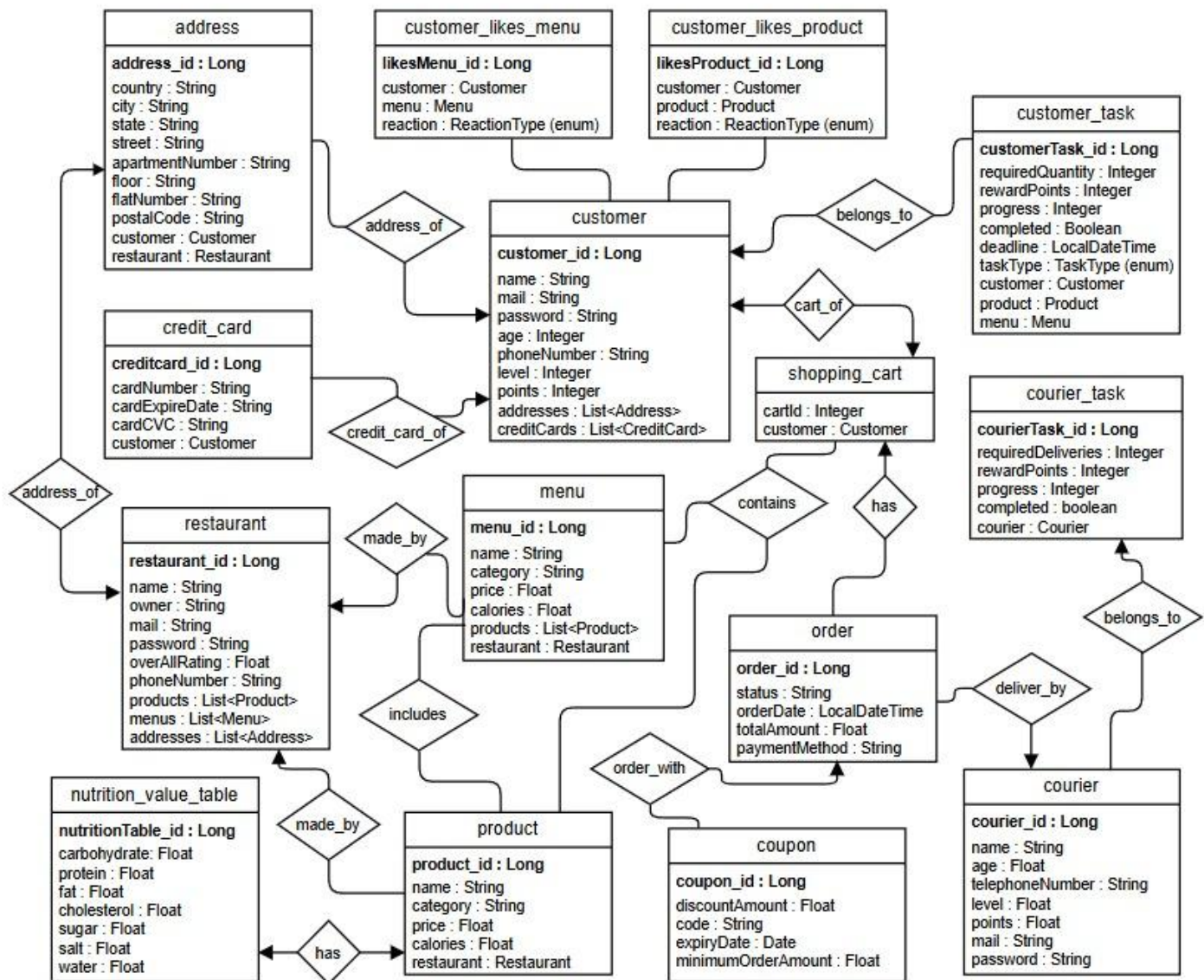<<HUFDS>>
Design Diagram Document

1 Introduction

This document contains some diagrams for tracking the progress of the project. These diagrams are updated ER Diagram, Updated Class Diagram, State Chart Diagram for Usecase: Menu Management, and Sequence Diagrams for critical usecases.
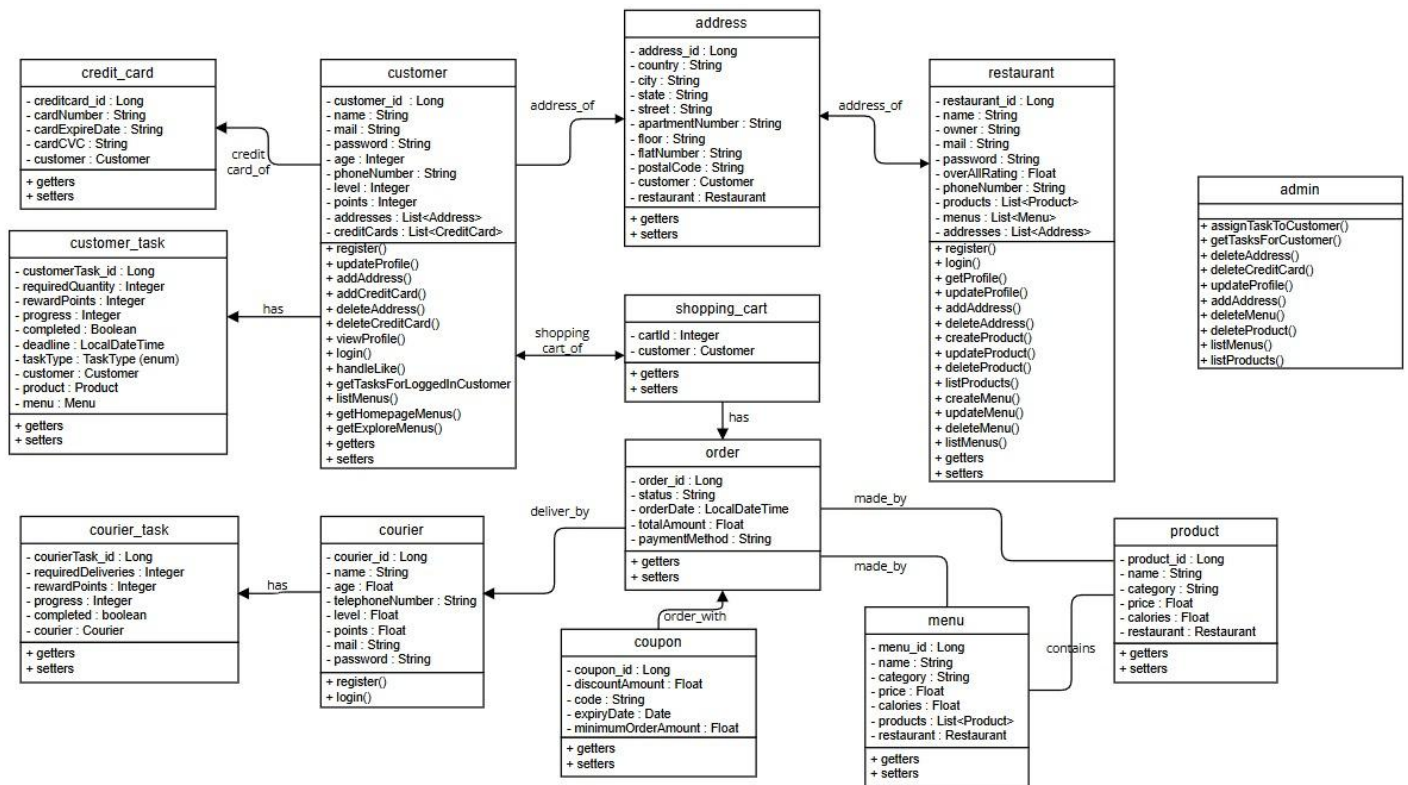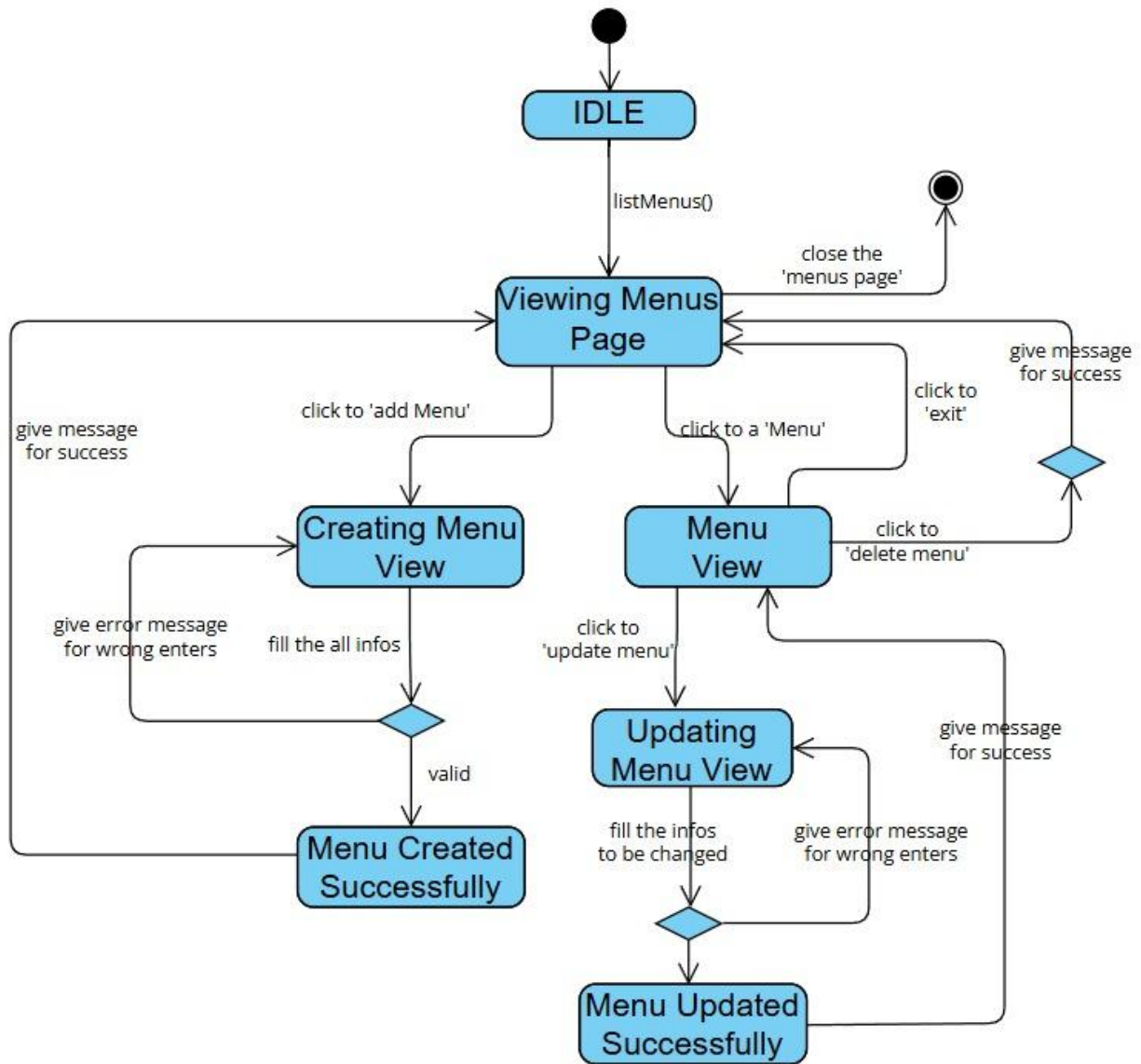
2 Diagrams

- **Updated ER Diagram:**

- **Updated Class Diagram:**

credit_card
- creditcard_id : Long
- cardNumber : String
- cardExpireDate : String
- cardCVC : String
- customer : Customer

+ getters
+ setters

customer
- customer_id : Long
- name : String
- mail : String
- password : String
- age : Integer
- phoneNumber : String
- level : Integer
- points : Integer
- addresses : List<Address>
- creditCards : List<CreditCard>

+ register()
+ updateProfile()
+ addAddress()
+ addCreditCard()
+ deleteAddress()
+ deleteCreditCard()
+ viewProfile()
+ login()
+ handleLike()
+ getTasksForLoggedInCustomer()
+ listMenus()
+ getHomepageMenus()
+ getExploreMenus()
+ getters
+ setters

address
- address_id : Long
- country : String
- city : String
- state : String
- street : String
- apartmentNumber : String
- floor : String
- flatNumber : String
- postalCode : String
- customer : Customer
- restaurant : Restaurant

+ getters
+ setters

restaurant
- restaurant_id : Long
- name : String
- owner : String
- mail : String
- password : String
- overAllRating : Float
- phoneNumber : String
- products : List<Product>
- menus : List<Menu>
- addresses : List<Address>

+ register()
+ login()
+ getProfile()
+ updateProfile()
+ addAddress()
+ deleteAddress()
+ createProduct()
+ updateProduct()
+ deleteProduct()
+ listProducts()
+ createMenu()
+ updateMenu()
+ deleteMenu()
+ listMenus()
+ getters
+ setters

admin
+ assignTaskToCustomer()
+ getTasksForCustomer()
+ deleteAddress()
+ deleteCreditCard()
+ updateProfile()
+ addAddress()
+ deleteMenu()
+ deleteProduct()
+ listMenus()
+ listProducts()

customer_task
- customerTask_id : Long
- requiredQuantity : Integer
- rewardPoints : Integer
- progress : Integer
- completed : Boolean
- deadline : LocalDateTime
- taskType : TaskType (enum)
- customer : Customer
- product : Product
- menu : Menu

+ getters
+ setters

shopping_cart
- cartId : Integer
- customer : Customer

+ getters
+ setters

order
- order_id : Long
- status : String
- orderDate : LocalDateTime
- totalAmount : Float
- paymentMethod : String

+ getters
+ setters

product
- product_id : Long
- name : String
- category : String
- price : Float
- calories : Float
- restaurant : Restaurant

+ getters
+ setters

courier_task
- courierTask_id : Long
- requiredDeliveries : Integer
- rewardPoints : Integer
- progress : Integer
- completed : boolean
- courier : Courier

+ getters
+ setters

courier
- courier_id : Long
- name : String
- age : Float
- telephoneNumber : String
- level : Float
- points : Float
- mail : String
- password : String

+ register()
+ login()

coupon
- coupon_id : Long
- discountAmount : Float
- code : String
- expiryDate : Date
- minimumOrderAmount : Float

+ getters
+ setters

menu
- menu_id : Long
- name : String
- category : String
- price : Float
- calories : Float
- products : List<Product>
- restaurant : Restaurant

+ getters
+ setters

Relationships:
address_of, credit card_of, address_of, has, shopping cart_of, deliver_by, has, order_with, made_by, made_by, has, contains

- **State Chart – Usecase: Management Menu**

- ## **Sequence Diagrams**

Login

Search and Browse

sd [search & browse]


```
: Scroll View        : CustomerHomeController        ;Products

Customer

1: Enter product keys

2: Enter product keys

2.1: getProducts(keys): Vector

2.1.1: new Vector<Product>

2.1.2: return(products)

2.1.2.1: viewProducts(Vector)

if product couldn't found:

Error message is
displayed:Product not found!
```
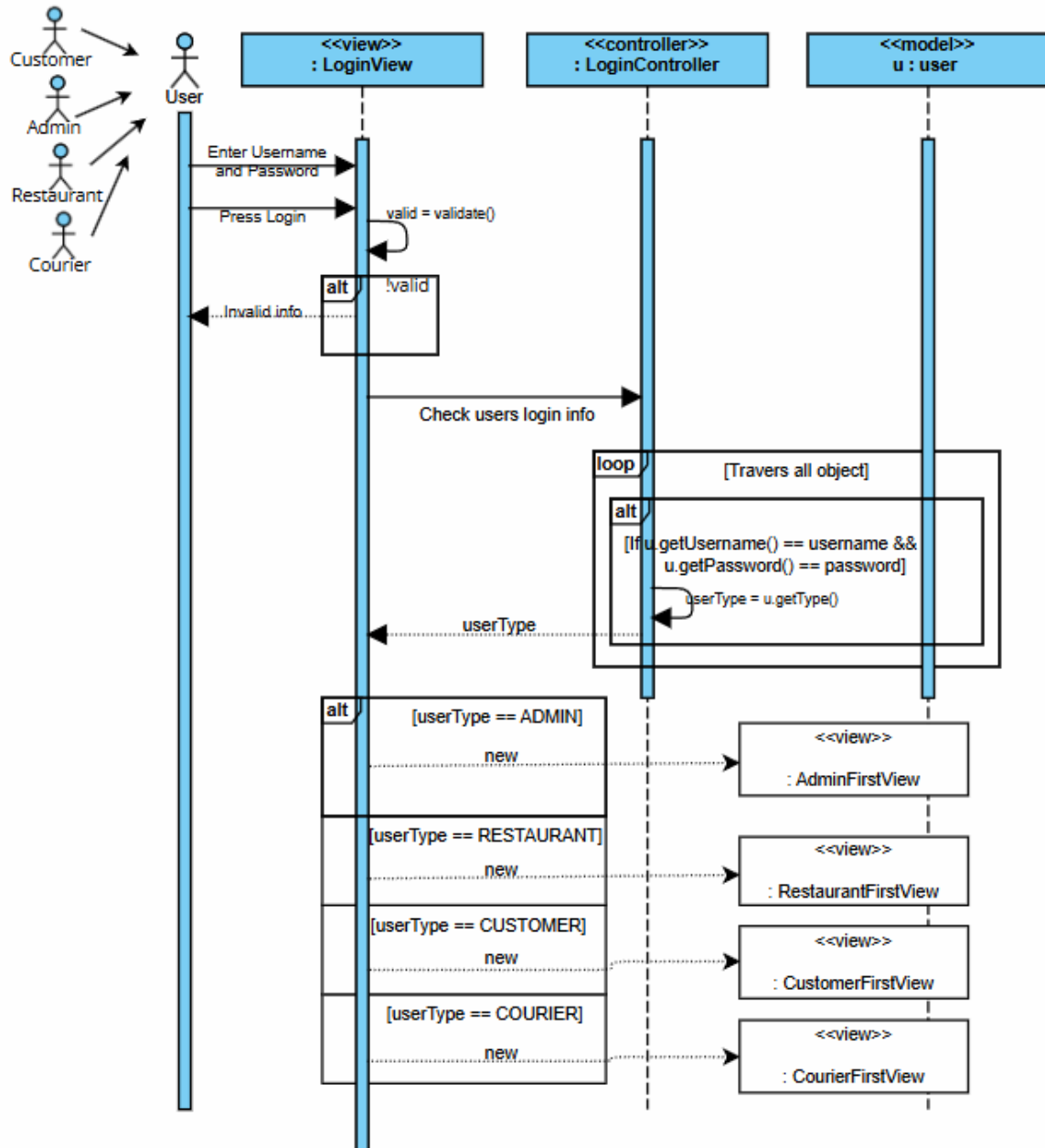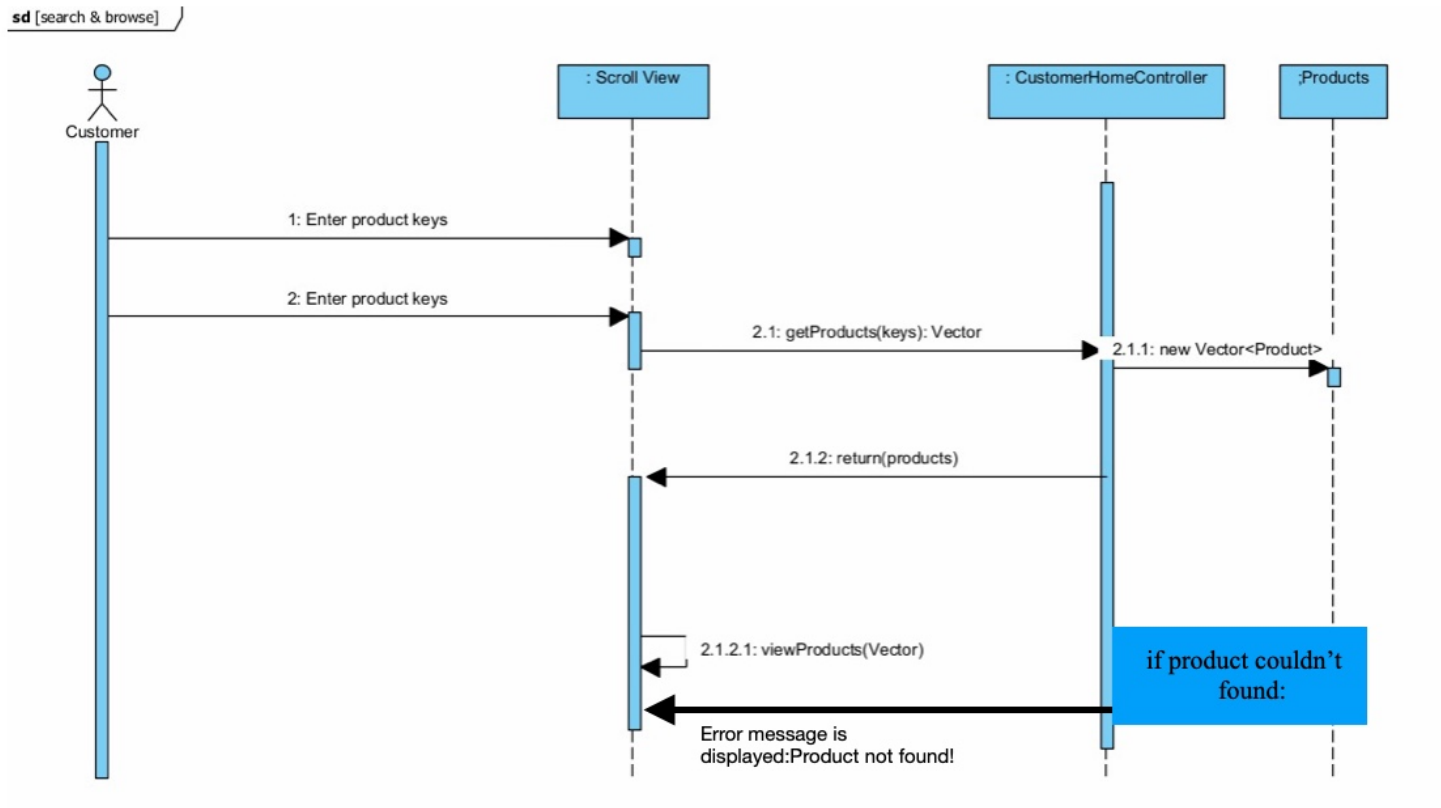
Menu Management

HUFDS	
Design Diagram Document	Date: 04/05/2025
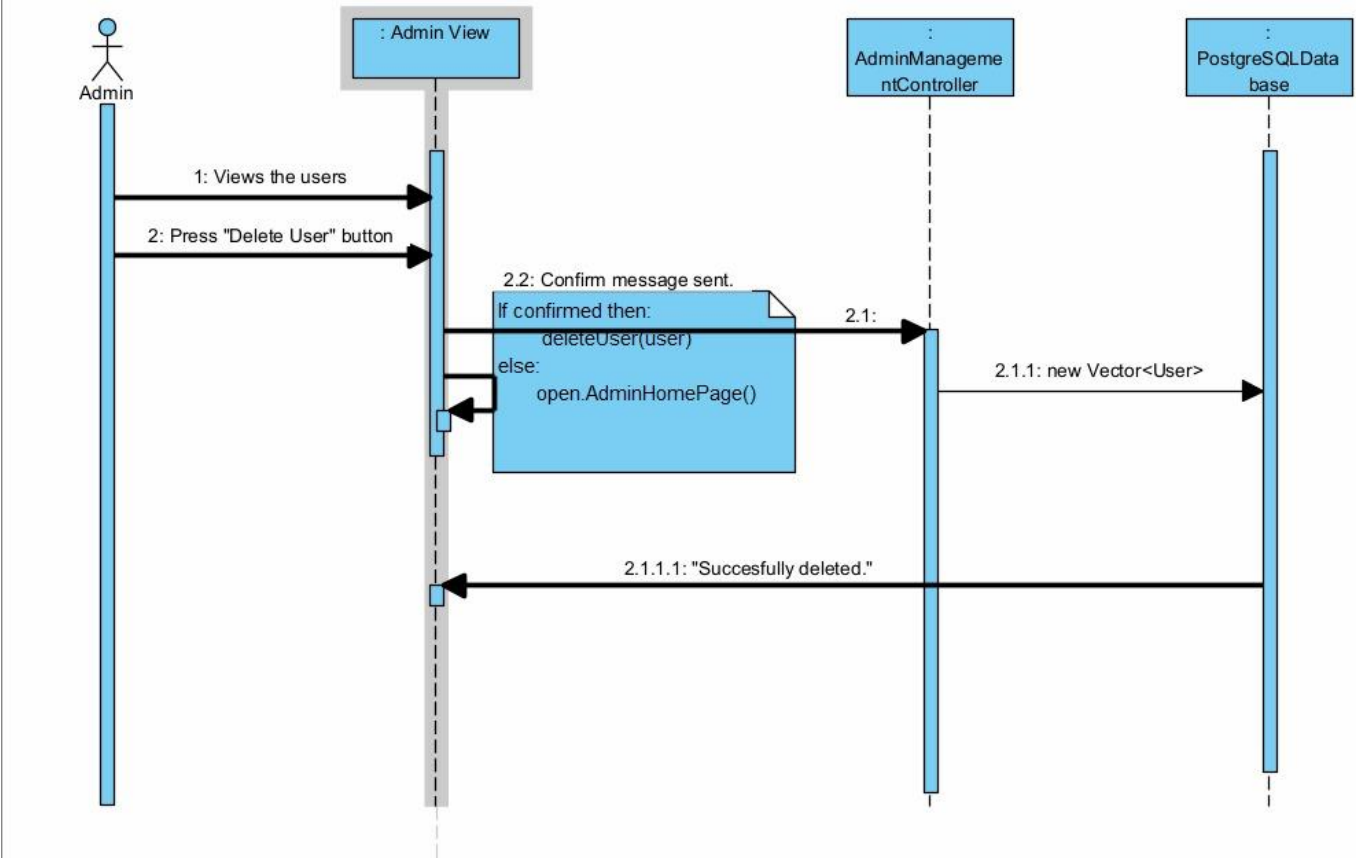
sd [Menu Management]

Admin Management System

Assign Derby Tasks

HUFDS	
Design Diagram Document	Date: 04/05/2025

sd [Assign derby task]
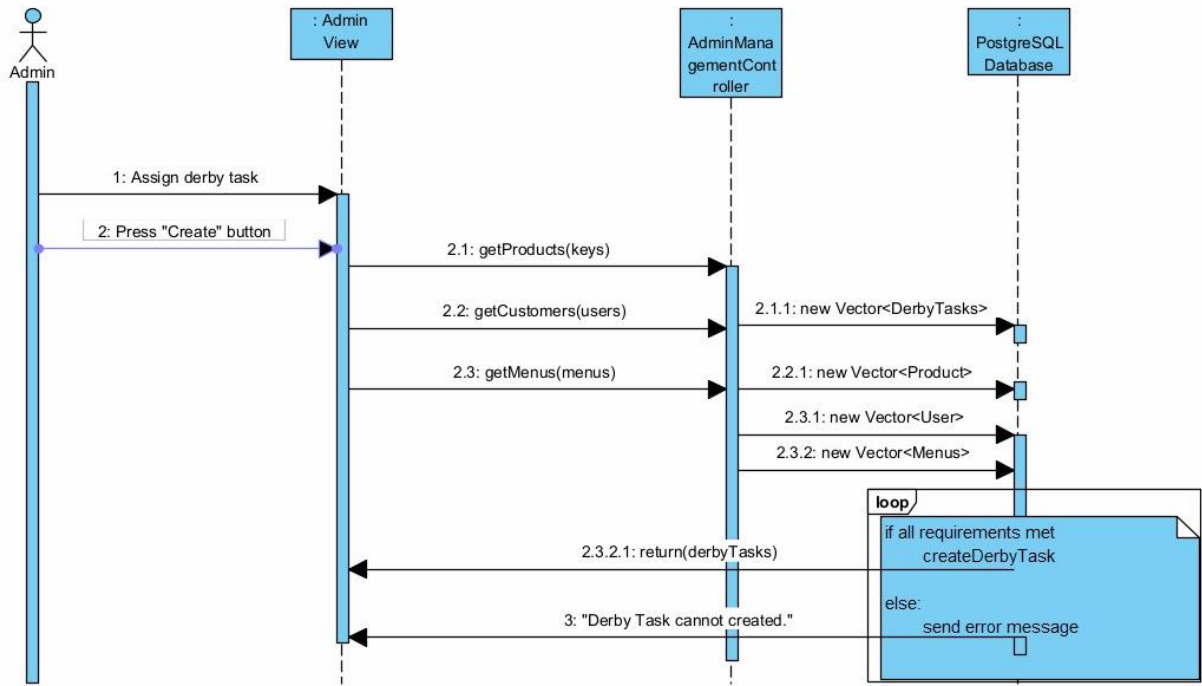
HUFDS	
Design Diagram Document	Date: 04/05/2025

Update Menu

sd [update menu]

Restaurant Owner

: Menu View

: MenuContr oller

: PostgreSQL Database

1: Update menu information

2: Press "Edit" button

2.1: updateMenu(calorie, price, name, product)

2.1.1: new Vector<Menus>

alt

if no change:
    send error message
else if the change is not in the
correct format:
    send error message
else:
    return(updatedMenu)

2.1.2: "There is no change."

2.1.3: "All parts must be filled."

2.1.1.1: return(updatedMenu)
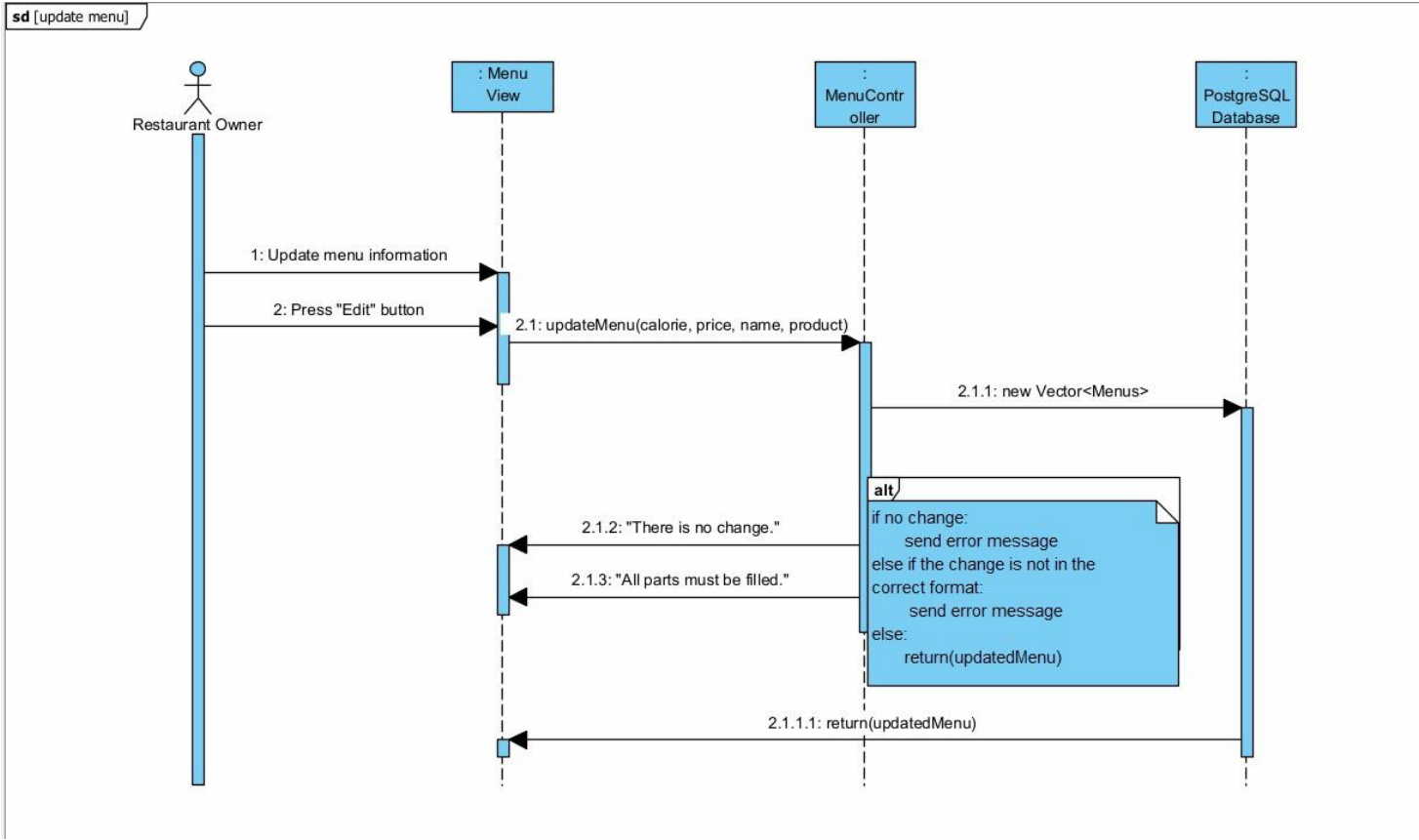
Scroll and Swipe
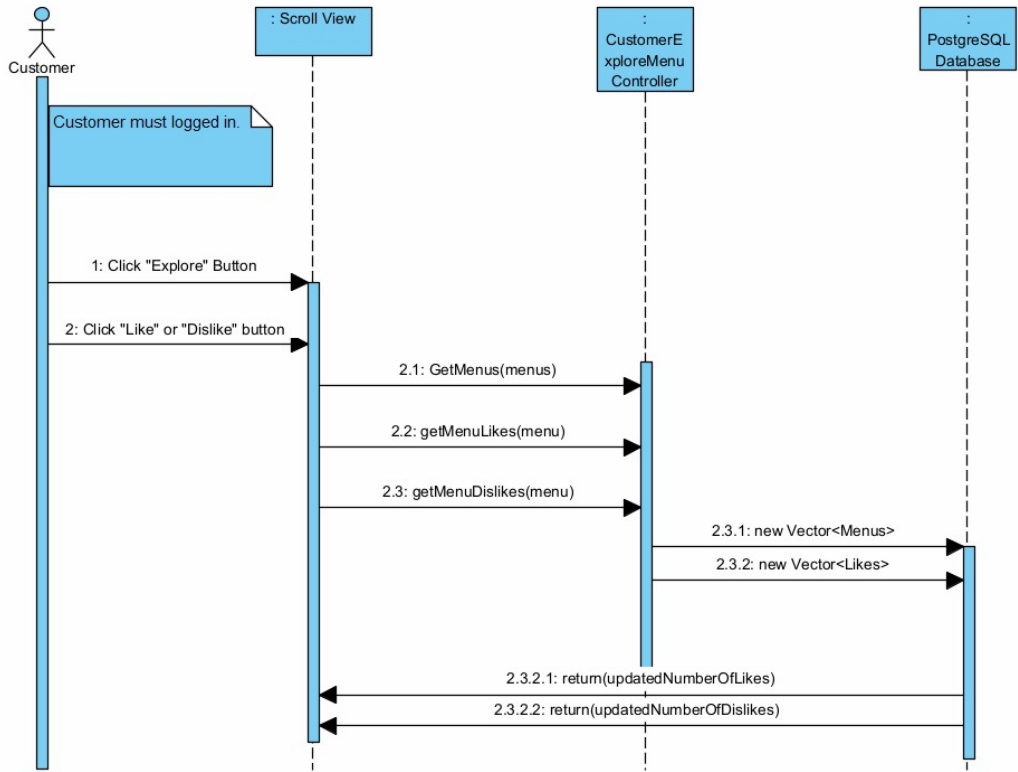
Traceability Table

	Gökdeniz Şimşek	Zeynep Nisa Karataş	Cansu Aslan	Mert Can Köseoğlu
Software Test Report		X		
Coding Standarts				X
Design Diagram Document (other than sequences)	X			
Design Diagram Document (sequences)			X	