Albanian Courier WebApp Requirements Specification

Final Version

June 17, 2022

AC ALBANIAN COURIER

POSTË EKSPRES

Group Members:

Arb Koçi

Romeisa Aliu

Vanesa Myftari

Xhoi Muça

Table of Contents

1. Executive Summary

1.1 Project Overview

Albanian Courier Service is a software product that aims to facilitate the current postal services by working as an intermediary between the agencies that want to ship their products and the couriers. Both the agencies, and the couriers, will be able to connect with each other through a user-friendly interface where each user will have their own menus and where everything will be administered by the admin, which in our case will be the CEO of the Albanian Courier company. The agencies will be able to create orders, gather them, print the bill and keep track or the orders created. Overall, the agencies themselves are the clients of the postal firm that will be using our software, and as such they will be provided with all the options required in the software. On the other side, the couriers will be able to take the orders from the agencies, deliver them to the respective buyer, and at the end of the day generate a daily report to keep track of their finances.

1.2 Purpose and Scope of this Specification

In scope

The following specifications describe the purpose of this product and how it intends to be used:

● The purpose of this project is to facilitate the process of delivering packages by acting as a medium of connection between the agencies and the couriers of the postal firm.

● Target users are the agencies that choose to work with the postal firm and the employed couriers of the postal firm.

● All the users of this software must be within the Albanian borders.

● A user-friendly interface and a software that can be accessed from the web and used through different smart devices will be provided.

● The admin will register or deactivate users' accounts.

● Users' personal information is guaranteed to be secure and not misused.

Out of Scope

The following specifications aim to elucidate the terms and conditions of using this product:

● This software does not take responsibility over the damaged packages gathered by the agencies. The postal firm will only be aware of the price and weight of the package.

● In case of inconveniences with the product, maintenance services will be offered immediately to the party experiencing the troubles.

2. Product/Service Description

2.1 Product Context

This software is independent and self-contained. It serves the context of creating a comfortable digital connection for the postal service companies and their clients. The product will offer flexibility for its users, since it will be able to be accessed from everyone with an internet connection.

2.2 User Characteristics

Courier Service will have three types of users that will be differentiated by their role:

- Agencies
- Couriers
- Admin
  - **Agencies** will be the clients of the postal firm. They will have their own unique interface and their menus for logging in and changing their password, creating an order, gathering the orders, printing the bills as well as a dashboard for keeping track of the orders.
  - **Couriers** will be the employed staff of the postal firm. They will have a unique interface as well as the menus for logging in, opening the bags, delivering the orders and generating a daily report at the end of their shift to keep track of their finances.
  - **Admin** will preferably be the CEO of the postal firm, but it can be anyone with the competence to surveil the overall work of all the users. The main duty of the admin will be to create or deactivate users according to the circumstances, and have a general overview of the daily reports of his employees.

2.3 Assumptions

- It is assumed that a reliable internet connection is available from every user that wishes to access the application.
- It is assumed that the product will be able to run and work efficiently in every browser.
- We expect the project to be fully delivered in 12 weeks.

2.4 Constraints

- Lack of cooperation within the group members may lead to project failure.
- Lack of cooperation from the admin may lead to delays in user registering or deleting.
- Further constraints may be encountered.

2.5 Dependencies

- This software product is dependent in the fact that the users are connected to the internet.
- Every user is dependent on having a smart device to access and perform operations on the product.
- Login and authentication are required for every user so that it can use the software.

3. Requirements

3.1 Functional Requirements

CS – The courier service software product

FR – Functional requirements for the product

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
CS_FR_01	User registration	The registration of each user can only be done by the admin. The users will be identified by a unique username, a password for their account and the user role. Users must be registered in order to access the services of the software.	1	15/04/22	Arb Koçi
CS_FR_02	User Login	The user will login to the system through itsusername and password for his identified role. The user won't be able to login if the password is incorrect and if his account hasn't been registered by the admin for hisspecific role.	1	15/04/22	Arb Koçi
CS_FR_03	Friendly, distinct user-interface	A unique user-interface will be provided for the different types of users. Ease of usabilityand navigation through the individual menusof each role will be provided.	1	15/04/22	Arb Koçi
CS_FR_04	User update account details	The software will provide the users the abilityto change their passwords at any time.	1	15/04/22	Arb Koçi
CS_FR_05	Admin User Role	Admin's user-interface will consist of three menus that represent its role functionality. The first menu will be to register users. The second menu will be to deactivate users according to the circumstance, for example a courier that no longer works for the postal firm. The third menu will be a general overview of the daily reports from each employee(courier).	1	15/04/22	Arb Koçi

CS_FR_06	Agency User Role	Agencies that have chosen to work with the postal firm operating on our software will have a user-interface that will consist of 5 menus that represent its role functionality. The first one is authentication menu and interface to change account password. The second one is to create the order. The third one is to print the bill for each order with a unique QR code to be scanned by the courierafter delivery. The fourth one is to gather the orders in such way that every agency has an employee from the postal firm to handle the packaging. One agency may have one or more couriers. At last, a menu for keeping track of the orders through the report dashboard will be provided.	1	15/04/22	Arb Koçi
CS_FR_07	Courier User Role	The courier employees of the postal firm will be responsible for delivering the packages from the agencies to the respective buyer. They will be able to interact with the agencies through the presence of three menus on theiruser-interface. The first one will be a menu for accessing the orders that are created by the agencies. A courier may deliver packages from one or more agencies. The second menu will be for the delivery process.The courier should scan the QR code on the printed bill from the agency if he has delivered the order to the buyer. If the order gathered from the agency hasn't been delivered in the expected day for several circumstantial reasons, the order should be gathered again and a reason for not delivering must be	1	15/04/22	Arb Koçi

		mentioned. The last menu consists of a daily report dashboard that will generate financial overviews for each courierat the end of their day.			

3.2 Non-Functional Requirements

3.2.1 Product Requirements

3.2.1.1 User Interface Requirements

Given that the software will be a web application, the user interface of the software shall be compatibleto any browser and operating system. Initially, a login authentication page will be displayed to every user, distinct for each role, where they can insert their registered credentials: username and password.If the login is successful, the user will be redirected to the unique functionalities for its role. Administrator can create new users or deactivate current ones

3.2.1.2 Usability

Learnability

- The regular user interface will be of a minimal user-friendly design.
- Every role will be guided on how to use the interface.
- The software will be designed to be easy to learn and intuitive

3.2.1.3 Efficiency

3.2.1.3.1 Performance Requirements

- The performance shall depend upon the internet connection speed of each user.
- The software shall support unlimited users and data to be loaded.

3.2.1.3.2 Space Requirements

The server is built by the client, so the space required is not a problem. Since there is only information and no pictures, it doesn’t take too much space. The test database is built with a 250 GB space and can be enough for 5 years.

3.2.1.4 Dependability

Availability

The software will be available 24/7 for every user wishing to access the product. It will cover and operate in the whole Albanian region, given that agencies may have orders from every city of Albaniaand the postal firm will have employees covering specific geographical regions within Albania.

Reliability

The system should be reliable during working hours for all the users to access the system, provided that the packages are delivered during the day. Downtime time should be scheduled for early hours of the day, 01:00-05:00 since the traffic would be the lowest.

Monitoring

Any changes on the system will be kept through logs for user access and user changes. CPU usage and RAM usage of the server must be kept below 80%.

Maintenance

Simplicity of coding structure and documentation throughout the whole development process will beprovided so that going back to improve and maintain the software is easily doable.

Integrity

The integrity of the system will be maintained through logs for any changes that may happen on the system.

3.2.1.5Security

The factors that will protect the system from malicious or accidental access, modification, disclosure,destruction, or misuse for our product are as follows:

- Encryption of data
- Authentication Middleware in C#
- Token for tracking the timeline for every user activity given that all users will be within Albania

3.2.2 Organizational Requirements

3.2.2.1 Environmental Requirements

The software will be operating 24/7. The users will be able to access the software at any time as it will be able to operate on any device with an internet connection, provided that it is Web based. The users to access the system must be within Albanian territory, since the Albanian Courier company operates in this state.

3.2.2.2 Operational Requirements

The software is a web application that will serve as a platform for the courier company to manage and track the work of its employees and its clients, the couriers and the agencies respectively. The admin will have the responsibility of registering its users, while both users will benefit from this transparent and efficient system as they will be able to provide details and generate reports for every package being packed or delivered.

3.2.2.3 Development Requirements

The technologies that will be used to develop the software are listed below:

- Front-End Development: HTML, CSS, Bootstrap, React
- Database: MS Azure
- Back-End Development: C# ASP .NET Core Web App

3.2.3 External Requirements

3.2.3.1 Regulatory Requirements

The system shall not disclose any sensitive information regarding its users and the details of the packages being delivered.

3.2.3.2 Ethical Requirements

Users are expected to share a certain amount of their data within the system. Agencies should provide data about their packages, their legal name, their address. Couriers should provide data about their deliveries, and if not delivered, state their reason for not doing so.

3.2.3.3 Legislative Requirements

The software that we will produce will be in compliance with Albanian laws, standards and regulations imposed by law enforcement agencies of our government.

3.2.3.3.1 Accounting Requirements

No financial transactions will be required to be conducted from the users. The platform will serve as a facilitating interface between the Albanian Courier company and its end users.

3.2.4 System Interface/Integration

The software will be available for access from every machine, despite of the operating system provided that it is Web based. No special installation will be needed to be made in order for its end users to use it.

3.2.5 Data Management

There will be different types of data that will be stored in our database:

- The agencies will be identified with attributes that correspond to their unique ID, their name, NIPT number, address.
- The couriers will be identified by their unique ID, username, password and their clients.
- Moreover, data regarding deliverable packages such as their weight, price, tax price, state and person to be delivered to will be stored.

- Daily reports consisting of number of delivered packages, the area covered and total price summarizing general overview of the courier's work will be stored.

3.2.6 Portability

The product is independent of portability requirement because it is accessed through web from any device and any browser. Meanwhile, the database will be saved in the Albanian Courier's server.

3.3 Domain Requirements

Domain requirements apply to the calculation of the price of the package to be delivered from the couriers. We worked closely with the finance department of Albanian Courier company to better understand and correctly implement the formula for the taxation process and price generation. The delivery price for the package will be implemented as a function of its weight, fuel tax, and distance to be delivered from the place of collection. For 0 to 2 kg packages, the price is 150 ALL, and for every additional kg above the standard weight, the price will increase by 30 ALL.

4. User Scenarios/Use Cases

4.1 User Scenarios

1. **Scenario:** Admin deactivates user
    a. Admin logs in to the admin dashboard.
    b. Admin searches for the registered users.
    c. Admin selects the user/s he wants to deactivate from the grid.
    d. Admin deactivates the selected user.
2. **Scenario:** Agent successfully creates a bag
    a. Agent logs in to the agency's dashboard
    b. Agent generates the bag's code.
    c. Agent fills the bag with valid created orders.
    d. Agents presses the save button to save the process.
    e. Agent presses the create bag button to create the bag.
    f. The status of all the orders included in the bag is changed.
    g. Bag is created successfully.
3. **Scenario:** Agent successfully adds to the bag
    a. Agent logs in to the agency's dashboard
    b. Agent generates the bag's code.
    c. Agent fills the bag with a created order.
    d. If the order exists, the status of the order is checked.
    e. If order has a valid status, order is added to the bag successfully.
4. **Scenario:** Agent fails to add to the bag
    a. Agent logs in to the agency's dashboard
    b. Agent generates the bag's code.

  c. Agent fills the bag with an order.
  d. The order doesn’t exist or its status is not valid.
  e. Agent can’t add to the bag.

5. **Scenario:** Agent tracks the orders
   a. Agent logs in to the agency’s dashboard
   b. Agent selects the menu for reports from the dashboard
   c. Agent has all the generated orders displayed in the screen.
   d. The orders are differentiated between creation, collection, added to the bag and delivered.

6. **Scenario:** Agent recreates an order
   a. Agent logs in to the agency’s dashboard
   b. Agent selects the menu for the collection from the dashboard
   c. Agent checks the order’s status.
   d. If status is not delivered, agent picks to recreate order.
   e. Order is available again for collection.

7. **Scenario:** Agency collects orders
   a. Agent logs in to the agency’s dashboard.
   b. Agent enters the orders to be delivered from the collection menu
   c. Agent enters the weight and amount of order.
   d. Agent selects destination of order which determines the delivery code.
   e. Agent selects the Courier to deliver the bag.
   f. Agent selects the client subject.
   g. Agent selects method of payment.
   h. Agent gills receiving side information.

   i. Agent enters order delivery price.

   j. Agent begins the collection process of the orders.

   k. Agent collects the bags with valid price and orders.

8. **Scenario:** Courier opens the bag created by the Agents

   a. Courier logs in to the courier dashboard.

   b. Courier enters the order he has to deliver.

   c. Courier opens the bag

   d. Courier checks details of delivery provided by the agent.

9. **Scenario:** Courier doesn’t deliver the collected order to the client

   a. Courier travels to deliver the order.

   b. Client doesn’t show up.

   c. Courier fails to deliver the collected order.

   d. Courier states reason of not delivering.

10. **Scenario:** Courier successfully delivers the collected order to the client

   a. Courier travels to deliver the order.

   b. Client shows up to take the order.

   c. Courier fails to deliver the collected order.

   d. Courier enters the order status as delivered.

4.1.1 Use Cases Diagram

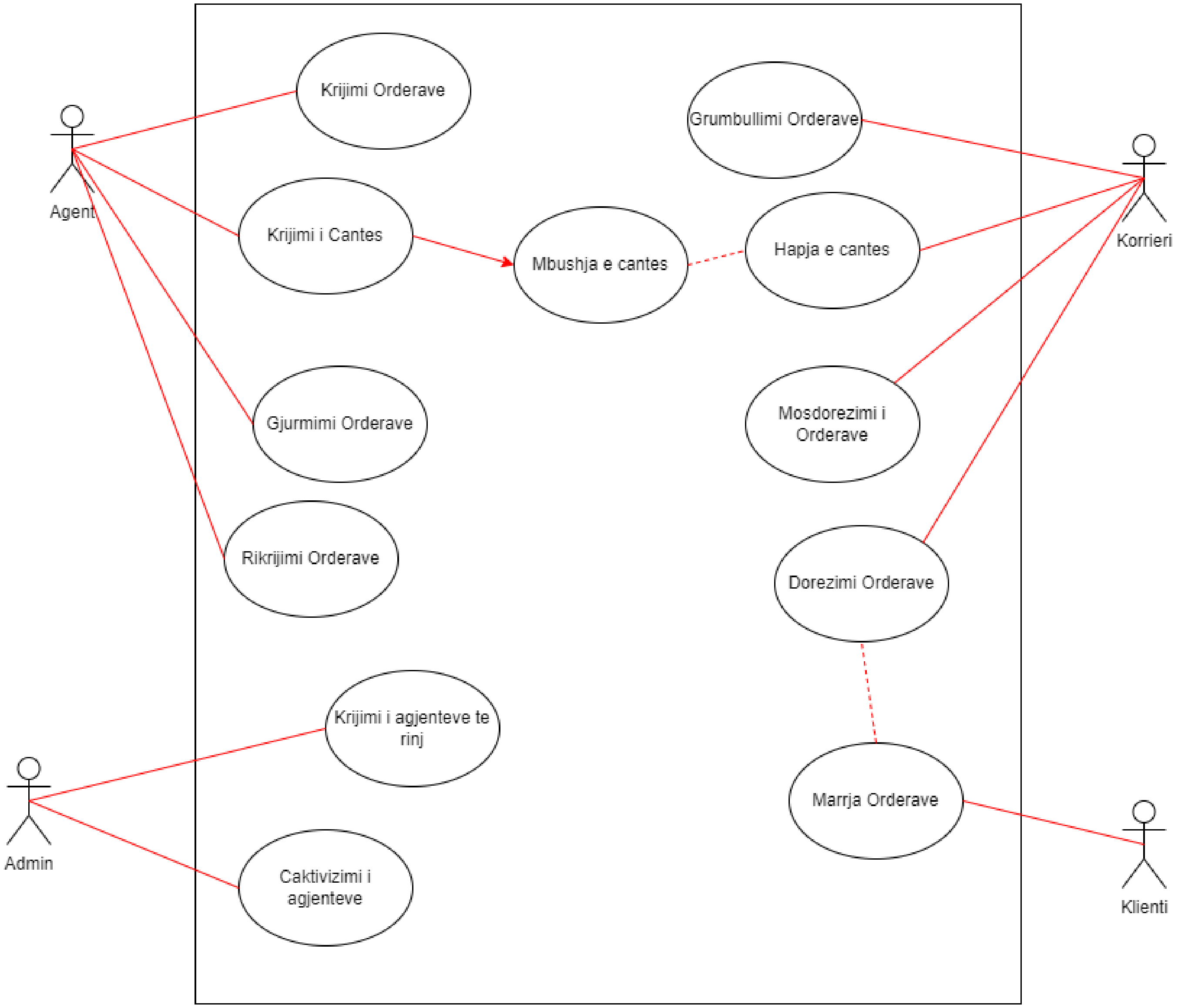


Fig. 1 – General Use Case Diagram

4.1.2 Use Cases Tables (Extended)

UC_1	**Admin Deactivates User**
Summary	- The admin is the only role in the system with the functionality of deactivating users.
Dependency	-
Actors	- Administrator
Preconditions	- The users must be already registered in the system and displayed in the admin’s dashboard, which will be done by a HR employee directly through the database.
Description of the main sequence	- Admin will be able to deactivate users by selecting from the list of all registered users, the ones that he wants to deactivate.
Description of the Alternate Sequence	- In cases when there are no registered users in the system, the admin cannot deactivate anything because the displayed view will be empty.
Non-functional Requirements	- Provided the hierarchy of the institution, end-users can’t interfere with each other’s account. This is done for security purposes.
Postconditions	- Admin will view a newly generated list of the registered users without the one/s he deactivated.

UC_2	**Agent Successfully Creates a Bag**
Summary	- When an agent receives an order, they will be able to create a bag through their dashboard menus.
Dependency	-
Actors	- Agent
Preconditions	- The agent must have logged in successfully into the agent's page and received valid orders.
Description of the main sequence	- Agent will generate a code for the new bag and fill the bag with valid orders, save the process and create the bag.
Description of the Alternate Sequence	- In the other case, when the orders are not valid, the agent won't be able to create the bag.
Non-functional Requirements	-
Postconditions	- The agent will be redirected to its dashboard.

UC_3	**Agent Successfully Adds to the Bag**
Summary	- Agent will be able to add orders to the bag.
Dependency	-
Actors	- Agent
Preconditions	- The agent must have logged in successfully into the agent's page and received valid orders.
Description of the main sequence	- Provided that the order exists and the status it holds is valid, the agent will add the order to the bag.
Description of the Alternate Sequence	- In the other case, the task will fail. Agent will need to check the process again.
Non-functional Requirements	-
Postconditions	- The agent will be redirected to his dashboard.

UC_4	**Agent Fails to Add to The Bag**
Summary	- Agent will not be able to add orders to the bag.
Dependency	-
Actors	- Agent
Preconditions	- The agent must have logged in successfully into the agent's page.
Description of the main sequence	- Provided that the order does not exist and/or the status it holds is invalid, the agent will not be able to add an order to the bag.
Description of the Alternate Sequence	- In the other case, the order is added to the bag
Non-functional Requirements	-
Postconditions	- An error message will be displayed and the agent will be directed to his dashboard.

UC_5	Agent Tracks the Orders
Summary	- Each user with the role of agent will be able to track all the orders he has created.
Dependency	-
Actors	- Agent
Preconditions	- The agent must have logged in successfully into the agent's page and he must have successfully created at least one order.
Description of the main sequence	- All the generated orders will be displayed to the agent differentiated by different filters.
Description of the Alternate Sequence	- In the other case, where the agent has not generated any orders yet, they won't be able to track anything.
Non-functional Requirements	-
Postconditions	- The agent will be redirected to the report menu in the agent's dashboard.

UC_6	**Agent Recreates an Order**
Summary	- Users in the role of agent will be able to replace an order in case its delivery was refused for the first time.
Dependency	-
Actors	- Agent
Preconditions	- The agent must have logged in successfully into the agent’s page and at least one of his orders has not been delivered from the first collection.
Description of the main sequence	- Once the agent verifies the status of the order, it may pick it to recreate it so that it can be collected again.
Description of the Alternate Sequence	- In the other case, where there are no returned packages, recreation of an order won’t be possible.
Non-functional Requirements	-
Postconditions	- The agent will be redirected to his dashboard.

UC_7	Agency Collects Orders
Summary	- Each user in the role of the agent will be able to collect orders.
Dependency	-
Actors	- Agent
Preconditions	- The agent must have logged in successfully into the agent's page.
Description of the main sequence	- Each agent will enter the orders to be delivered from the collection menu by providing details about its weight, destination, the courier to deliver it and client side information.
Description of the Alternate Sequence	- In the other case, where there are no created orders from the bag, collection of orders won't be possible.
Non-functional Requirements	-
Postconditions	- The agent will be redirected to his dashboard.

UC_8	Courier Opens the Bag Created by the Agents
Summary	- Courier will be able to check the orders to be delivered from his own dashboard.
Dependency	-
Actors	- Courier, Agent
Preconditions	- The users must have been logged in successfully to their respective dashboards and a bag must have been created previously from the agent.
Description of the main sequence	- Each courier will be able to open the bag and checking the details of delivery provided by the agent.
Description of the Alternate Sequence	- In the other case, where a bag hasn't been created previously by the agent, there is nothing to be delivered.
Non-functional Requirements	-
Postconditions	- Courier will be redirected to his dashboard.

UC_9	**Courier doesn’t deliver an order**
Summary	- Due to no-shows from the client side, couriers may not be able to deliver all the collected orders.
Dependency	-
Actors	- Courier, Client
Preconditions	- The users must have been logged in successfully to their respective dashboards and a bag must have been created previously from the agent and opened by the courier.
Description of the main sequence	- Couriers will be allowed the option of No show if a delivery of the package hasn’t taken place with the need to state the reason for it.
Description of the Alternate Sequence	- In the other case, the status of the order would show as delivered.
Non-functional Requirements	-
Postconditions	- The courier will be redirected to his dashboard.

UC_10	Courier Successfully Delivers order
Summary	- Each user will be able to login into the system by entering their respective credentials.
Dependency	-
Actors	- Courier, Client
Preconditions	- The users must have been logged in successfully to their respective dashboards and a bag must have been created previously from the agent and opened by the courier.
Description of the main sequence	- Client shows up and receives his order.
Description of the Alternate Sequence	- In the other case, where the client doesn't respond to the courier's enquiry for the delivery of package, the status od the order would be set as not delivered.
Non-functional Requirements	-.
Postconditions	- Courier will be redirected to his dashboard.

4.2 Diagrams

4.2.1 Activity Diagrams

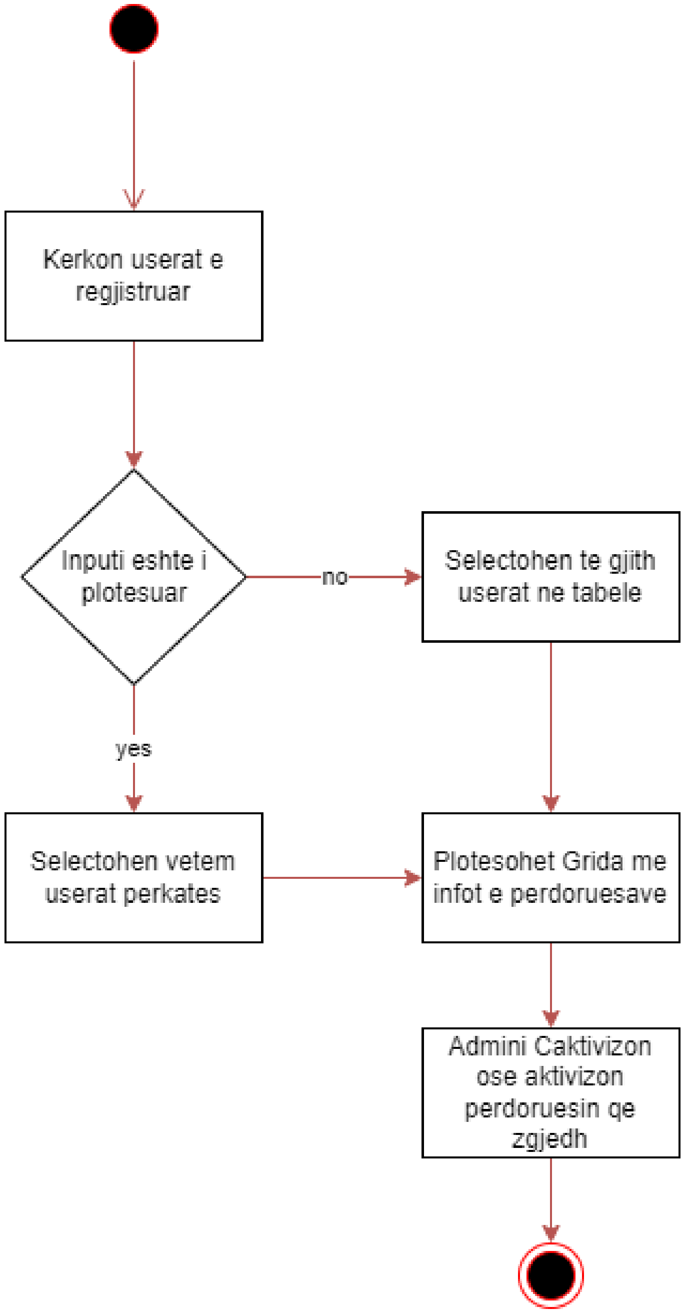

Kerkon userat e regjistruar
Inputi eshte i plotesuar
no
Selectohen te gjith userat ne tabele
yes
Selectohen vetem userat perkates
Plotesohet Grida me infot e perdoruesave
Admini Caktivizon ose aktivizon perdoruesin qe zgjedh

Fig. 2 – Admin Activity Process

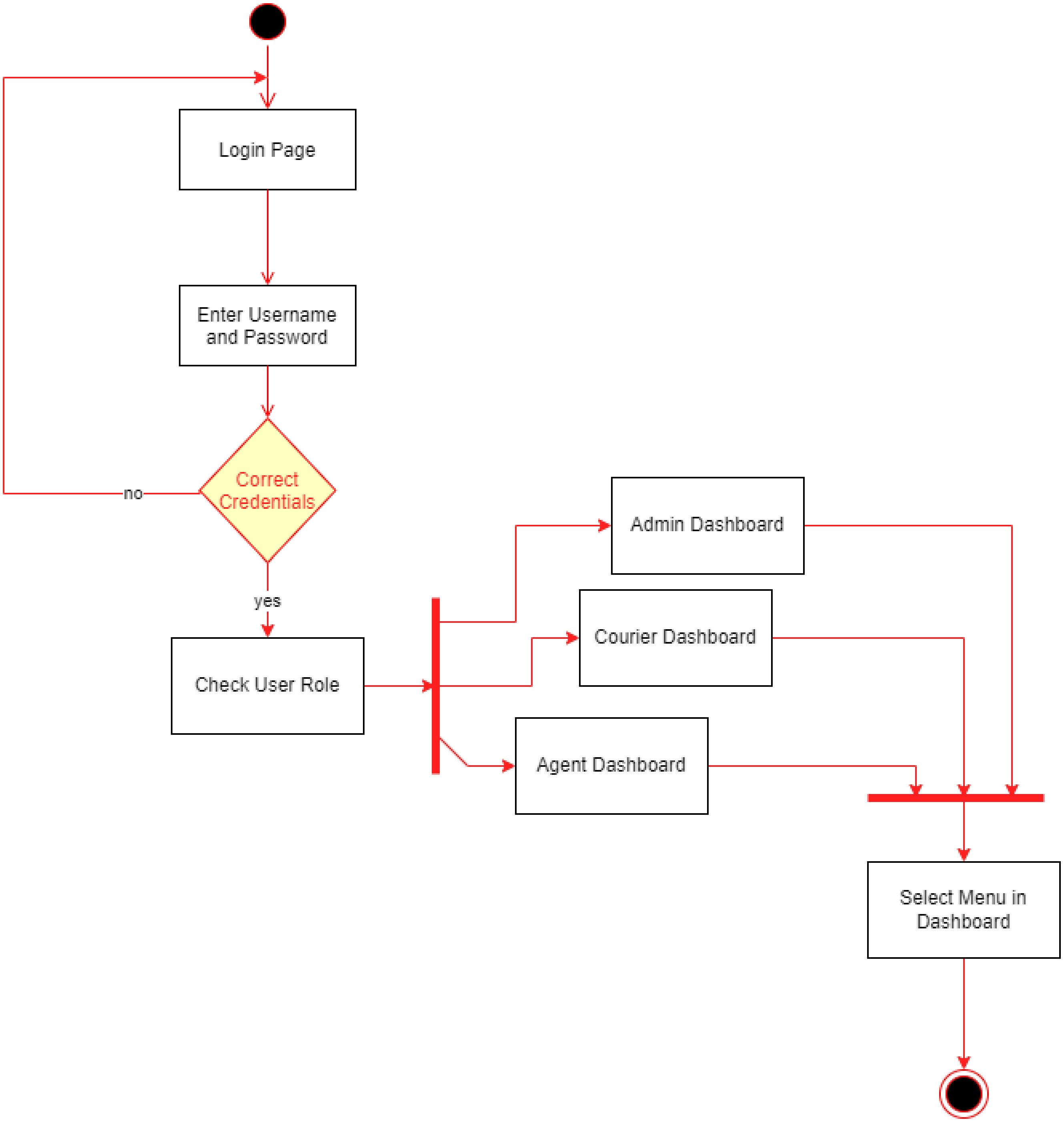


Fig. 3 – Log in User Page

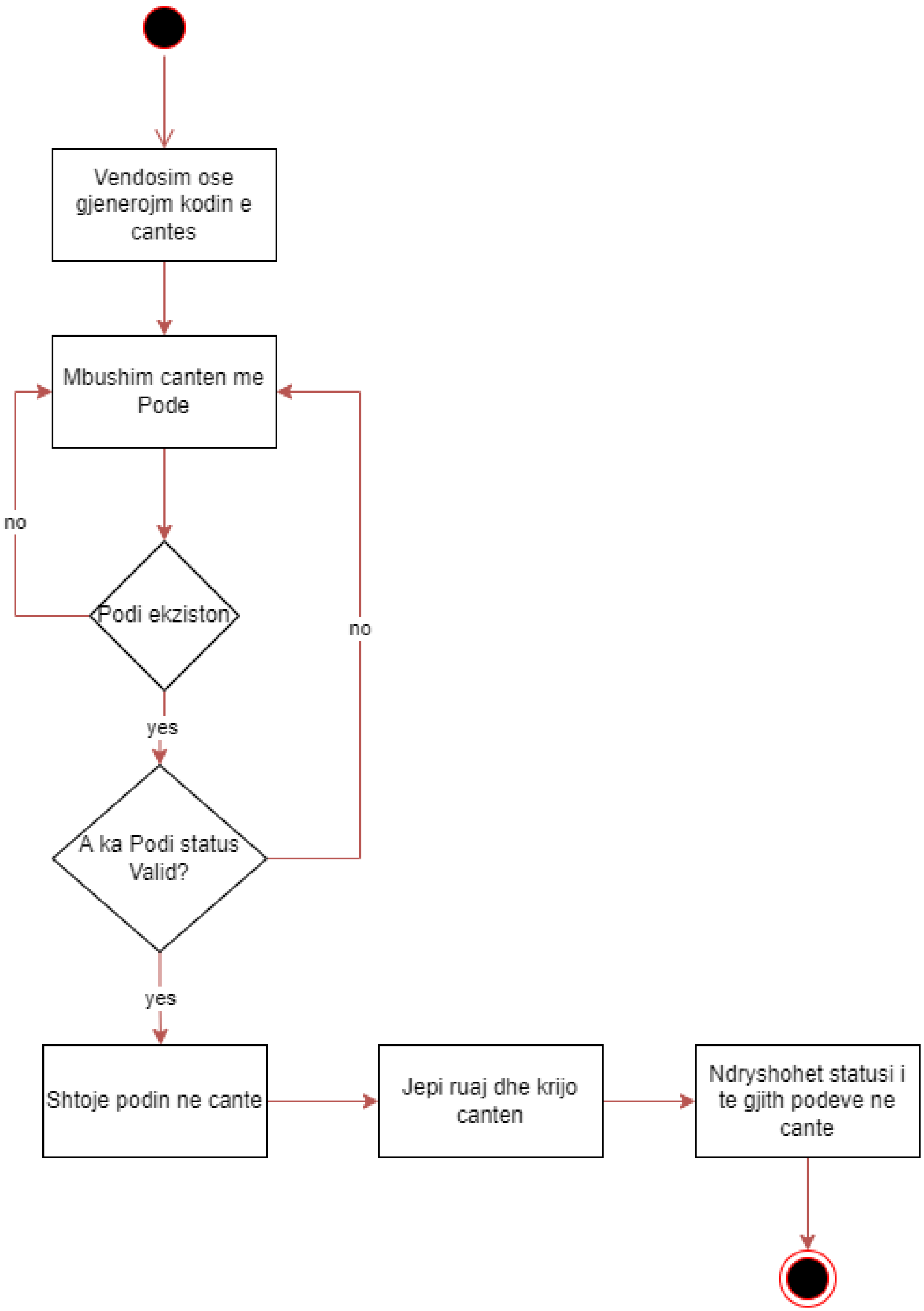

Vendosim ose gjenerojm kodin e cantes
Mbushim canten me Pode
no
Podi ekziston
no
yes
A ka Podi status Valid?
yes
Shtoje podin ne cante
Jepi ruaj dhe krijo canten
Ndryshohet statusi i te gjith podeve ne cante

Fig. 4 – Bag Creation Process

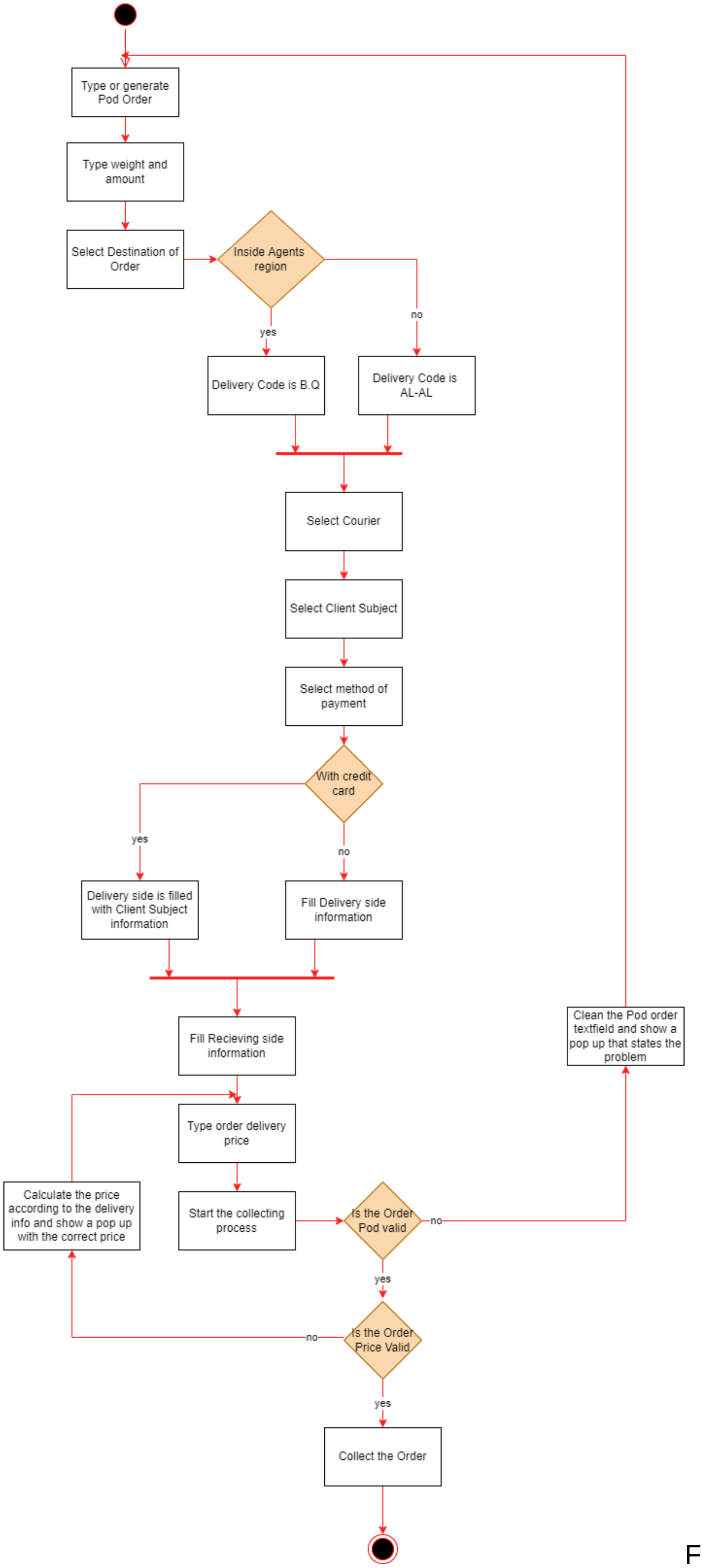

Type or generate Pod Order
Type weight and amount
Select Destination of Order
Inside Agents region
yes
no
Delivery Code is B.Q
Delivery Code is AL-AL
Select Courier
Select Client Subject
Select method of payment
With credit card
yes
no
Delivery side is filled with Client Subject information
Fill Delivery side information
Fill Recieving side information
Type order delivery price
Calculate the price according to the delivery info and show a pop up with the correct price
Start the collecting process
Is the Order Pod valid
no
Clean the Pod order textfield and show a pop up that states the problem
yes
Is the Order Price Valid
no
yes
Collect the Order

Fig. 5 – Order Collection Process

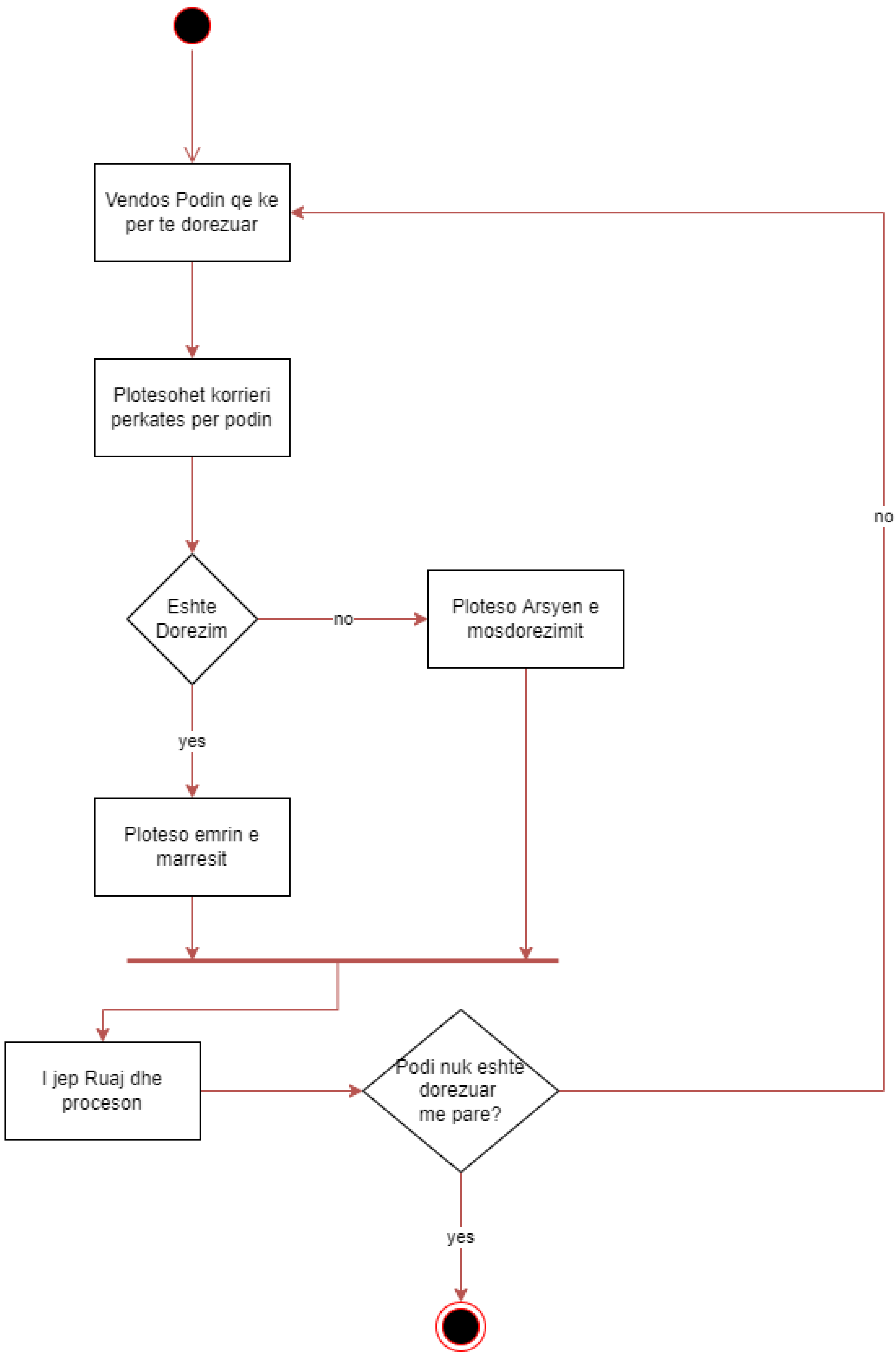

Vendos Podin qe ke per te dorezuar
Plotesohet korrieri perkates per podin
Eshte Dorezim
no
Ploteso Arsyen e mosdorezimit
yes
Ploteso emrin e marresit
I jep Ruaj dhe proceson
Podi nuk eshte dorezuar me pare?
no
yes

Fig. 6 – Courier’s package delivery process

4.2.2 State Diagrams

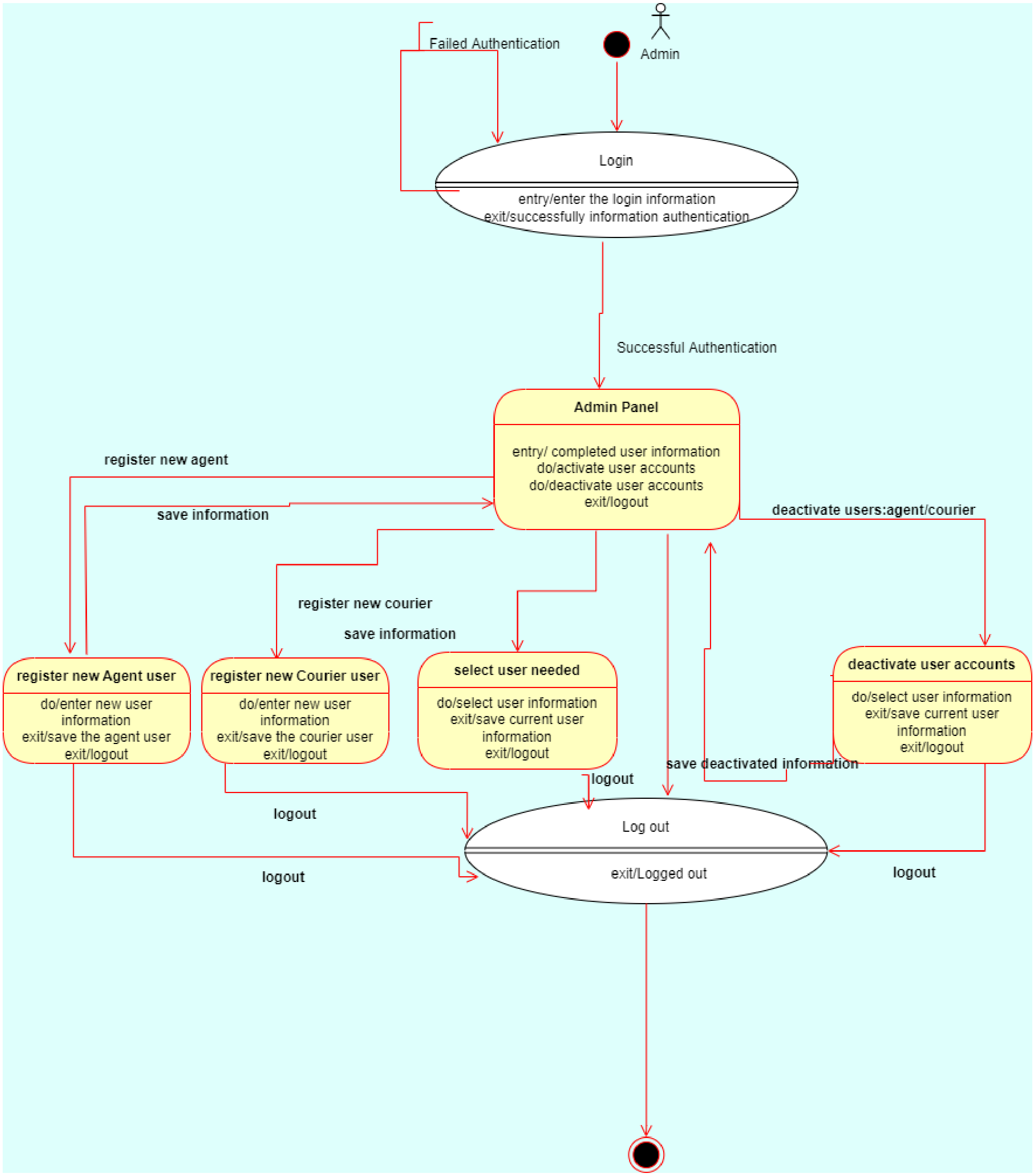

Failed Authentication
Admin
Login
entry/enter the login information
exit/successfully information authentication
Successful Authentication
Admin Panel
entry/ completed user information
do/activate user accounts
do/deactivate user accounts
exit/logout
register new agent
save information
deactivate users:agent/courier
register new courier
save information
register new Agent user
do/enter new user information
exit/save the agent user
exit/logout
register new Courier user
do/enter new user information
exit/save the courier user
exit/logout
select user needed
do/select user information
exit/save current user information
exit/logout
deactivate user accounts
do/select user information
exit/save current user information
exit/logout
save deactivated information
logout
logout
logout
logout
Log out
exit/Logged out

Fig. 7 – Admin State Diagram

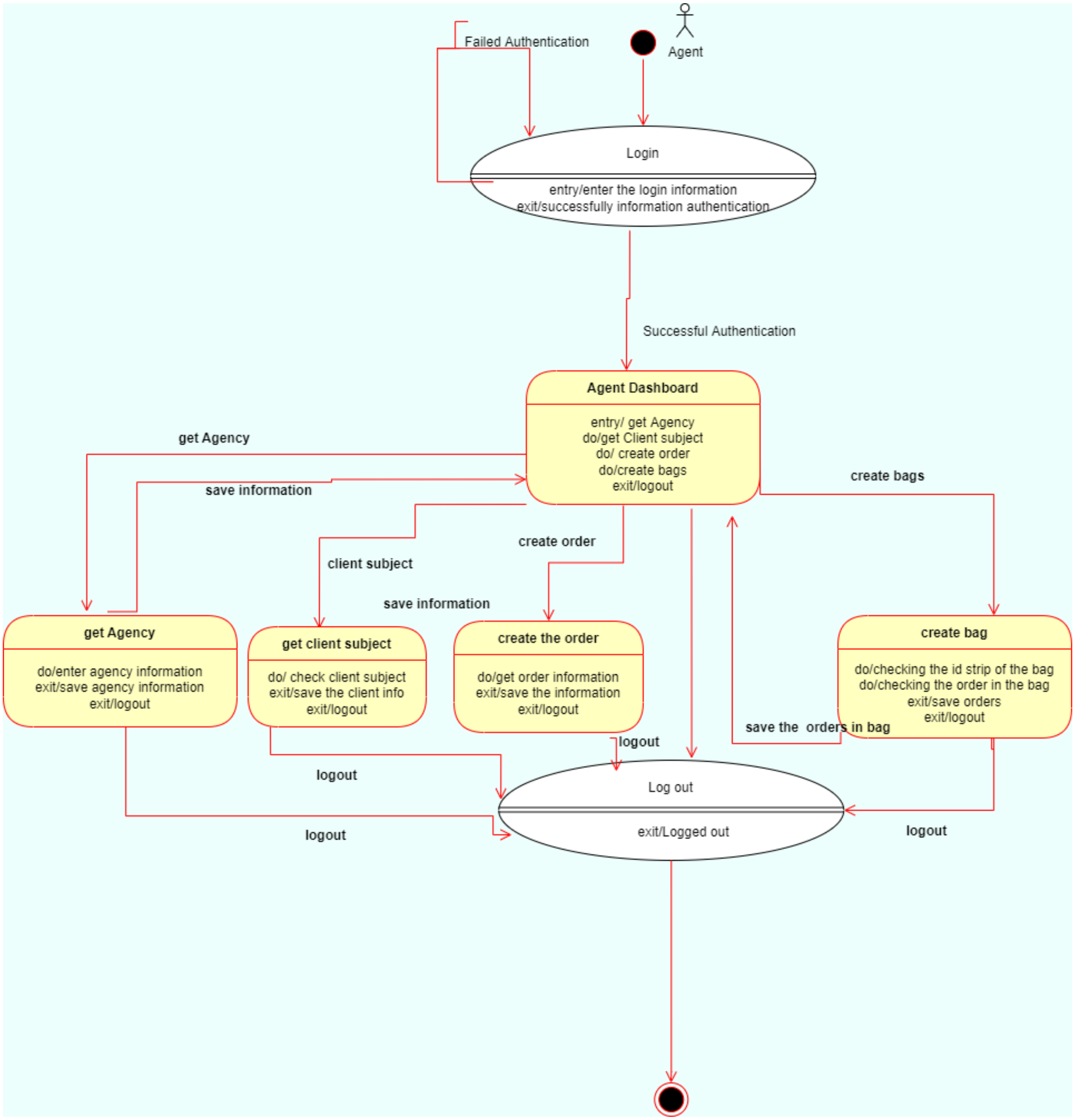


Fig. 8 – Agent State Diagram

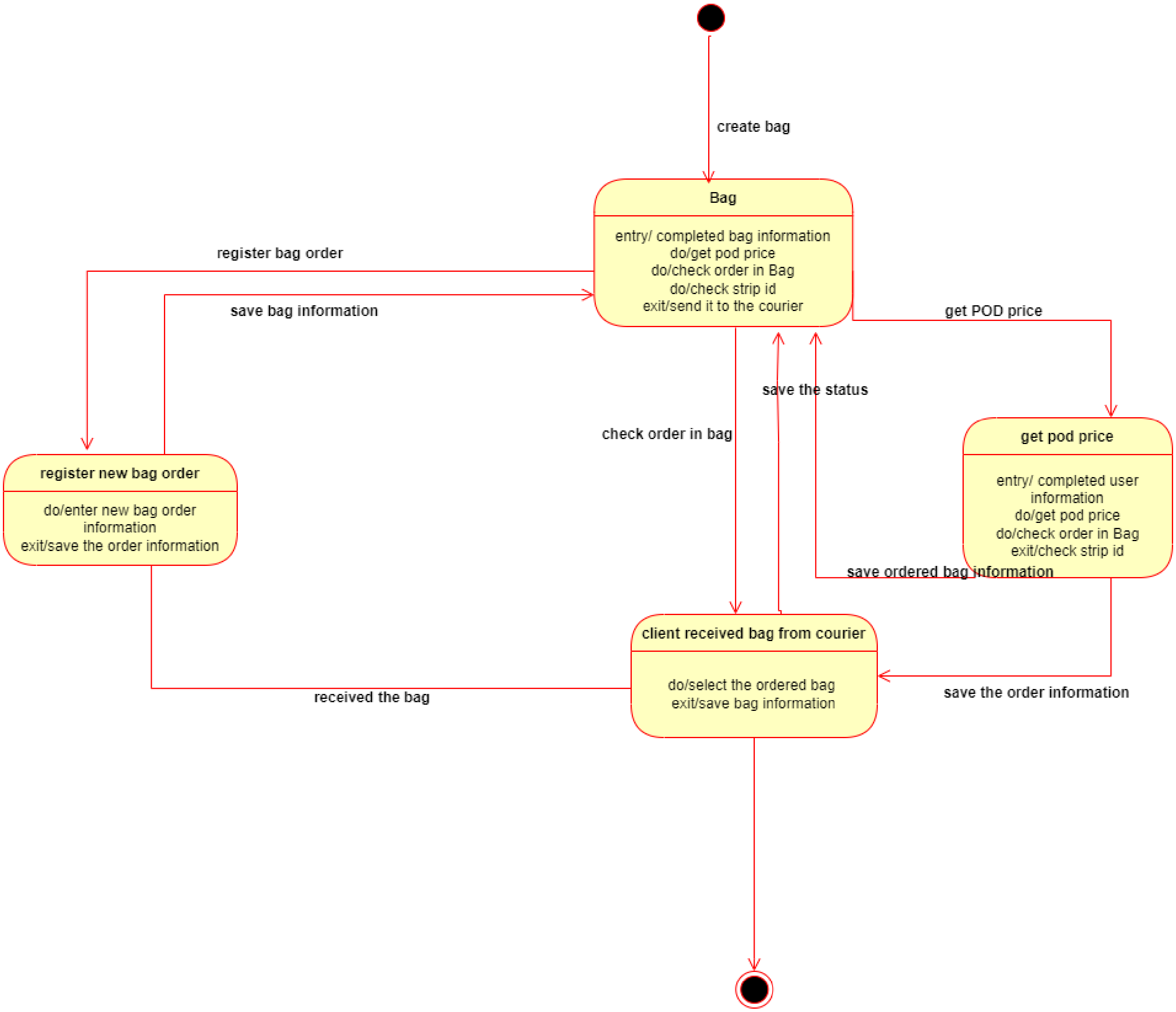

create bag
Bag
entry/ completed bag information
do/get pod price
do/check order in Bag
do/check strip id
exit/send it to the courier
register bag order
save bag information
get POD price
save the status
check order in bag
register new bag order
do/enter new bag order information
exit/save the order information
get pod price
entry/ completed user information
do/get pod price
do/check order in Bag
exit/check strip id
save ordered bag information
client received bag from courier
do/select the ordered bag
exit/save bag information
received the bag
save the order information

Fig. 9 – Bag Creation State Diagram

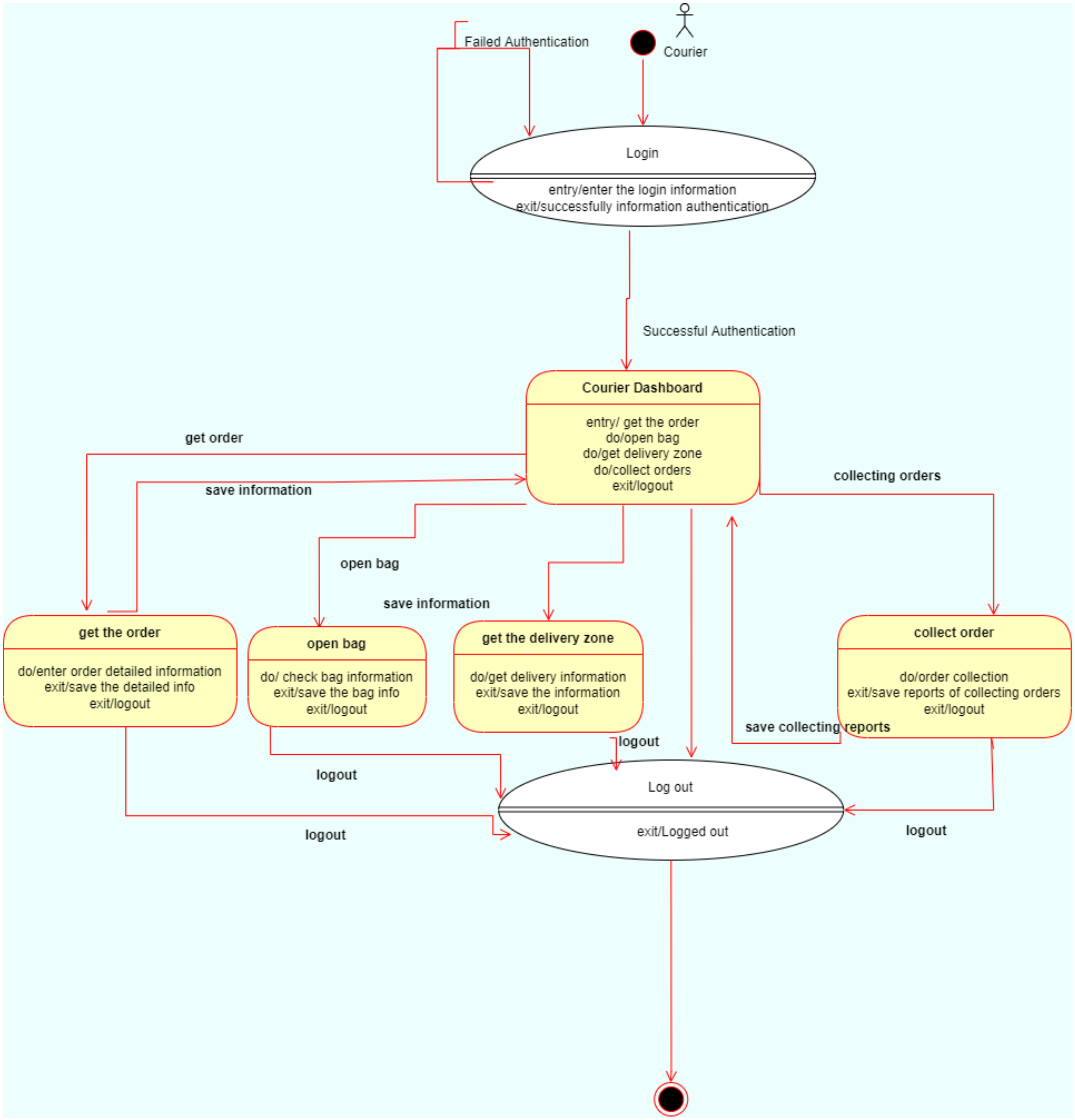


Fig. 10 – Courier State Diagram

4.2.3 Swimlane Diagram

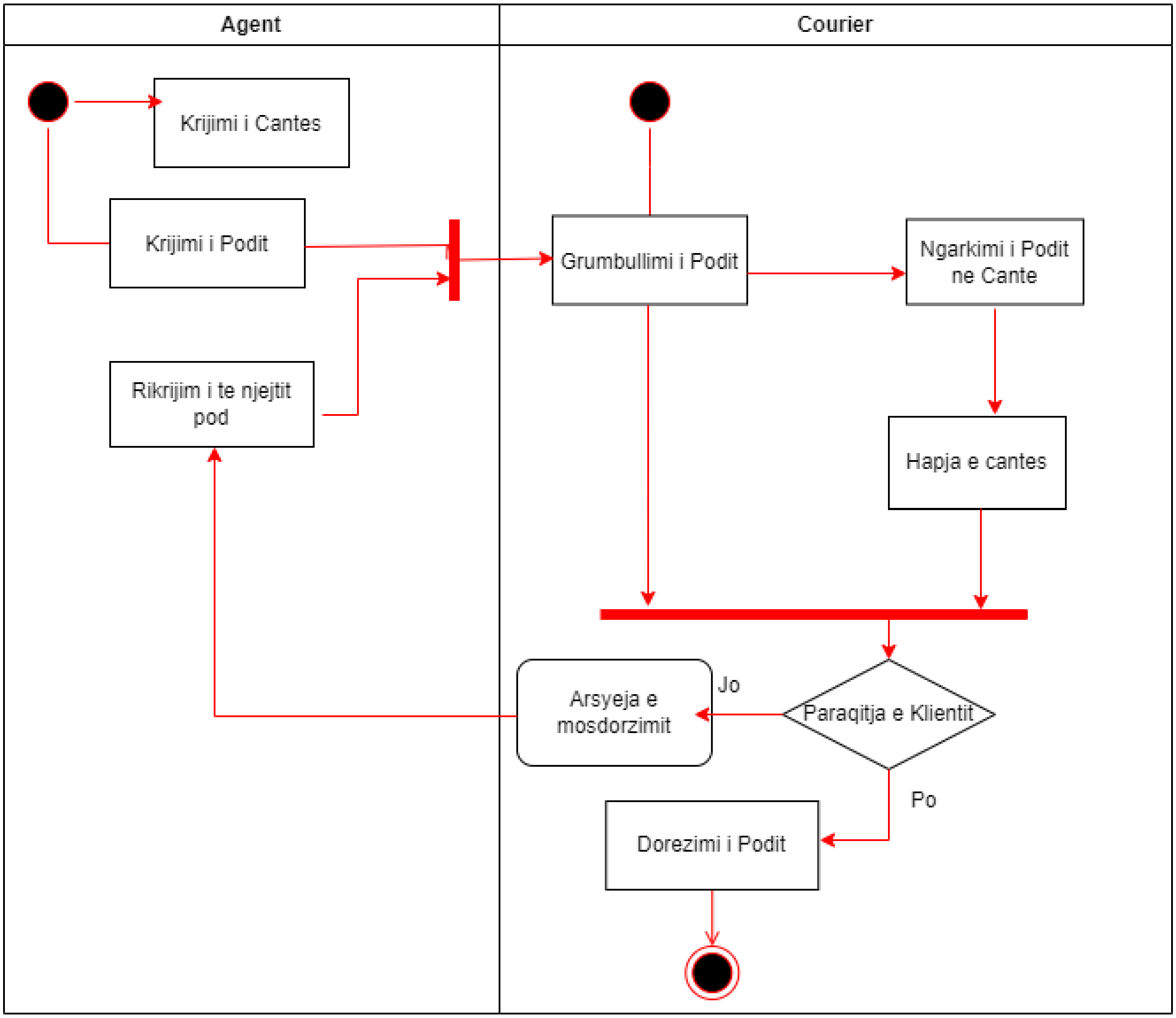


Fig. 11 – Relation between Agent and Courier activities

4.2.4 Sequence Diagrams

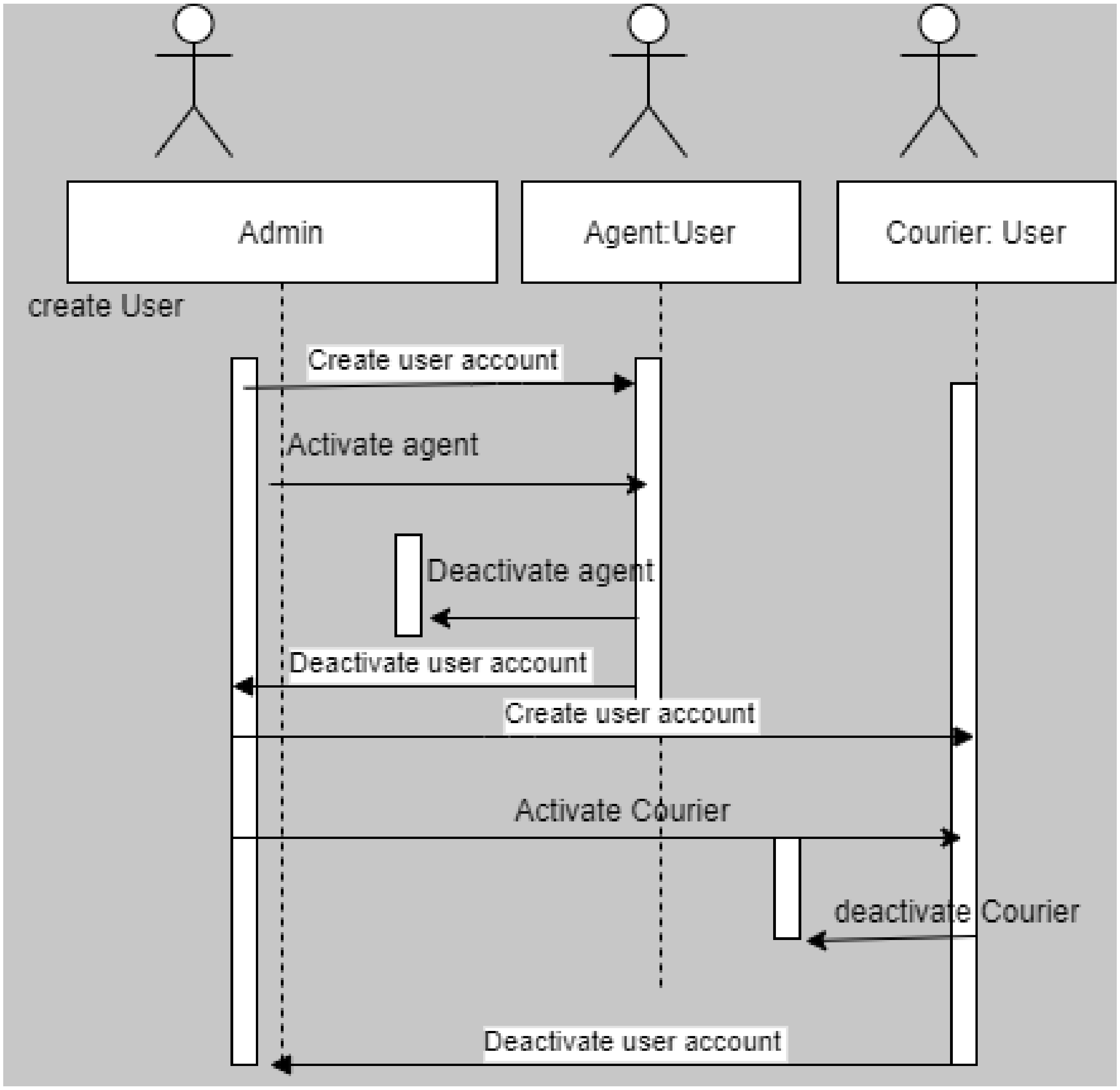


Fig. 12 – Admin’s Sequence Diagram

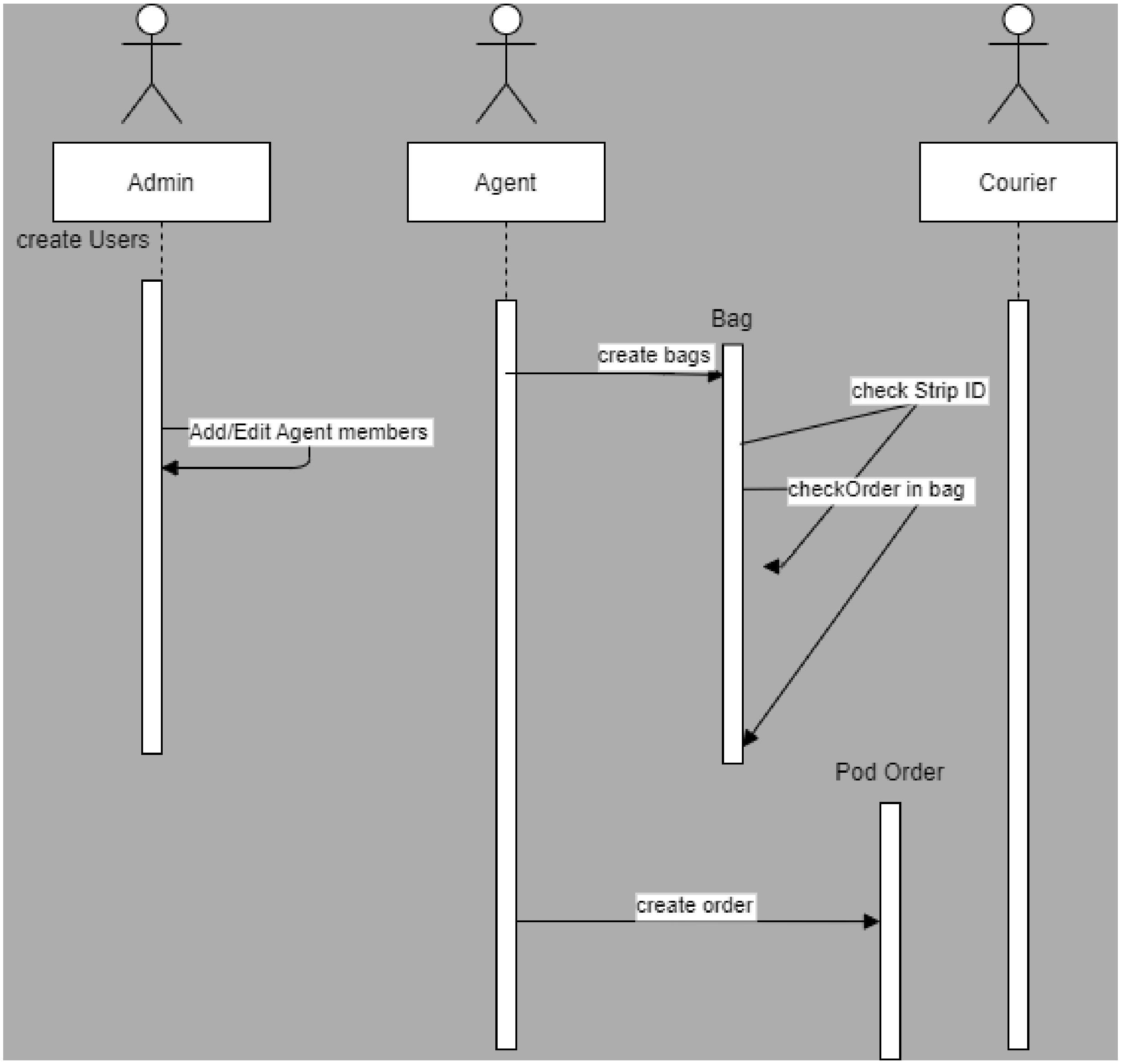


Fig. 13 – Agent’s Sequence Diagram

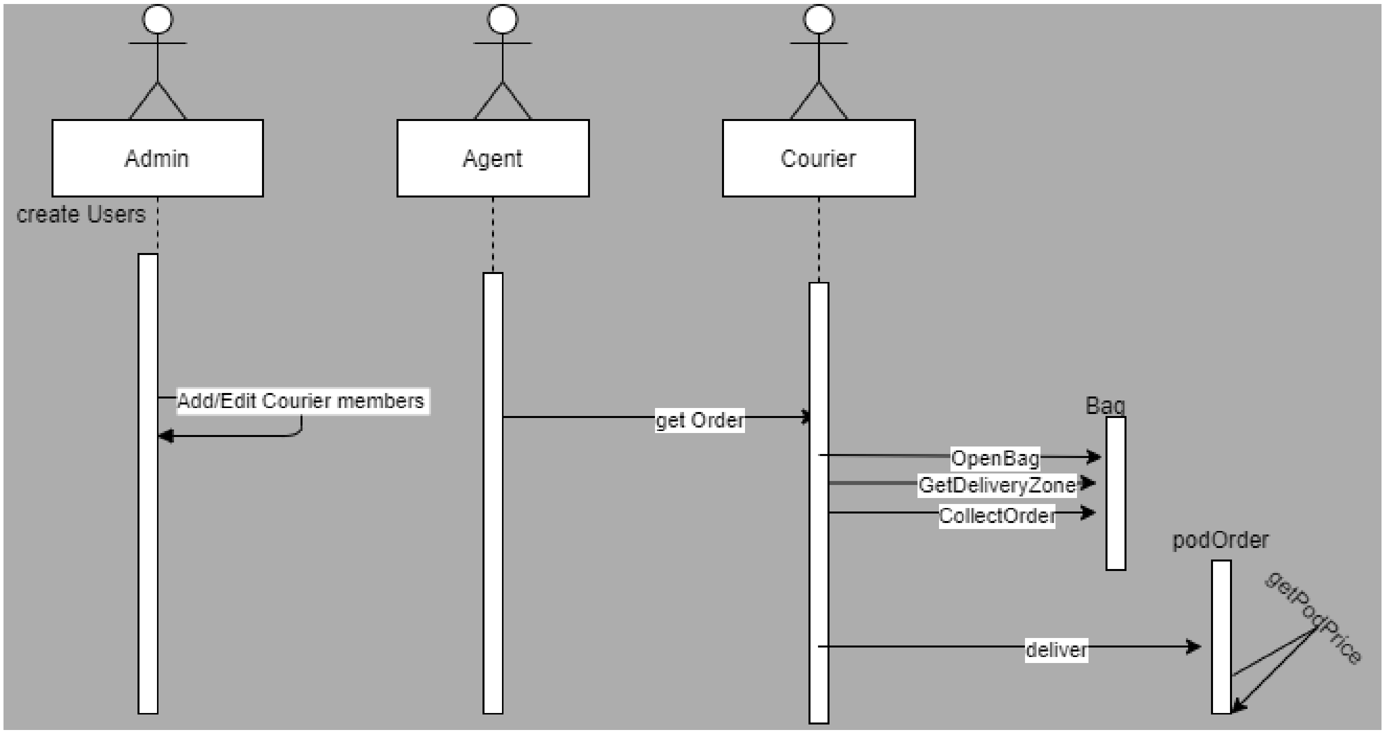


Fig. 14 – Courier’s Sequence Diagram

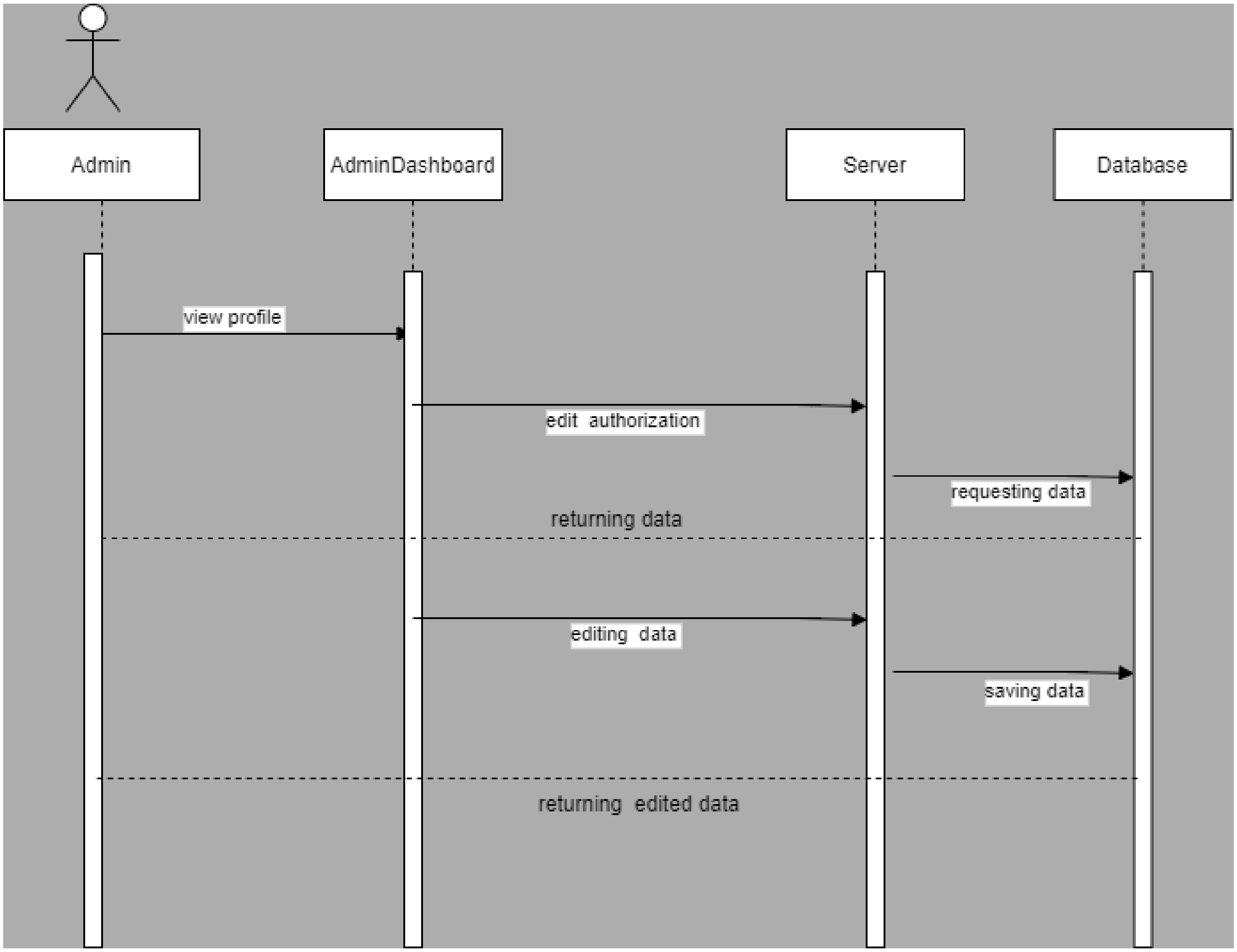


Fig. 15 – Admin Server Sequence Diagram

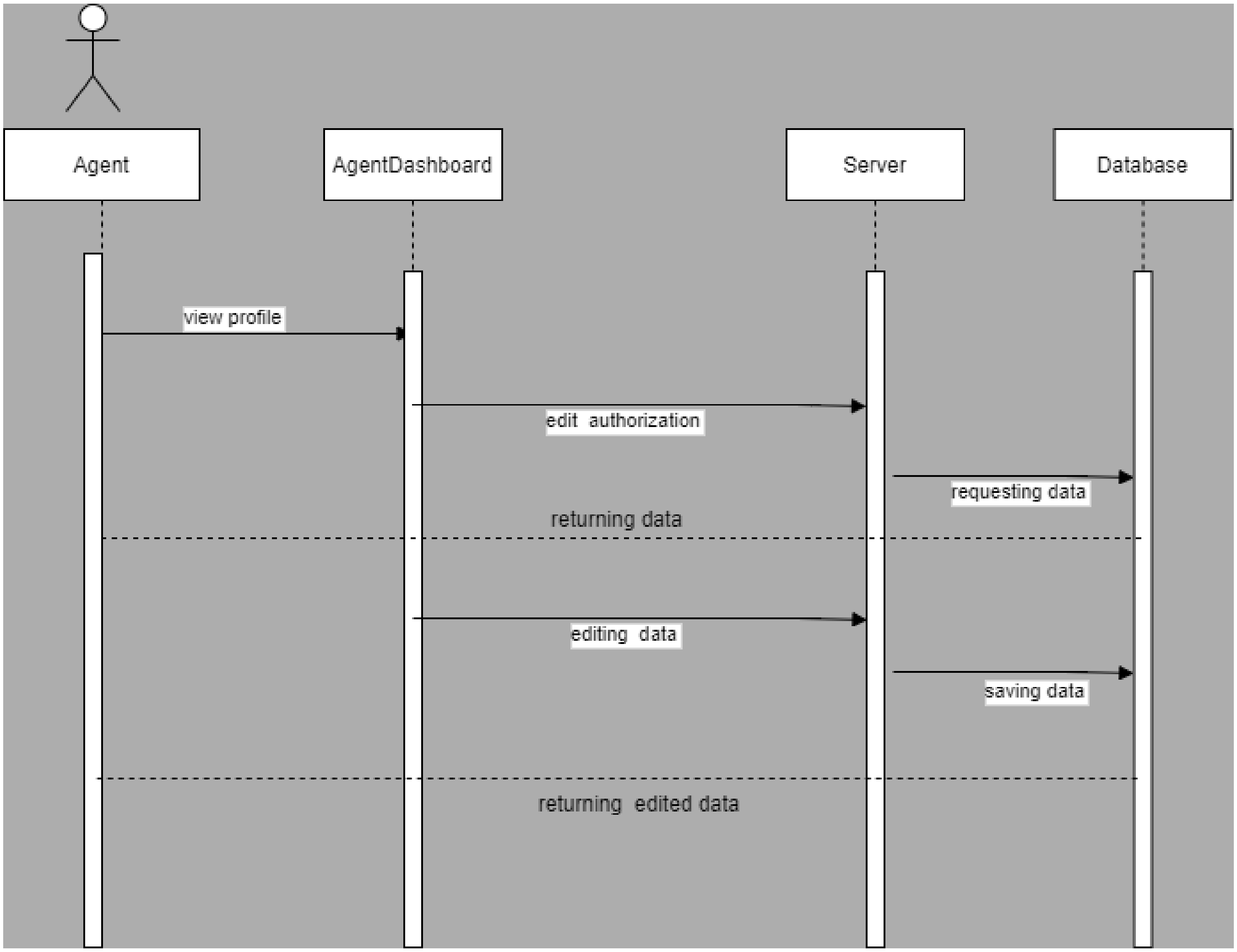


Fig. 16 – Agent’s Server Sequence Diagram

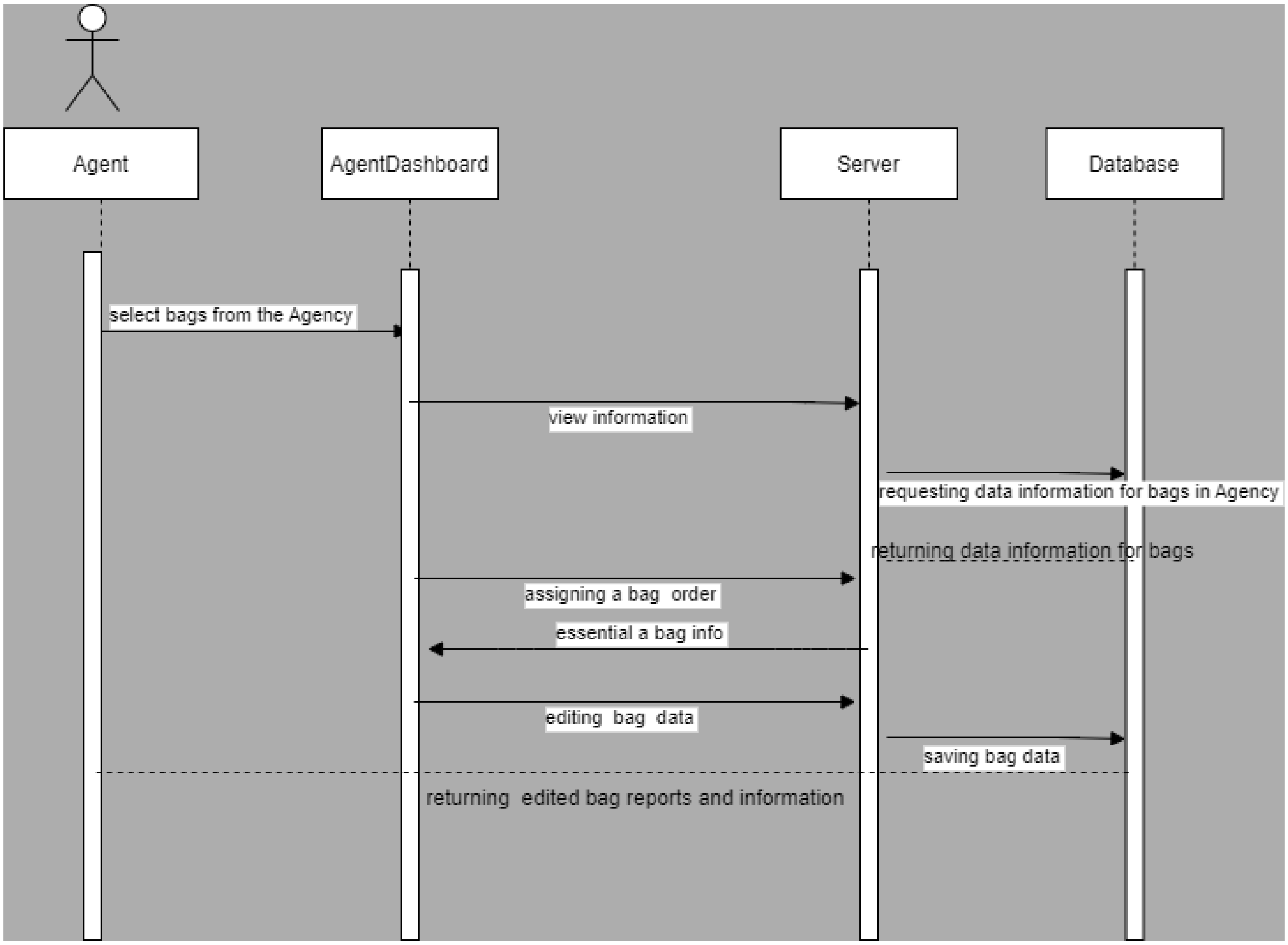

Agent
AgentDashboard
Server
Database
select bags from the Agency
view information
requesting data information for bags in Agency
returning data information for bags
assigning a bag order
essential a bag info
editing bag data
saving bag data
returning edited bag reports and information

Fig. 17 – Agent’s Bag Sequence Diagram

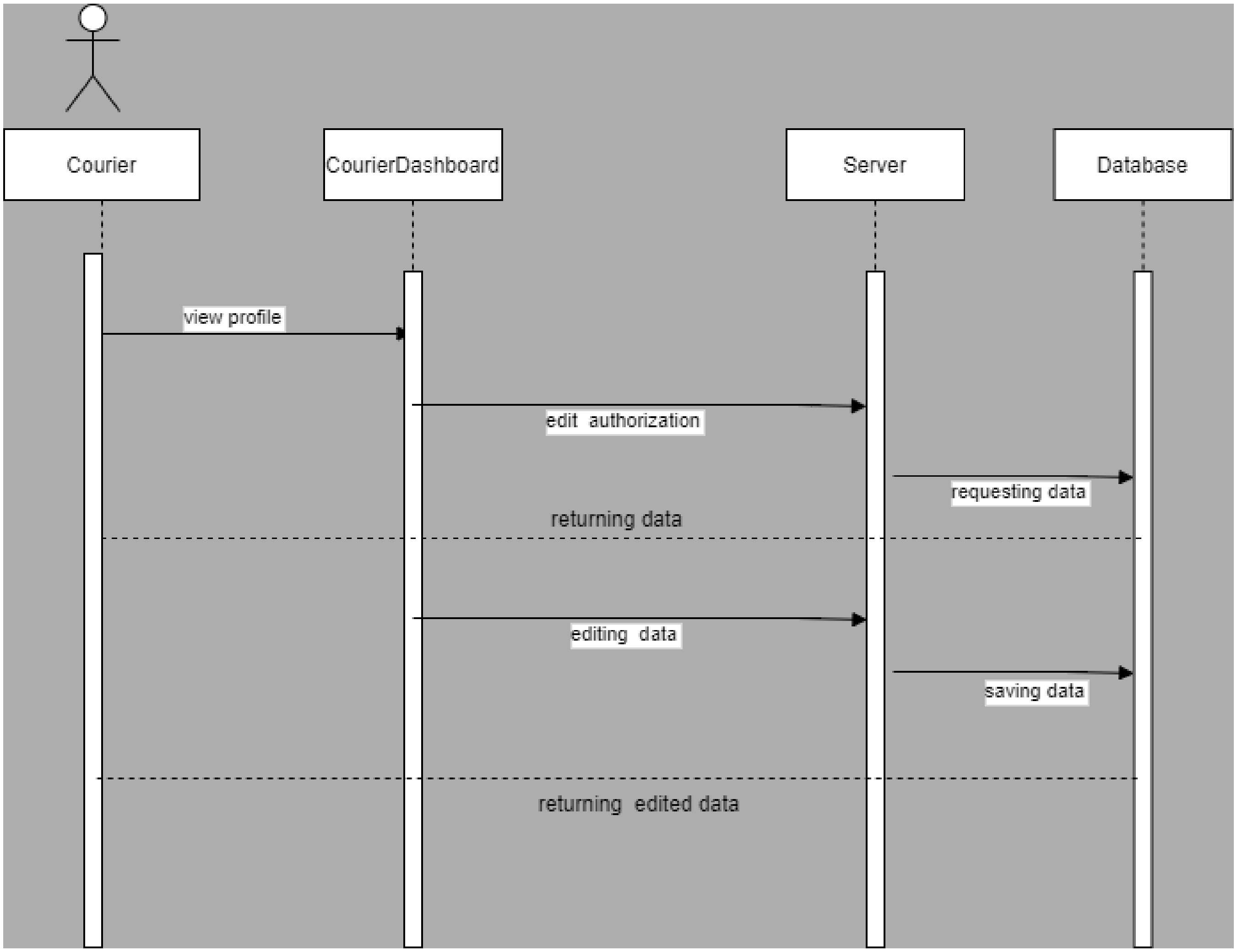


Fig. 18 – Courier’s Server Sequence Diagram

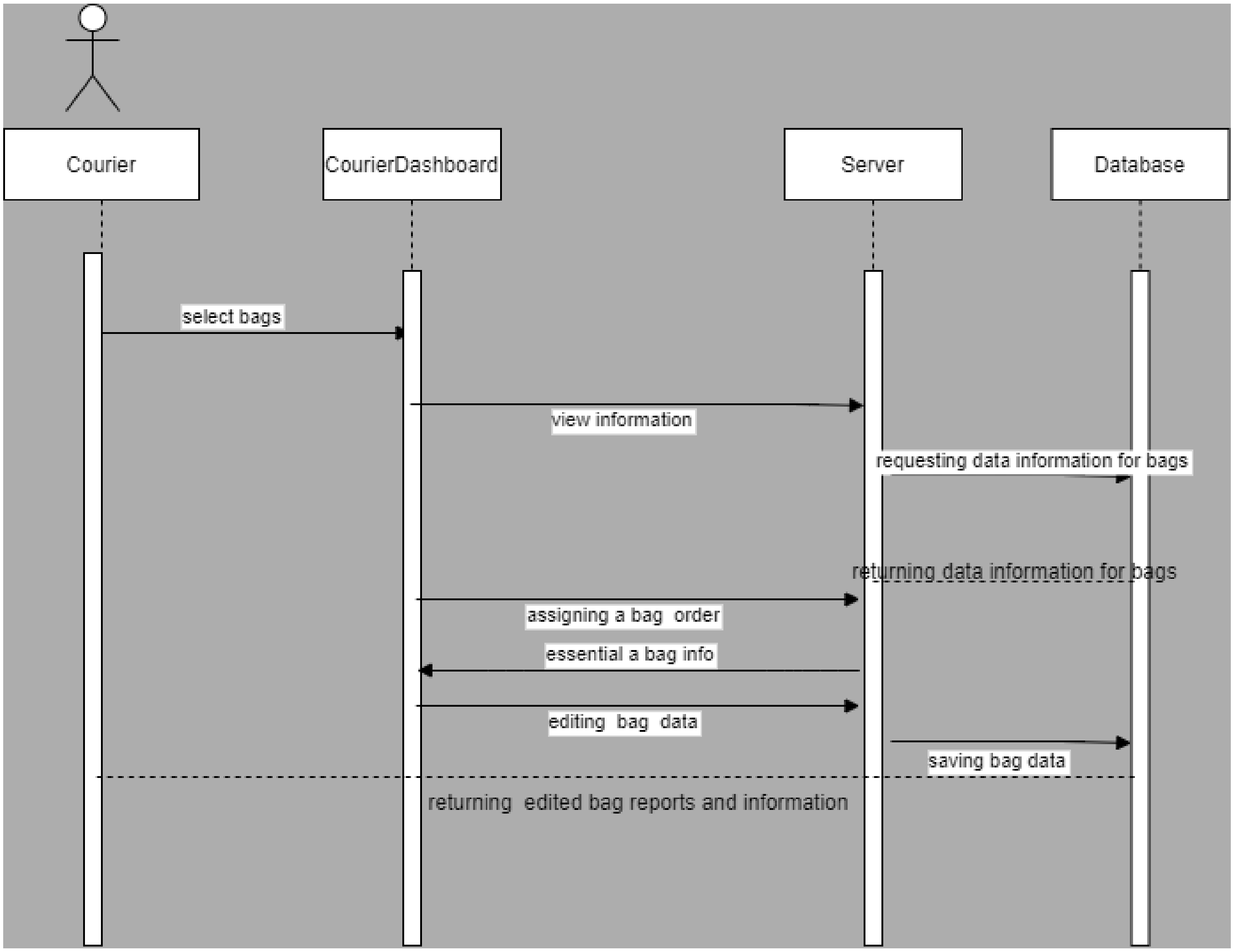


Fig. 19 – Courier Bag Sequence Diagram

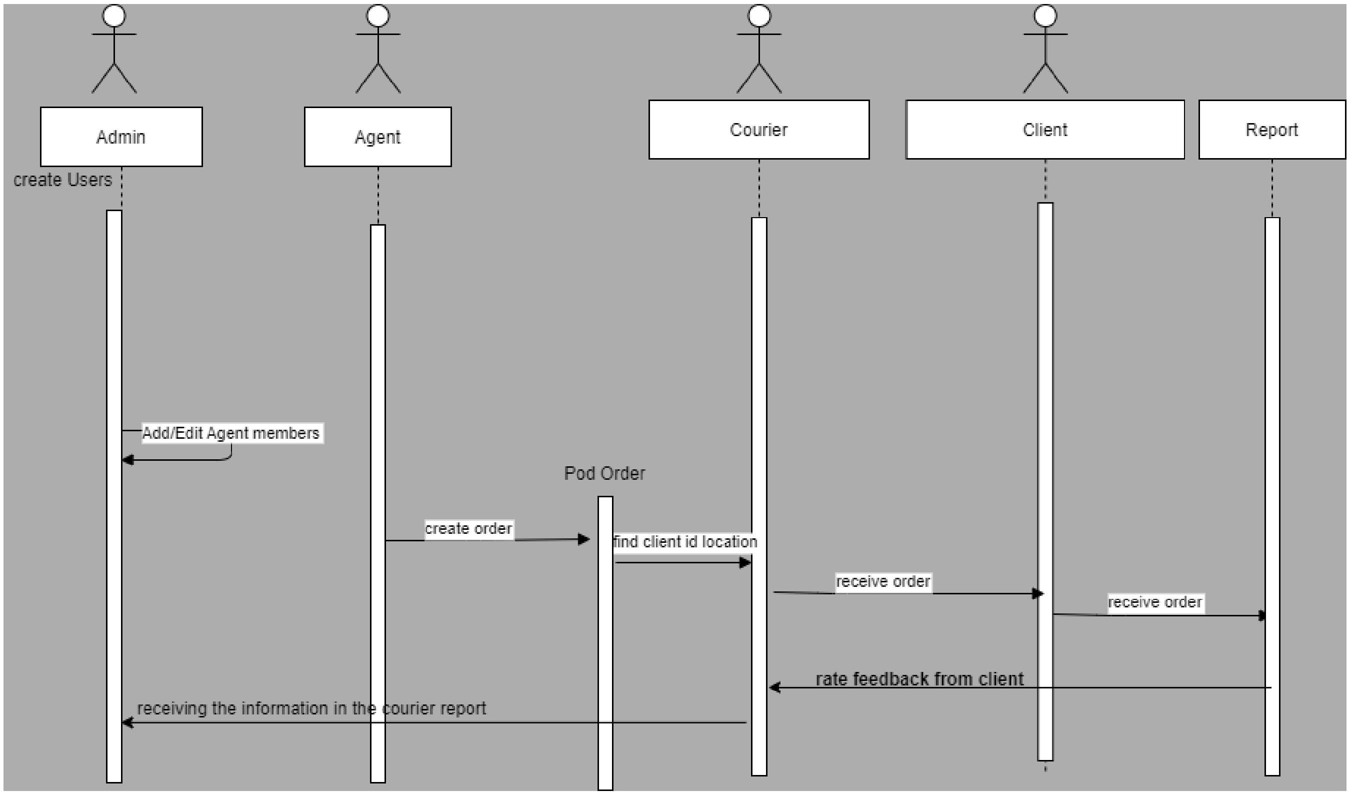


Fig. 20 – Report Sequence Diagram

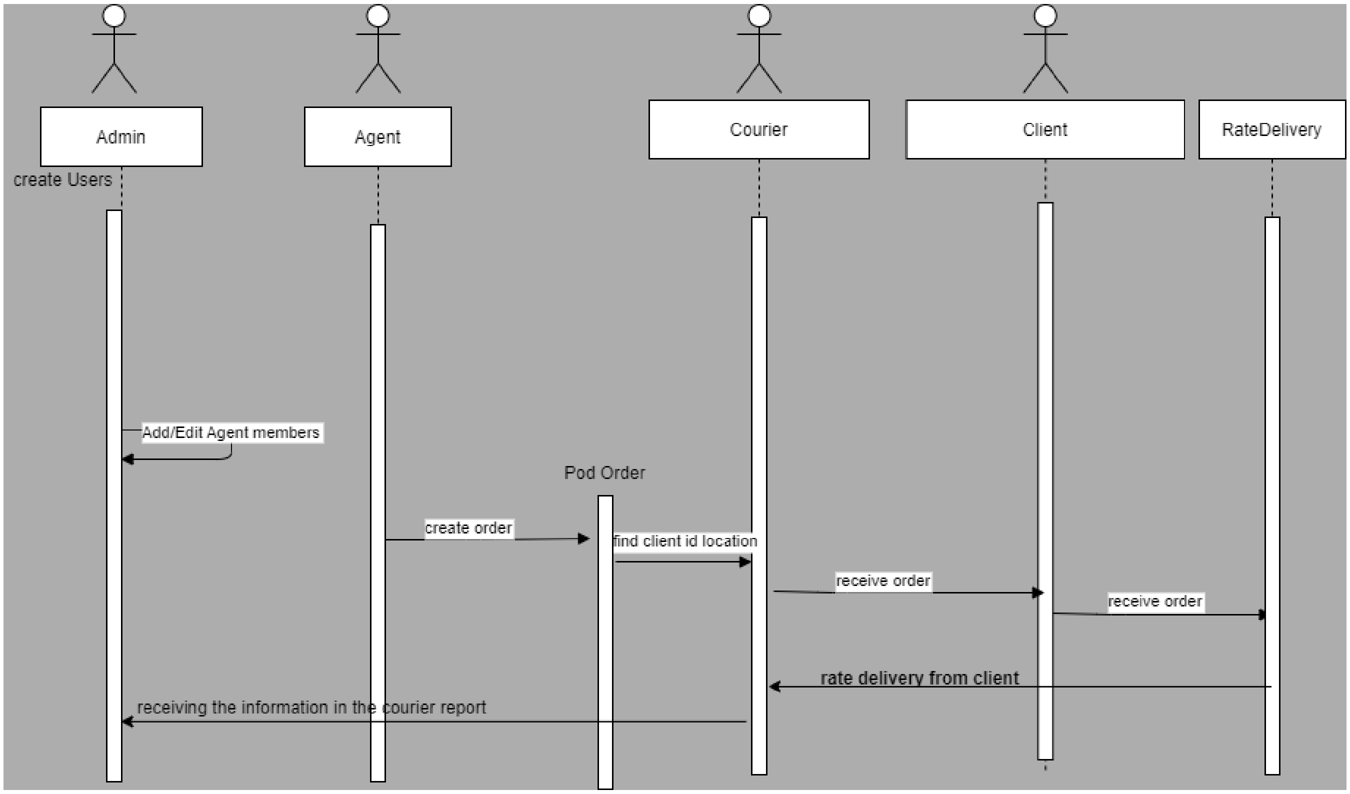


Fig. 21 – Client Rate Delivery Sequence Diagram

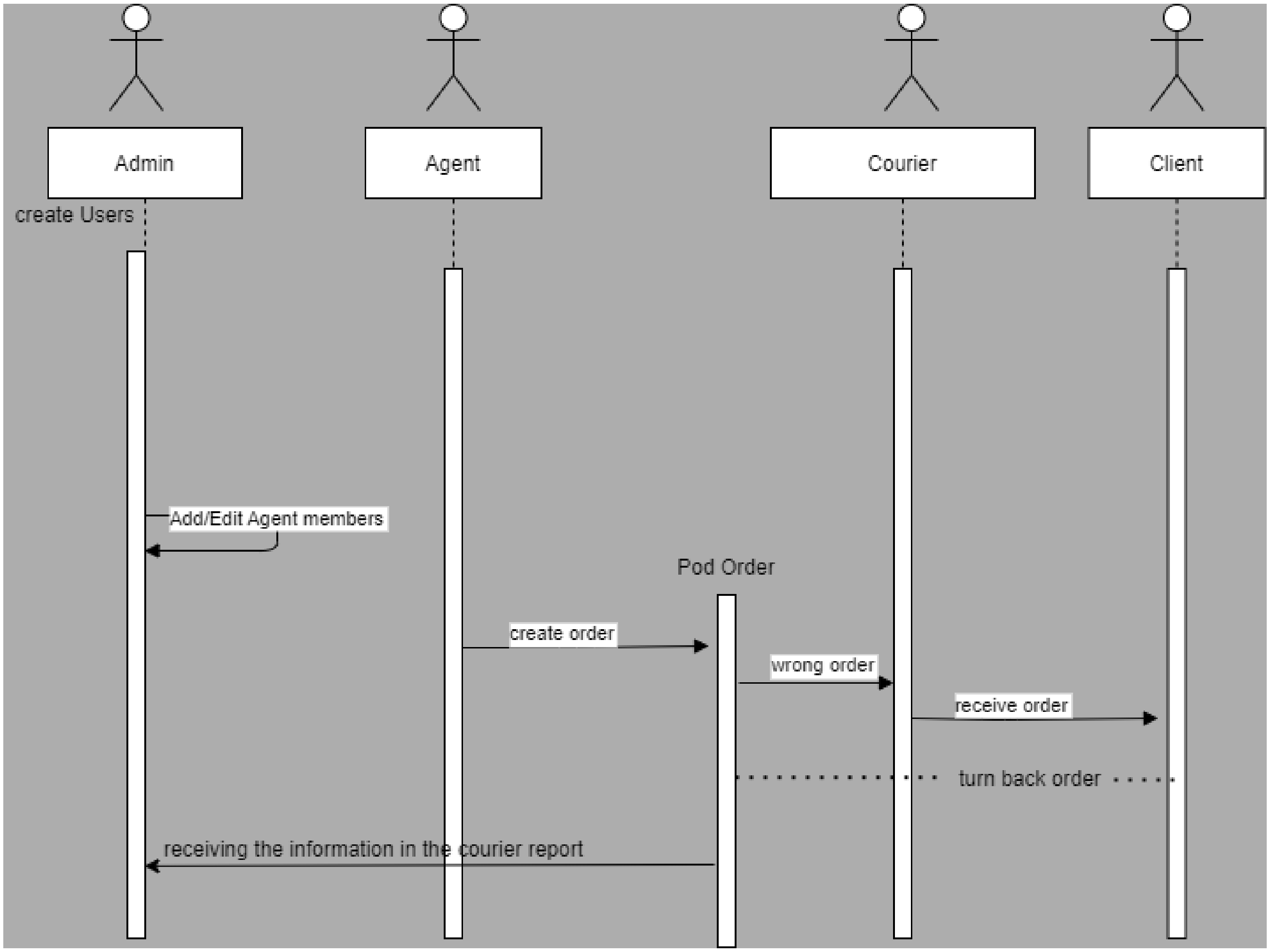


Fig. 22 – Client Sequence Diagram

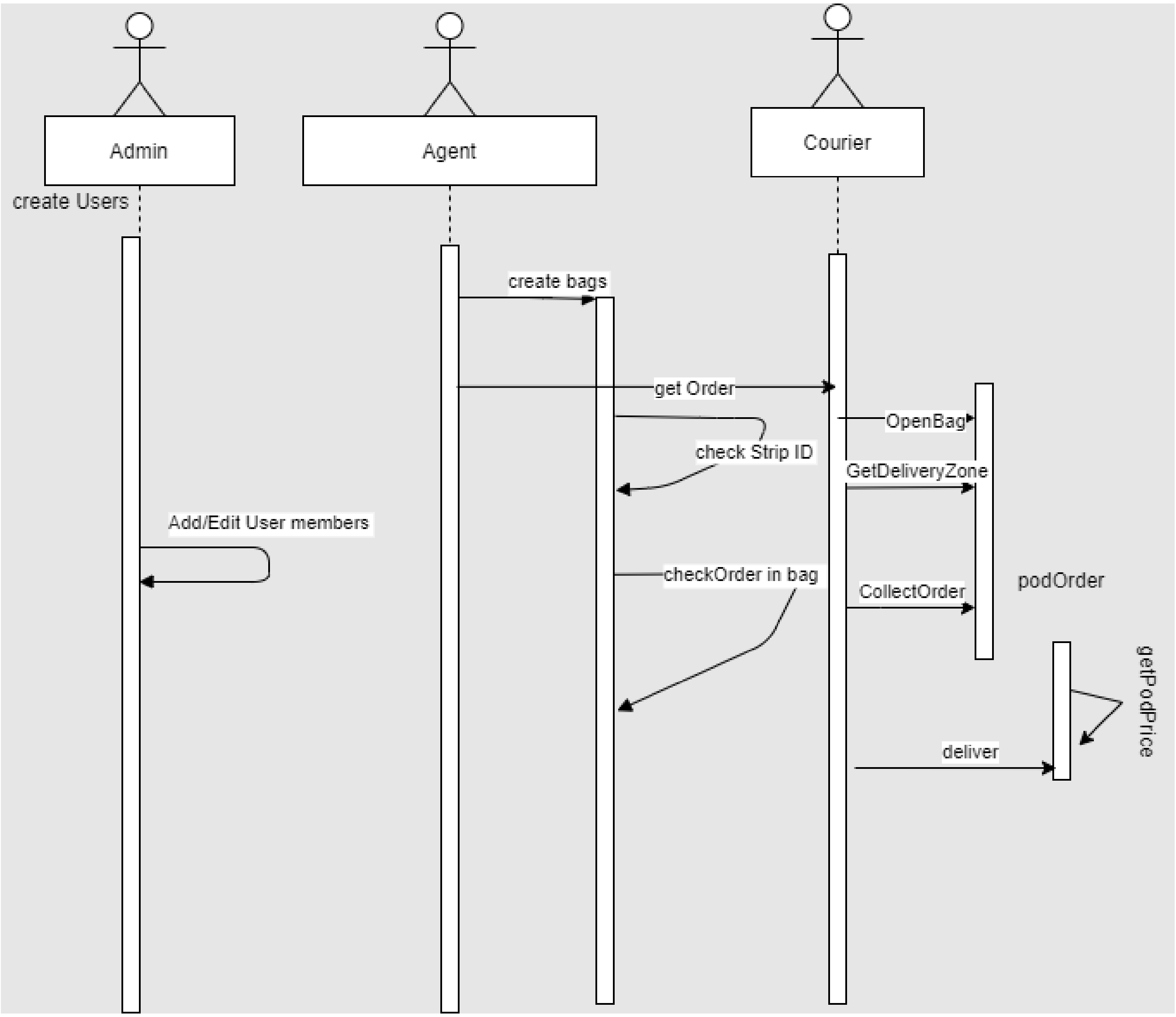


Fig. 23 – General Sequence Diagram

4.2.5 Data Flow Diagram

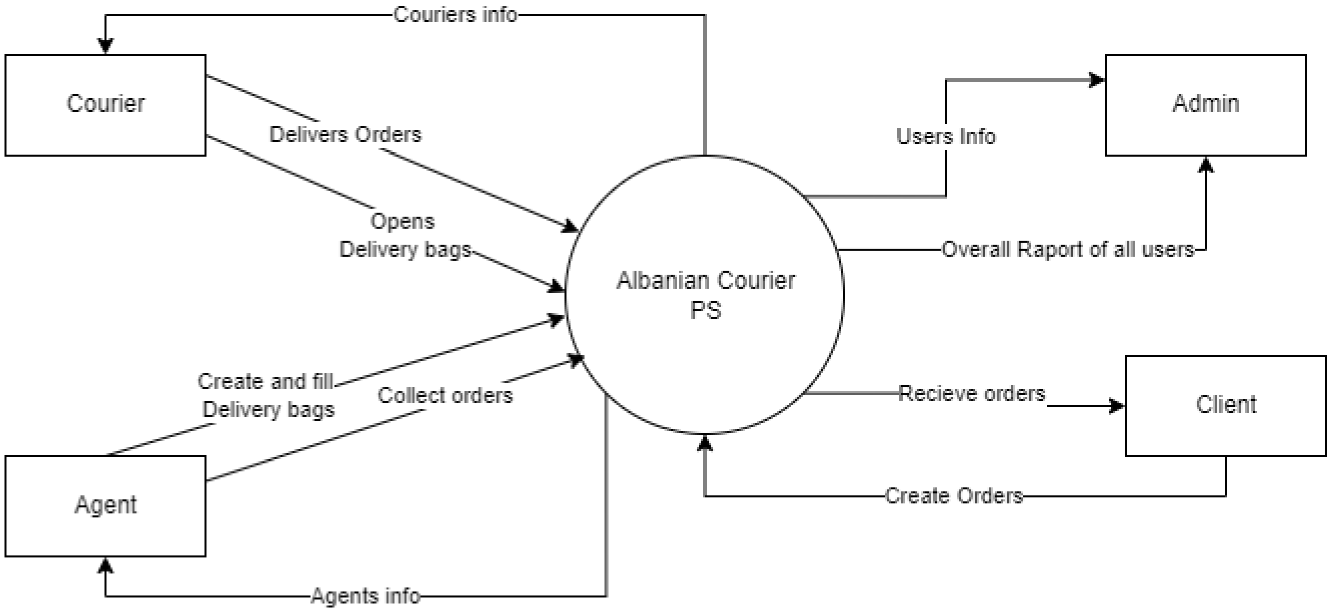


Fig. 24 – All Users Interaction Level 0

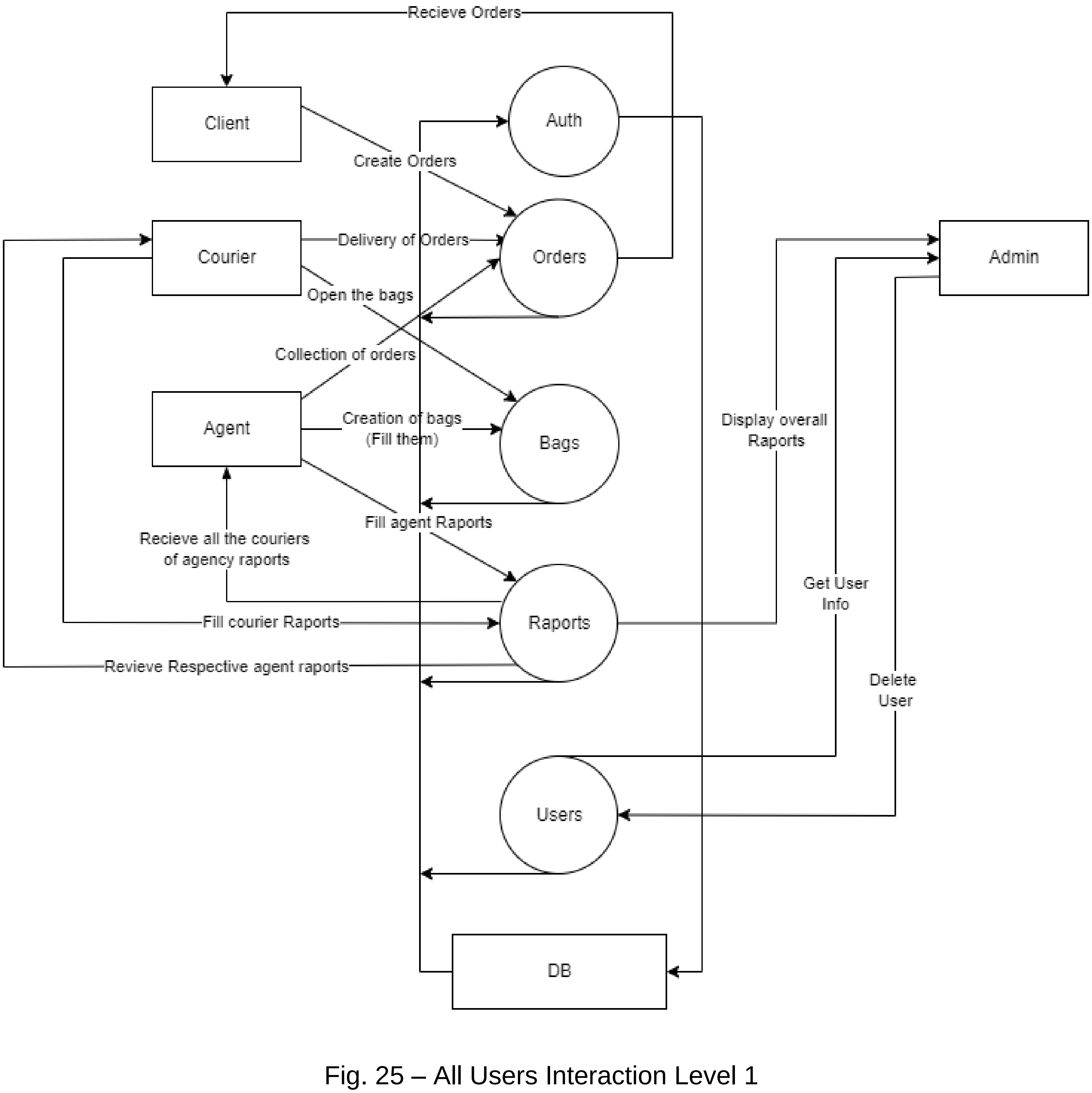


Fig. 25 – All Users Interaction Level 1

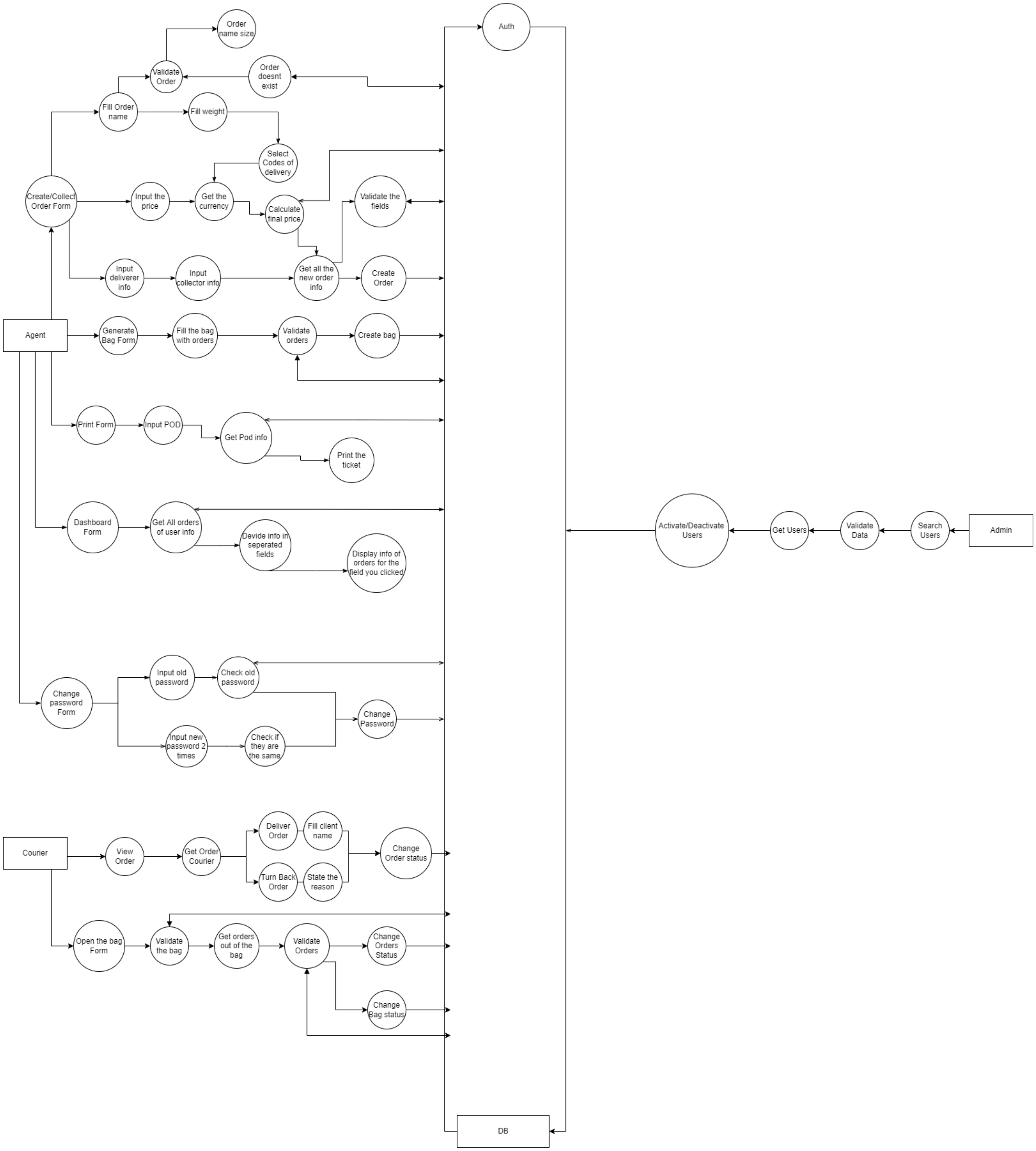


Fig. 26 – All Users Interaction Level 2

4.2.6 Collaboration Diagrams

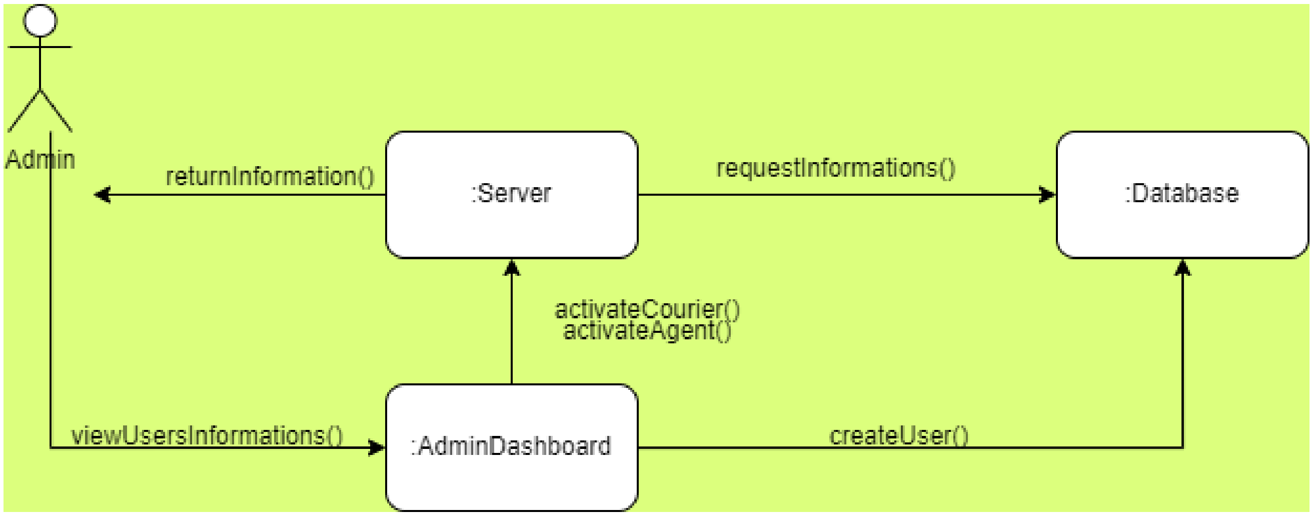


Fig. 27 – Admin Collaboration Diagram

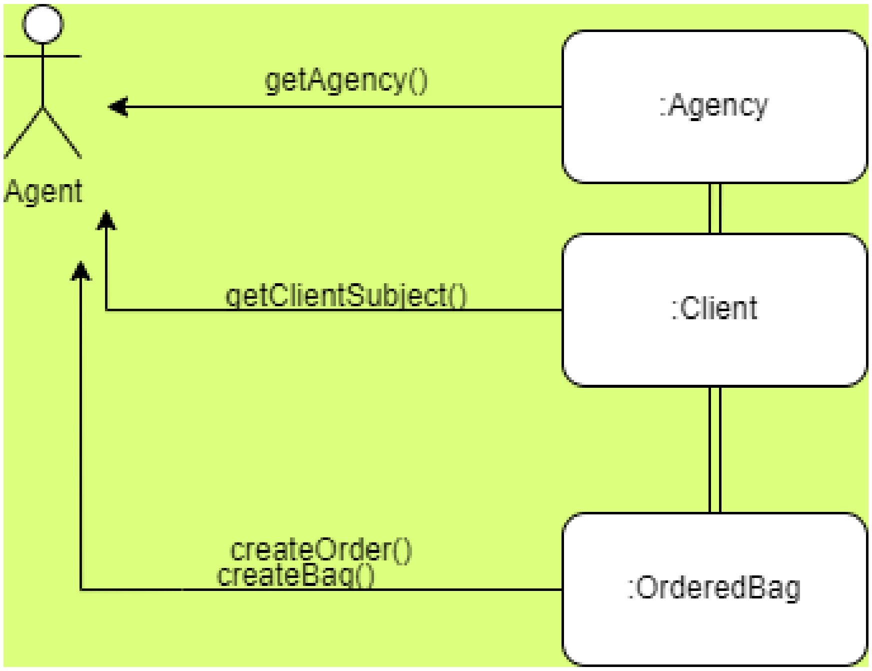


Fig. 28 – Agent Collaboration Diagram

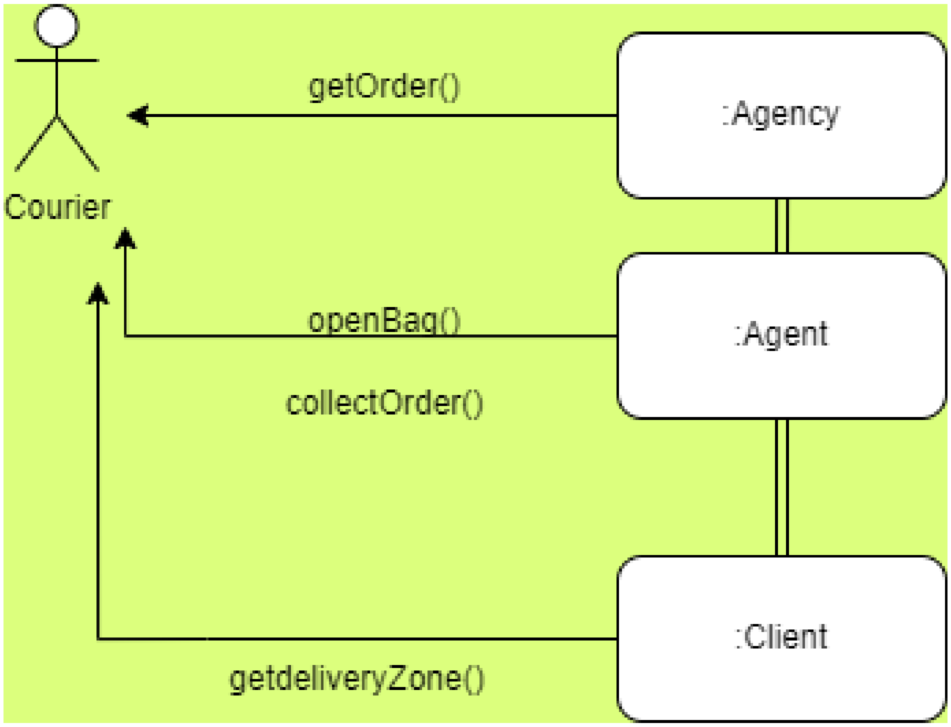


Fig. 29 – Courier Collaboration Diagram

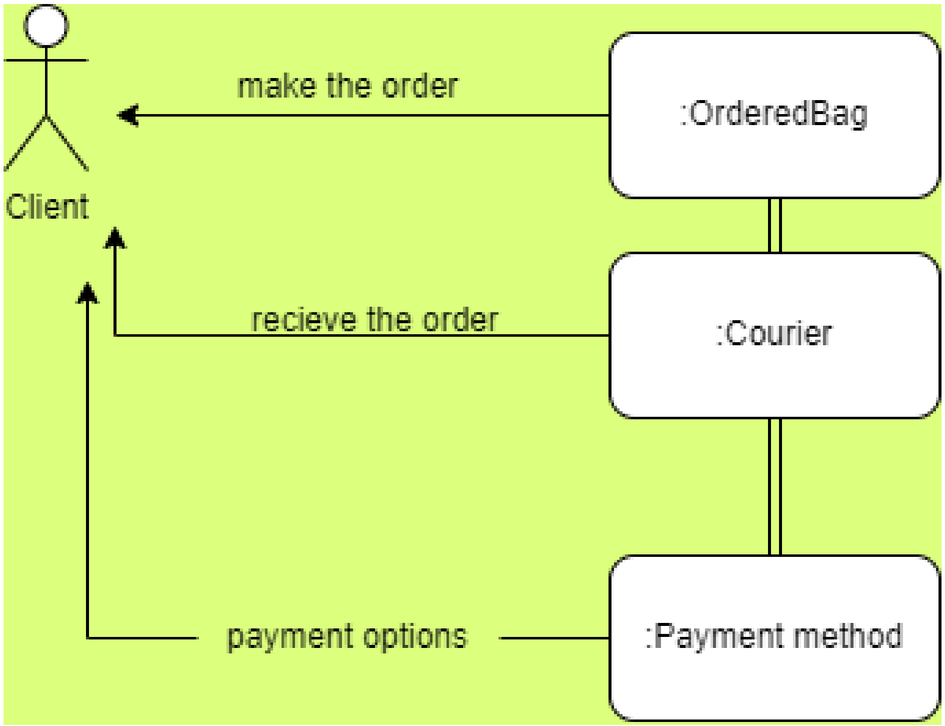


Fig. 30 – Client Collaboration Diagram

4.2.7 Class Diagram

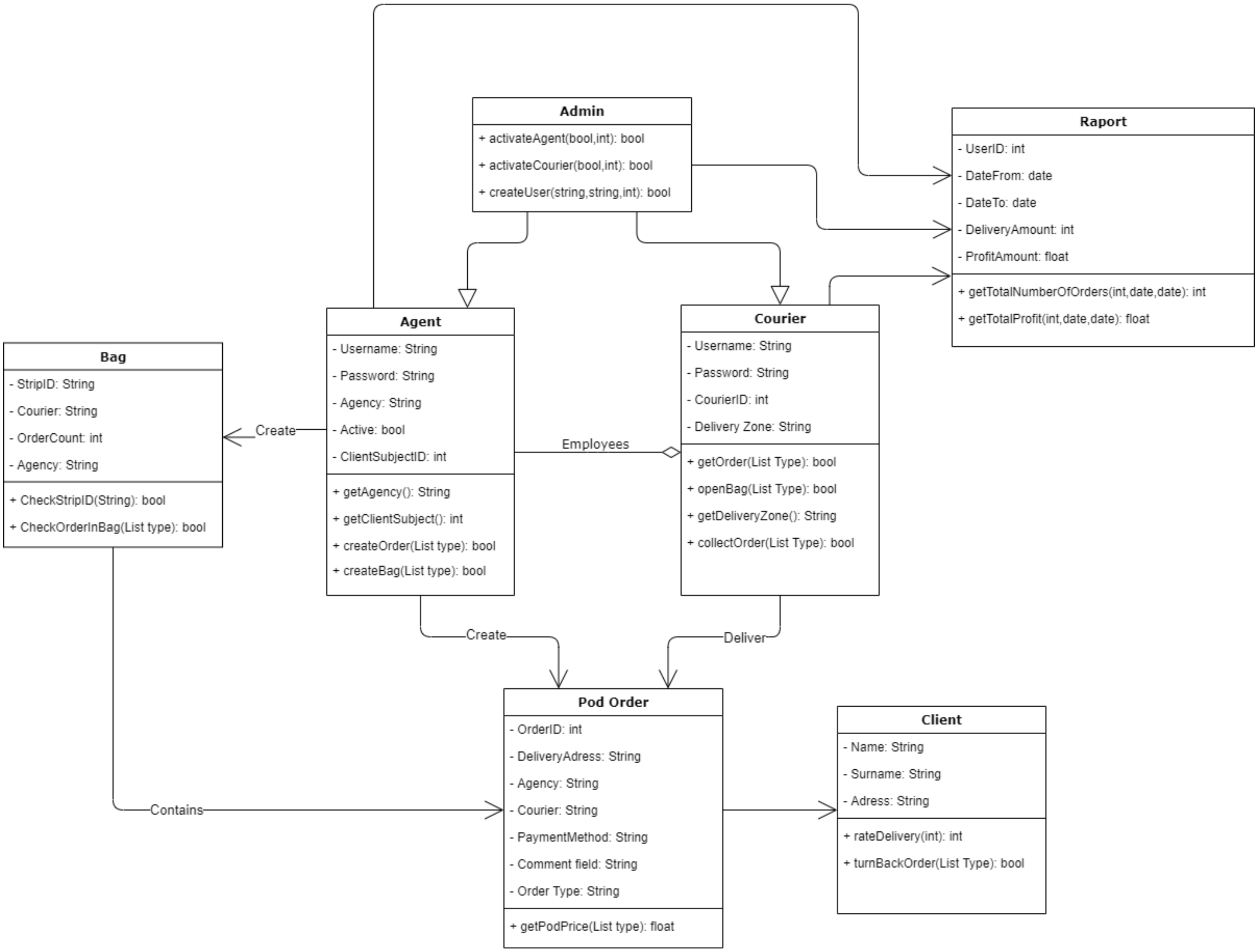


Fig. 31 – Processing of an order Class Diagram

4.2.8 Object Diagram

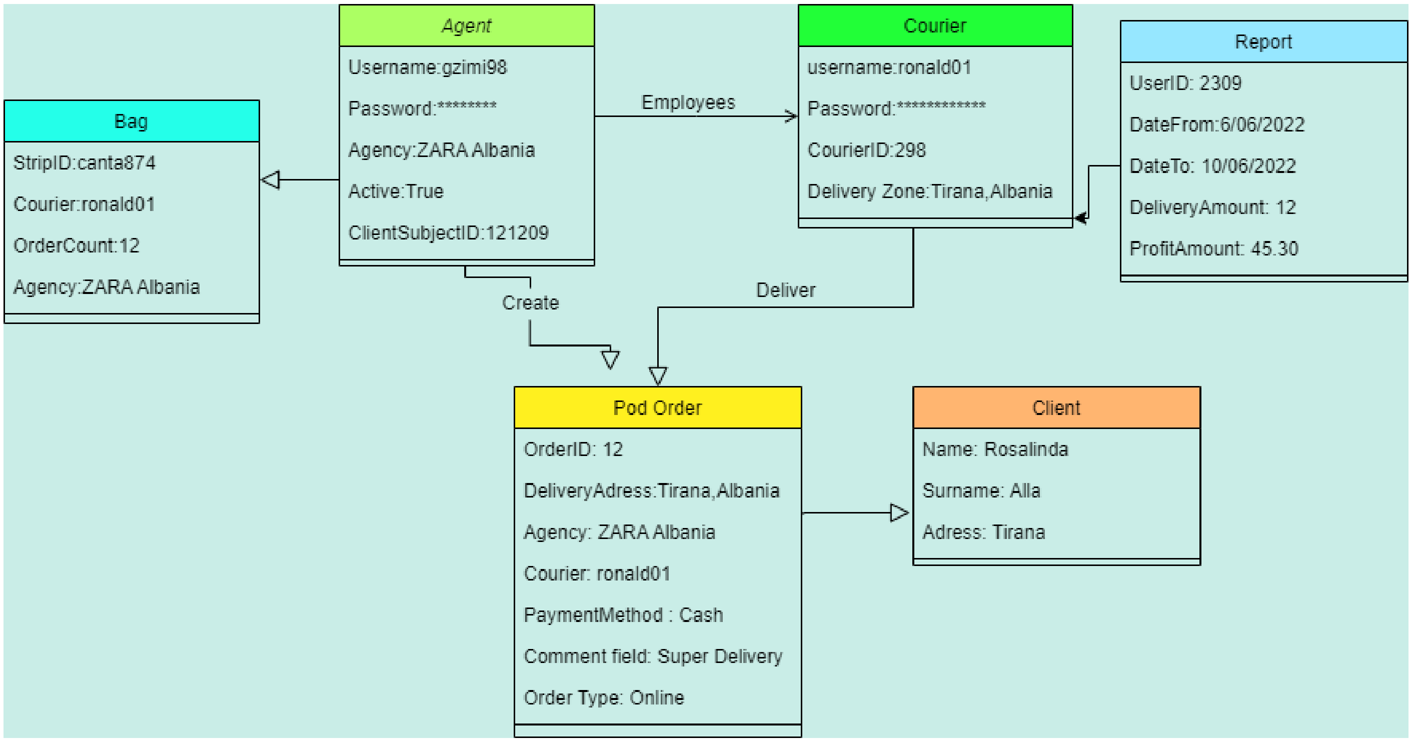
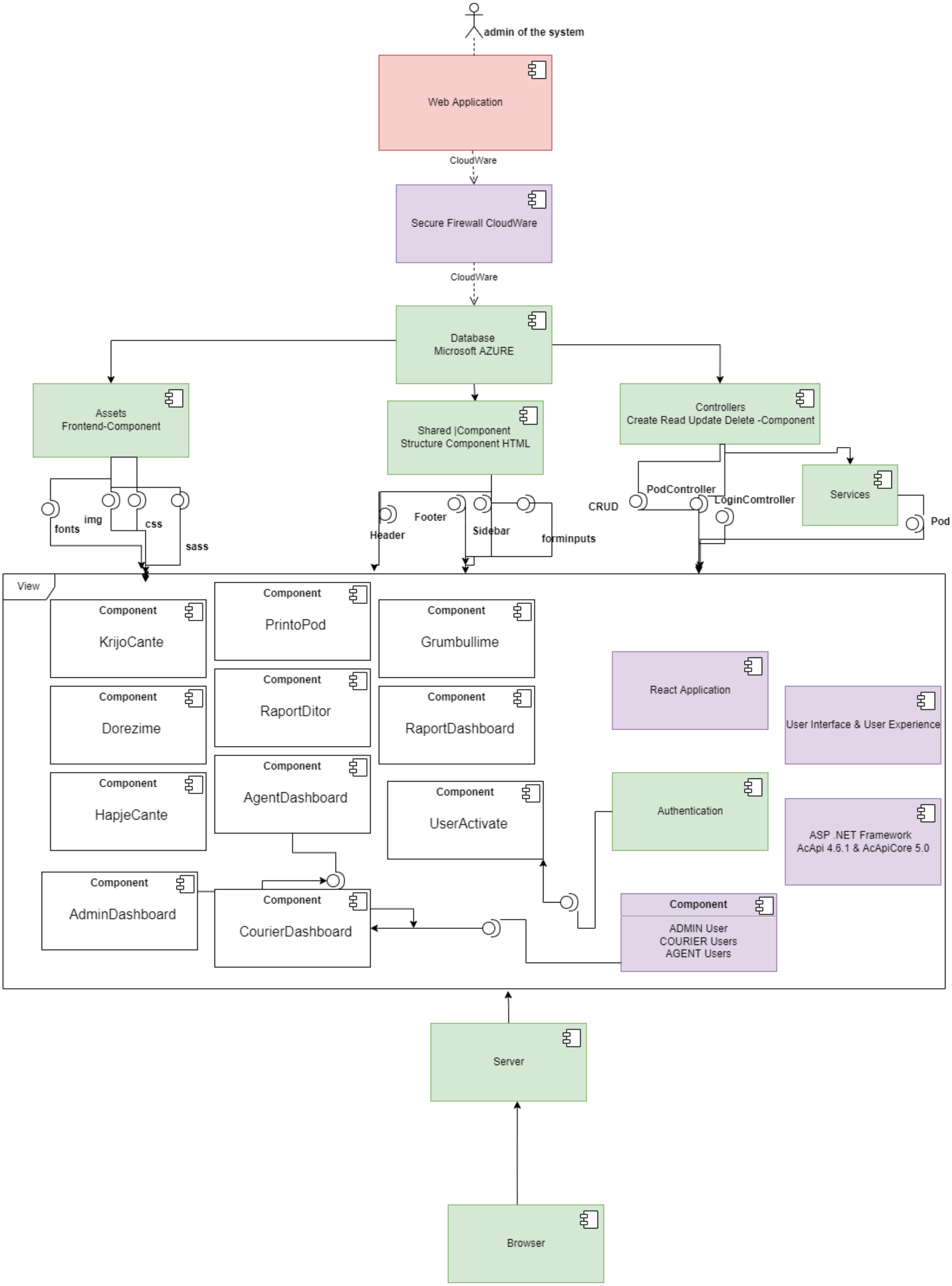


Fig. 32 – Object Diagram

4.2.9 Component Diagram



4.2.10 Deployment Diagram

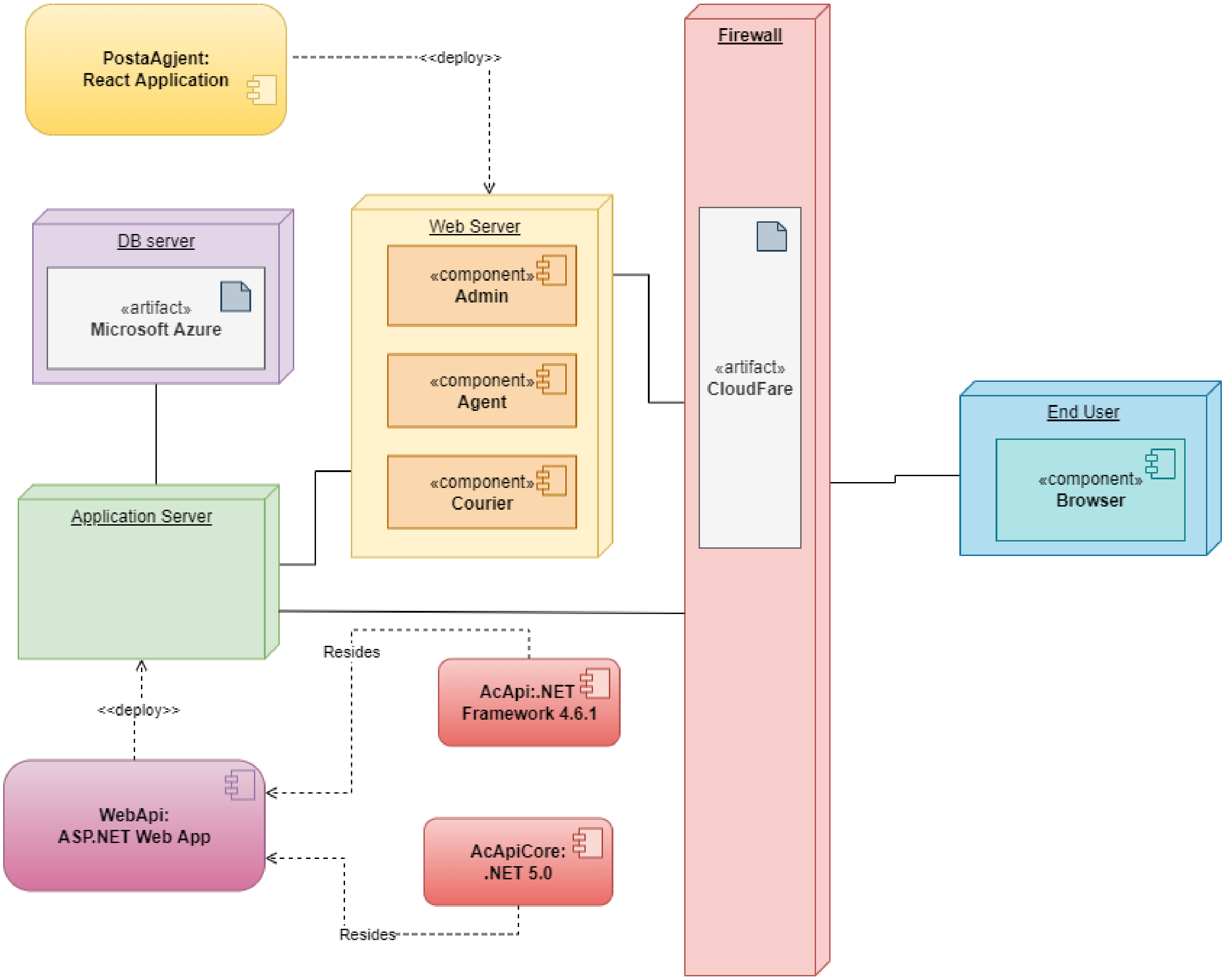


Fig. 34 – Deployment Diagram

4.2.11 Entity Relationship Diagram

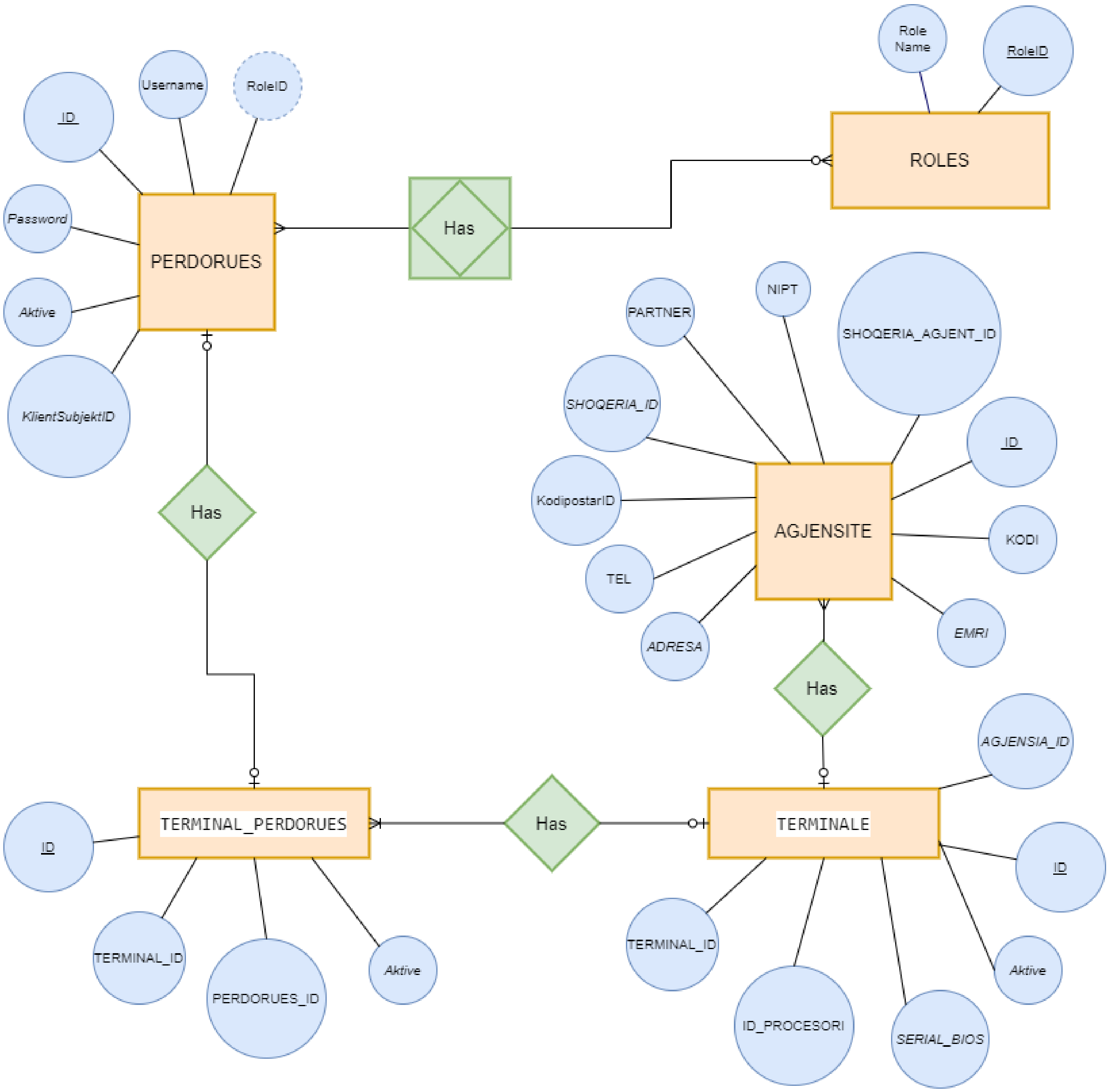


Fig. 35 – ERD Diagram

4.2.12 Relational Schema

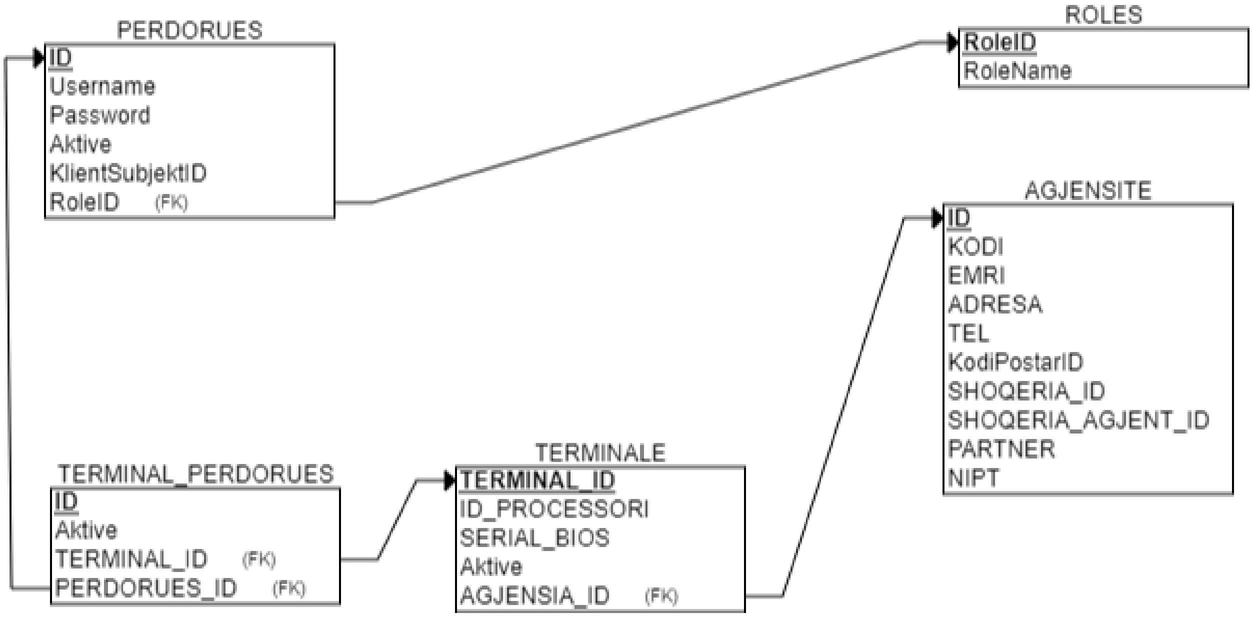


Fig. 36 – RS derived from ERD Diagram

5. Implementation

5.1 Technology Used

5.1.1 Frontend Development

HTML5 – A markup language used for the structuring and displaying content in the WWW.
CSS – A styling language, we have used for the design
React JS – A JavaScript framework for building user interfaces

5.1.2 Backend Development

C# ASP .NET – A modern, object-oriented, and type-safe programming language that provides tons of libraries, components, UI class libraries and other resources that speed up the development.

5.1.3 Database

MS Azure – A fully managed platform as a service database engine that handles most of the database management functions without user involvement

5.2 User Manual

Admin:

- Username: testuser
- Password: 12345678

Agent

- Username: testagent
- Password: asdfghjkl

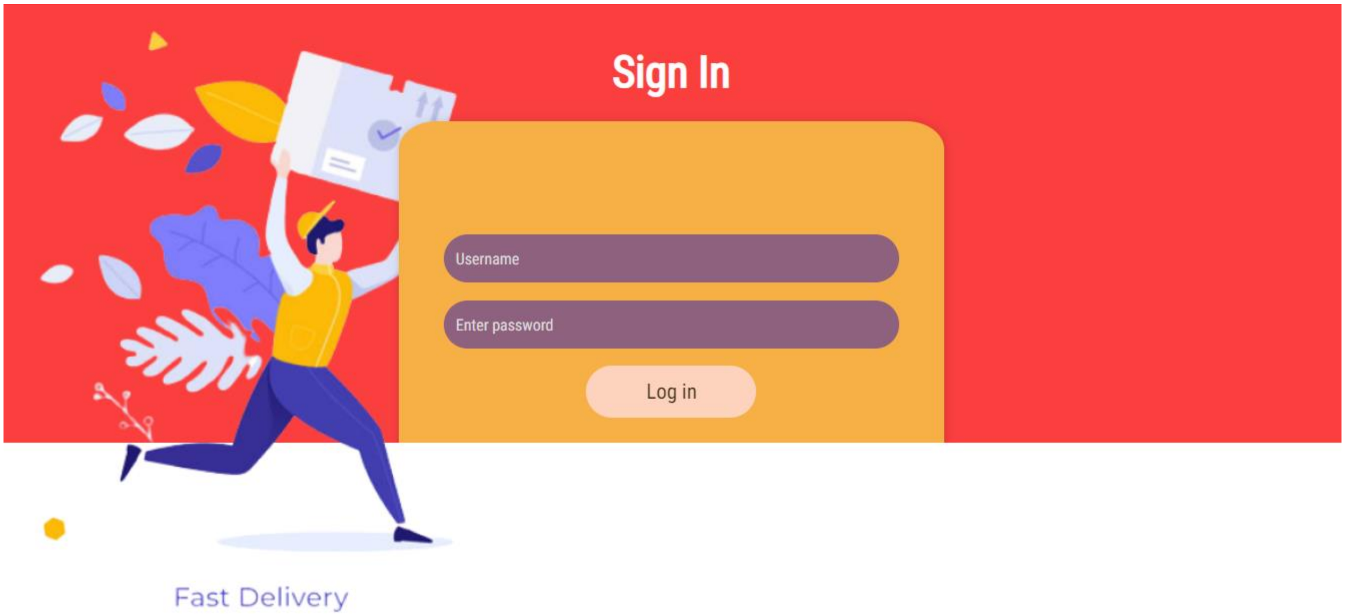
Courier

- Username: testcourier
- Password: police

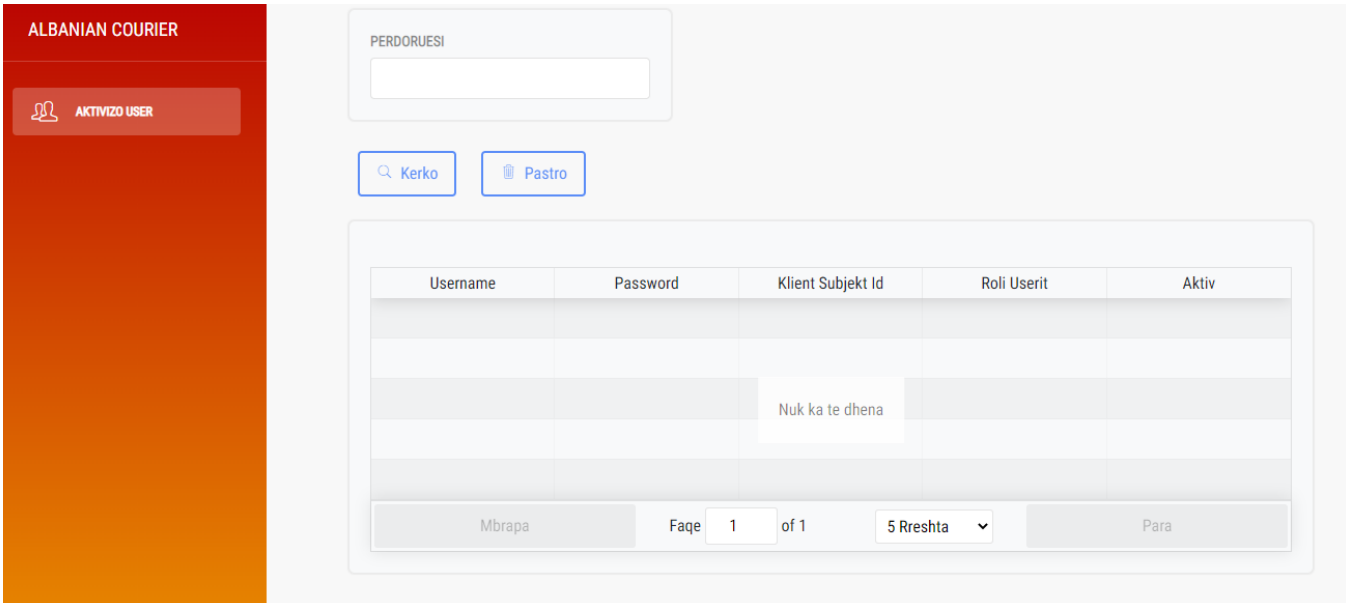
Database

- IP: groupsix.database.windows.net
- User: arbserver
- Password: wvh3r8f1ytjk.

5.3 Software Screenshots



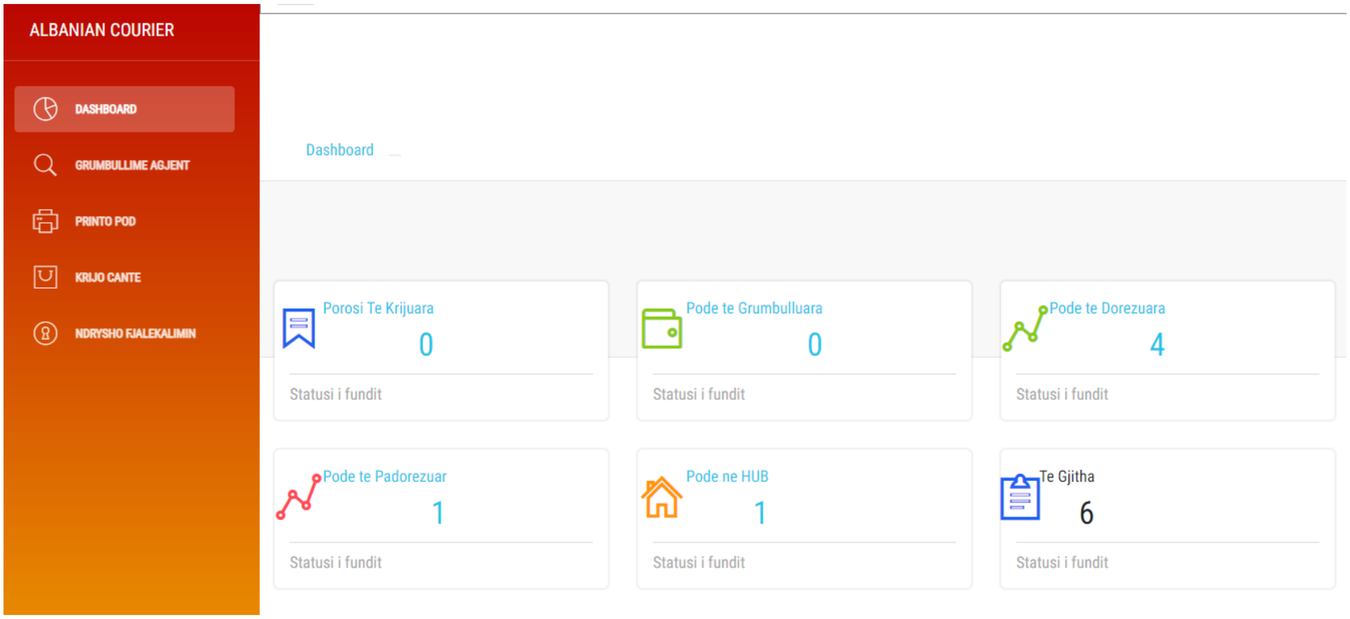
User Sign In Page



Username	Password	Klient Subjekt Id	Roli Userit	Aktiv

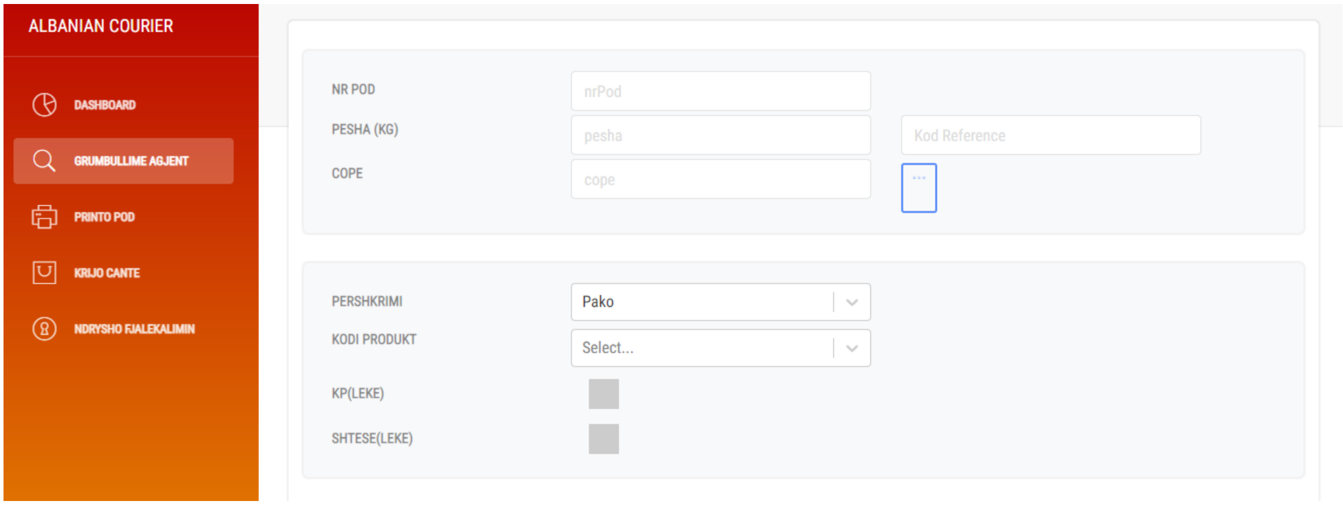
Nuk ka te dhena

Admin Page



Porosi Te Krijuara 0 Statusi i fundit	Pode te Grumbulluara 0 Statusi i fundit	Pode te Dorezuara 4 Statusi i fundit
Pode te Padorezuar 1 Statusi i fundit	Pode ne HUB 1 Statusi i fundit	Te Gjitha 6 Statusi i fundit

Agent Dashboard



NR POD: nrPod

PESHA (KG): pesha — Kod Reference

COPE: cope

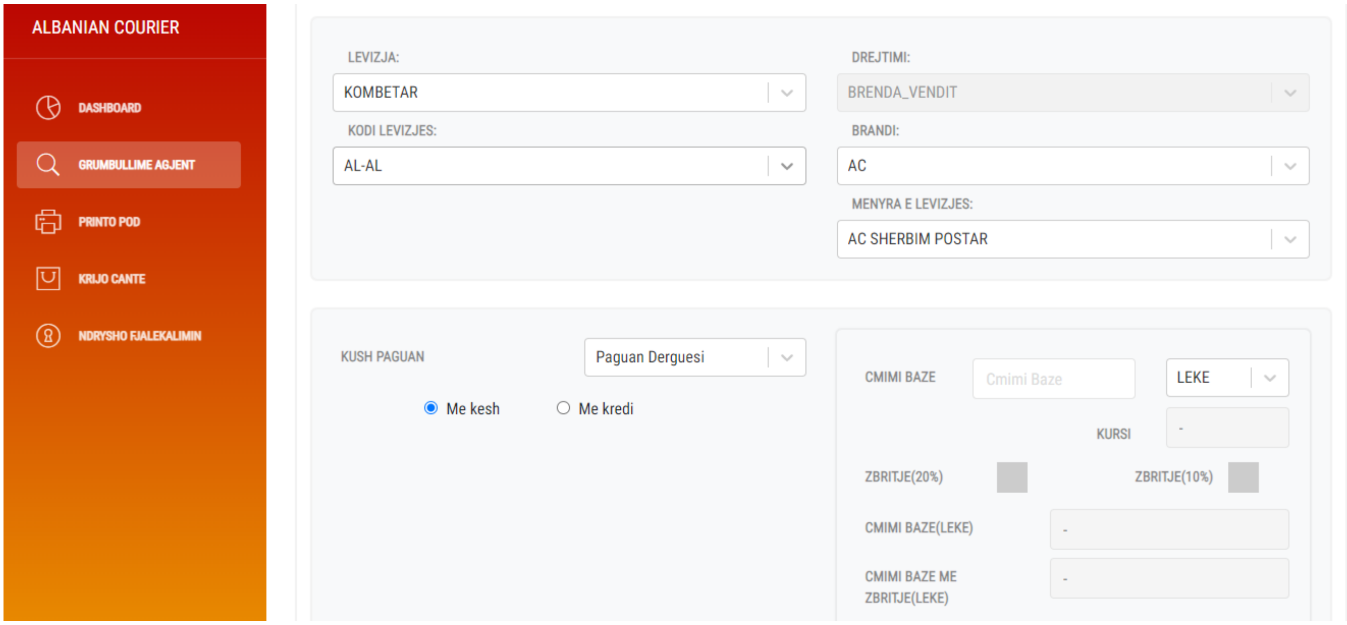
PERSHKRIMI: Pako

KODI PRODUKT: Select...

KP(LEKE)

SHTESE(LEKE)

Agent’s Order Collection Page (1)



LEVIZJA: KOMBETAR

DREJTIMI: BRENDA_VENDIT

KODI LEVIZJES: AL-AL

BRANDI: AC

MENYRA E LEVIZJES: AC SHERBIM POSTAR

KUSH PAGUAN: Paguan Derguesi

● Me kesh ○ Me kredi

CMIMI BAZE: Cmimi Baze LEKE

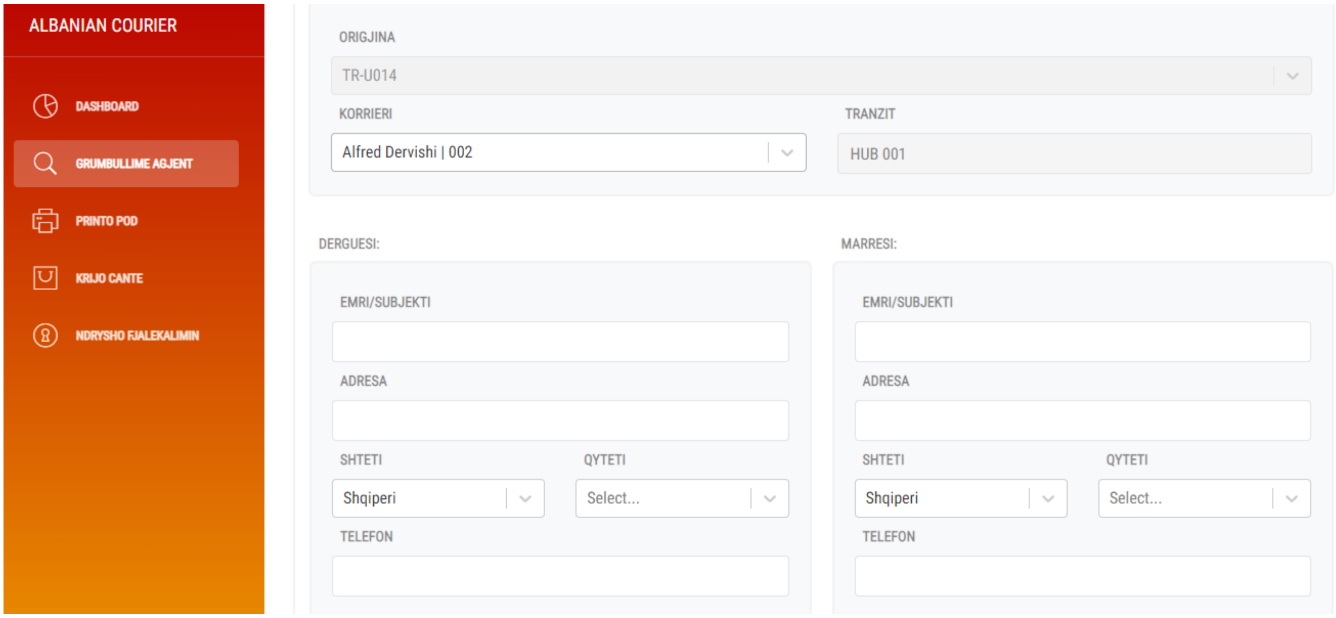
KURSI: -

ZBRITJE(20%) □ ZBRITJE(10%) □

CMIMI BAZE(LEKE): -

CMIMI BAZE ME ZBRITJE(LEKE): -

Agent’s Order Collection Page (2)



ORIGJINA: TR-U014

KORRIERI: Alfred Dervishi | 002

TRANZIT: HUB 001

DERGUESI:

EMRI/SUBJEKTI

ADRESA

SHTETI: Shqiperi

QYTETI: Select...

TELEFON

MARRESI:

EMRI/SUBJEKTI

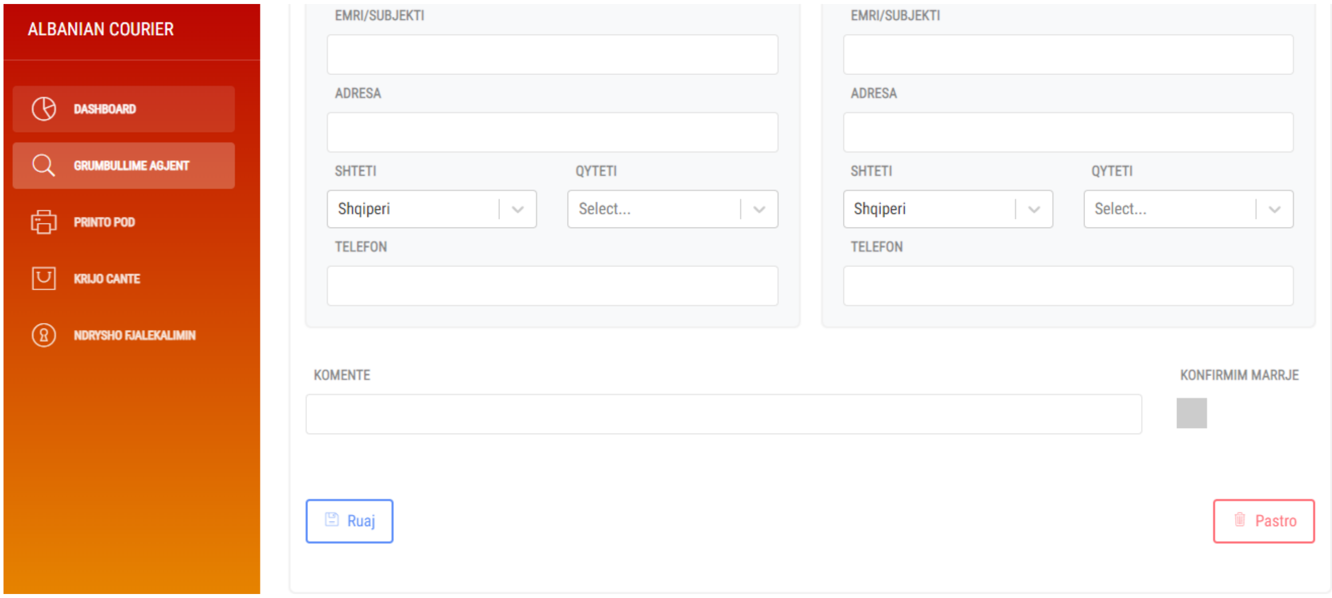
ADRESA

SHTETI: Shqiperi

QYTETI: Select...

TELEFON

Agent’s Order Collection Page (3)



EMRI/SUBJEKTI

ADRESA

SHTETI: Shqiperi

QYTETI: Select...

TELEFON

EMRI/SUBJEKTI

ADRESA

SHTETI: Shqiperi

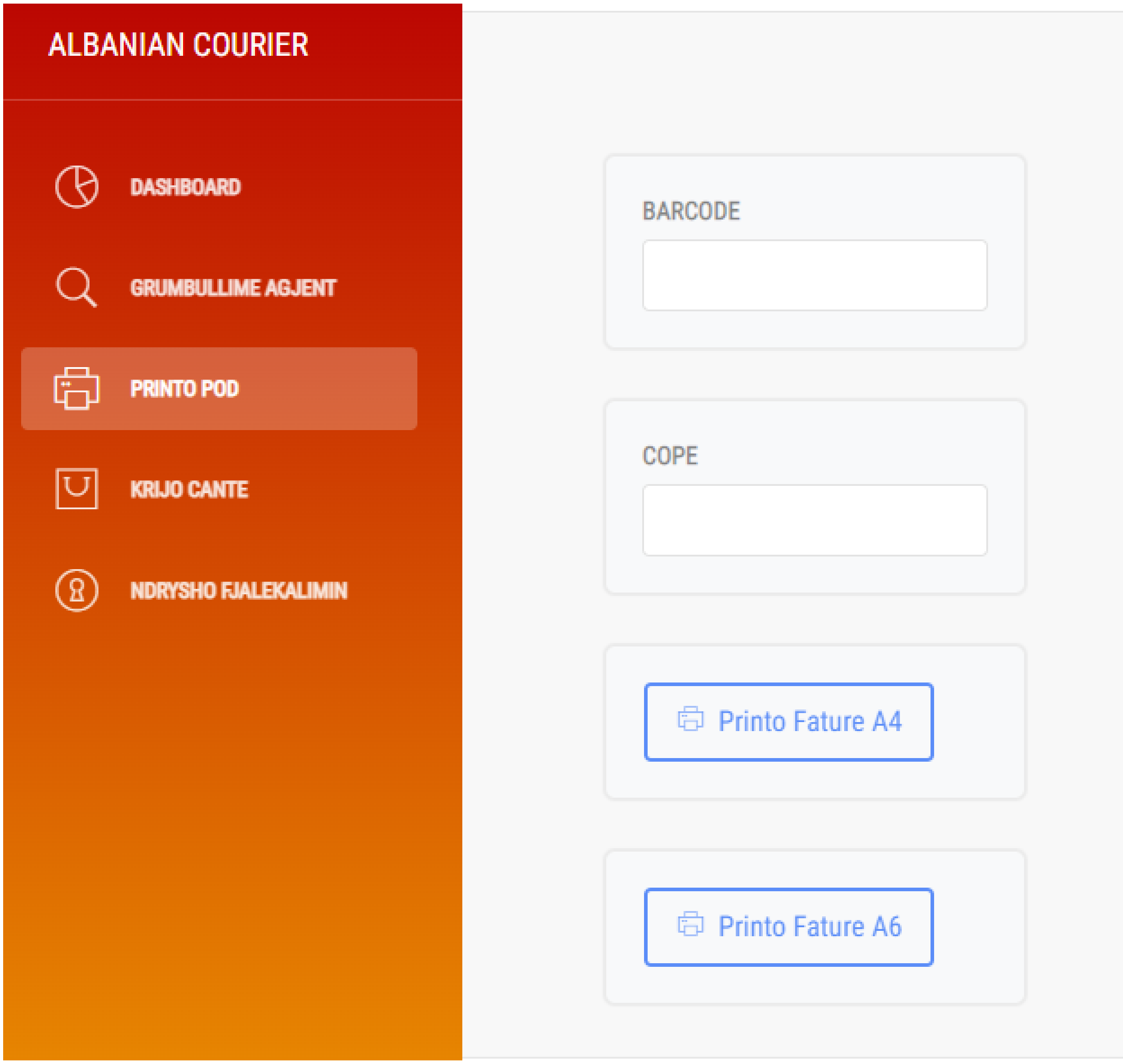
QYTETI: Select...

TELEFON

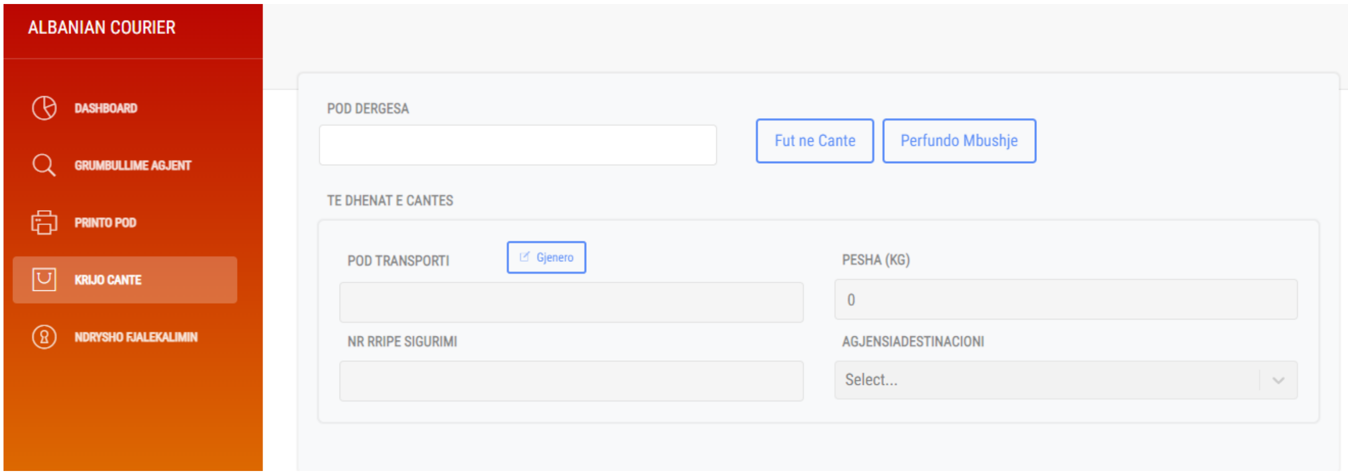
KOMENTE

KONFIRMIM MARRJE □

Agent’s Order Collection Page (4)



Agent’s Print Order Bill Page



POD DERGESA

TE DHENAT E CANTES

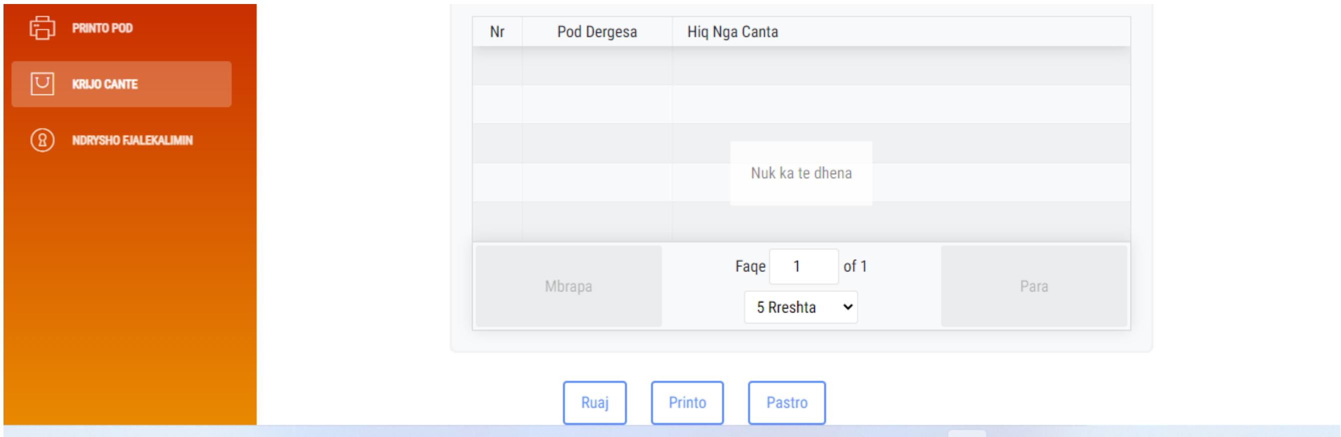
POD TRANSPORTI

PESHA (KG): 0

NR RRIPE SIGURIMI

AGJENSIADESTINACIONI: Select...

Agent’s Create Bag Page (1)



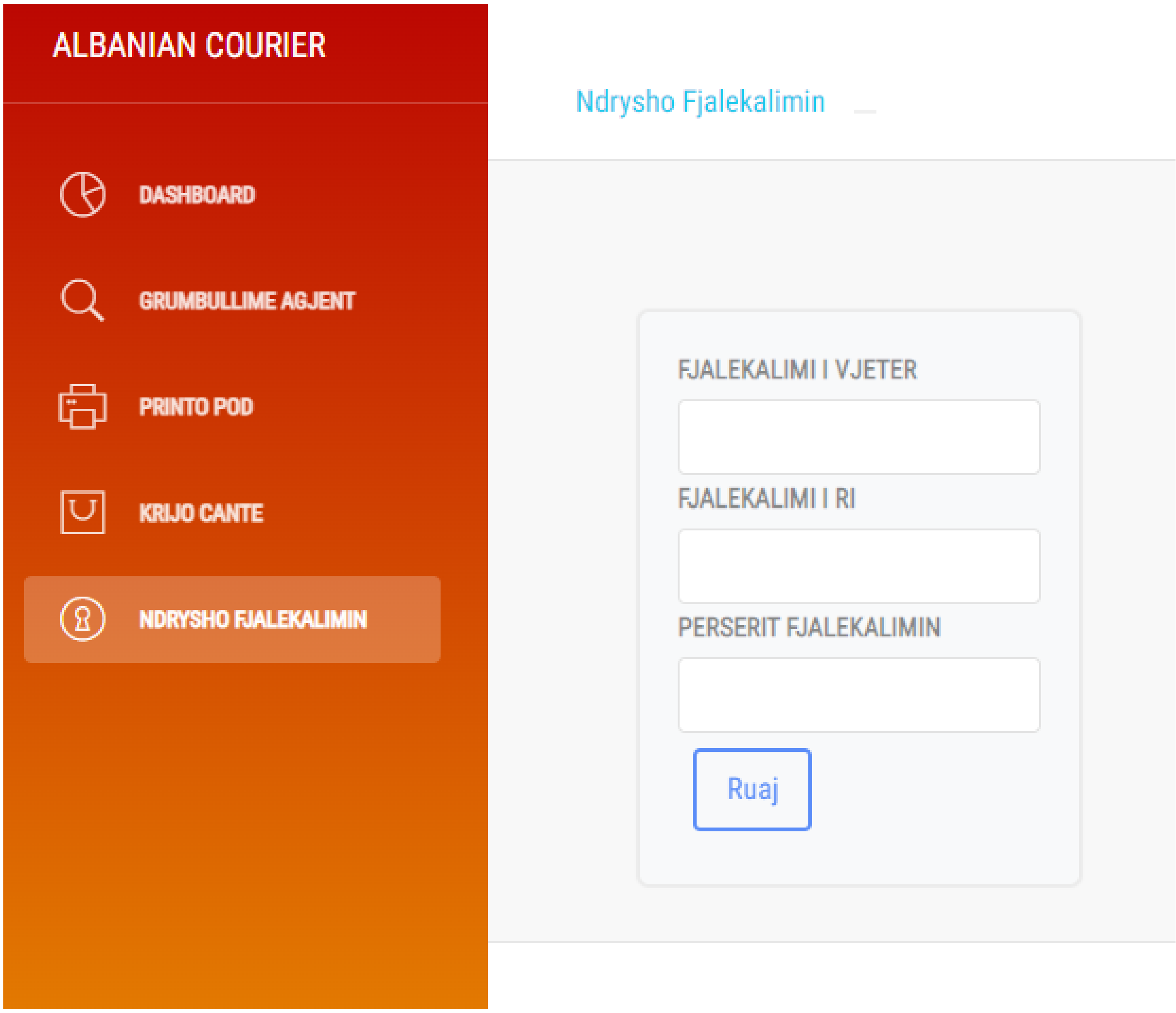
Nr	Pod Dergesa	Hiq Nga Canta

Nuk ka te dhena

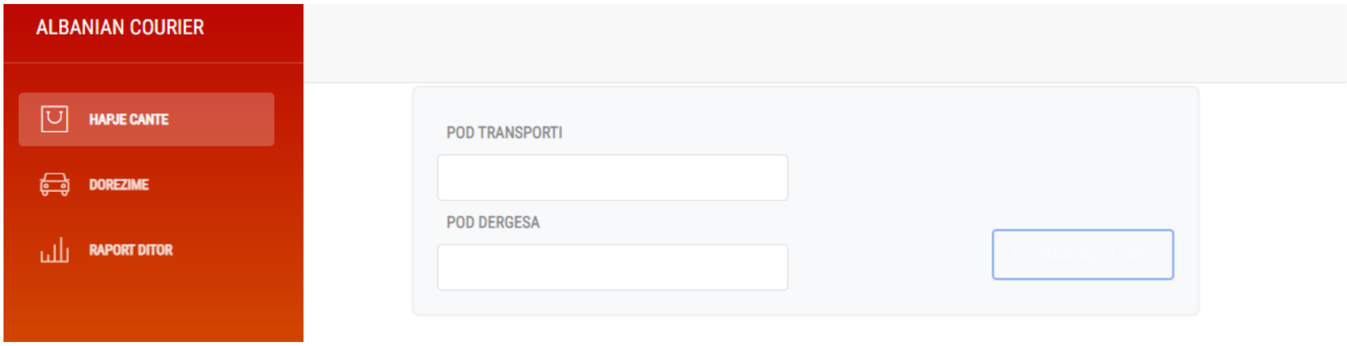
Faqe 1 of 1

5 Rreshta

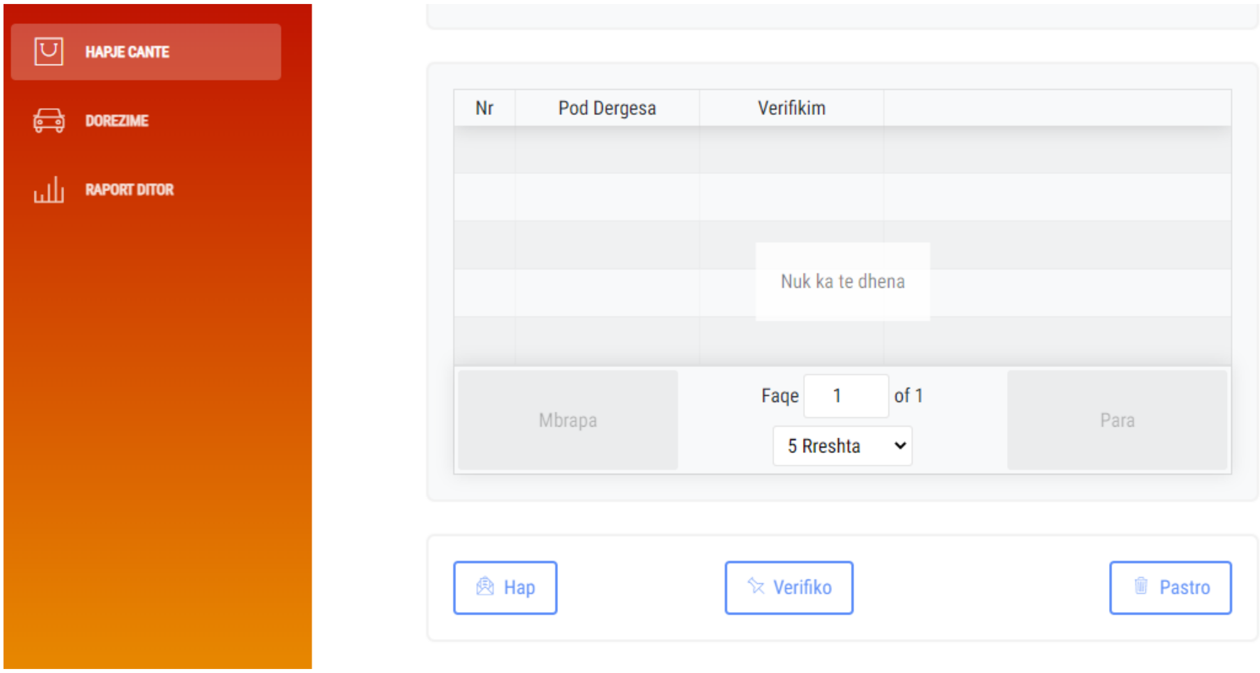
Agent’s Create Bag Page (2)



Agent’s Change Password Page



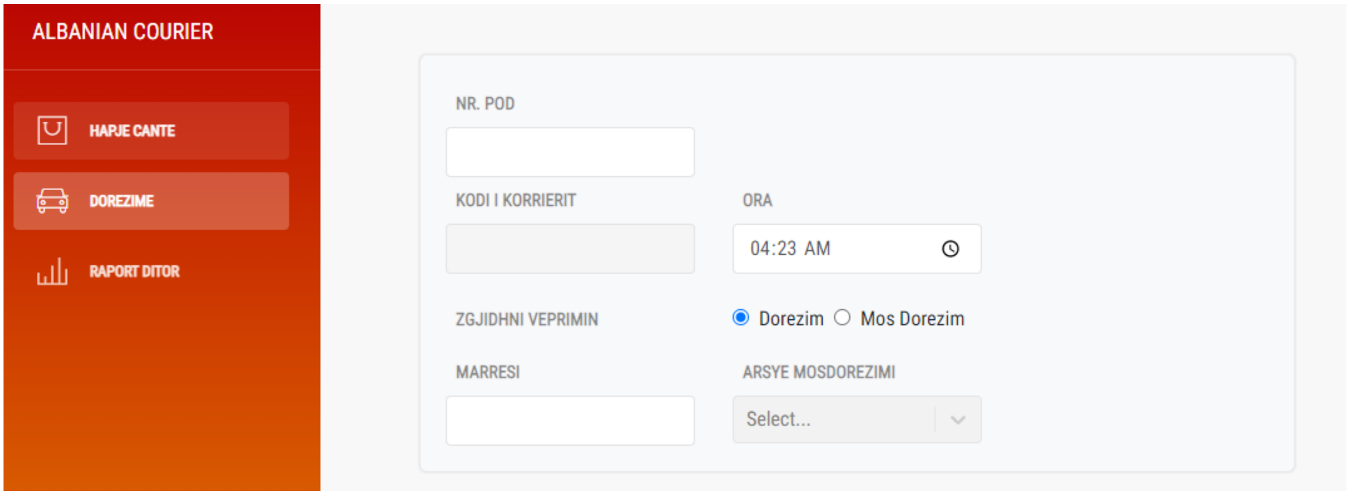
Courier Open Bag Page (1)



Nr	Pod Dergesa	Verifikim
Nuk ka te dhena		

Mbrapa — Faqe 1 of 1 — 5 Rreshta — Para

Courier Open Bag Page (2)



NR. POD

KODI I KORRIERIT

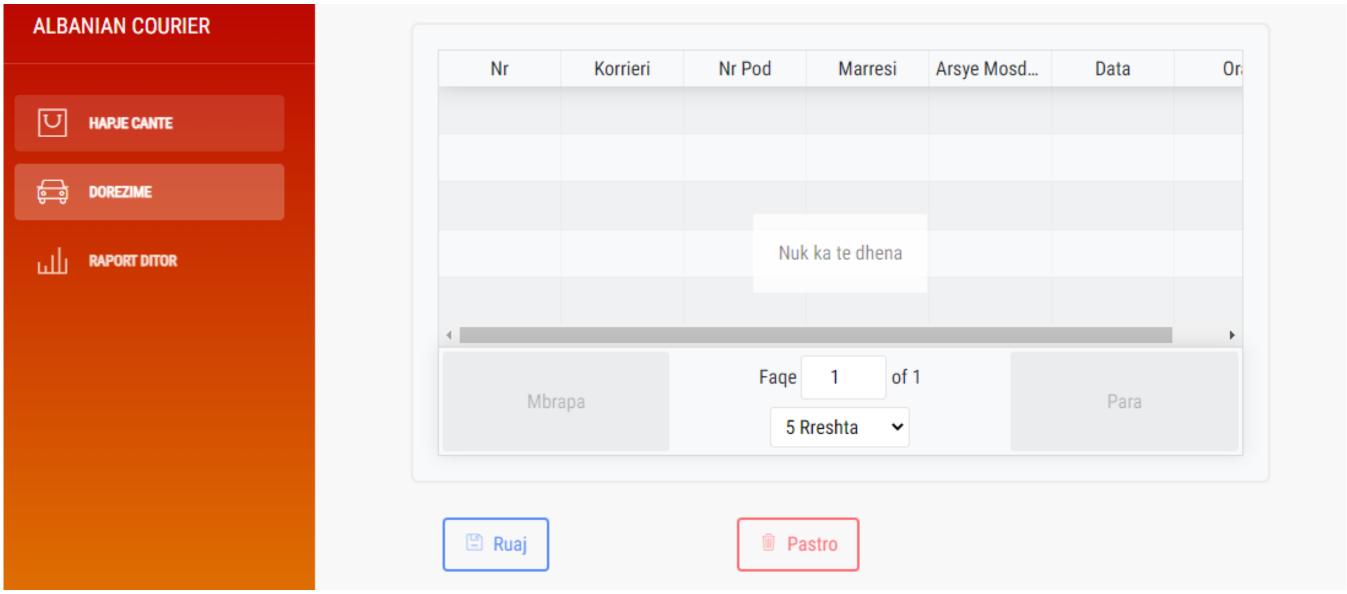
ORA: 04:23 AM

ZGJIDHNI VEPRIMIN: ● Dorezim ○ Mos Dorezim

MARRESI

ARSYE MOSDOREZIMI: Select...

Courier Delivery Page (1)



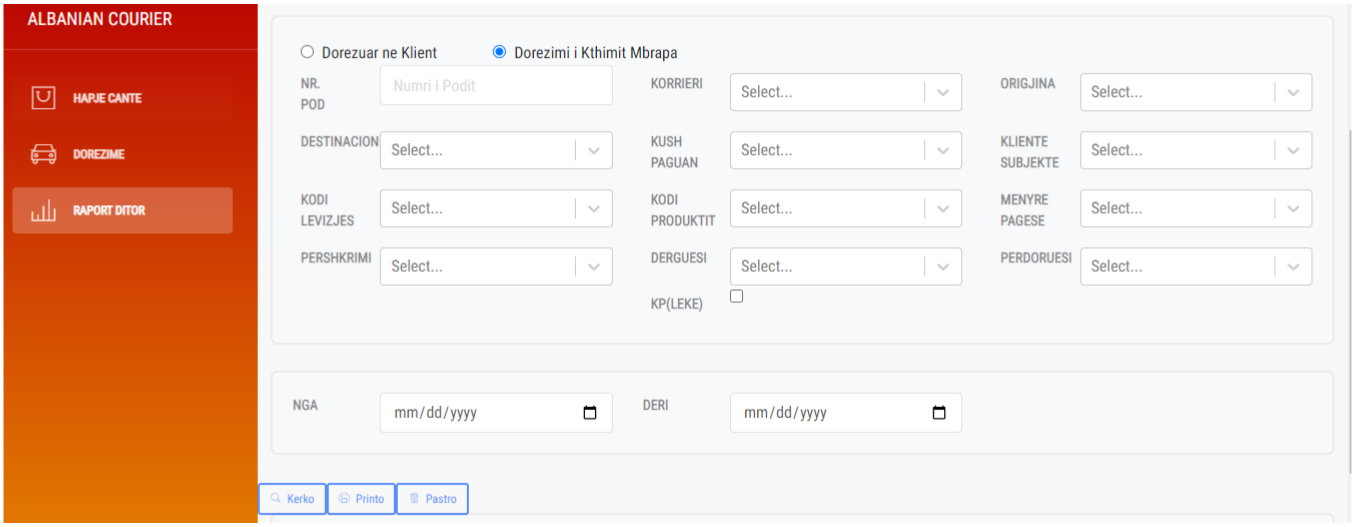
Nr	Korrieri	Nr Pod	Marresi	Arsye Mosd...	Data	Or

Nuk ka te dhena

Faqe 1 of 1

5 Rreshta

Courier Delivery Page (2)



○ Dorezuar ne Klient ● Dorezimi i Kthimit Mbrapa

NR. POD: Numri i Podit

KORRIERI: Select...

ORIGJINA: Select...

DESTINACION: Select...

KUSH PAGUAN: Select...

KLIENTE SUBJEKTE: Select...

KODI LEVIZJES: Select...

KODI PRODUKTIT: Select...

MENYRE PAGESE: Select...

PERSHKRIMI: Select...

DERGUESI: Select...

PERDORUESI: Select...

KP(LEKE) □

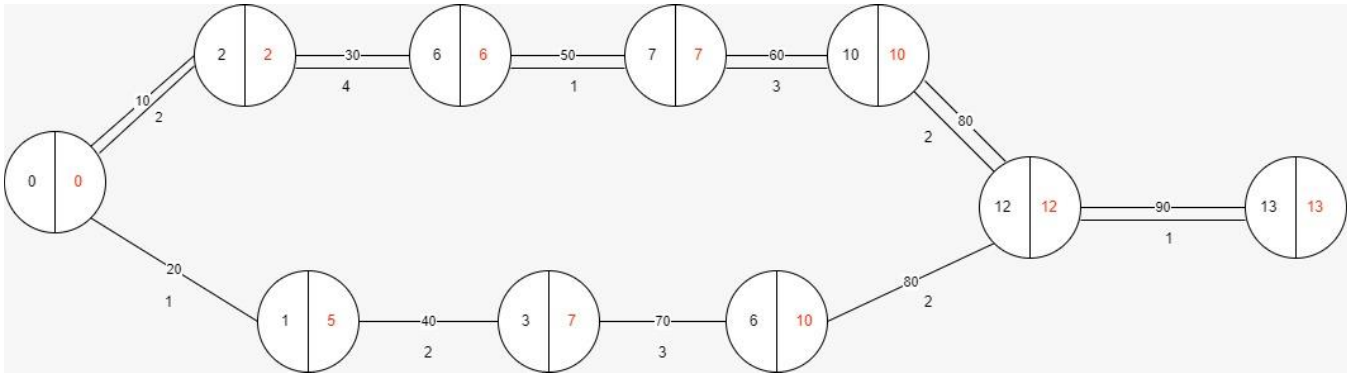
NGA: mm/dd/yyyy

DERI: mm/dd/yyyy

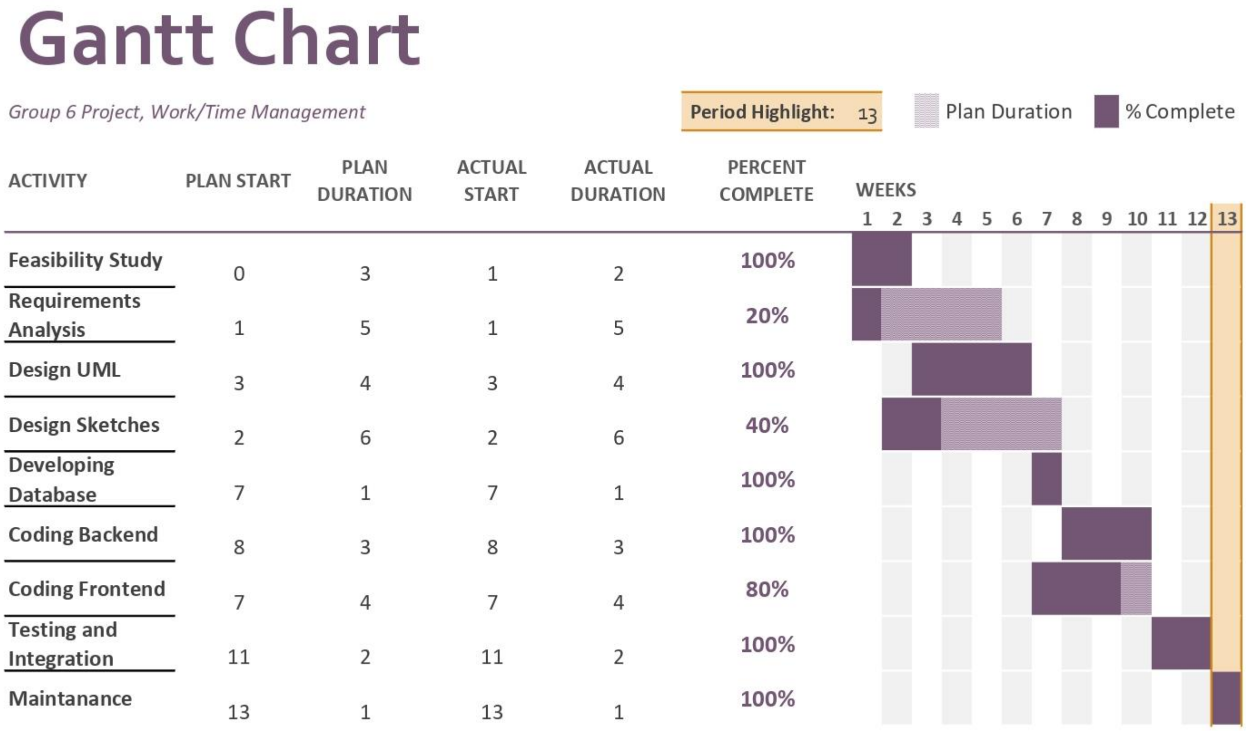
Courier Generate Daily Report Page

6. Project Planning

6.1 Activity Network



6.2 Gantt Chart and Activity Table



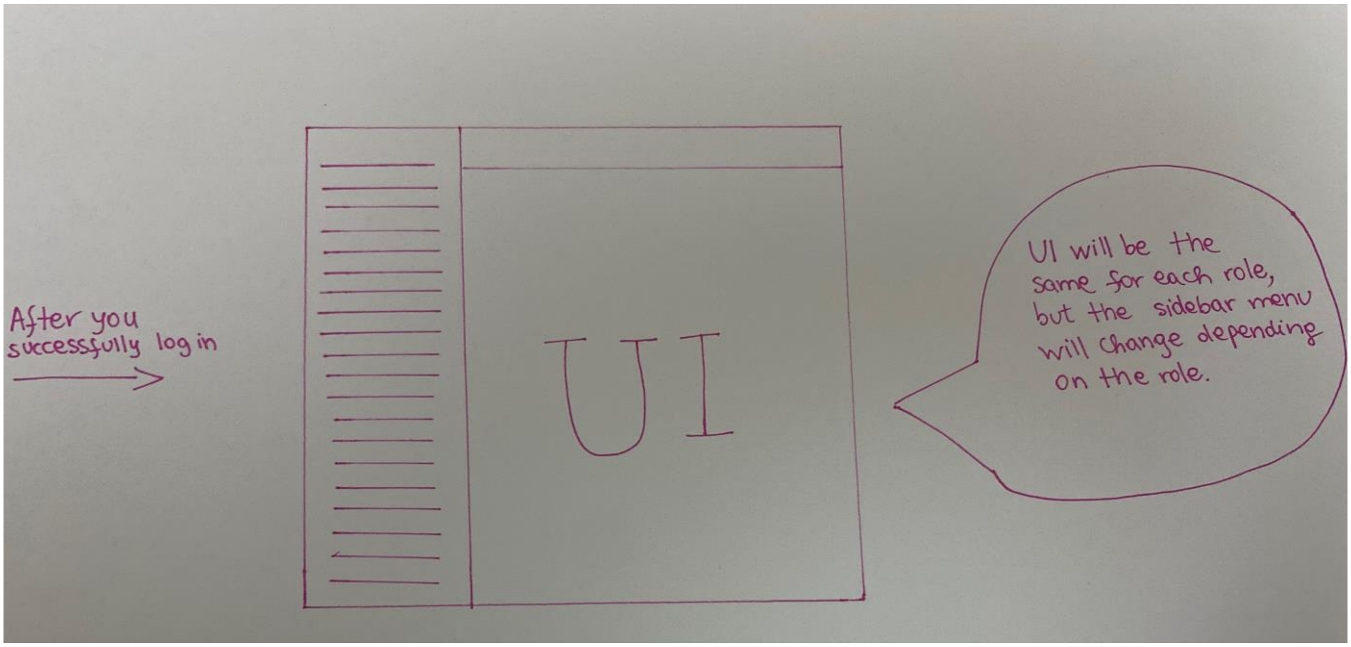
Gantt Chart

Group 6 Project, Work/Time Management

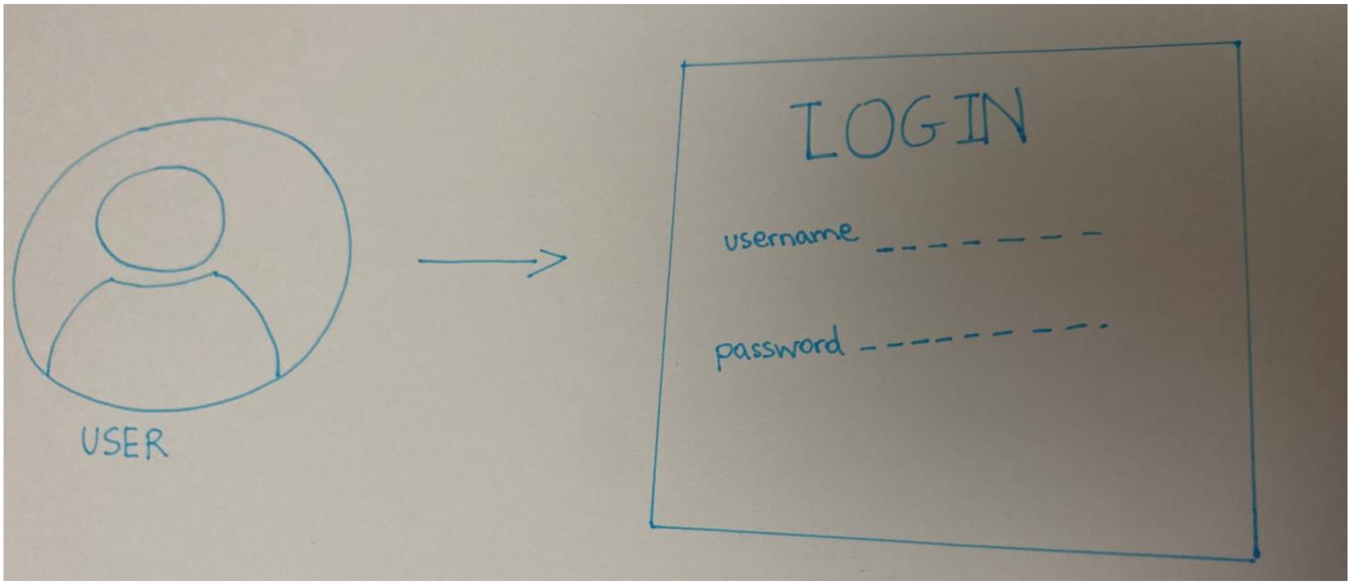
Period Highlight: 13 ▨ Plan Duration ■ % Complete

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Feasibility Study	0	3	1	2	100%
Requirements Analysis	1	5	1	5	20%
Design UML	3	4	3	4	100%
Design Sketches	2	6	2	6	40%
Developing Database	7	1	7	1	100%
Coding Backend	8	3	8	3	100%
Coding Frontend	7	4	7	4	80%
Testing and Integration	11	2	11	2	100%
Maintanance	13	1	13	1	100%

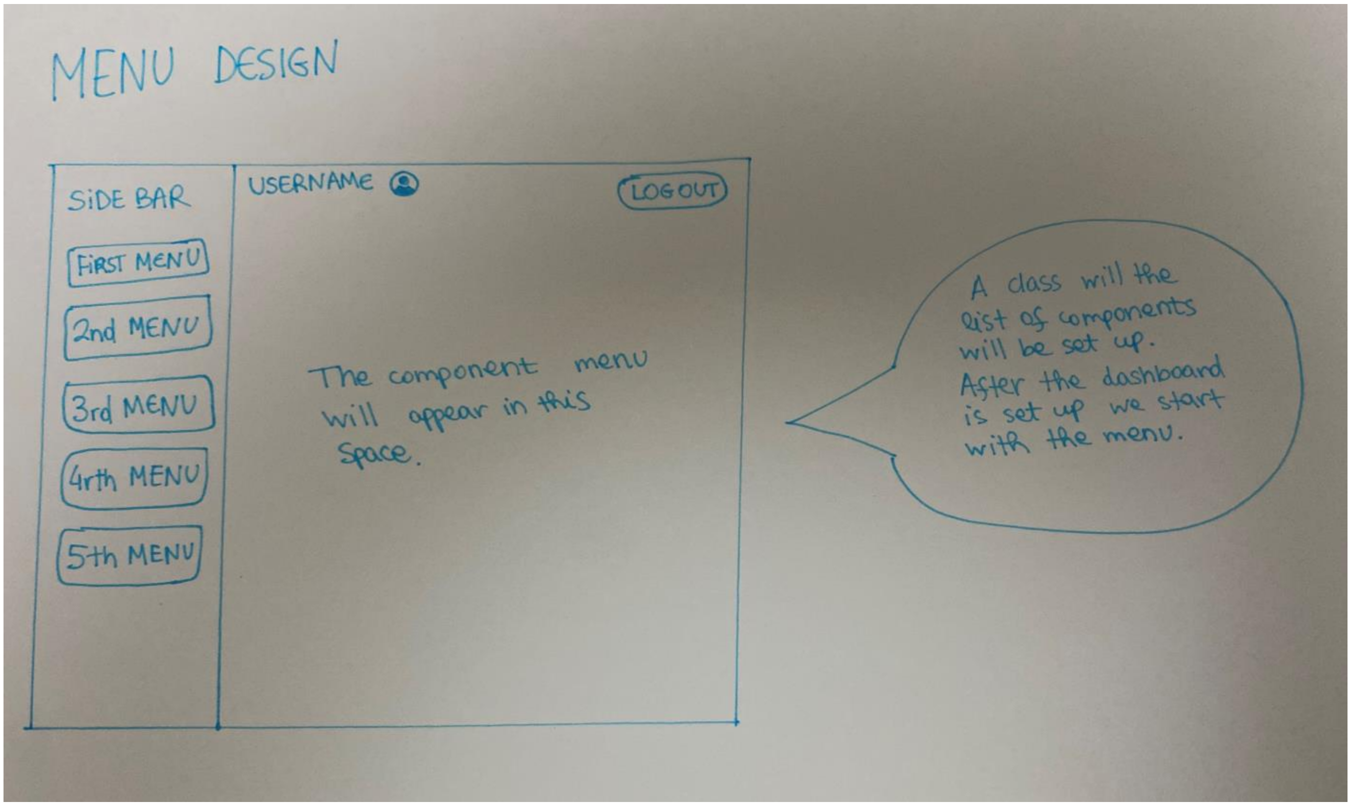
Appendix A. Sketches


After you successsfully log in
UI
UI will be the same for each role, but the sidebar menu will change depending on the role.

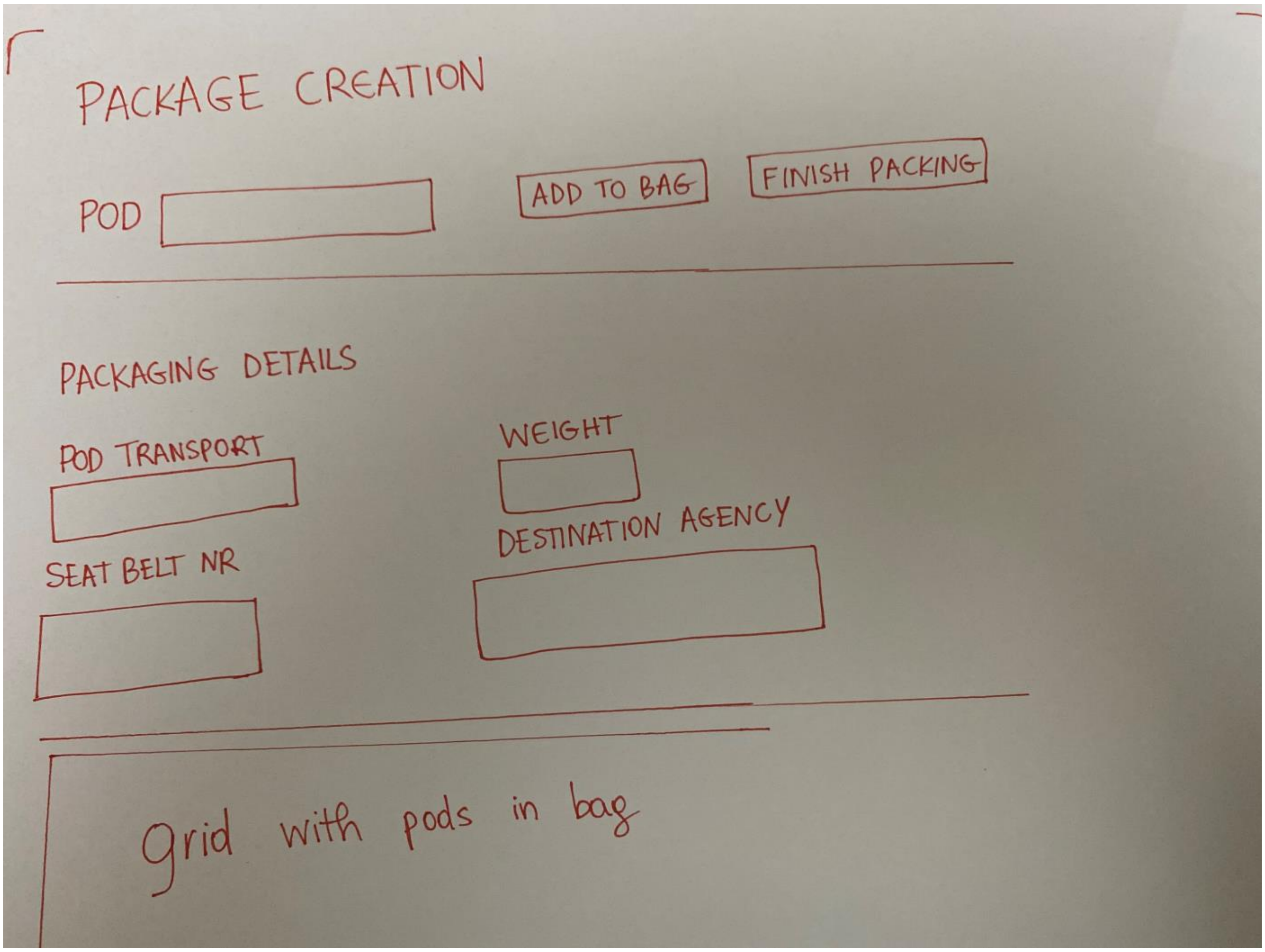
User Interface Sketch



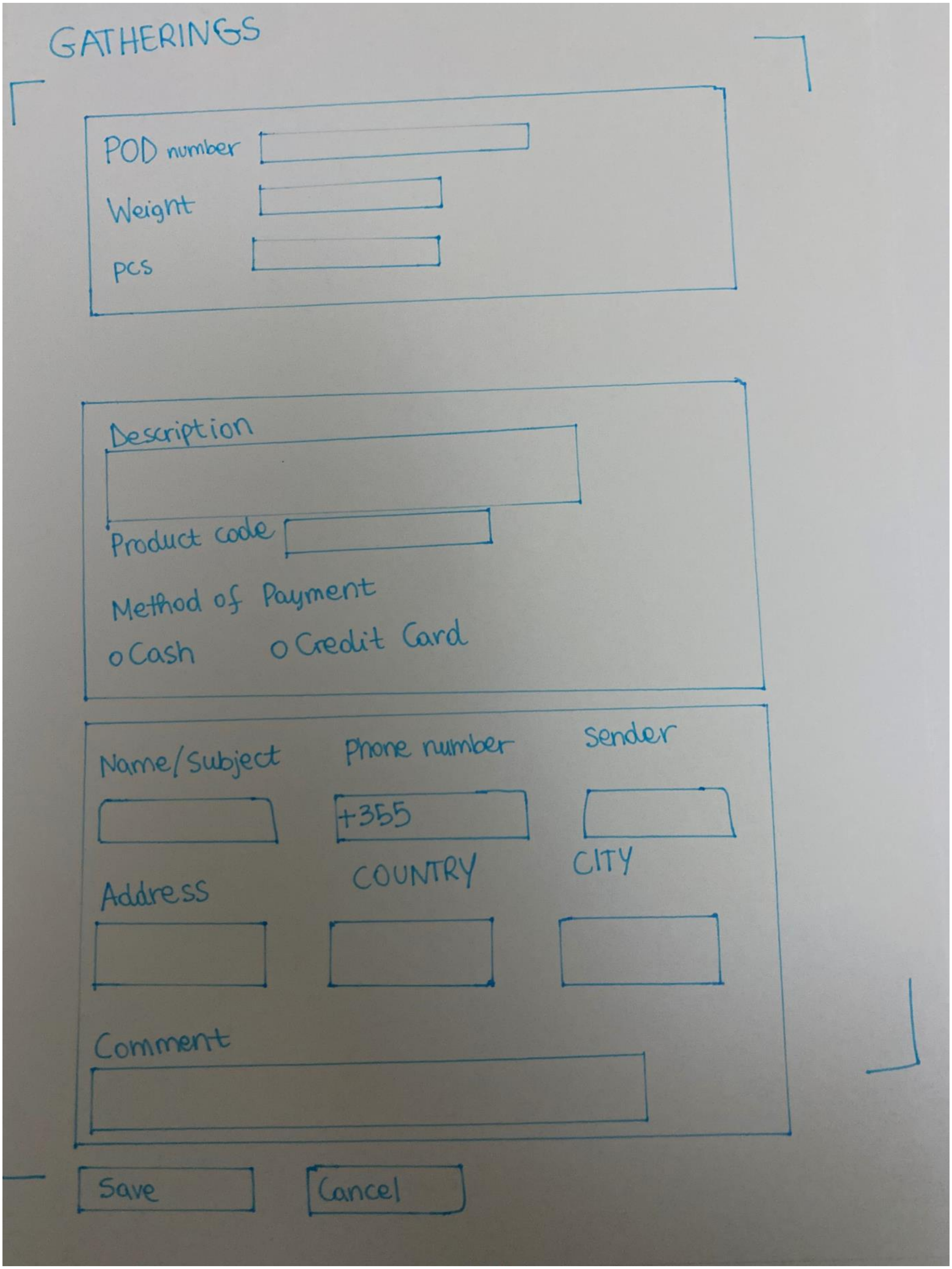
User Log In Sketch


MENU DESIGN
SIDE BAR
FIRST MENU
2nd MENU
3rd MENU
4rth MENU
5th MENU
USERNAME
LOG OUT
The component menu will appear in this Space.
A class will the list of components will be set up. After the dashboard is set up we start with the menu.

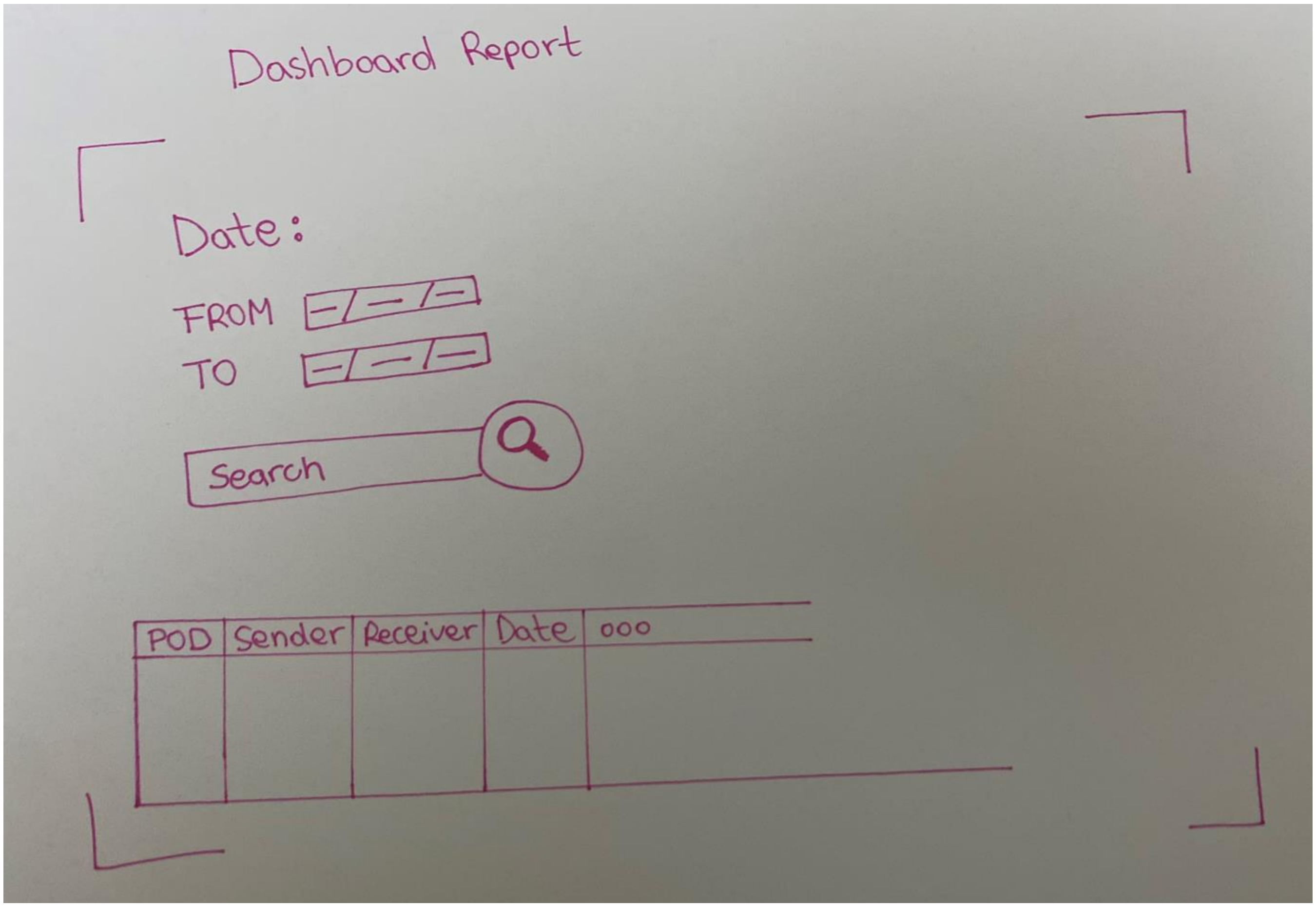
Menu Design Sketch



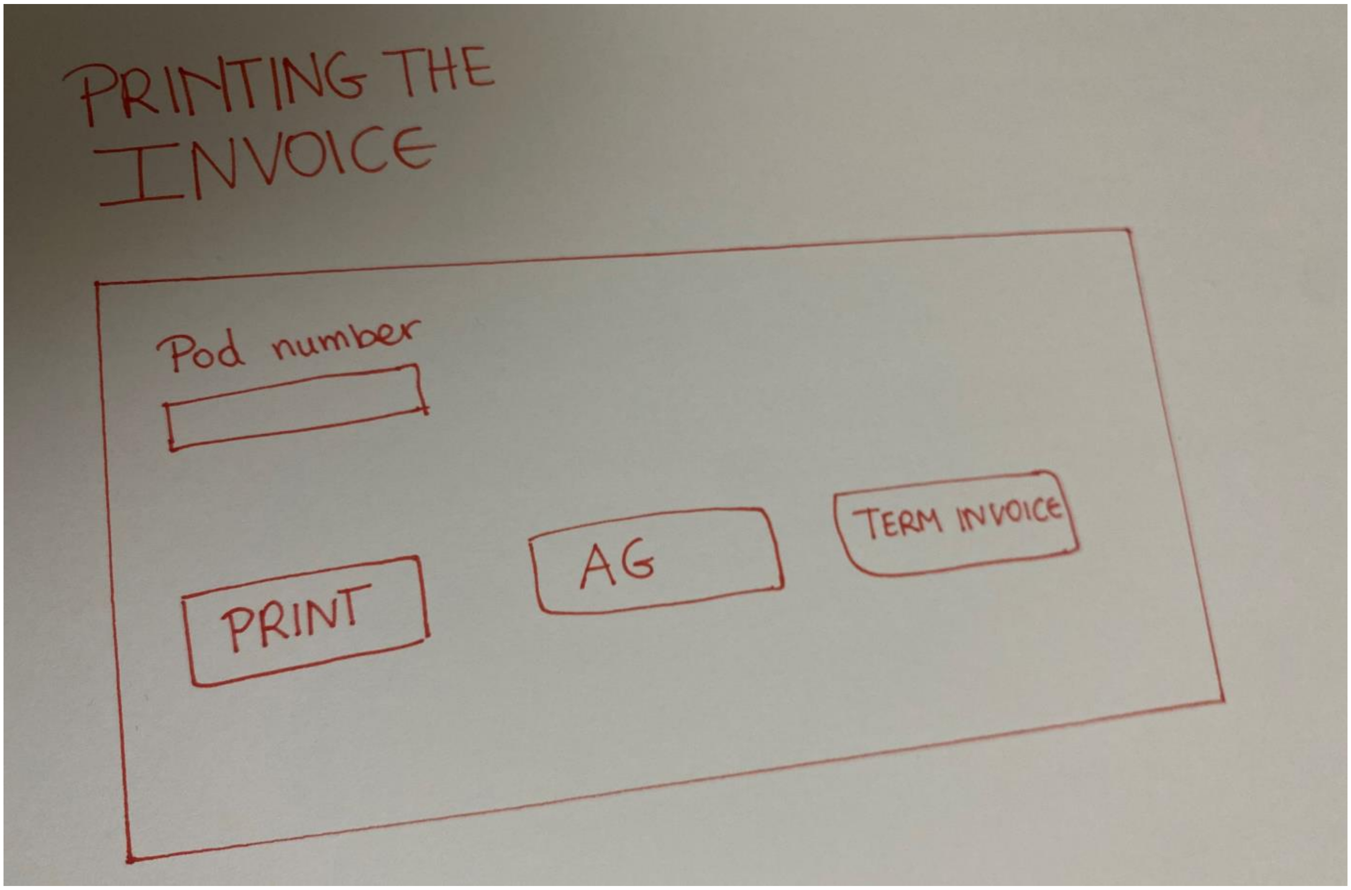
Package Creation Sketch



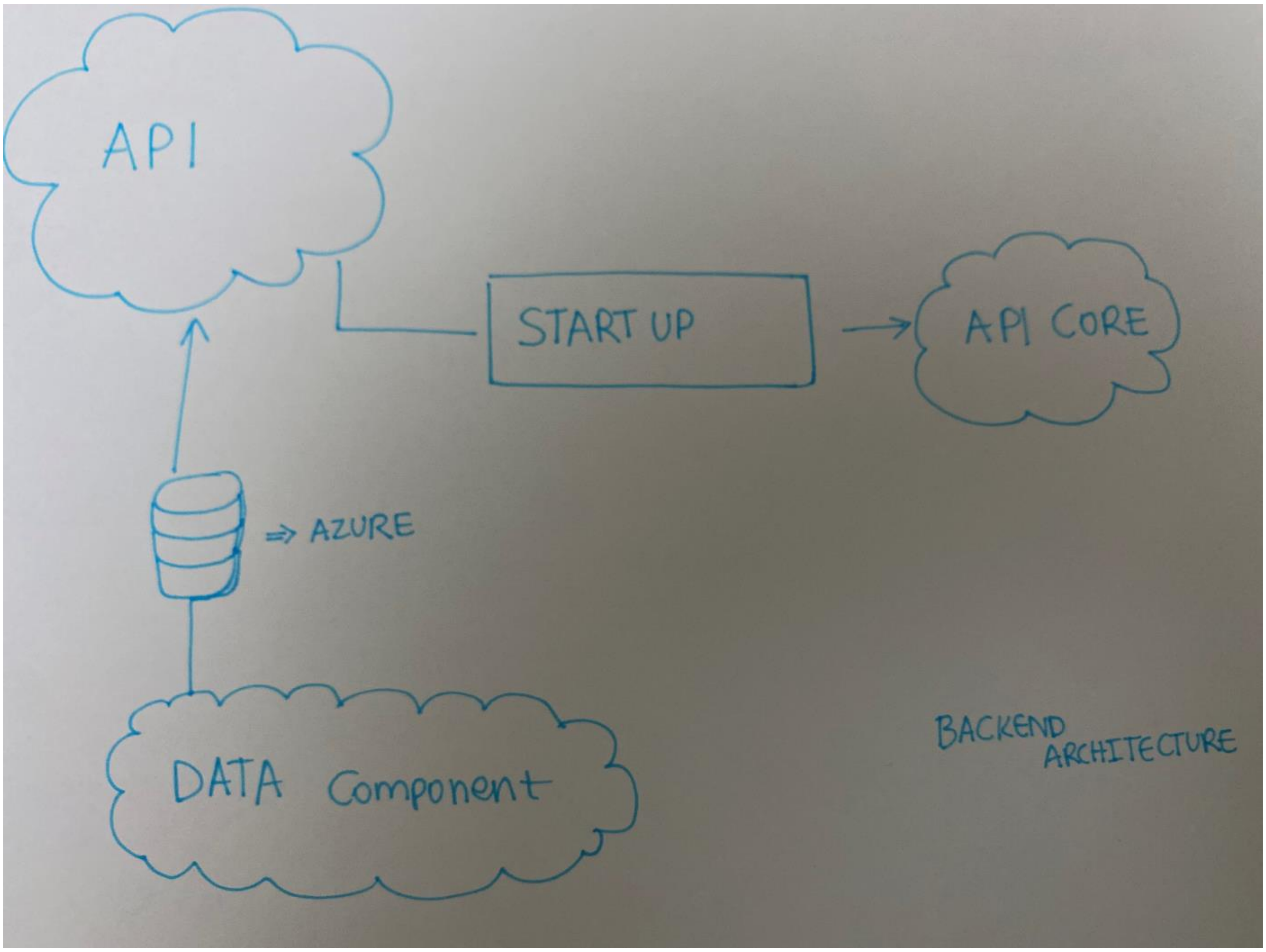
Collection Details Implementation Sketch



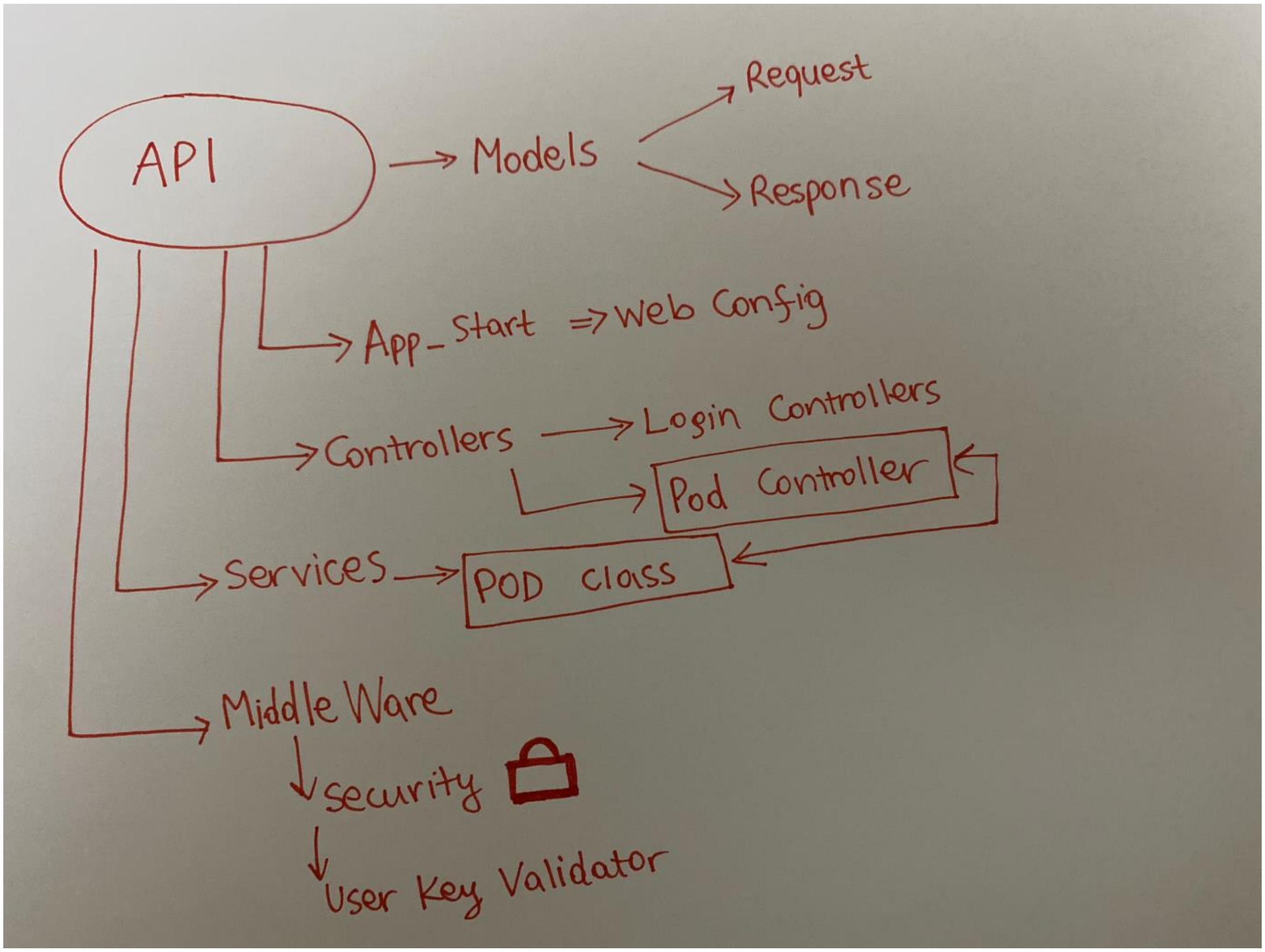
Dashboard Report Sketch



Invoice Printing Sketch

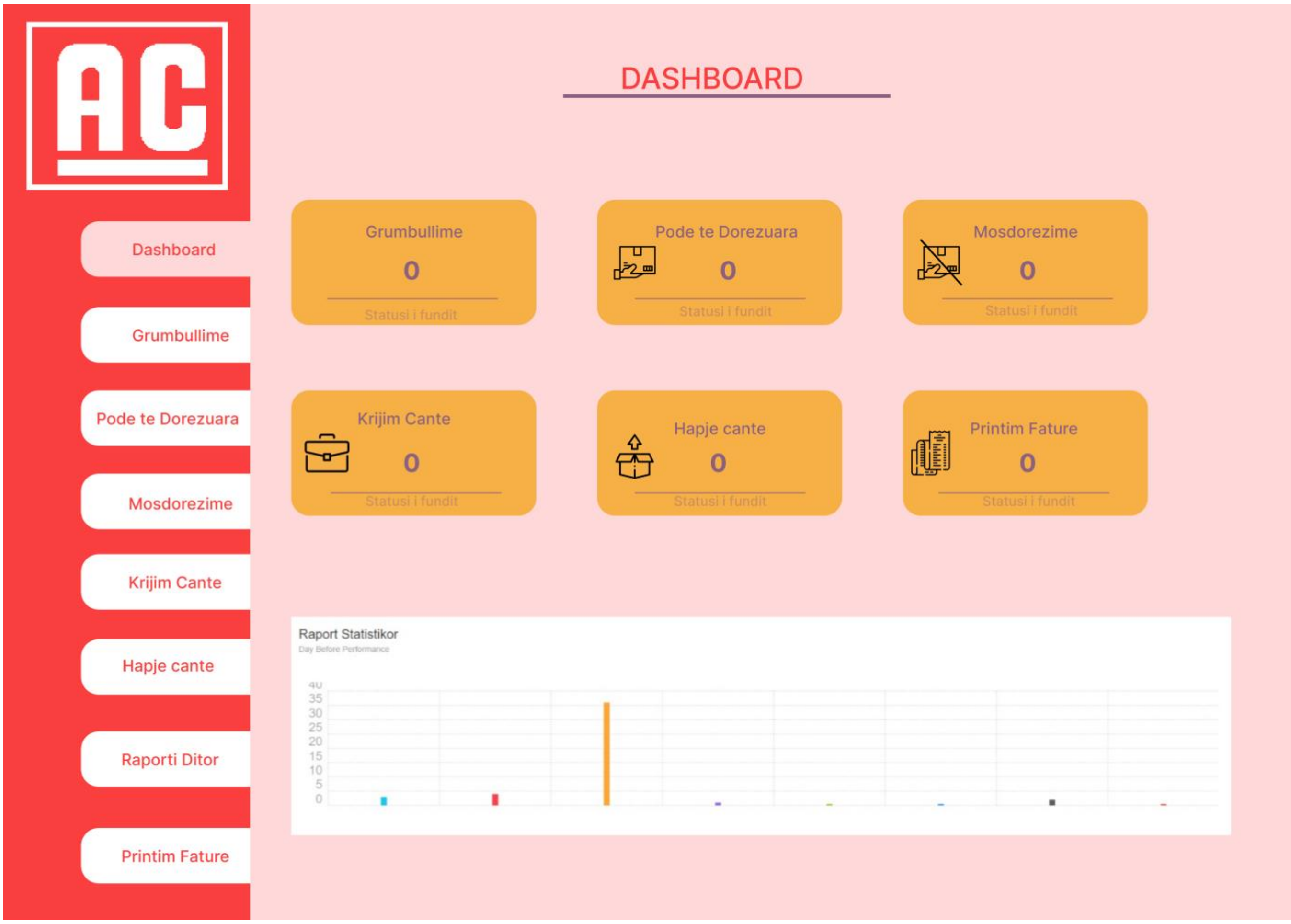


Backend Architecture Sketch 1

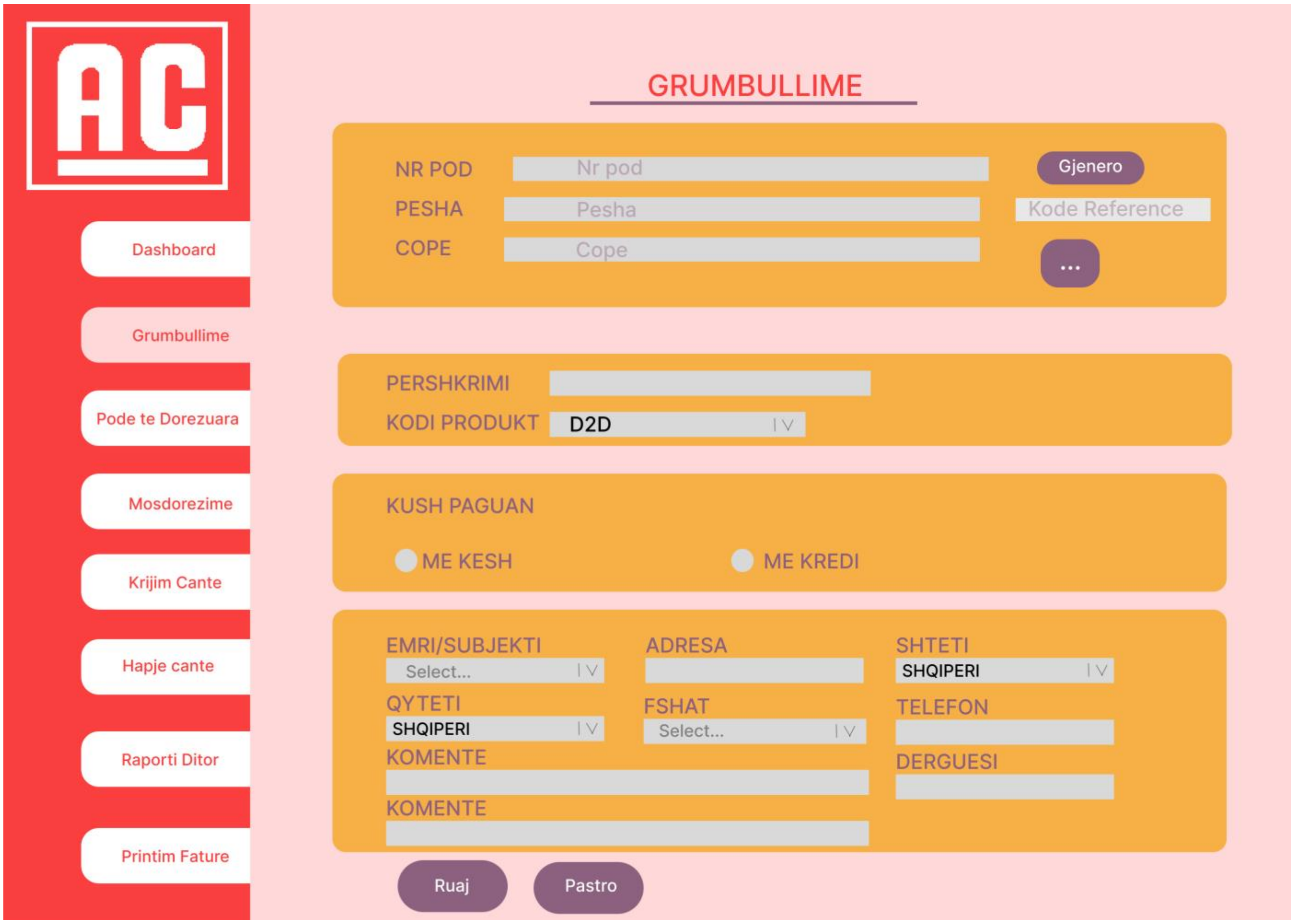


Backend Architecture Sketch 2

Appendix B. Prototype



Dashboard Design Prototype



GRUMBULLIME

NR POD Nr pod

PESHA Pesha

COPE Cope

Kode Reference

PERSHKRIMI

KODI PRODUKT D2D

KUSH PAGUAN

○ ME KESH ○ ME KREDI

EMRI/SUBJEKTI Select...

ADRESA

SHTETI SHQIPERI

QYTETI SHQIPERI

FSHAT Select...

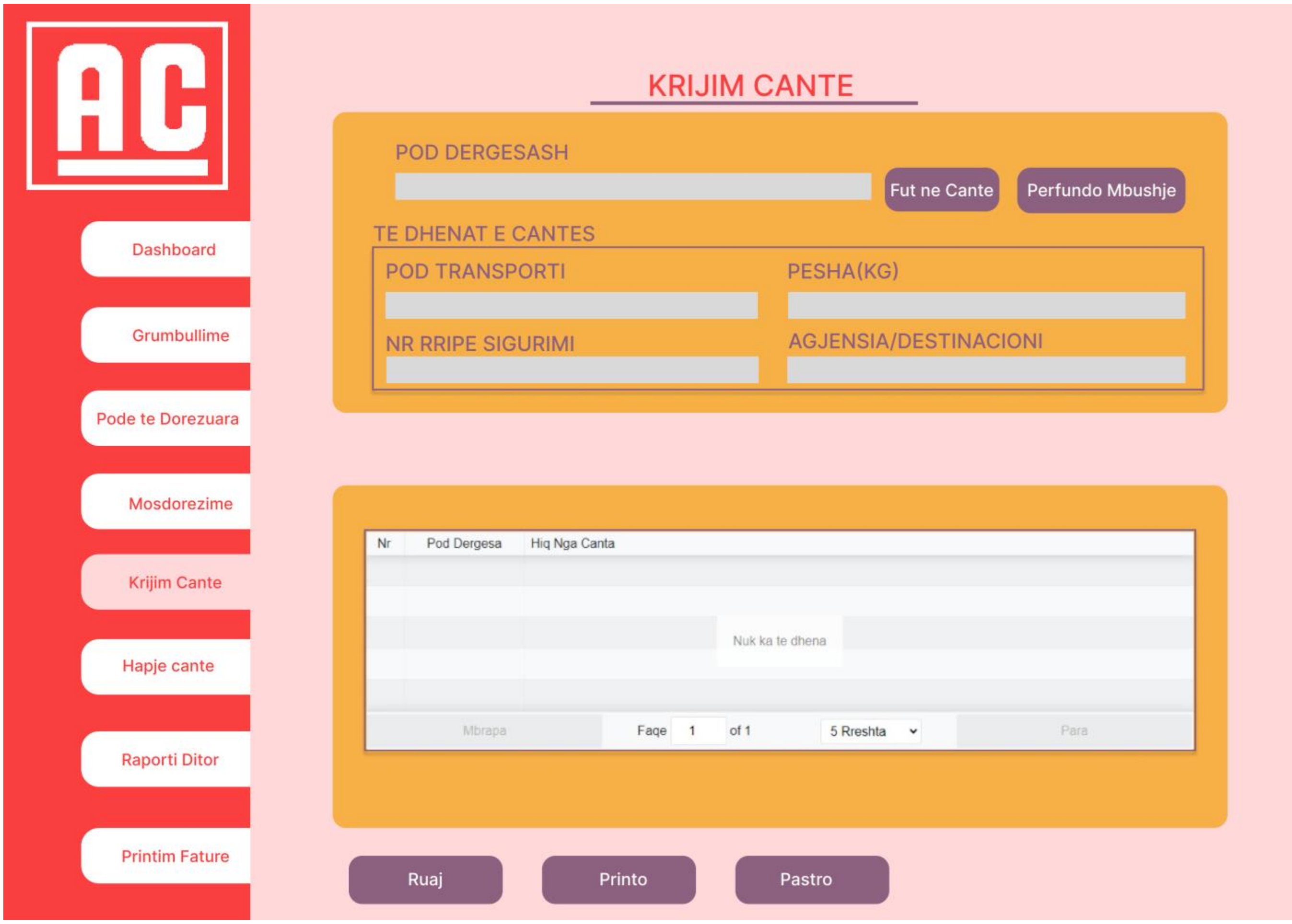
TELEFON

KOMENTE

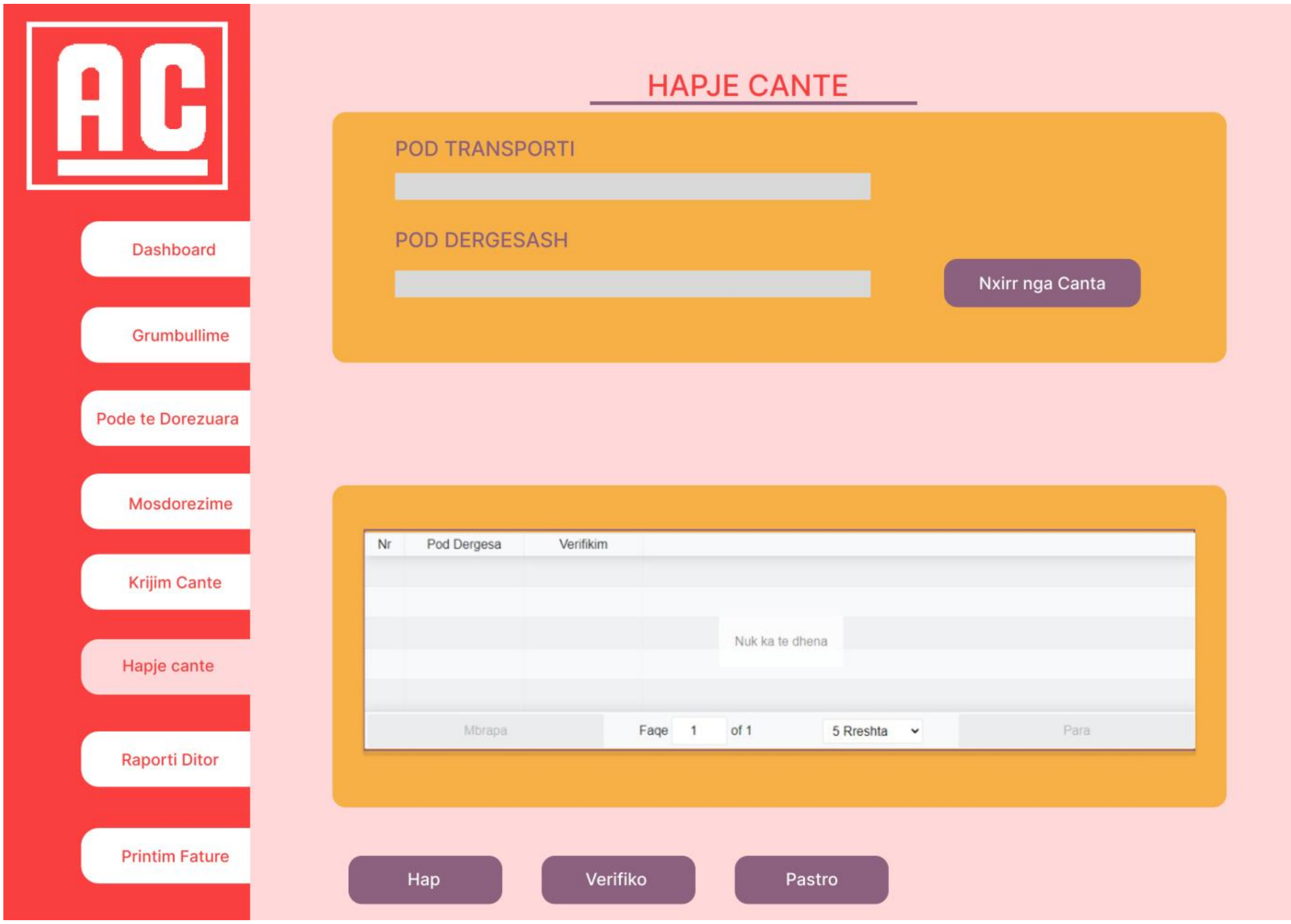
DERGUESI

KOMENTE

Grumbullime Design Prototype



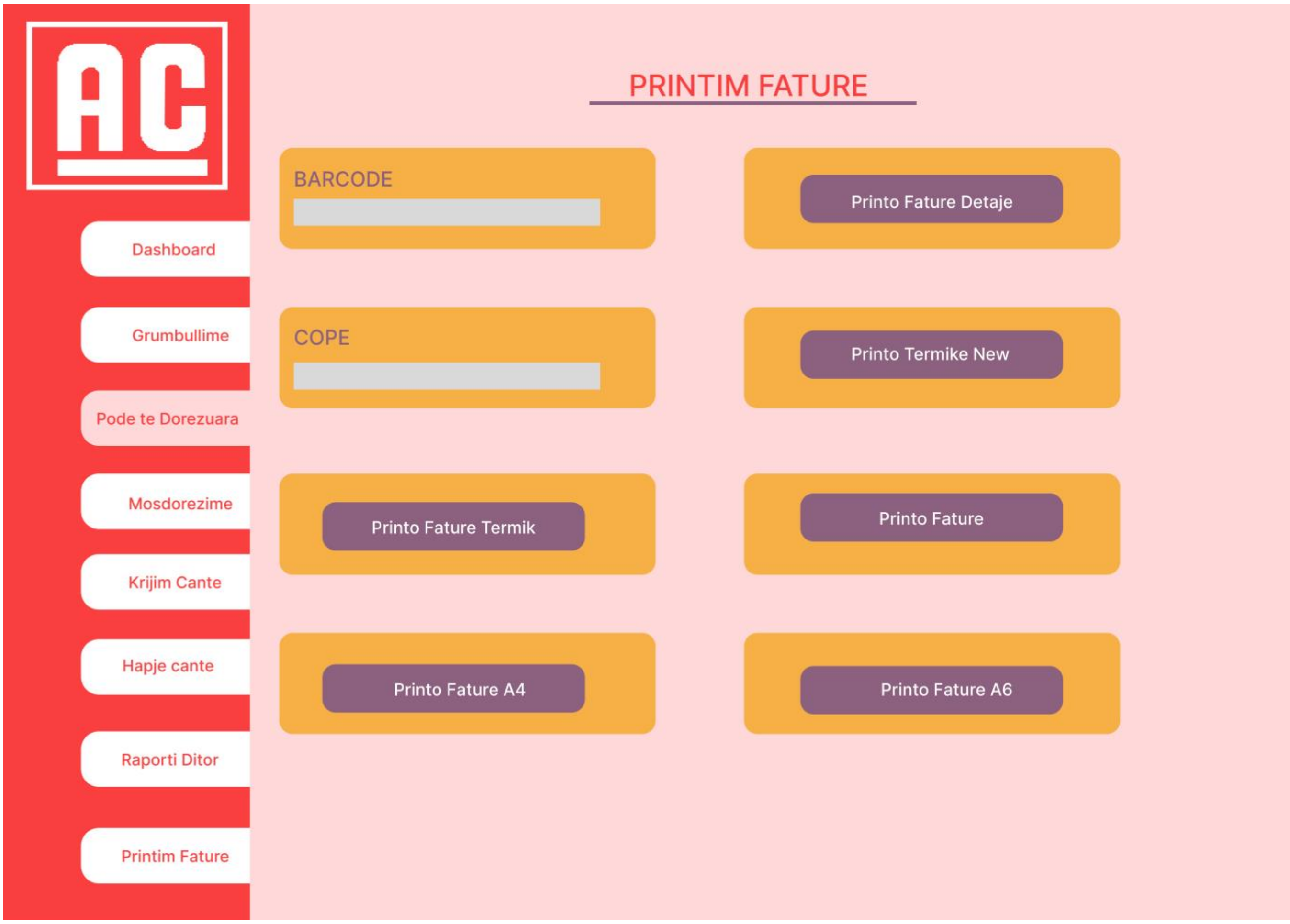
Bag Creation Design Prototype



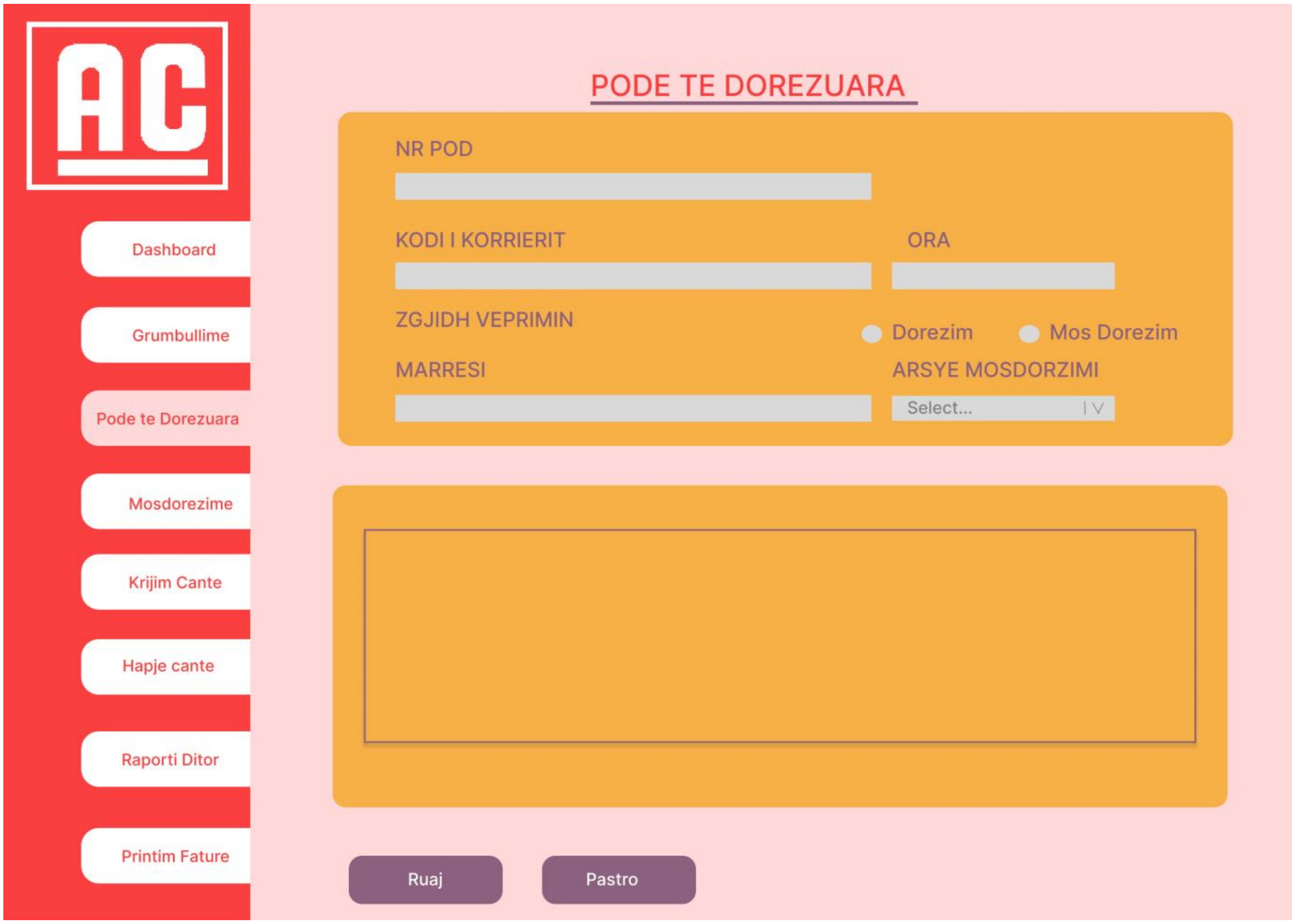
Nr	Pod Dergesa	Verifikim

Nuk ka te dhena

Bag Opening Design Prototype



Bill Printing Design Prototype



Delivered Orders Design Prototype

Appendix C. Pseudocode

a. Fetches a post method from the frontend that sends parameters as request to validate the call and receive a response. The token and username are always present for the security validation check.

```
fetch(window.UserP.url + "POD/GetKodLevizjeBrand", {
            method: "Post",
            headers: {
              Accept: "application/json",
              "Content-type": "application/json",
              "Access-Control-Allow-Origin": "*",
            },
            body: JSON.stringify({
              Token: HmacSHA256(
                Math.round(new Date().getTime() /
1000).toString(),
                window.UserP.key
              ).toString(),
              username: this.state.username,
              KodLevizje: data[1],

            }),
          })
            .then((res) => res.json())
            .then((data) => {
            this.setState({
              dataKodLevizjeBrand: data ,
              kodiLevizjesBrandNew:{label:data[0],value:data[0]}
              });
            });
```

b. Controller captures post method from the frontend. Gets the request as json from the frontend. Then it processes the information and sends back the response to the frontend.

```
  [HttpPost]
        [Route("api/POD/GetKodLevizjeDrejtim")]
        public List<string> GetKodLevizjeDrejtim([FromBody]
PodSaveReq param)
        {
            List<string> ret = null;

            try
            {

                ret = pod.GetKodLevizjeDrejtim(param.KodLevizje);

            }
            catch (Exception ex)
            {
                throw ex;
            }
            return ret;
        }
```

c. Functions for selecting information from the database and updating it.

```
public PodSaveRes NdryshoPassword(NdryshoPassReq pReq)
        {
            PodSaveRes res = new PodSaveRes();
            var check = (from p in dbContext.PERDORUES
                        where (p.USERNAME == pReq.username)
                        select p.PASSWORD).FirstOrDefault();

            if (check.ToString() != pReq.Tag)
            {
                res.Result = false;
                res.ResultDescription = "Ka ndodhur nje gabim. 
Fjalekalimi nuk eshte ndryshuar!";
            }
            else
            {

                var recordToUpdate = (from p in 
dbContext.PERDORUES
                                      where (p.USERNAME == 
pReq.username)
                                      select p).Single();
                recordToUpdate.PASSWORD = pReq.Password;
                dbContext.SaveChanges();

                res.Result = true;
                res.ResultDescription = "Fjalëkalimi u modifikua 
me sukses.";
            }

            return res;
        }
```

d. Functions for selecting the output of the store procedure as response:

```
ObjectParameter objData1 = new ObjectParameter("pData",
typeof(DateTime));
            ObjectParameter objLastId1 = new
ObjectParameter("pLastId", typeof(string));
            ObjectParameter objParamMsg1 = new
ObjectParameter("pMessage", typeof(string));
            ObjectParameter objParamMsgType1 = new
ObjectParameter("pMessageType", typeof(string));

            using (var transaction =
dbContext.Database.BeginTransaction())
            {

                try

                {
                    var result = dbContext.PROC_KRIJO_CANTE(

pReq.CantaKodi,

pReq.AgjensiaDestinacion,

pReq.AgjensiaDestinacion,

pReq.Pesha,

pReq.PerdoruesId,

pReq.NrRripSigurimi, //pReq.StatusFizikPod,

"JOANULLUAR",

"0004",

"",

objData,

objLastId,

objParamMsg,

objParamMsgType


);
```

```
                    if (result == 0 || result == null)
                    {
                        res.Result = false;
                        res.ResultMessage = "Ka ndodhur nje gabim.
Fjalekalimi nuk eshte ndryshuar!" + objParamMsg.Value.ToString();
```