Concordia University

SOEN 6481 - Software Design Methodologies

Deliverable 2

Requirement Analysis for Ticket Vending Machine

Supervisor
Prof. Pankaj Kamthan

Team K
Authors
Devanshi Piyushkumar Patel - 40172139
Hetul Patel - 40225667
Jay Bharatbhai Patel - 40197003
Juhi Birju Patel - 40190446
Krishna Patel - 40206701
Mahavir Patel - 40198619

GitHub Address: `https://github.com/mahavir0/iGo---SOEN-6461-SDM`

Contents

Problem 6

6.1 Description about Cards

The Ticket class represents a ticket that can be purchased from the iGO system. It sends a message to the TicketType class to determine the type of ticket that was purchased, and stores the validity period and purchase date of the ticket. It also sends a message to the Payment class to store the payment details of the ticket purchase.

The TicketType class represents the types of tickets that can be purchased from the iGO system. It stores the name, description, validity period, and fare of the ticket type. It sends a message to the Ticket class to determine which tickets have been sold and their validity period.

The Payment class represents a payment made by the user to purchase a ticket from the iGO system. It stores the amount of money paid, payment method, and payment status. It sends a message to the Ticket class to store the payment details of the ticket purchase and to the Transaction class to store the payment details of the transaction.

The User class represent different types of users of the iGO system. It stores the name, email, password, and account validity of the user. It also send a message to the Transaction class to store the user details of the transaction. It is the parent class of Student, General ans SeniorCitizen class.

The Student class represents a student user of the iGO system. It sends a message to the User class to inherit the user attributes and methods.
The General class represents a general user of the iGO system. It sends a message to the User class to inherit the user attributes and methods.
The SeniorCitizen class represents a senior citizen user of the iGO system. It sends a message to the User class to inherit the user attributes and methods.

The Ticket Vending Machine class represents the TVM itself. It stores the physical location, status, and serial number of the machine. It sends a message to the Transaction class to store the transactions made by users on the TVM, and sends a message to the User class to store the user details.

The Transaction class represents a transaction made by a user at the TVM to purchase a ticket. It stores the ticket, payment, user, date, and machine used for the transaction. It sends a message to the Ticket class to store the ticket purchased in the transaction, sends a message to the Payment class to store the payment details of the transaction, and sends a message to the Ticket Vending Machine class to store the machine used for the transaction.

6.2 CRC Card Model

Ticket	
• Represent a ticket that can be purchased from the iGO system for public transportation • Send message to TicketType to determine the type of ticket purchased • Store the validity period and purchase date of the ticket • Send message to Payment to store the payment details of the ticket purchase	• TicketType • Payment

TicketType	
• Represent the types of tickets that can be purchased from the iGO system • Store the name, description, validity period, and fare of the ticket type • Send message to Ticket to to determine which tickets have been sold and their validity period	• Ticket

Payment	
• Represent a payment made by the user to purchase a ticket from the iGO • Store the amount of money paid, payment method, and payment status • Send message to Ticket to store the payment details of the ticket purchase • Send message to Transaction to store the payment details of the transaction	• Ticket • Transaction

User	Student, General, SeniorCitizen
• Represent a user of the iGO system • Store the name, email, password, and account validity of the user • Send a message to Transaction to store the user details of the transaction • Send message to Student, General, and SeniorCitizen to represent the type of user.	• Transaction • Student • General • SeniorCitizen

Student	User
• Represent a student user of the iGO system • Send message to User to inherit user attributes and methods	• User

General	User
• Represent a general user of the iGO system • Send a message to User to inherit user attributes and methods	• User

SeniorCitizen	User
• Represent a senior citizen user of the iGO system • Send a message to User to inherit user attributes and methods	• User

User	Student, General, SeniorCitizen
• Represent a user of the iGO system • Store the name, email, password, and account validity of the user • Send a message to Transaction to store the user details of the transaction • Send message to Student, General, and SeniorCitizen to represent the type of user.	• Transaction • Student • General • SeniorCitizen

Student	User
• Represent a student user of the iGO system • Send message to User to inherit user attributes and methods	• User

General	User
• Represent a general user of the iGO system • Send a message to User to inherit user attributes and methods	• User

SeniorCitizen	User
• Represent a senior citizen user of the iGO system • Send a message to User to inherit user attributes and methods	• User

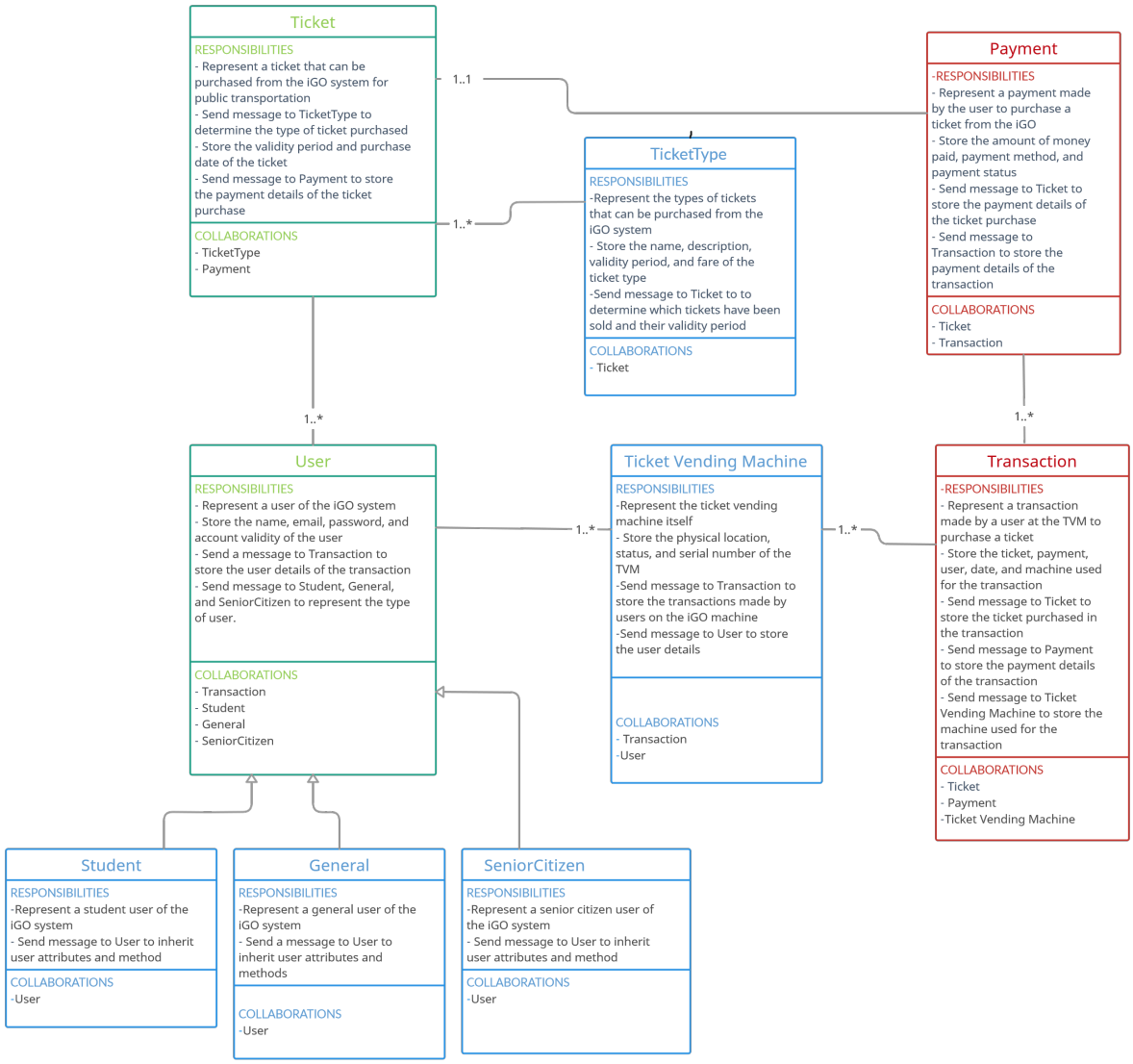


Figure 6.2.1: CRC Card Model

Problem 7

7.1 UML Class Diagram

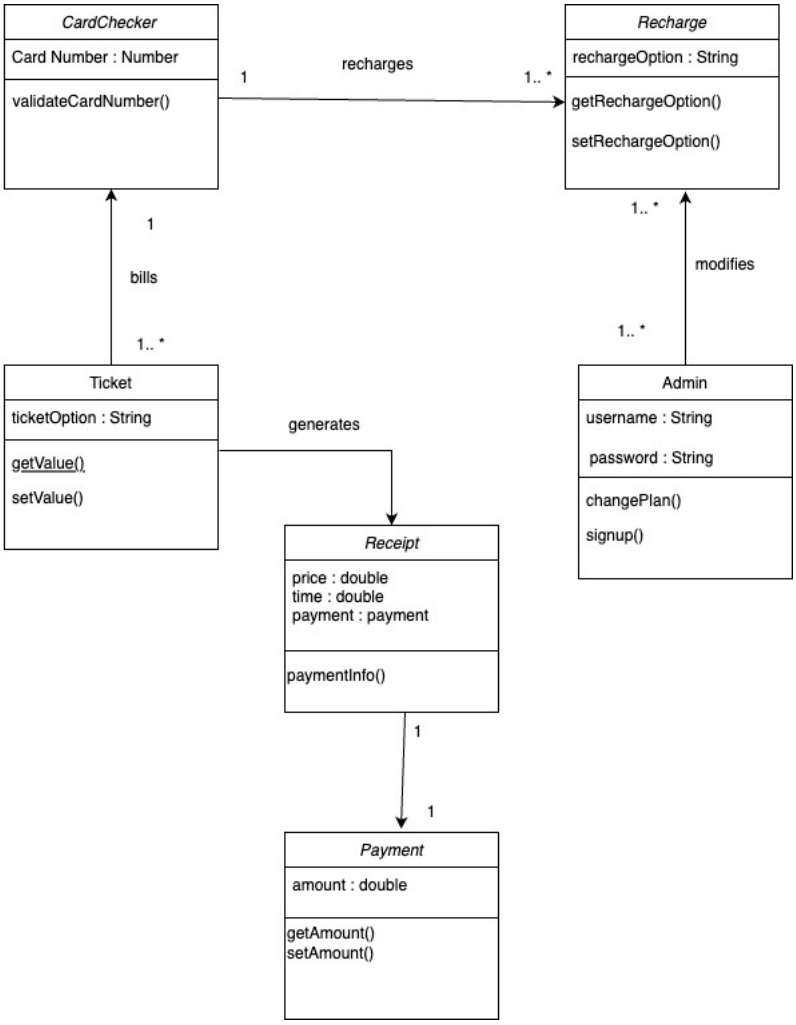


Figure 7.1.1: UML Class Diagram

A UML class diagram is a type of diagram used to depict the structure of a system by showing the classes of objects, their attributes, methods, and the relationships among them. Based on this requirement, I assume that iGo is a software system, and we need to create a class diagram to represent its structural design.

To begin with, we need to identify the classes in the iGo system. A class is a blueprint for creating objects, so we need to determine what objects we will need in iGo.

- CardChecker: This class has a property called "card number" which is of type number. It also has a method called "validatecardnumber()" which presumably checks if the card number is valid.
- Recharge: This class has a property called "rechargeoption" which is of type string. It also has methods called "getrechargeoption()" and "setrechargeoption()" which are used to get and set the value of the "rechargeoption" property.
- Ticket: This class has a property called "ticketOption" which is of type string. It also has methods called "getvalue()" and "setvalue()" which are used to get and set the value of the "ticketOption" property.
- Admin: This class has two properties: "username" and "password", both of which are of type string. It also has methods called "changeplan()" and "signup()" which presumably allow the admin to change the plan and sign up new users.
- Receipt: This class has three properties: "price" and "time" both of which are of type double, and "payment" which is of type Payment (another class). It also has a method called "paymentinfo()" which is used to retrieve information about the payment.
- Payment: This class has one property called "amount" which is of type double. It also has methods called "getAmount()" and "setAmount()" which are used to get and set the value of the "amount" property.

7.2 Sequence Diagram

7.2.1 Login

The Login Sequence Diagram for iGo Ticket Vending Machine depicts the process of logging into the system. The diagram provides a clear overview of the steps involved in the login process, including the entry of user credentials, authentication, and the granting of system access. By following the flow of the diagram, users can gain a deeper understanding of the login process and the different stages involved.

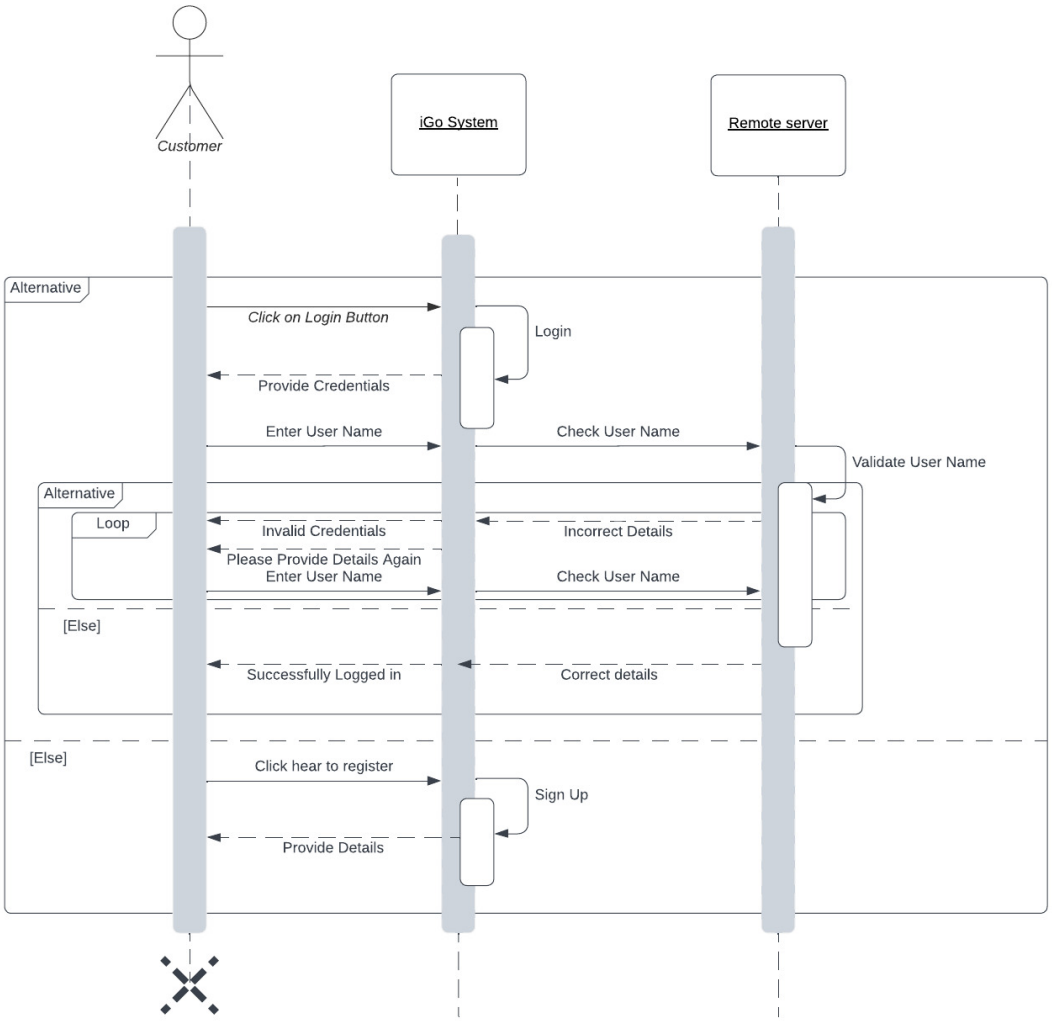


Figure 7.2.1: Login Sequence Diagram

7.2.2 Sign Up

The Sign up Sequence Diagram for iGo System illustrates the process of registering a new user with the system. This diagram provides a step-by-step guide for users to easily understand how they can sign up for the iGo System and gain access to its features. The diagram outlines the various stages involved in the registration process, including the entry of personal information, validation of the information provided, and the creation of a user account. By following the flow of the diagram, users can easily navigate the registration process and become familiar with the various steps involved. Once registered, users can take full advantage of the iGo System's features and benefits.

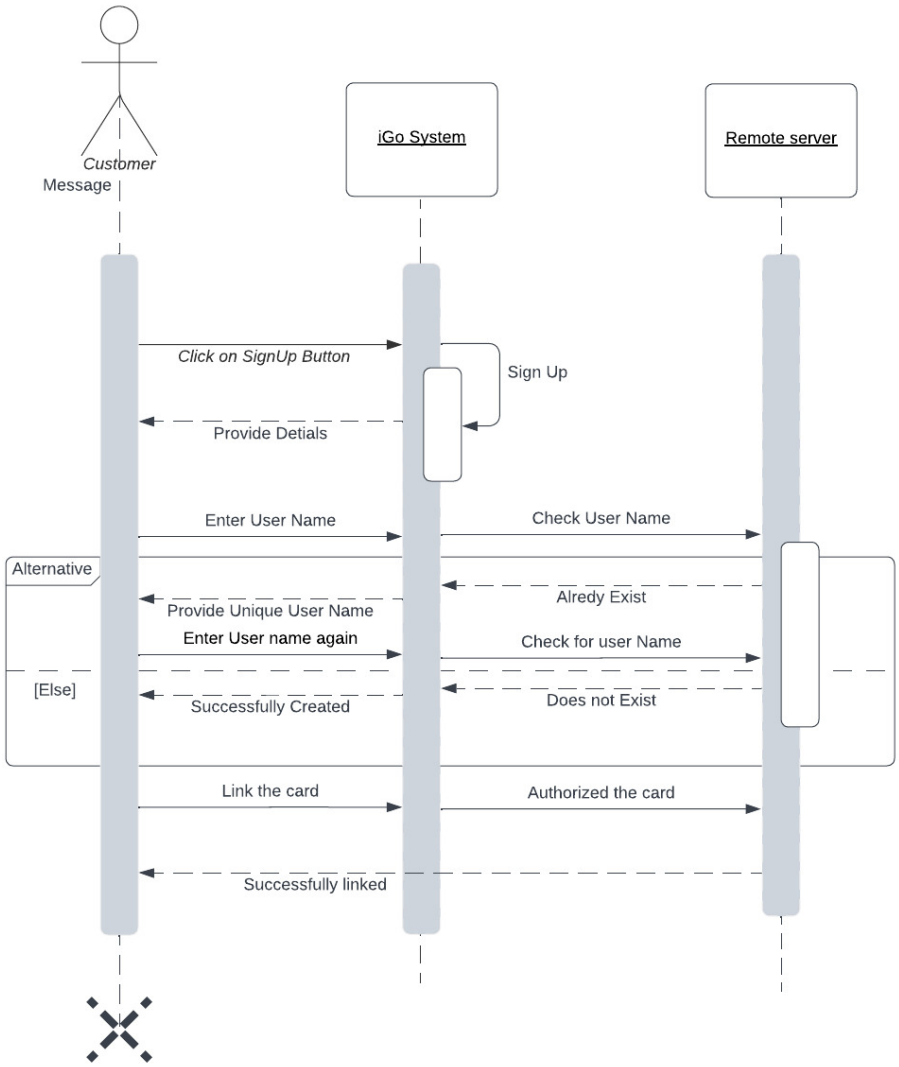


Figure 7.2.2: Sign Up Sequence Diagram

7.2.3 View Activity Details

This Sequence Diagram outlines the processes involved in both logging into and signing up for the iGo System, as well as the ability for users to update their profile. To log in, users must have valid credentials, which they can obtain through the sign-up process. To sign up, new users must follow the steps outlined in the Sign up Sequence Diagram, which includes entering personal information and creating a user account. Once logged in, users can access their profile, which contains their personal information and other details. Users can also update their profile as needed, making changes to their information or preferences. By following the flow of the diagram, users can easily navigate these processes and take full advantage of the features and benefits of the iGo System.

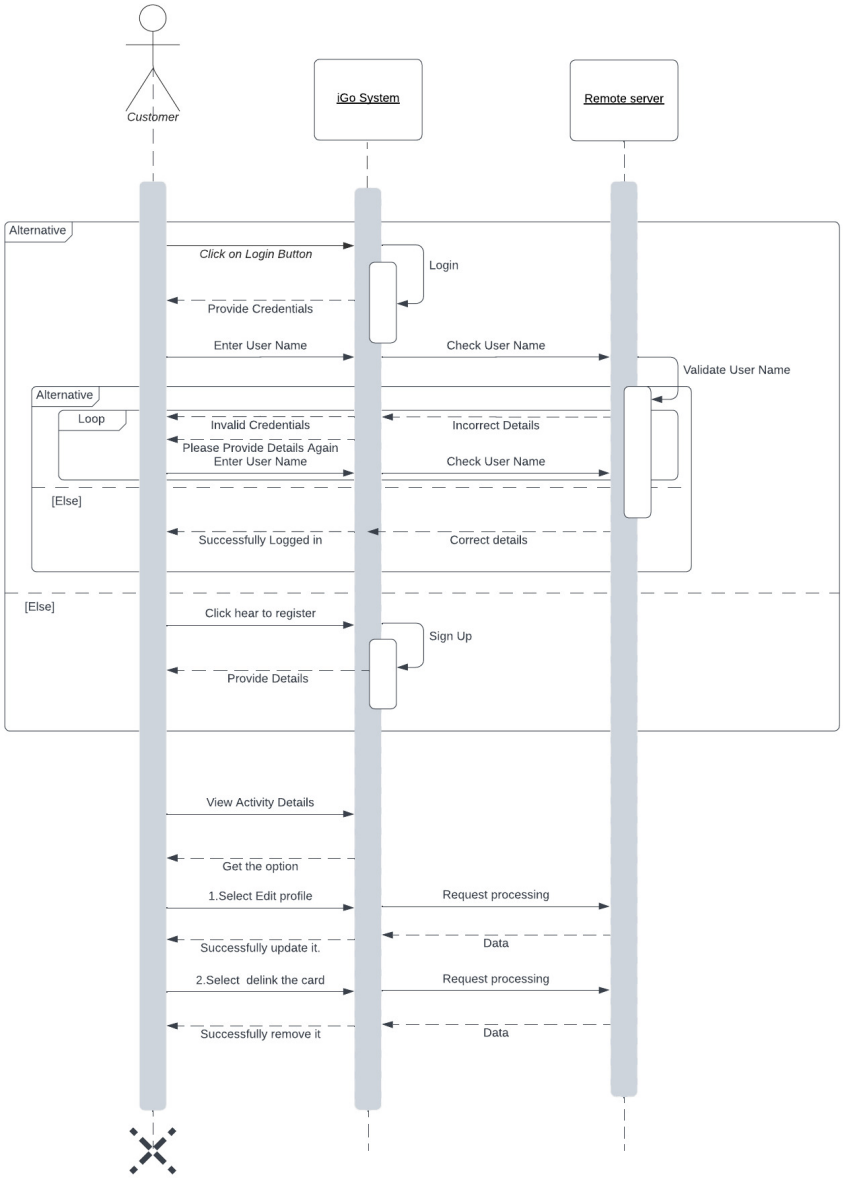


Figure 7.2.3: View Activity Details Sequence Diagram

7.2.4 View Transaction History

The View Transaction History feature of the iGo System provides users with an easy way to access details about their past transactions within the system. By selecting the View Transaction History option, users can see a list of their previous transactions, including details such as the transaction date, time, and amount. Users can also select individual transactions to view more detailed information about each one. Additionally, the View Transaction History feature offers the option to print transaction records, providing users with a hard copy of their transaction history if needed. This feature is a valuable tool for users who need to keep track of their transaction history for accounting or other purposes.

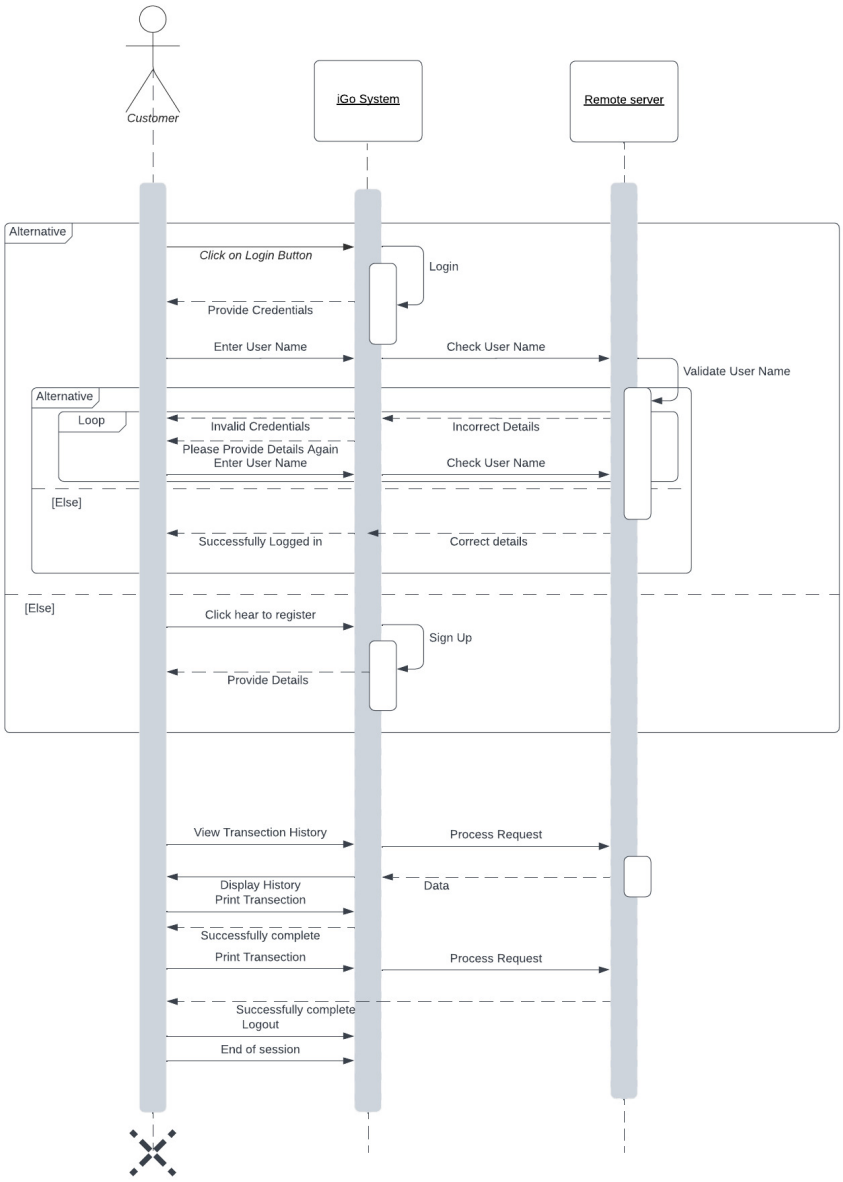


Figure 7.2.4: View Transection History Sequence Diagram

7.2.5 Purchase Ticket

The ticket purchase process in the iGo System is represented in this Sequence Diagram. To purchase a ticket, users must first select the desired ticket type, and then proceed to the payment stage. The system provides users with various methods of payment, including credit card, debit card, and cash payment options. Once the payment has been processed and confirmed, the user receives a receipt for the transaction. This Sequence Diagram provides a clear and detailed overview of the ticket purchase process, from ticket selection to payment and receipt.

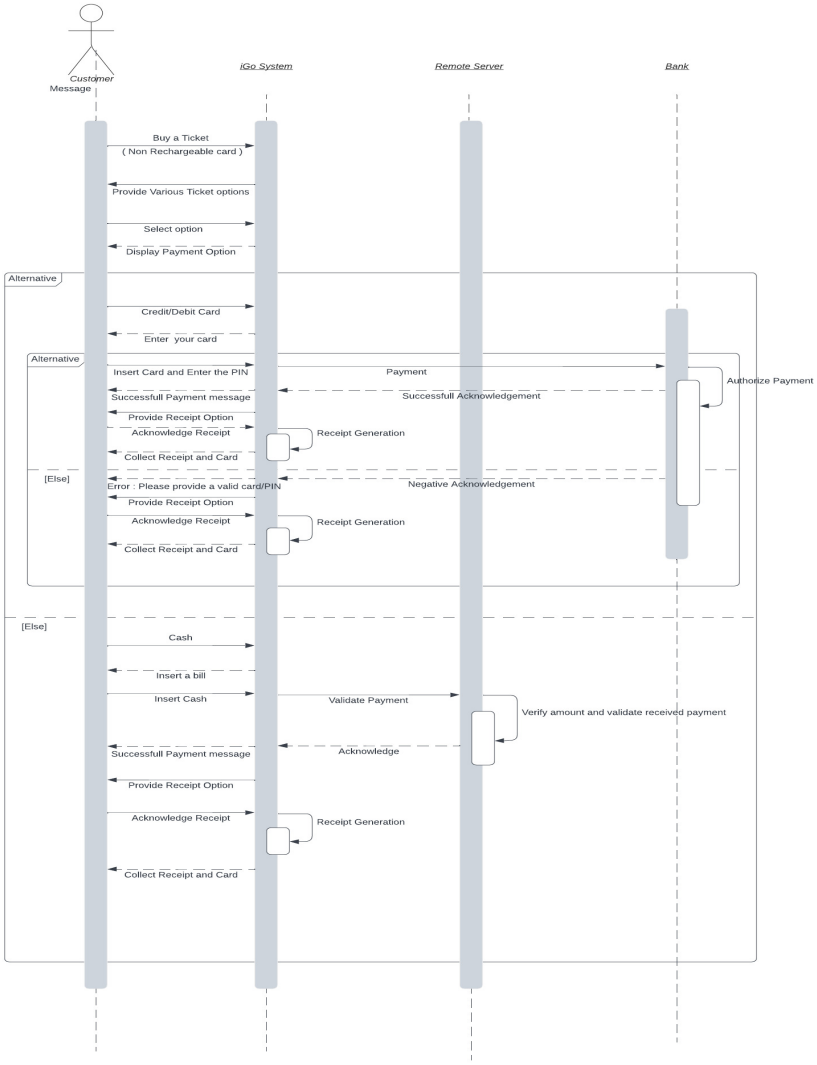


Figure 7.2.5: Purchase Ticket Sequence Diagram

7.2.6 Make Payment

This Sequence Diagram illustrates the payment process for the iGo System. To make a payment, users must first verify their identity through a secure authentication process. Once verified, users can select from a variety of payment options, including credit card, debit card, and cash. The system processes the payment and confirms the transaction, providing the user with a receipt for their records. The iGo System's user-friendly design ensures that making payments is a simple and straightforward process for all users.

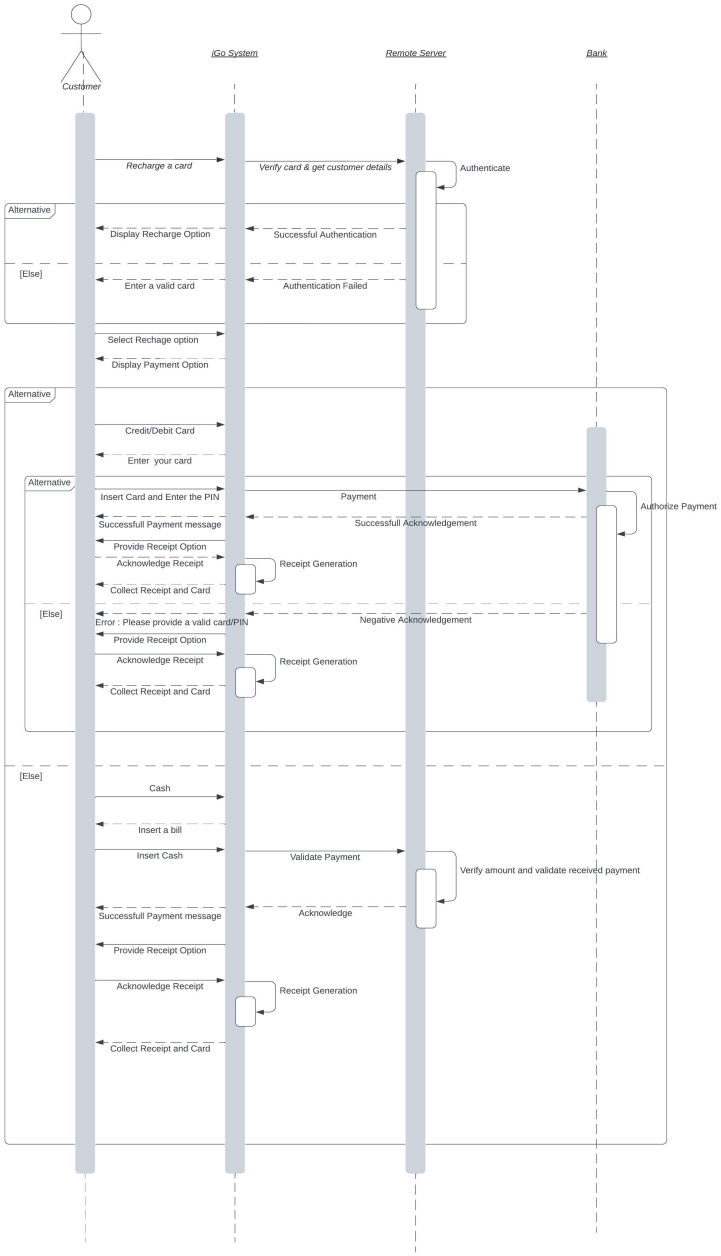


Figure 7.2.6: Make Payment Sequence Diagram

Problem 8

Implementation of the code can be found at the following GitHub link:
https://github.com/mahavir0/iGo—SOEN-6461-SDM

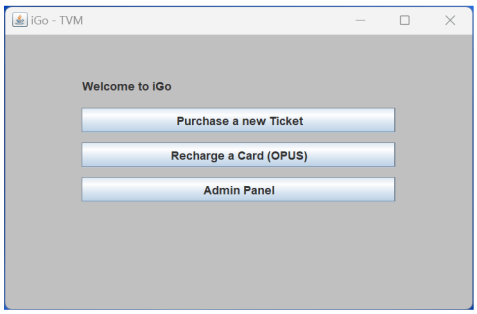
We have used Java Swing GUI for the front end development. It provides a graphical user interface for users to purchase a new ticket, recharge a card, or access the admin panel.
There are several methods to create different screens for the TVM application. The mainPage method creates the main screen with buttons to purchase a new ticket, recharge a card, or access the admin panel. The rechargeCard method creates a screen with options for card recharge. The buyTicket method creates a screen with options for buying a new ticket. The adminPanel method creates a screen with options for the admin user to manage ticket and recharge plans. The paymentMethod method creates a screen with payment options for the user to complete their purchase.

Problem 9

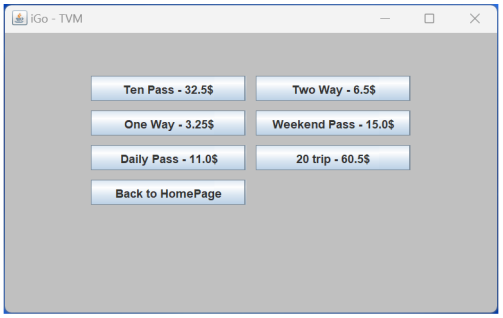
9.1 Test Cases

9.1.1 Purchase New Ticket

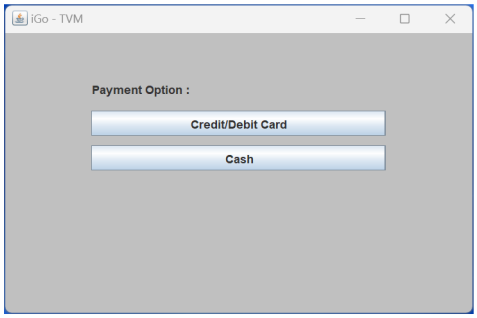
To purchase a new ticket, the user must navigate to the homepage and select the "Purchase a New Ticket" option. From there, the user will be presented with various ticket types to choose from. Next, the user will be prompted to select a method of payment, which includes the option to pay by card or cash. If the user selects the card payment option, they must input valid card details to complete the transaction. Conversely, if the user opts for cash payment, they will need to enter the exact amount of cash and collect any change due. Users will then be given the option to receive the receipt via email or print it at the iGO machine. Once all necessary requirements have been fulfilled, the iGO machine will display a "Thank You" message.



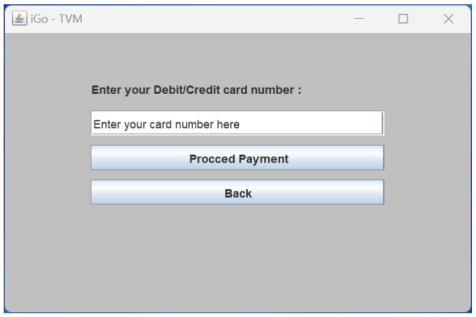
Home Page



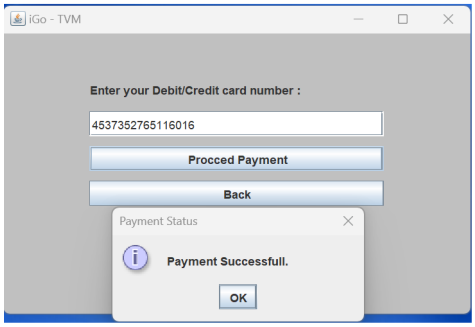
Select Ticket Type



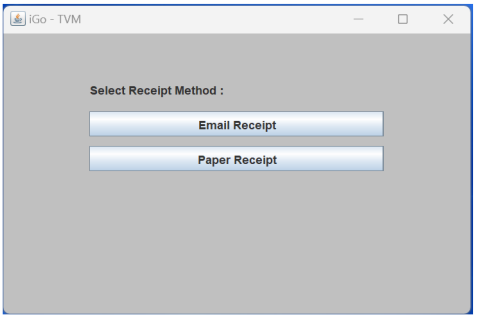
Select Payment Method



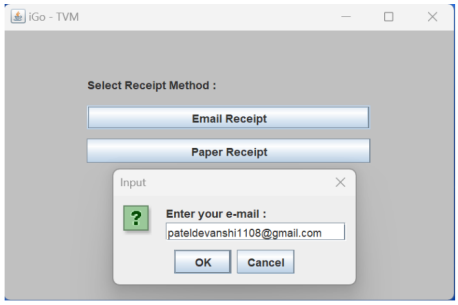
Enter Card Details



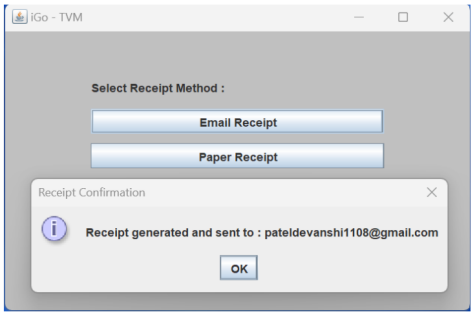
Payment Successful



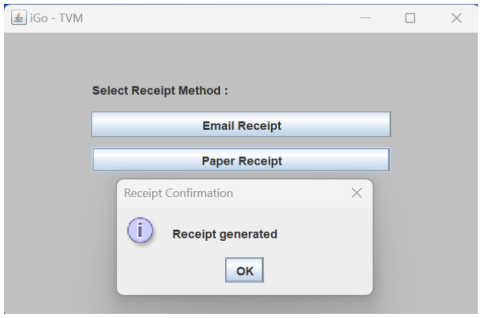
Select Receipt Method



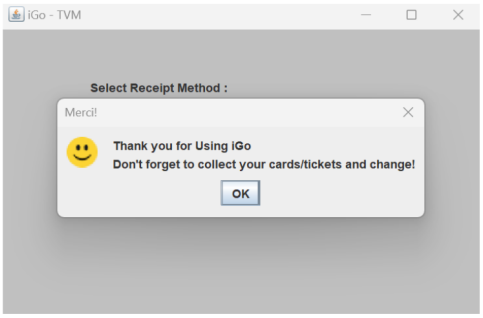
Enter Valid Email Address



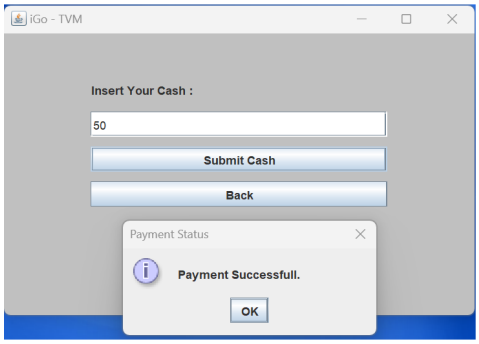
Receipt sent to registered email address



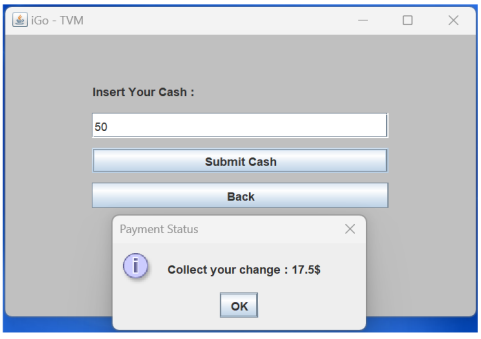
Selecting paper receipt option



Thankyou message at the end



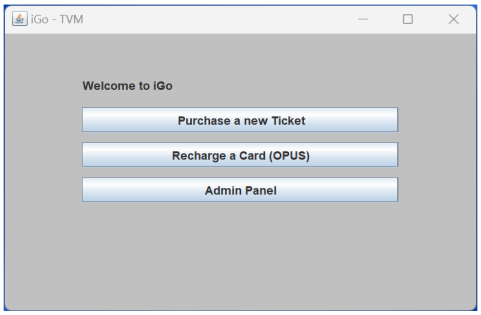
Selecting cash payment option



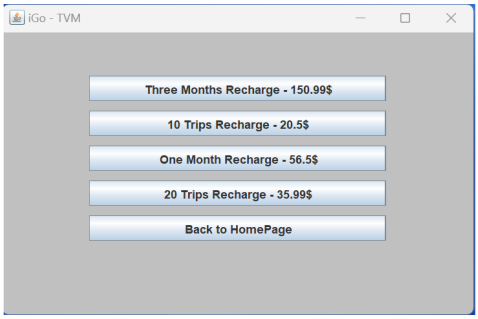
Withdrawing change

9.1.2 Recharge Opus Card

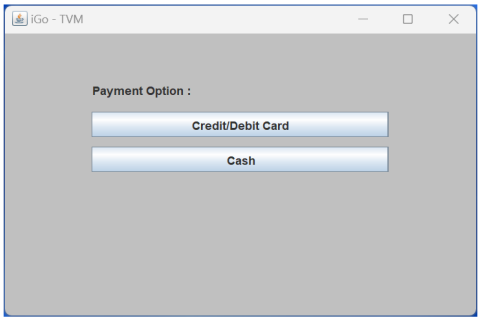
To recharge an Opus card, the user must first access the homepage and select the "Recharge a Card (Opus)" option. From there, the user will be presented with a variety of recharge options to choose from. Next, the user will need to select a payment method, which includes the option to pay via card or cash. If the user chooses to pay via card, they will need to provide valid card information to complete the transaction. Alternatively, if the user opts for cash payment, they will be required to enter the exact amount and collect any change due. Once all necessary requirements are met, a message confirming a successful recharge will be displayed.



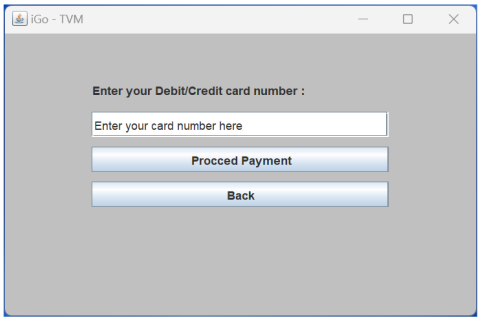
Select Recharge option



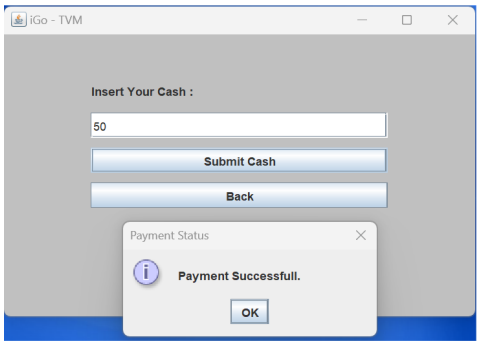
Select recharge option



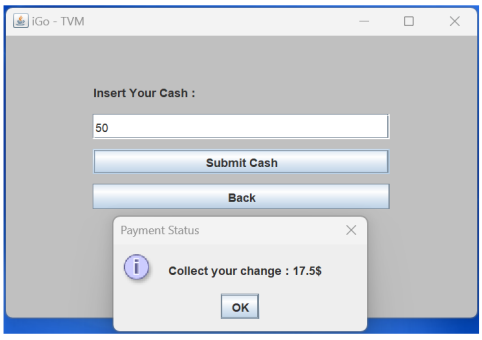
Select Payment Option



Payment using credit card



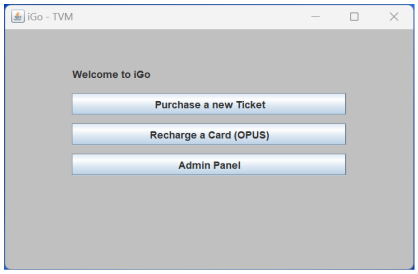
Payment using cash



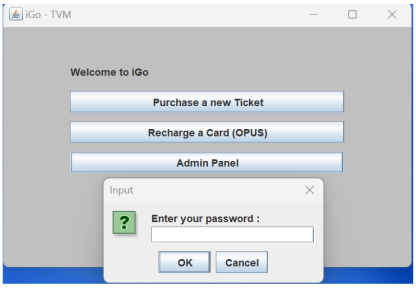
Withdrawing change

9.1.3 Admin Panel

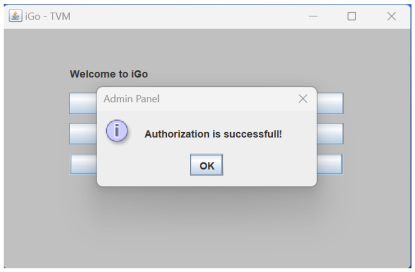
To access the admin panel, the administrator must select the "Admin Panel" option located on the homepage. Afterward, the administrator must verify their identity by entering the correct password to complete the authorization process. Once authorization is successful, the administrator can proceed to modify the existing ticket plans or recharge plans as required. This will involve entering the updated price for the chosen plan. Once the prices have been successfully updated, a message will be displayed confirming the same.



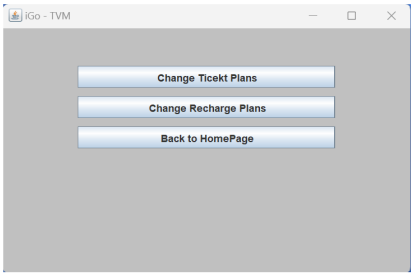
Select Admin panel option



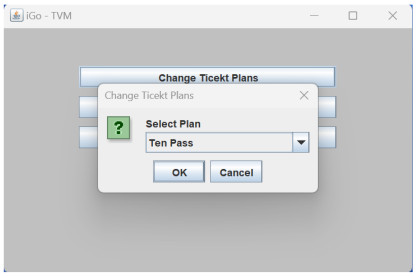
Enter password for Admin



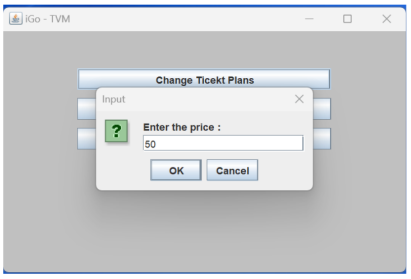
Authorization Successful



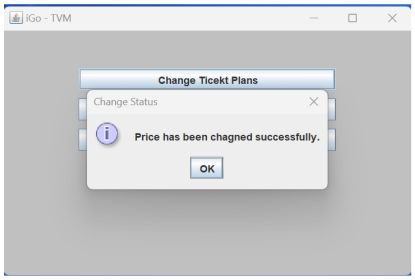
Select one option



Change Ticket Plan



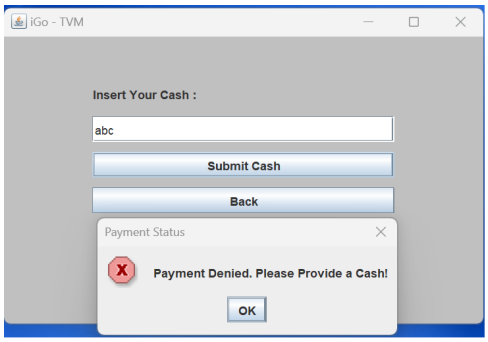
Change Price of selected plan



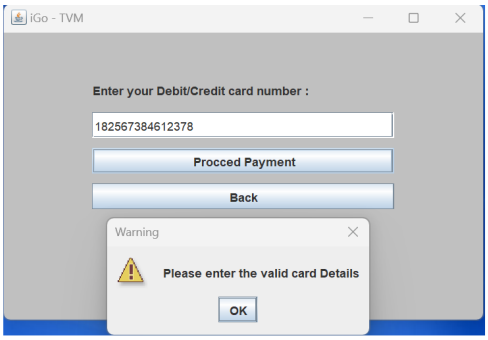
Price changed sucessfully

9.1.4 Negative Test Cases

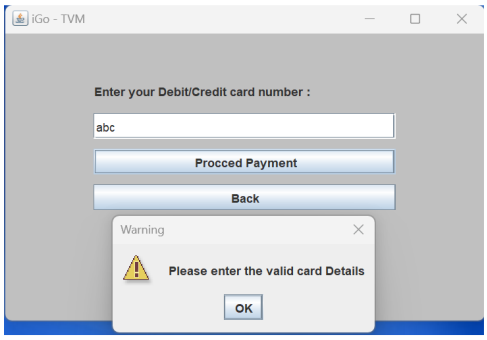
In the case of negative use cases, if the user attempts to enter non-numeric values when inserting the cash amount, a message will be displayed indicating that payment has been denied and prompting the user to enter a valid cash amount. Similarly, for card payments, the inputted card details must be numeric and correct for the payment to be successful. If the email address provided is not in the proper format, the iGO system will not accept it, and the user will be prompted to enter a valid email address. For admin authorization, the password is case sensitive, and any deviation from the correct password format will result in a failed login attempt. Therefore, the administrator must enter the exact password to complete the authorization process successfully.



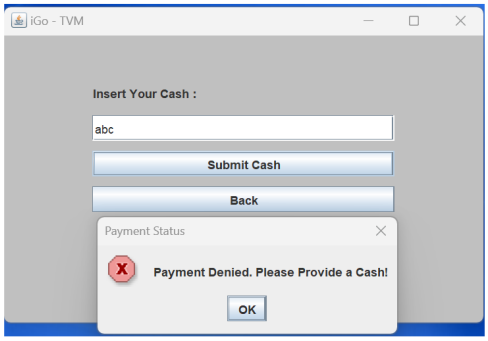
Cash value needs to be numeric



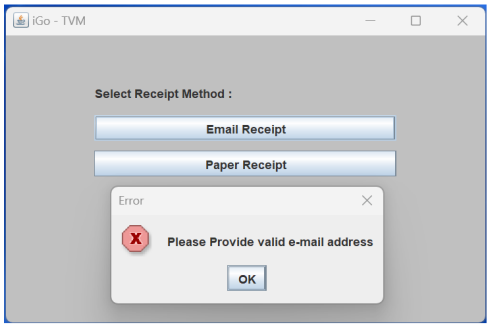
Enter valid card details



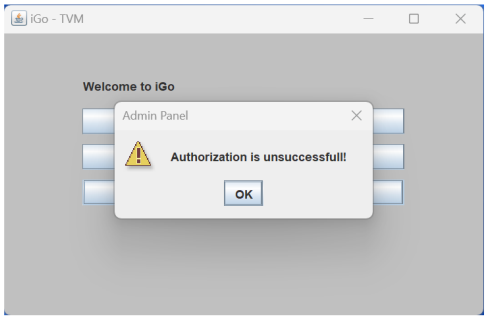
Card number needs to be numeric



Cash value needs to be numeric



Enter valid email address



Admin password need to be correct

Tools Used

- Overleaf
- Xmind
- LucidChart
- GitHub
- Google drive

References

1. PANKAJ KAMTHAN (2023) “Introduction to Responsibility-Driven Design ”
2. PANKAJ KAMTHAN (2023) “Introduction to Diagramming”
3. PANKAJ KAMTHAN (2023) “A Case Study of the Object-Oriented Design of an Elevator Control System”
4. PANKAJ KAMTHAN (2023) “Introduction To Domain Modeling”
5. PANKAJ KAMTHAN (2023) “Introduction To Use Case Modeling”
6. `https://www.lucidchart.com/pages/uml-activity-diagram`
7. `https://echeung.me/crcmaker/`
8. `http://www.stm.info/en`