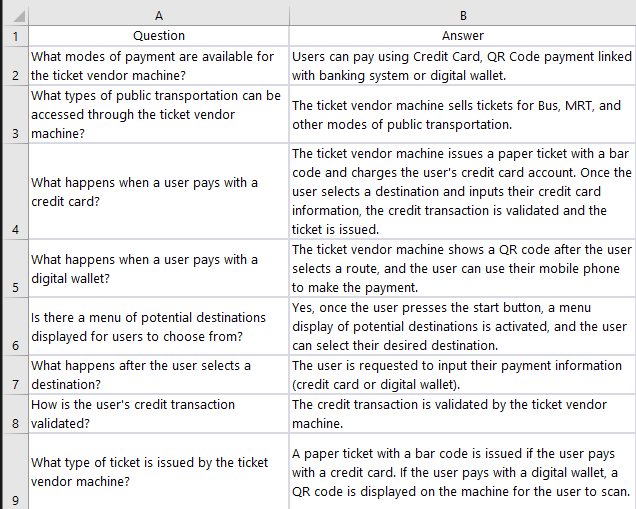
Name: Đặng Lữ Anh Kiệt

Student\_ID: 521H0090

Ticket Vendor Machine

**Question 1:**



**Question 2:**

**Functional Requirements:**

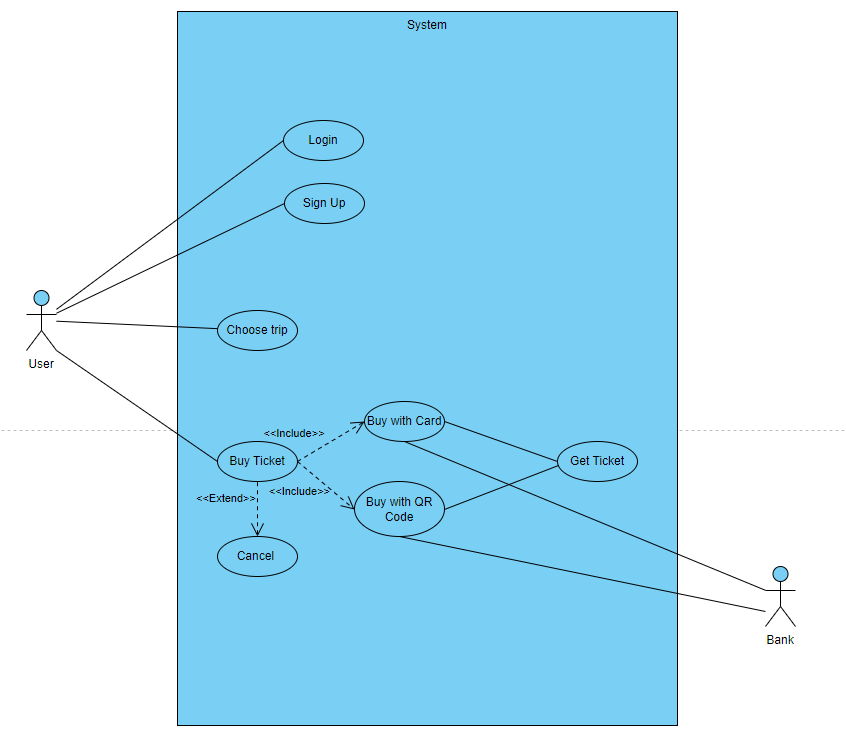
* The ticket vendor machine should sell tickets for various modes of public transportation, such as buses, trains, and subways.
* The machine should allow users to select their destination from a menu of potential destinations.
* The machine should allow users to select their payment method, which may include credit card, QR Code payment linked with banking system or digital wallet.
* The machine should issue a paper ticket with a bar code if the user pays with a credit card.
* The machine should display a QR code if the user pays with a digital wallet.
* The machine should validate credit transactions and issue tickets once payment is accepted.
* The machine should be able to print and dispense tickets quickly and accurately.
* The machine should be easy to use and provide clear instructions to users.

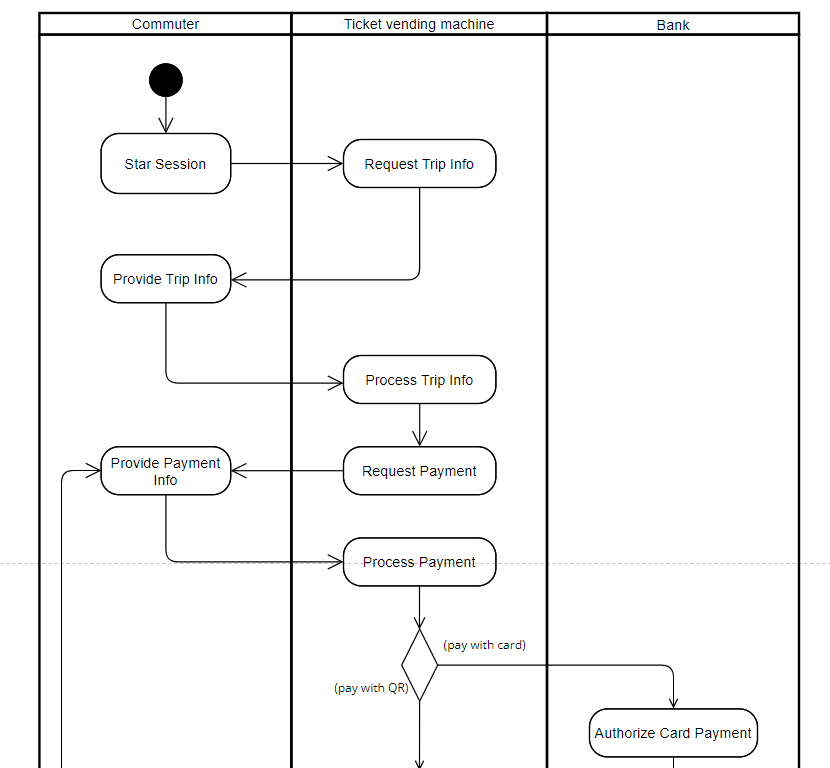
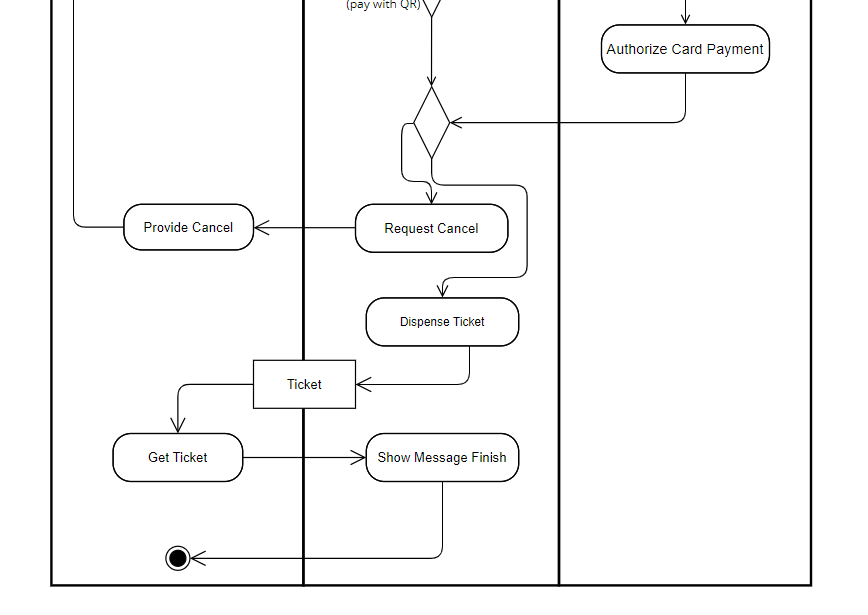
**Non-Functional Requirements:**

* The machine should be reliable and have a high level of uptime to ensure that users can purchase tickets at all times.
* The machine should be able to process payments quickly and efficiently to minimize wait times for users.
* The machine should have a user-friendly interface that is easy to navigate and understand.
* The machine should be secure and protect users' payment and personal information from unauthorized access.
* The machine should be easy to maintain and service to ensure that it remains functional and reliable.

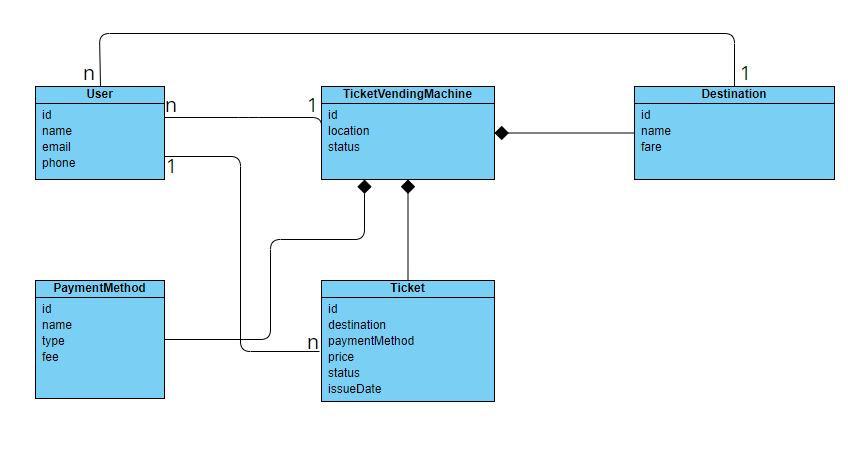
**Domain Requirements:**

* The machine should be compatible with the payment systems used by various transportation providers.
* The machine should be able to handle different types of currency and payment methods.
* The machine should be able to communicate with other parts of the smart ticketing system to ensure that users have access to up-to-date information on routes and schedules.
* The machine should be able to provide users with information on fares, routes, and schedules.
* The machine should be designed to withstand various weather conditions and be able to operate in both indoor and outdoor environments.
* The machine should be designed to be accessible to all users, including those with disabilities.

**Question 3:**

**Question 4:**

**Question 5:**



Class Diagram: A class diagram is a UML diagram that shows the relationships between classes in a system. For the ticket vending machine, the classes would include "user," "ticket vending machine," "destination," "payment method," and "ticket." The relationships between these classes would show how they interact with each other, for example, the user interacts with the ticket vending machine to purchase a ticket, and the ticket vending machine interacts with the payment method to process the payment and issue the ticket.

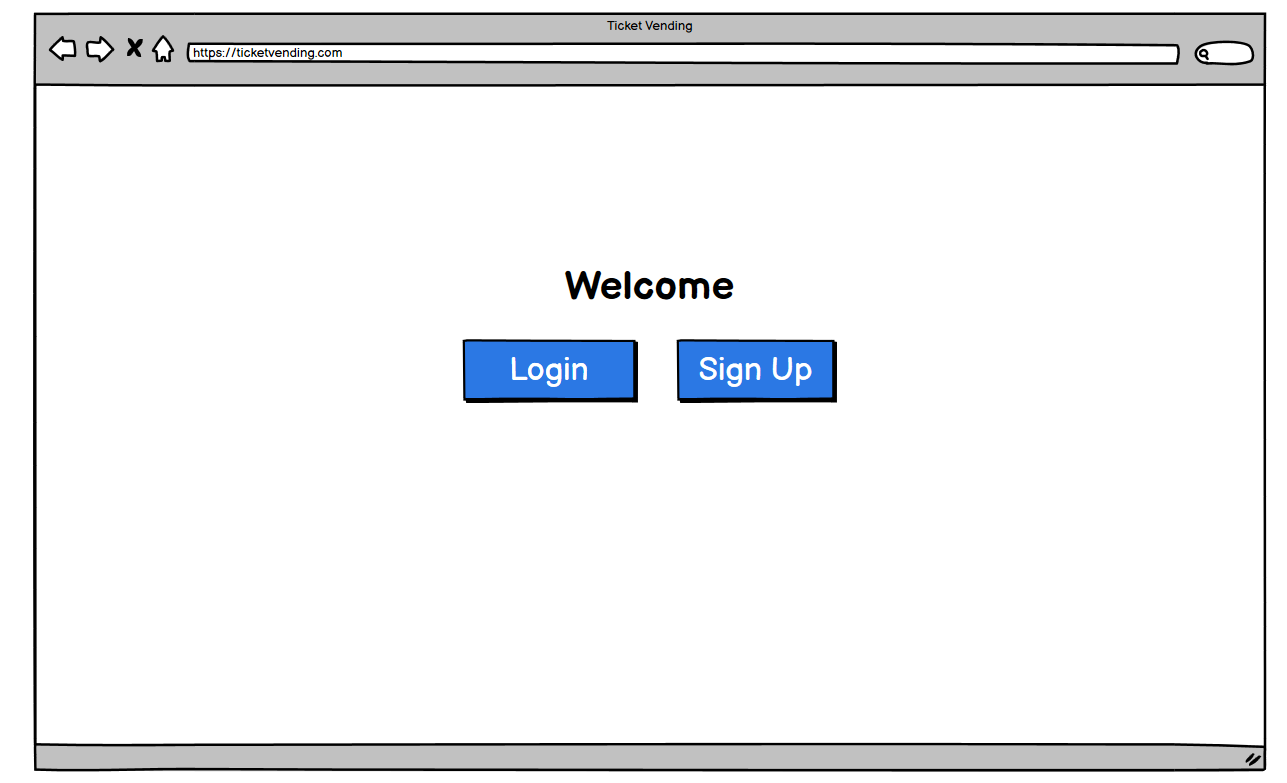
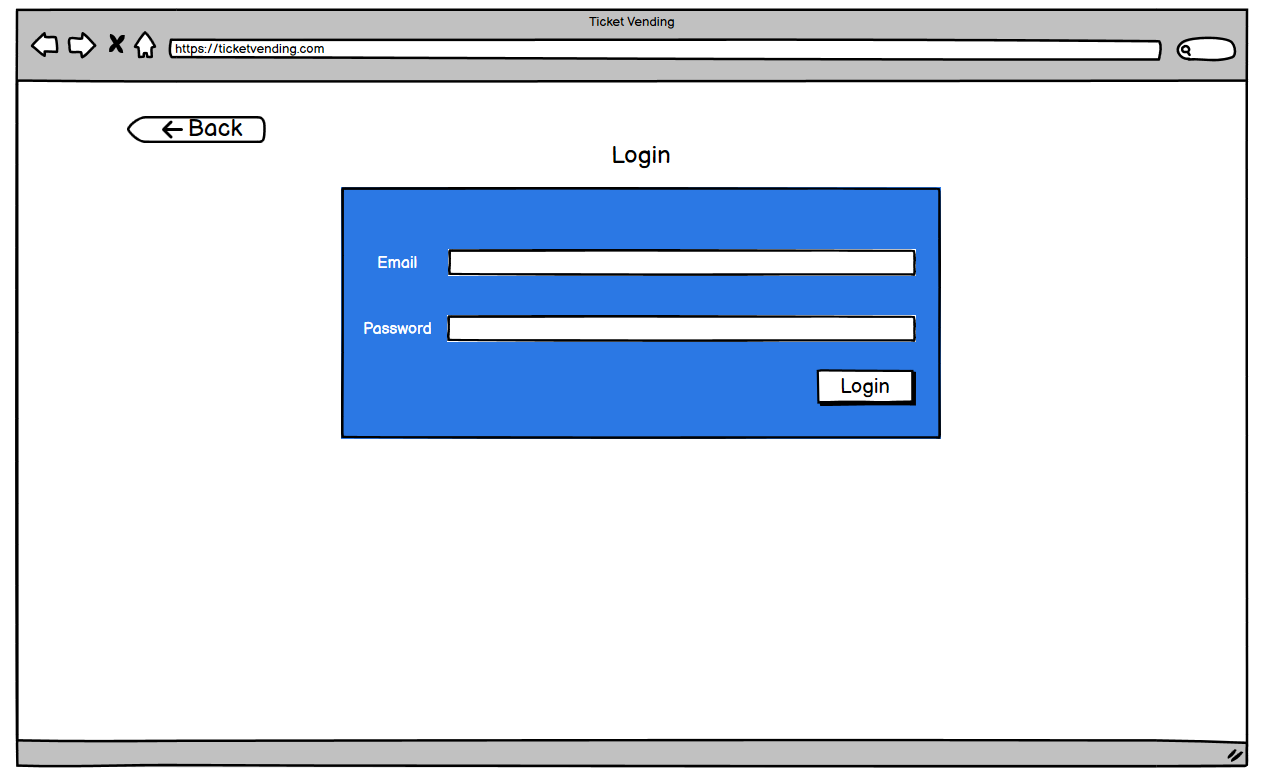
A User can purchase many Tickets: 1-n (One-to-Many) relationship between User and Ticket.

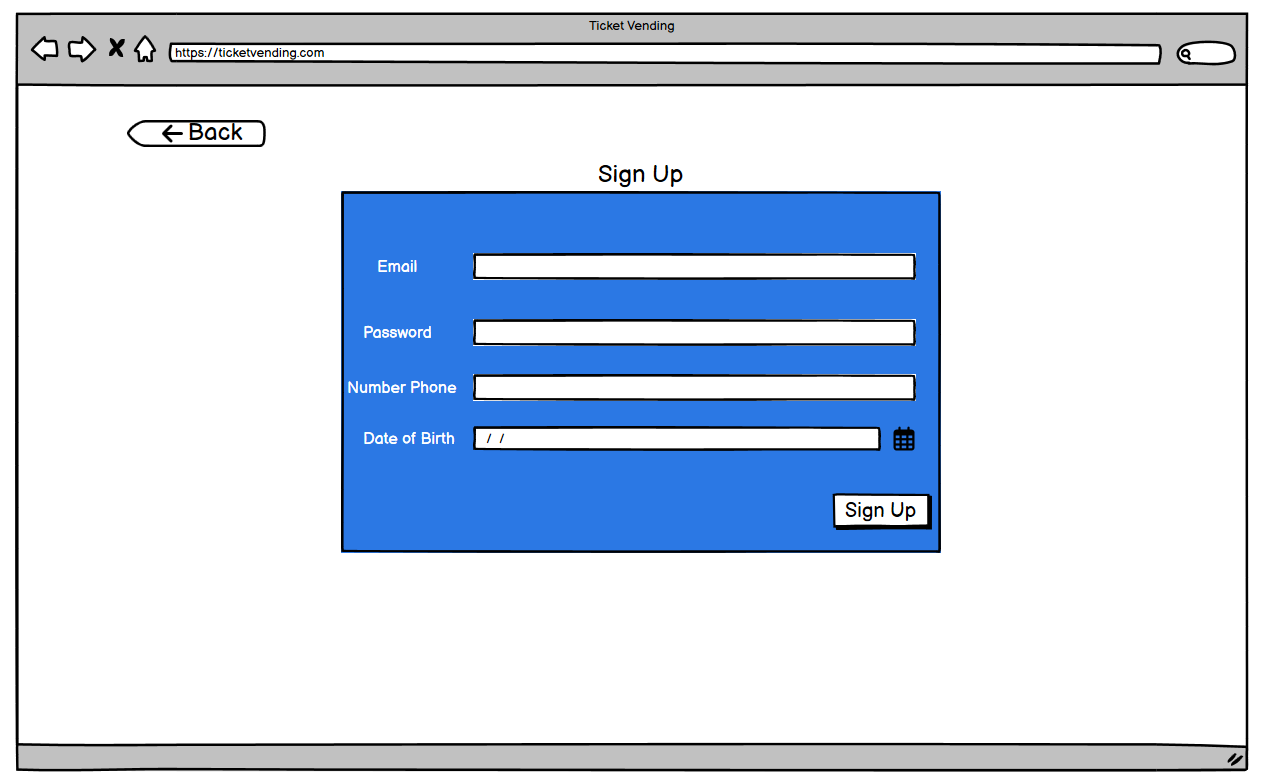
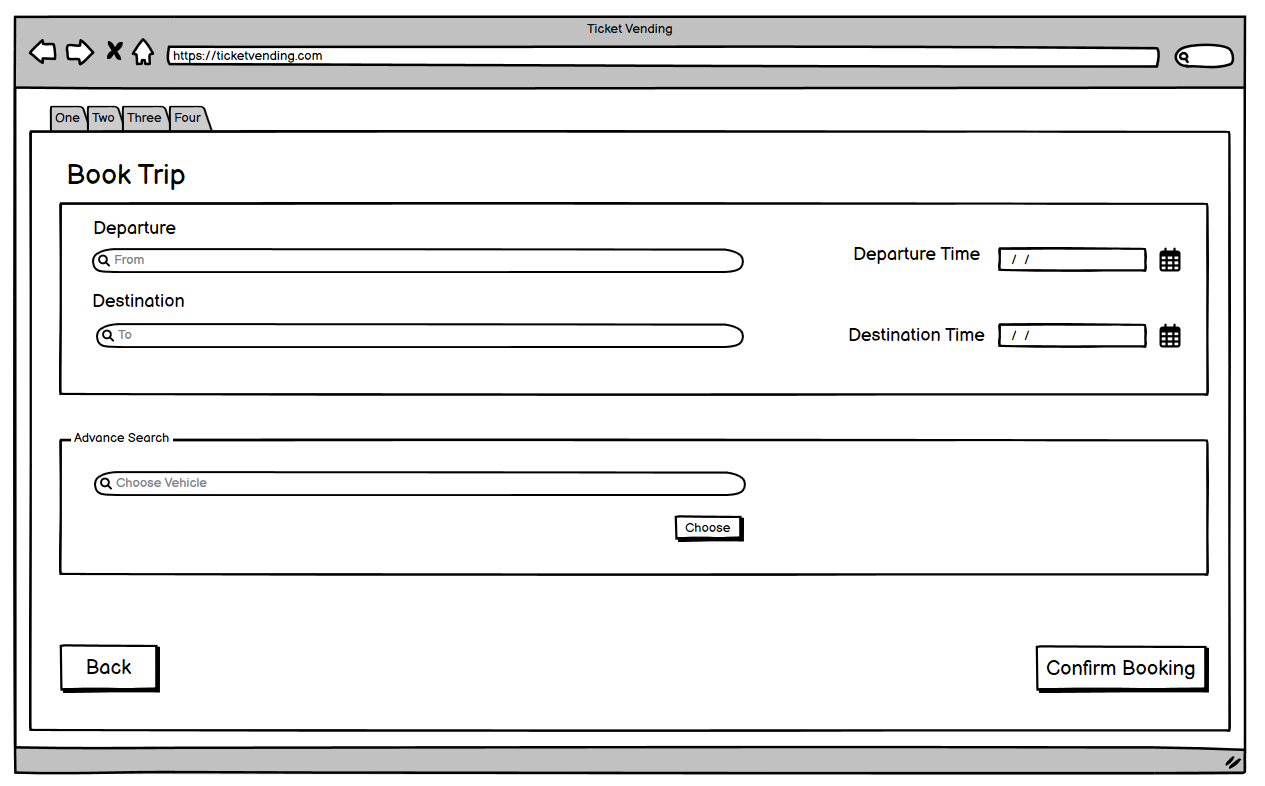
A TicketVendingMachine can serve many Users: 1-n (One-to-Many) relationship between TicketVendingMachine and User.

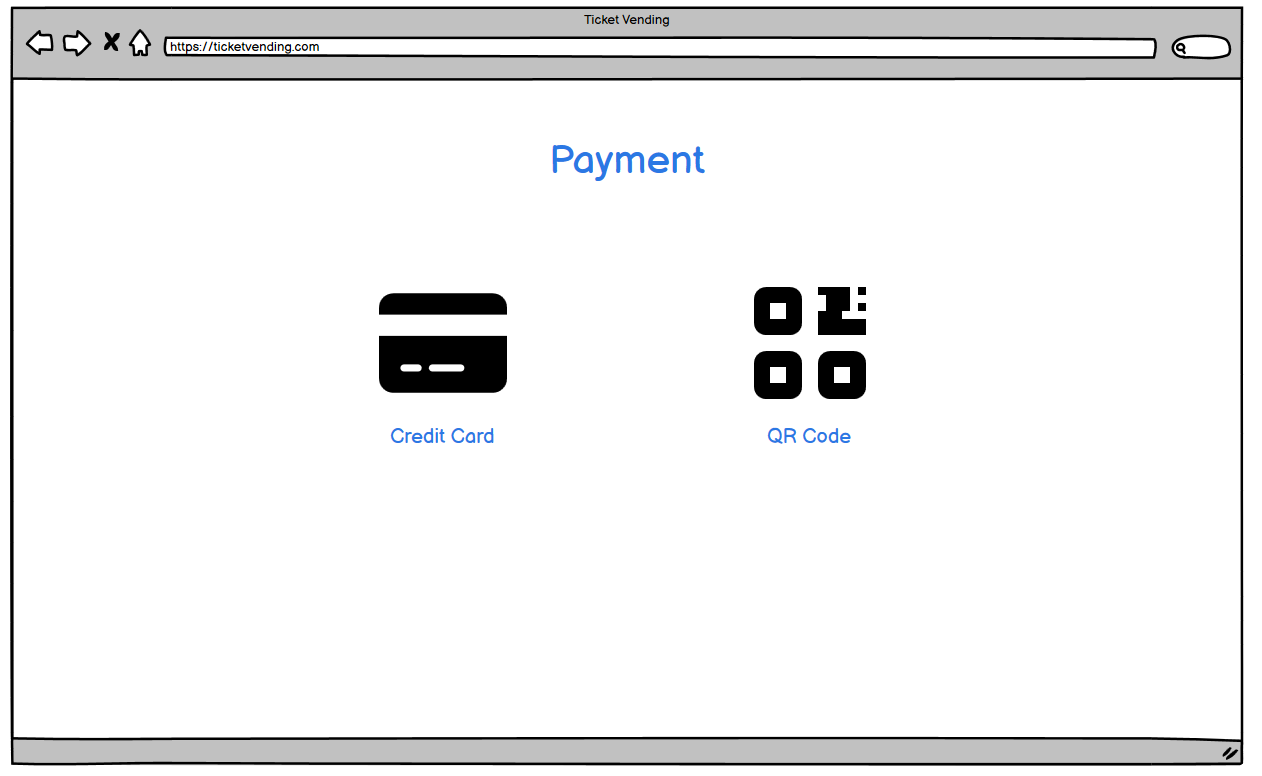
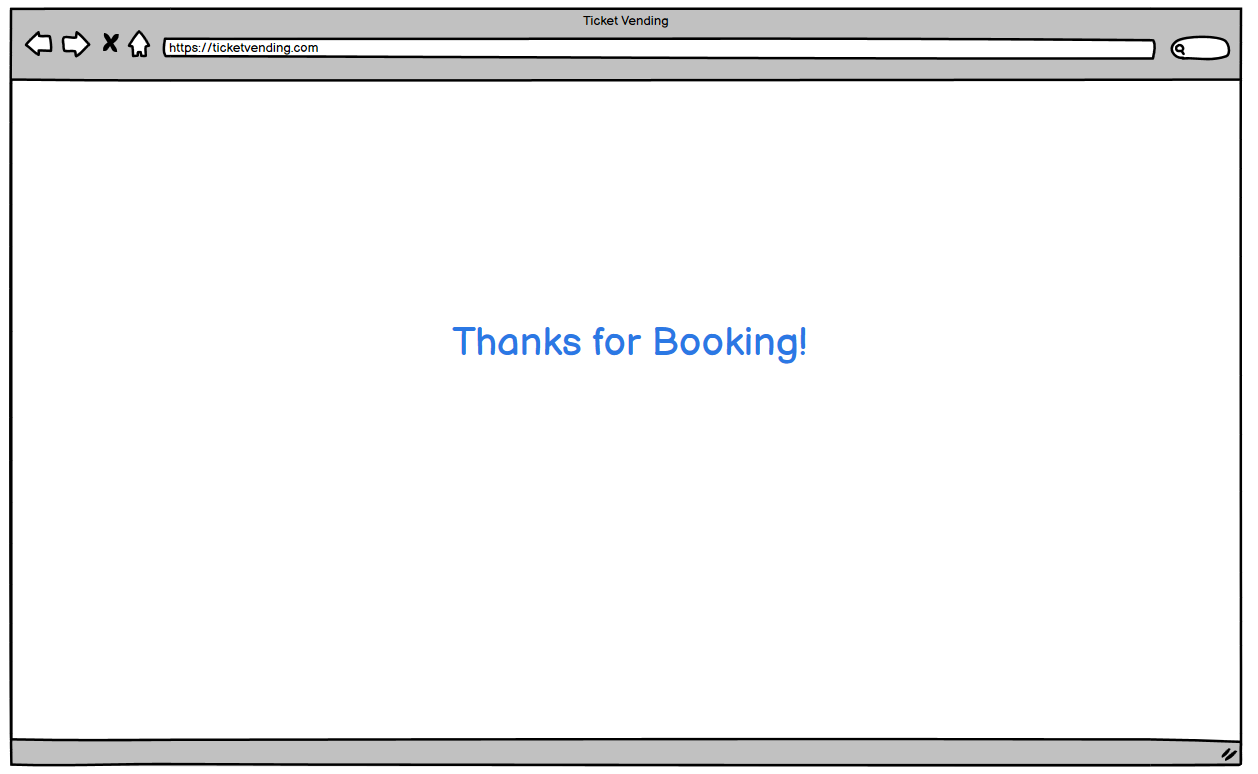
A Ticket can be issued for a Destination and use a PaymentMethod: Many-to-One-Many (n-1-n) relationship between Ticket, Destination, and PaymentMethod.

A PaymentMethod can be used to purchase many Tickets: 1-n (One-to-Many) relationship between PaymentMethod and Ticket.

A Destination can be selected by many Users: Many-to-One (n-1) relationship between Destination and User.

**Question 6:** Balsamiq





**Question 7:**

* SQl:
* CREATE TABLE User (
* id INT PRIMARY KEY,
* name NVARCHAR(255),
* email NVARCHAR(255),
* phone VARCHAR(255)
* );
* CREATE TABLE TicketVendingMachine (
* id INT PRIMARY KEY,
* location NVARCHAR(255),
* status NVARCHAR(255)
* );
* CREATE TABLE Destination (
* id INT PRIMARY KEY,
* name NVARCHAR(255),
* fare FLOAT
* );
* CREATE TABLE PaymentMethod (
* id INT PRIMARY KEY,
* name NVARCHAR(255),
* type NVARCHAR(255),
* fee FLOAT
* );
* CREATE TABLE Ticket (
* id INT PRIMARY KEY,
* destination\_id INT,
* payment\_method\_id INT,
* price FLOAT,
* status NVARCHAR(255),
* issue\_date DATETIME,
* FOREIGN KEY (destination\_id) REFERENCES Destination(id),
* FOREIGN KEY (payment\_method\_id) REFERENCES PaymentMethod(id)
* );
* INSERT INTO User (id, name, email, phone)
* VALUES (1, 'John Smith', 'john.smith@gmail.com', '84941234567');
* INSERT INTO TicketVendingMachine (id, location, status)
* VALUES (1, 'Metro Bến Thành - Suối Tiên', 'Operational');
* INSERT INTO Destination (id, name, fare)
* VALUES (1, 'Ho Chi Minh City', 2.50);
* INSERT INTO PaymentMethod (id, name, type, fee)
* VALUES (1, 'Credit Card', 'Card', 0.20);
* VALUES (2, 'QR Code', 'QR', 0.50);
* INSERT INTO Ticket (id, destination\_id, payment\_method\_id, price, status, issue\_date)
* VALUES (1, 1, 1, 3.00, 'Issued', '2023-03-19 10:30:00');
* SELECT \* FROM User

Question 9:

Link Github: <https://github.com/DG-Kit/QT1-SE>