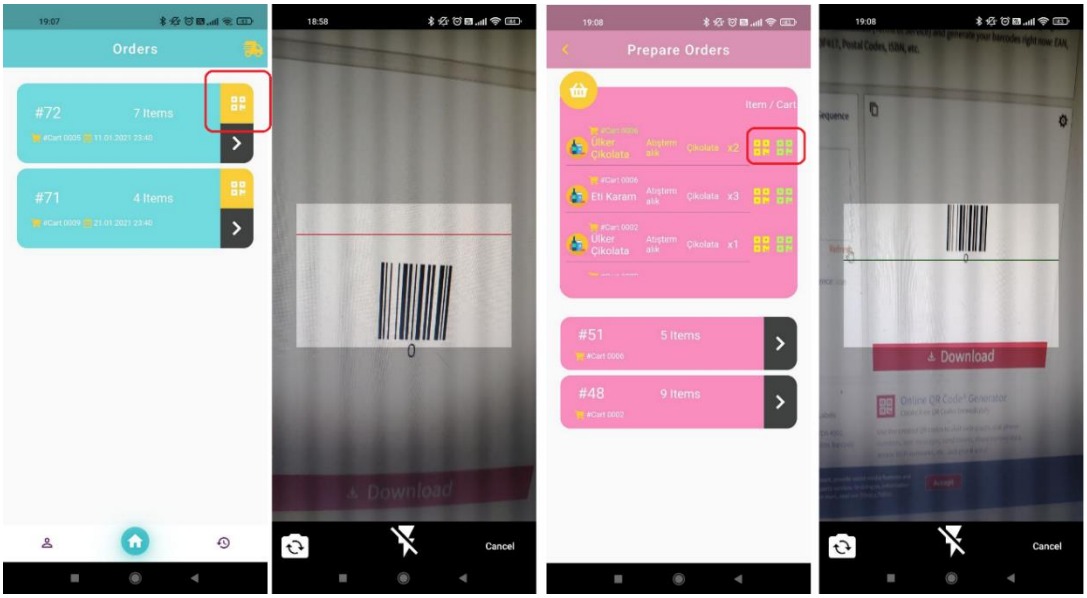

Figure 45 Scan Barcode

5.3.2. Map Integration

Packer can view the relevant address on the Google Map application and get route directions by pressing the Show Map button on all relevant pages. In addition, on the Delivery page, it can display its current location, mark the order addresses on the map, and display the distance and time to the address. The geolocator module in Flutter is used to get the current location. The map launcher module in Flutter was used for the Show Map button. Directions and Distance Matrix APIs provided by Google Map were used for Map-related functions. [2] [3] [7]

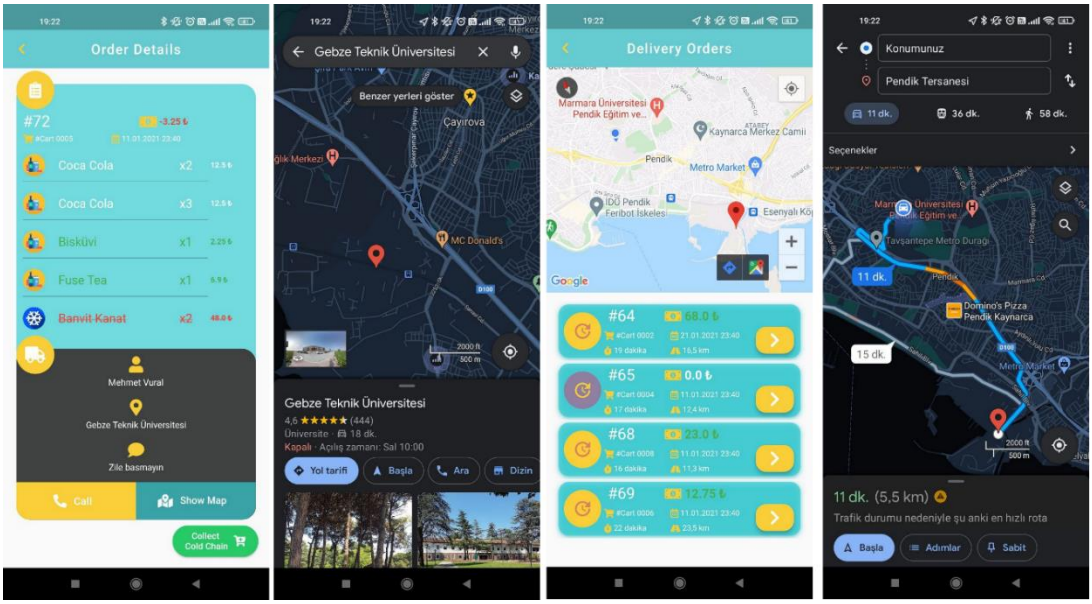

Figure 46 Google Map Integration

5.4. OPTIMIZE (CRITERIA - 4)

- The application provides optimal product collection suggestion and the most optimal route. (All permutations will be taken and compared.).

5.4.1. Optimize Route

Packer may have more than one order to deliver to addresses. Delivering orders as soon as possible from their location is a situation that will satisfy both the packer and the customer. The main criteria for optimizing orders are distance and traffic information. Today, the most well-known service that provides this data in a very effective way is the Google Map service. Google Map offers a wide variety of Map APIs. One of them is Directions API, with this API, starting and destination locations are selected, optionally Google offers this route in the most optimal way. While presenting this optimal route, it processes data such as distance to addresses, traffic information and sorts it in the most optimal way. Another API is the Distance Matrix API, which determines the distance and time from the instantaneous location to the destination locations. In this project, both services are provided for Packer. To prove this success criterion with a few examples; [7]

1. Sample – 1:

Starting Location: Pendik Center

Packet's Location:

- Kartal Center
- Pendik Marina
- Pendik Kaynarca
- Pendik Sapanbaglari
- Tuzla Shipyard
- Pendik Kurtkoy

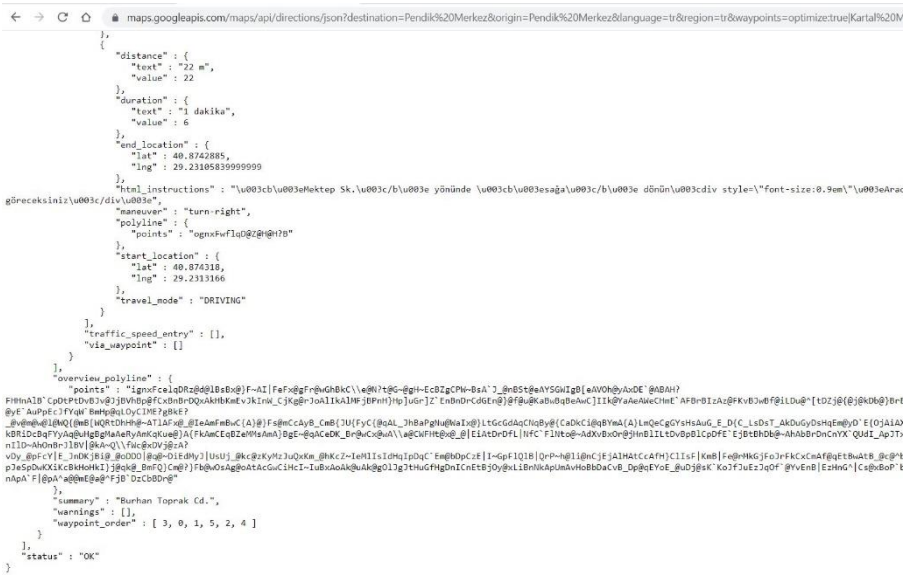

Figure 47 Google Map Rotation API Response

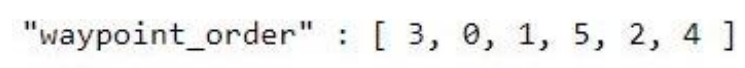

Figure 48 Optimal Rotation Response

The ranking according to the results is as follows:

- Pendik Sapanbaglari
- Kartal Center

- Pendik Marina
- Pendik Kurtkoy
- Pendik Kaynarca
- Tuzla Shipyard

This is the most optimal result for Packer to start distributing orders from Pendik Center and return to Pendik Center again. These data are suggestions created with 12.01.2022 20:00 traffic data. Suggestion order may change as there will be different traffic conditions at different times. When we look at the suggested data, the minimum time required to deliver a total of 6 orders and return to the starting position takes 1 hour 37 minutes, and the total distance is 49.5 km. The suggested route information is as follows:


```
{
   "distance": {
      "text":"1,6 km",
      "value":1640
   },
   "duration": {
      "text":"5 dakika",
      "value":301
   },
   "end_address":"Sapan Bağları, Pendik/İstanbul,
   Türkiye",
   "end_location": {
      "lat":40.8844001,
      "lng":29.221829
   },
   "start_address":"Batı Geziboyu Caddesi, Batı,
   Mektep Sk. No:18, 34890 Pendik/İstanbul,
   Türkiye",
   "start_location": {
      "lat":40.8742885,
      "lng":29.23105839999999
   },
```


Figure 49 Pendik Center – Sapanbaglari

```
{
   "distance": {
      "text":"4,9 km",
      "value":4901
   },
   "duration": {
      "text":"10 dakika",
      "value":599
   },
   "end_address":"Kordonboyu, Ankara Cd. No:1,
   34860 Kartal/İstanbul, Türkiye",
   "end_location": {
      "lat":40.8887682,
      "lng":29.18630599999999
   },
   "start_address":"Sapan Bağları, Pendik/İstanbul,
   Türkiye",
   "start_location": {
      "lat":40.8844001,
      "lng":29.221829
   },
```


Figure 50 Sapanbaglari – Kartal Center


```
{
   "distance": {
      "text":"4,8 km",
      "value":4765
   },
   "duration": {
      "text":"9 dakika",
      "value":538
   },
   "end_address":"Batı, Sahil Blv No:9 D:1, 34890
   Pendik/İstanbul, Türkiye",
   "end_location": {
      "lat":40.8712076,
      "lng":29.2330427
   },
   "start_address":"Kordonboyu, Ankara Cd. No:1,
   34860 Kartal/İstanbul, Türkiye",
   "start_location": {
      "lat":40.8887682,
      "lng":29.18630599999999
   },
```


Figure 51 Kartal Center – Pendik Marina

```
{
  "distance": {
    "text":"9,5 km",
    "value":9479
  },
  "duration": {
    "text":"20 dakika",
    "value":1201
  },
  "end_address":"Kurtköy, 34912 Pendik/İstanbul,
  Türkiye",
  "end_location": {
    "lat":40.9127659,
    "lng":29.2984069
  },
  "start_address":"Batı, Sahil Blv No:9 D:1, 34890
  Pendik/İstanbul, Türkiye",
  "start_location": {
    "lat":40.8712076,
    "lng":29.2330427
  },
```


Figure 52 Pendik Marina – Pendik Kurtkoy


```
{
  "distance": {
    "text":"7,7 km",
    "value":7702
  },
  "duration": {
    "text":"15 dakika",
    "value":896
  },
  "end_address":"Pendik, Kaynarca, 34890 Pendik/
  İstanbul, Türkiye",
  "end_location": {
    "lat":40.8793794,
    "lng":29.2581639
  },
  "start_address":"Kurtköy, 34912 Pendik/İstanbul,
  Türkiye",
  "start_location": {
    "lat":40.9127659,
    "lng":29.2984069
  },
```


Figure 53 Pendik Kurtkoy – Pendik Kaynarca

```
{
   "distance": {
      "text":"9,0 km",
      "value":8982
   },
   "duration": {
      "text":"15 dakika",
      "value":876
   },
   "end_address":"Evliya Çelebi, Tersaneler Cd.
   No:50, 34944 Tuzla/İstanbul, Türkiye",
   "end_location": {
      "lat":40.8410824,
      "lng":29.2667363
   },
   "start_address":"Pendik, Kaynarca, 34890 Pendik/
   İstanbul, Türkiye",
   "start_location": {
      "lat":40.8793794,
      "lng":29.2581639
   },
```


Figure 54 Pendik Kaynarca – Tuzla Shipyard


```
{
   "distance": {
      "text":"12,0 km",
      "value":11969
   },
   "duration": {
      "text":"23 dakika",
      "value":1361
   },
   "end_address":"Batı Geziboyu Caddesi, Batı,
   Mektep Sk. No:18, 34890 Pendik/İstanbul,
   Türkiye",
   "end_location": {
      "lat":40.8742885,
      "lng":29.23105839999999
   },
   "start_address":"Evliya Çelebi, Tersaneler Cd.
   No:50, 34944 Tuzla/İstanbul, Türkiye",
   "start_location": {
      "lat":40.8410824,
      "lng":29.2667363
   },
```


Figure 55 Tuzla Shipyard – Pendik Center

2. Sample – 2:

Starting Location: Gebze Center

Packet's Location:

- Gebze Technical University
- Gebze Eskihisar

- Tuzla Sifa
- Gebze Technical University

```
            },
            {
               "distance" : {
                  "text" : "48 m",
                  "value" : 48
               },
               "duration" : {
                  "text" : "1 dakika",
                  "value" : 24
               },
               "end_location" : {
                  "lat" : 40.8010247,
                  "lng" : 29.43156339999999
               },
               "html_instructions" : "\u003cb\u003eİbrahim Ağa Cd.\u003c/b\u003e yönünde \u003cb\u003esağa\u003c/b\u003e dönün\u003cdiv style=\'
gõreceksiniz\u003c/div\u003e",
               "maneuver" : "turn-right",
               "polyline" : {
                  "points" : "_``xFeisrDTILC\\KBALE"
               },
               "start_location" : {
                  "lat" : 40.801435,
                  "lng" : 29.43139499999999
               },
               "travel_mode" : "DRIVING"
            }
         ],
         "traffic_speed_entry" : [],
         "via_waypoint" : []
      }
   ],
   "overview_polyline" : {
      "points" : "k}_xFgjsrD|@WjA~AlCnDlDtFlBpFxAdE^W~@k@vCwBlFaEhCwBS_AGYDm@`BdAlBvAv@fAn@nApBxD`@~@tBiANGBM@QKaA?
a@D[`@\\fCpBfAx@hC`BtFvDvAzAfA`B~@nBz@`C^~@^vBZ`EDzBEjCSpCe@~CsCrO}@`Eo@dDiAfHoCfO{BrMyClPqB`LsA~GmBxL{B`MkJhh@sAdFkC~Gy@~AqCpEwIfLoRnWeh@jr@i\\pc@gE
@VSYsBoC}DgFyDqFj@}@k@|@~A~BrD~EvEhGRXt@WpA|DHXRYb@KPRTjAEHeA|CaCtFgAbDo@lBBRFVBDVRL?
`@SdAeD|EsNzBqG`@aB|AkD~A}BbCwDb@eAzBwCnFwHfBcC`EiGvAd@dATpE`A`K~BnOnDzJ~BjE~@~@}I_A|IgBa@eCm@cOgDgVwFuFkAcA[u@UhCgDpCkDnBuBn@a@vAe@l@EdAFn@?
v@Kj@Wj@c@Zo@Lg@HeAFsBHqBNmA\\cAfAeBFfuHjCiDvAoBnAaB`A{@bFgH@WdAeB`@g@D_@bSaXtNsRl^if@hBmC~AwCnBuEhAmDPi@nAwF`FyXlAcHh@yCdG]}vDiSVk@p@sCfAgGn@sC\\gAPv
`Hl@`EXfB@hAGxBg@nBoAnAyAdFqItAgBl@m@|@o@tFoC|@s@t@}@r@wAb@yARmBXmCXqAXk@`AeAl@]h@MdA?
`ARbDvA`@Jh@On@C\\Mj@g@\\q@j@uBb@qAZ\\cAzCIZARE~@MLBSEu@My@IgAd@yE^iDd@eCp@uBiA}@hA|@Sf@s@vCYhBq@rGI`AHfAPrAAn@PYBgAH[bA{C[]c@pAIVa@|A]p@k@f@]L}@Bc@G_
i@Lm@\\aAdAYj@Qp@SxA[vCEh@c@xAs@vAOPmAlAwBfAsCnA}@n@yBfCeE`H_AzA_BzAiAn@iAZqAPqA?aAOuAK}BSsCWkBOgA@iBP_AVcBv@s@h@i@v@_AlB}BvHaC|Hm@pB_@n@w@p@k@NkAFmC
yA\\iCtAgDP]S_@Wc@cBcD{CiEqCqCeBwBm@gAcCiFsCwGcC^a@FAy@iEfAyBwJuCmJiBsFqAqAyDkCuEwCRcAaBkBoBqBgAuAkA}@Vu@xA}DfAmBHYb@Hn@S"
      },
      "summary" : "D100",
      "warnings" : [],
      "waypoint_order" : [ 1, 2, 0 ]
   }
],
"status" : "OK"
}
```


Figure 56 Google Map Route API Response-2

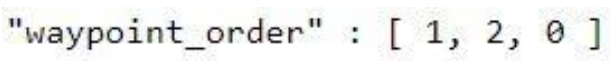

Figure 57 Optimal Route Response-2

The ranking according to the results is as follows:

- Tuzla Sifa
- Gebze Technical University
- Gebze Eskihisar

This is the most optimal result for Packer to start distributing orders from Gebze Center and return to Gebze Center again. These data are suggestions created with 12.01.2022 20:30 traffic data. Suggestion order may change as there will be different traffic conditions at different times. When we look at the suggested data, the minimum time required to deliver a total of 3 orders and return to the starting position takes 46 minutes, and the total distance is 29.8 km. The suggested route information is as follows:

When we extract all the information between the distances and look at all their permutations, we can see that the proposed route is the most optimal. Below we can see the distance and time information between both packages. As can be seen, the best solution is the suggested optimal solution. 46 minutes, with a distance of 29.8 km.

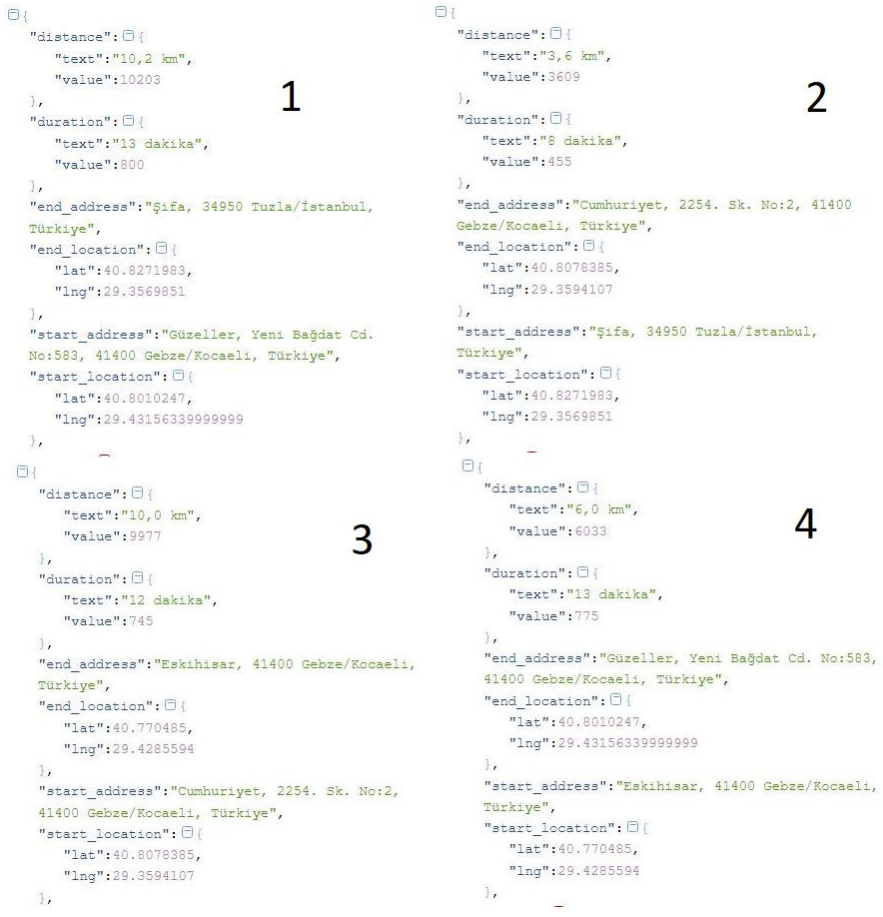

Figure 58 Optimal Route

The duration and distance of other routes can be seen below, there is also a comparison of them;

1.) Gebze Eskihisar | Tuzla Sifa | Gebze Technical University
Duration: 49 minutes, Distance: 30,7 kilometers

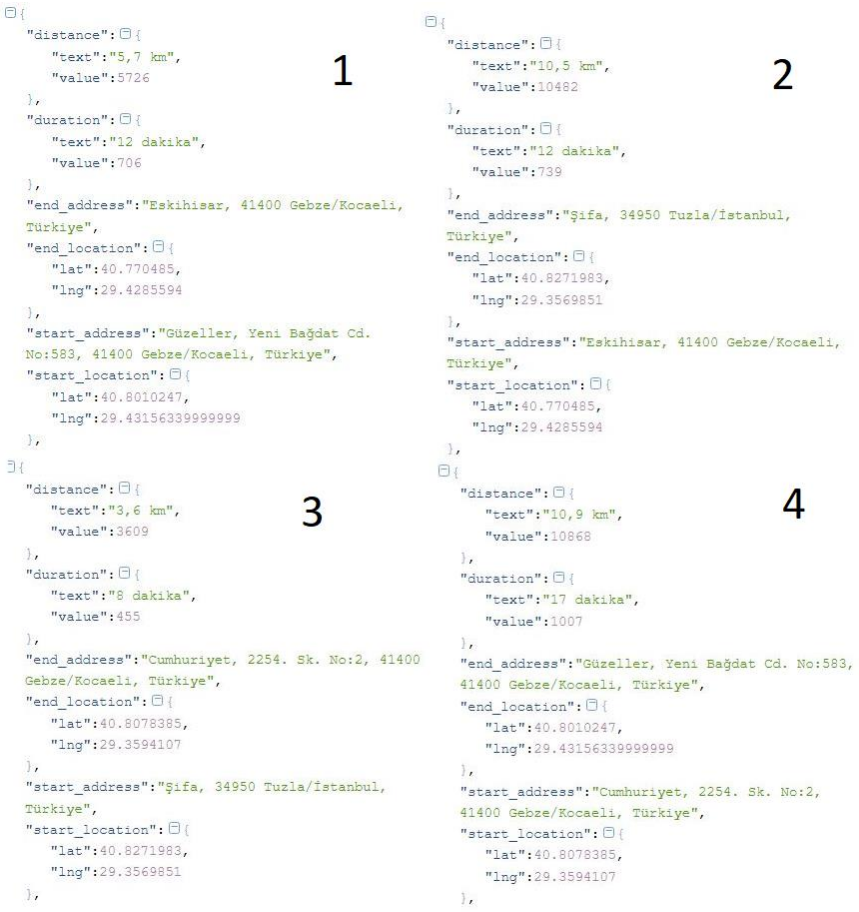

Figure 59 Optional Route-1

2.) Gebze Eskihisar | Gebze Technical University | Tuzla Sifa
Duration: 53 minutes, Distance: 33,8 kilometers

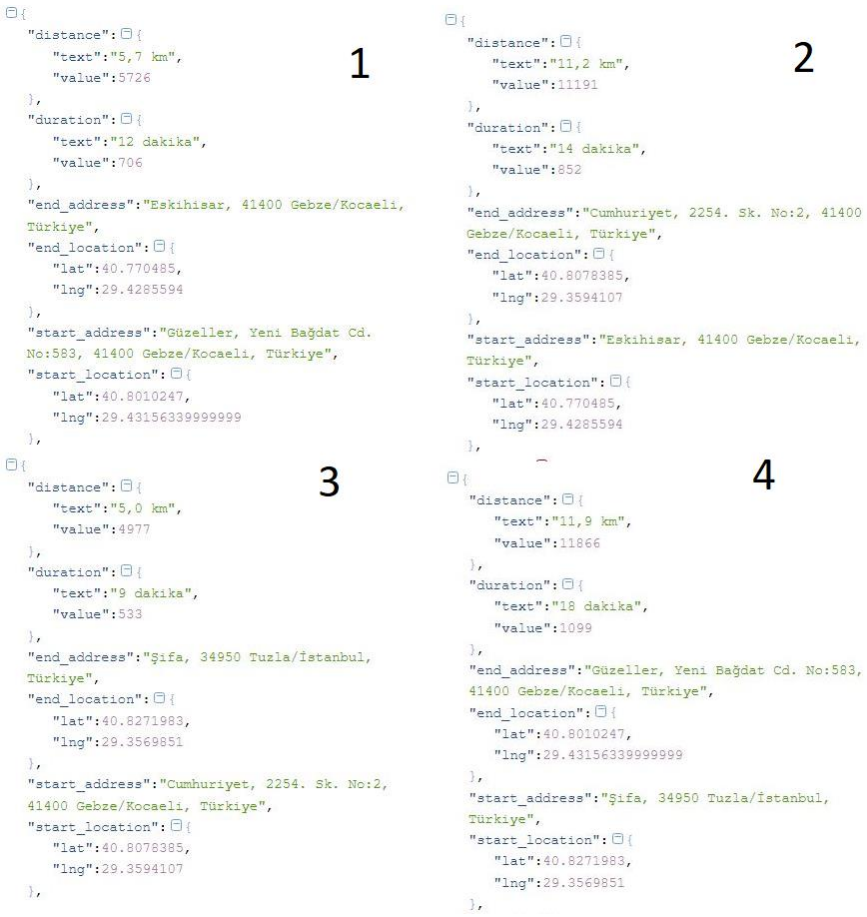

Figure 60 Optional Route-2

3.) Tuzla Sifa | Gebze Eskihisar | Gebze Technical University
Duration: 58 minutes, Distance: 44,3 kilometers

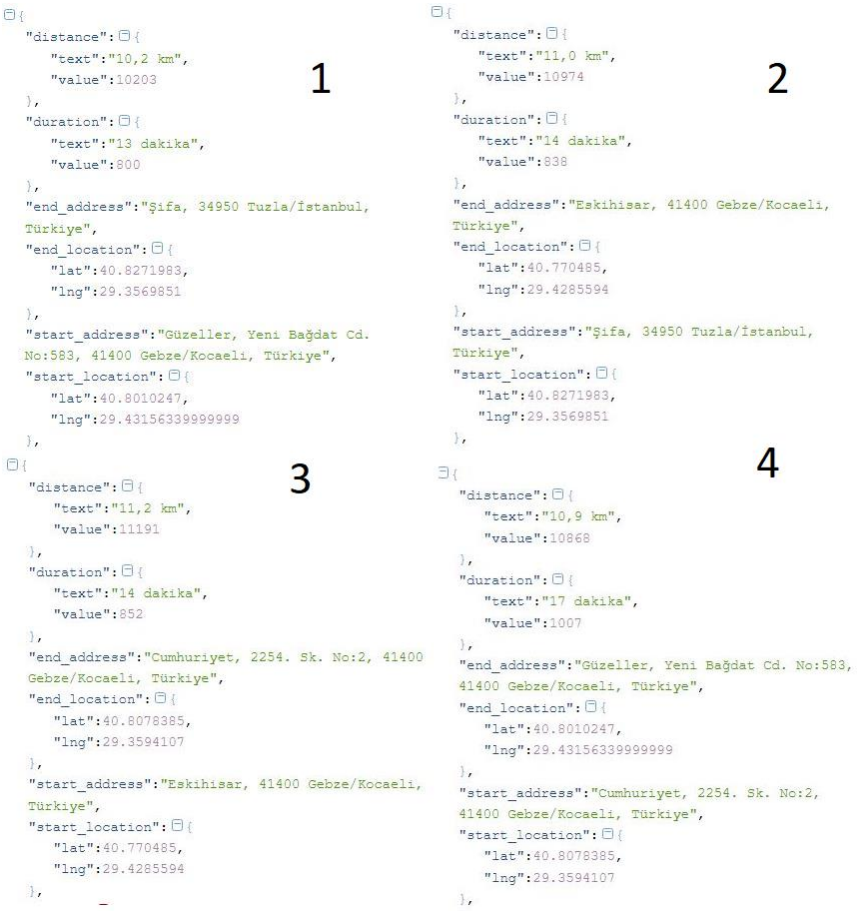

Figure 61 Optional Route-3

4.) Gebze Technical University | Gebze Eskihisar | Tuzla Sifa
Duration: 57 minutes, Distance: 43,3 kilometers

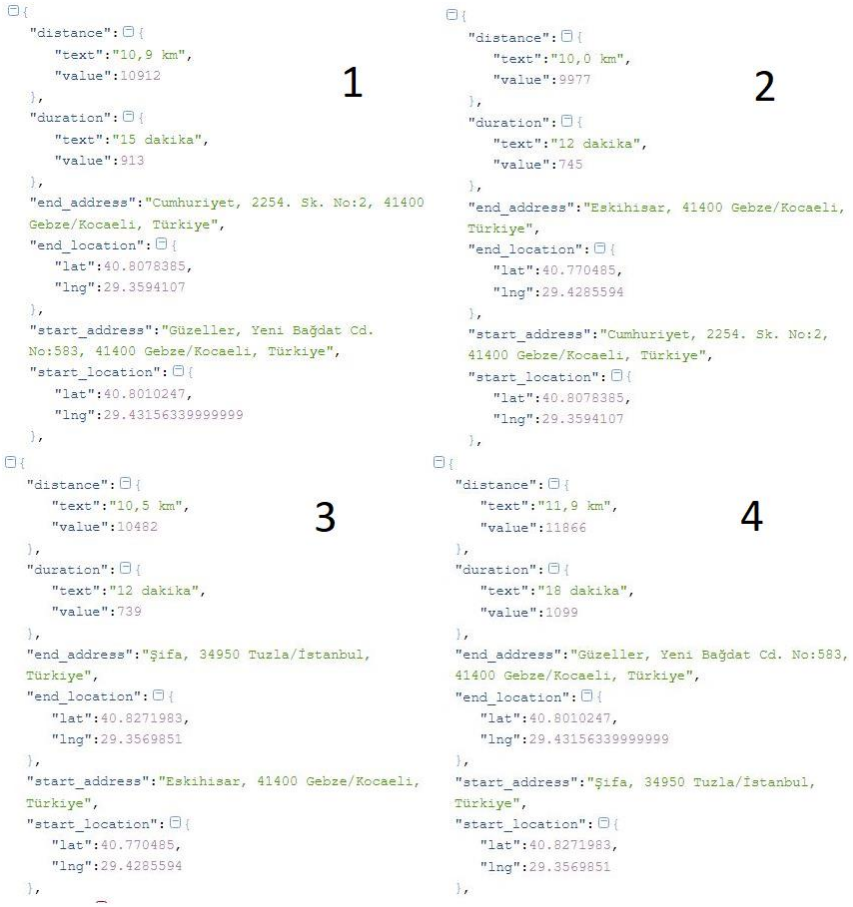

Figure 62 Optional Route-4

5.) Gebze Technical University | Tuzla Sifa | Gebze Eskihisar
Duration: 51 minutes, Distance: 32,9 kilometers

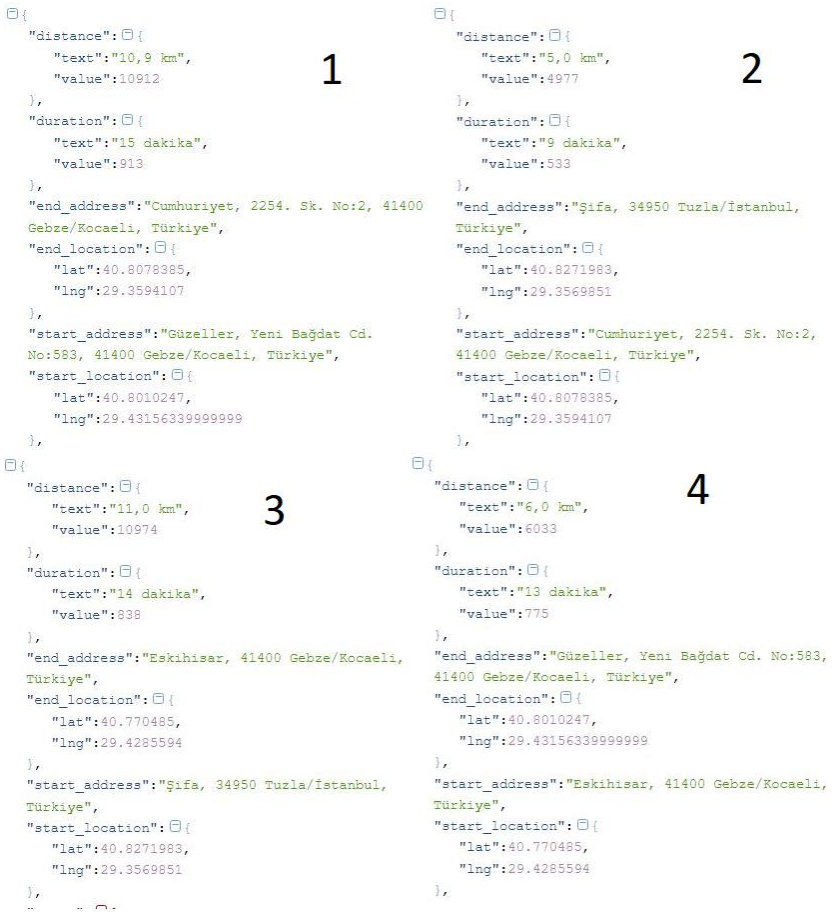

Figure 63 Optional Route-5

5.4.2. Optimize Product Collection

Picker can prepare more than one order at the same time, in which case he has to ensure that he delivers his products in the most optimal way. Otherwise, he will have to go around the market many times, this project was aimed to prevent this. It was ensured that Picker prepared the orders in one go, and did not spend any unnecessary effort. At this stage, after the products were pulled from the database, they were sorted as (category, sub-category, product name, order number). In other words, category-based sorting was done first, and for those with the same category name, sorting by sub-category, the same situation proceeded as product name and order number. In order to prevent the cold chain products from being exposed to heat and spoiled, a

suggestion was made as the last collection. Thus, the desired criterion was met. To prove this success criterion with a few examples;

pickerorderitems — Enter a SQL expression to filter results (use Ctrl+Space)

	id	order_id	item_id	amount	is_added	cart_amount	is_deleted	alternative
1	104	51	8	3	[ ]	0	[ ]	[NULL]
2	103	50	12	2	[ ]	0	[ ]	[NULL]
3	89	49	8	2	[ ]	0	[ ]	[NULL]
4	96	48	12	3	[ ]	3	[ ]	[NULL]
5	64	48	14	2	[ ]	0	[ ]	[NULL]
6	62	49	15	3	[ ]	0	[ ]	[NULL]
7	69	50	17	2	[ ]	1	[ ]	[NULL]
8	111	51	17	2	[ ]	1	[ ]	[NULL]
9	61	49	20	2	[ ]	0	[ ]	[NULL]
10	105	51	21	2	[ ]	0	[ ]	[NULL]
11	66	48	22	3	[v]	2	[ ]	[NULL]
12	98	50	20	2	[ ]	0	[ ]	[NULL]
13	99	50	23	1	[ ]	0	[ ]	[NULL]
14	83	51	24	2	[ ]	0	[ ]	[NULL]
15	107	48	26	1	[v]	0	[ ]	[NULL]
16	110	49	26	1	[v]	1	[ ]	[NULL]
17	109	49	27	4	[v]	0	[ ]	[NULL]
18	63	50	29	4	[ ]	0	[ ]	[NULL]
19	112	51	29	4	[ ]	0	[ ]	[NULL]
20	82	48	30	2	[ ]	1	[ ]	[NULL]

Figure 64 Orders' Items

item — Enter a SQL expression to filter results (use Ctrl+Space)

	name	price	barcode	item_id	category_id	create_date	update_date
1	Lipton Ice Tea	8.449999809	Item 0002	8	145	2021/12/13, 20:56:52	2021/12/13, 20:56:52
2	Ülker Çikolata	4.25	Item 0001	12	150	2021/12/13, 21:36:35	2021/12/13, 21:56:56
3	Coca Cola	12.5	Item 0003	14	148	2021/12/13, 21:36:35	2021/12/13, 21:56:56
4	Eti Kek	1.5	Item 0004	15	1	2021/12/13, 20:56:52	2021/12/13, 20:56:52
5	Banvit Kanat	48	Item 0005	17	149	2021/12/13, 20:56:52	2021/12/13, 20:56:52
6	Eti Karam	2.75	Item 00043	20	150	2021/12/13, 20:56:52	2021/12/13, 20:56:52
7	Fuse Tea	6.900000095	Item 00051	21	145	2021/12/13, 20:56:52	2021/12/13, 20:56:52
8	Uludağ Gazoz	8	Item 00052	22	148	2021/12/13, 20:56:52	2021/12/13, 20:56:52
9	Bisküvi	2.25	Item 00053	23	1	2021/12/13, 20:56:52	2021/12/13, 20:56:52
10	Banvit Göğüs	42	Item 00054	24	149	2021/12/13, 20:56:52	2021/12/13, 20:56:52
11	Godiva	19.5	Item 0005422	26	150	2021/12/13, 20:56:52	2021/12/13, 20:56:52
12	Didi	9.5	Item 000542	27	145	2021/12/13, 20:56:52	2021/12/13, 20:56:52
13	Kraker	1.25	24	29	1	2021/12/13, 20:56:52	2021/12/13, 20:56:52
14	Fanta	10.75	25	30	148	2021/12/13, 20:56:52	2021/12/13, 20:56:52

Figure 65 Items

	category	subcategory	is_cold	category_id	create_date	update_date
1	Atıştırmalık	Çikolata	[ ]	150	2021/12/13, 15:59:18	2021/12/18, 20:50:04
2	İçecek	Soğuk Çay	[ ]	145	f	2021/12/13, 15:53:26
3	İçecek	Gazlı İçecekler	[ ]	148	2021/12/13, 15:57:40	2021/12/18, 20:50:04
4	Atıştırmalık	Diğer	[ ]	1	2021/12/13, 15:21:42	2021/12/13, 15:53:26
5	Et&Tavuk&Balık	Tavuk Ürünleri	[v]	149	2021/12/13, 15:58:57	2021/12/18, 20:50:04

Figure 66 Category

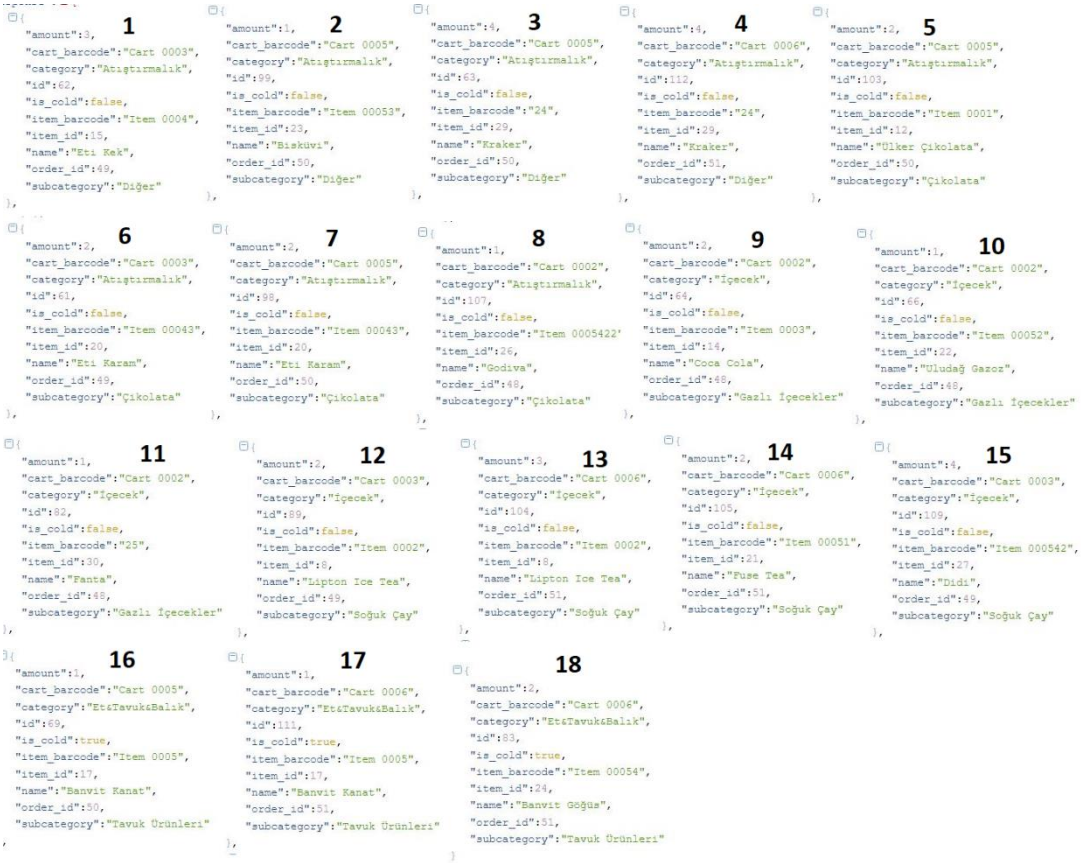

Figure 67 Optimal Item Suggestion

Figure 68 Optimal Item Suggestion App

6. DISCUSSION AND CONCLUSION

In this graduation project, a market employee application was made. The main and original purpose of this project is to have the necessary and predictable functions suitable for the use of 2 different types of personnel, to work in the Cloud environment, to have functions such as Optimize route and product recommendation. Employees are of two types, Picker and Packer, and contain many functions. Login to the application by e-mail or phone and password. Picker or Packer selection is made, he enters if he is authorized. Can view and update profile information, login information, photo. It can handle orders, past orders and order preparation, delivery processes. You can view detailed information about orders, information about products and make changes. Can view customer information. It is directly connected to the database and updates and controls are made continuously. Orders are automatically assigned to the employee with less open orders. It can change the employee status from active to passive. In his passive state, he is prevented from receiving any new orders. All these functions have been developed to be available for both Android mobile devices and IOS mobile devices. To provide these features, Flutter and Android Studio were used as development tools, Postgresql serving in Cloud environment as database system, Heroku and Render providing Cloud Service, Google Map API for Map applications and APIs, Python Flask for creating Backend RESTAPIs. [1] [2] [3][4] [5] [6] [7] [8]

As a result, this graduation project can be a good example or prototype for those who will work in this field. It can also be further developed and turned into a real commercial application with user feedback and various features. However, since it is an application that fulfills its purposes in the current situation, it can be easily used by the user.

In the last case, the product can be integrated and used in a commercial service, order-taking system. A store management system can easily be developed or integrated.

A new store management system connected to this application could be one of the future development plans.

7. REFERENCES

[1] Python Flask, Guide [online], https://flask.palletsprojects.com/en/2.0.x/
[2] Flutter Development, Guide [online], https://flutter.dev/
[3] Flutter Documents, Guide [online], https://docs.flutter.dev/development/
[4] RestAPI, Guide [online], https://restfulapi.net/
[5] Heroku, Guide [online], https://devcenter.heroku.com/start
[6] Render, Guide Documents [online], https://render.com/docs
[7] Google Map Console API, Guide [online], https://console.cloud.google.com/google/maps-apis
[8] Google, Android Studio Developer [online], https://developer.android.com/studio/*
[9] Stackoverflow [online], https://stackoverflow.com/questions/*
[10] Lucidchart [online], https://lucid.app/lucidchart
[11] Geeksforgeeks [online], https://www.geeksforgeeks.org/*
[12] Looka, Free Logo Maker, https://looka.com/
[13] Flaticon [online], https://www.flaticon.com
[14] FlutterFlow [online], https://flutterflow.io