INTERNATIONAL UNIVERSITY
VIETNAM NATIONAL UNIVERSITY, HCM CITY

School of Computer Science & Engineering

PROJECT REPORT

Topic 08: COACH TICKET SELLING

Lecturer: Nguyen Thi Thuy Loan
Course: Principle of Database Management

Group members:

Nguyễn Huỳnh Phương Thanh (Leader)	(35%)	ITITIU18115
Trần Công Mẫn	(35%)	ITITIU18285
Dương Thành Công	(30%)	ITITIU18229

Contents

I. Introduction

Nowadays, buying something over the Internet is very popular. Conveniently, you can sit at home to order and buy a ticket for your trip over the Internet. You do not need to go to a station to buy a ticket. Sometimes, after coming there, you can disappointedly discover that the ticket for your favorite trip is sold out.

With a new e-commerce site, you can buy and pay for your ticket over the Internet. You only need to print your ticket out, carry it with you on the traveling day, and then enjoy your trip. This kind of ticket called "E-Ticket" that is very popular in Western countries.

To adapt market demand, our group is making an application for buying coach tickets over the Internet. It based on the principle of database management and querying the database using C# Database Connectivity.

This application is built on three main subjects. Each participant in the app is provided with an account. Through the initial login interface, the users depending on the role, are given different functions.

a. *User:*

Users will be provided for the ability to look up trips based on place and date. The booking will be made when the user chooses the desired trip. The payment also happens online, via online transfer using a debit or credit card, directly on the application. Moreover, the users can search their tickets by ID Ticket if they forget.

b. *Driver:*

Drivers will search the trip in a period time that they have to drive giving by manager.

c. *Manager:*

Managers can access and modify information about outdated and upcoming trips, manage information of user and driver, assign trips to driver, collect the report of the ticket's amount, revenue each month or year. The management of trip income is also done by the managers.

II. Entity – Relationship Diagram

1. Requirement

The Coach E-Ticket system allows users to order and buy tickets over the Internet. The manager can manage all the trip's information, customer's information, drivers so on.

The user must log in to access the system by the account. Each account has a unique username and password.

Each User has a unique number, a name, address, mail, phone, date of birth, gender, and a particular role of each user.

The User system is organized into 3 roles: Manager, Driver, User.

The Manager can manage trips through many tools: add, delete, search, adjust, check the status from the system. Similarly, managers are given the same right to operate these acts to the drivers' information.

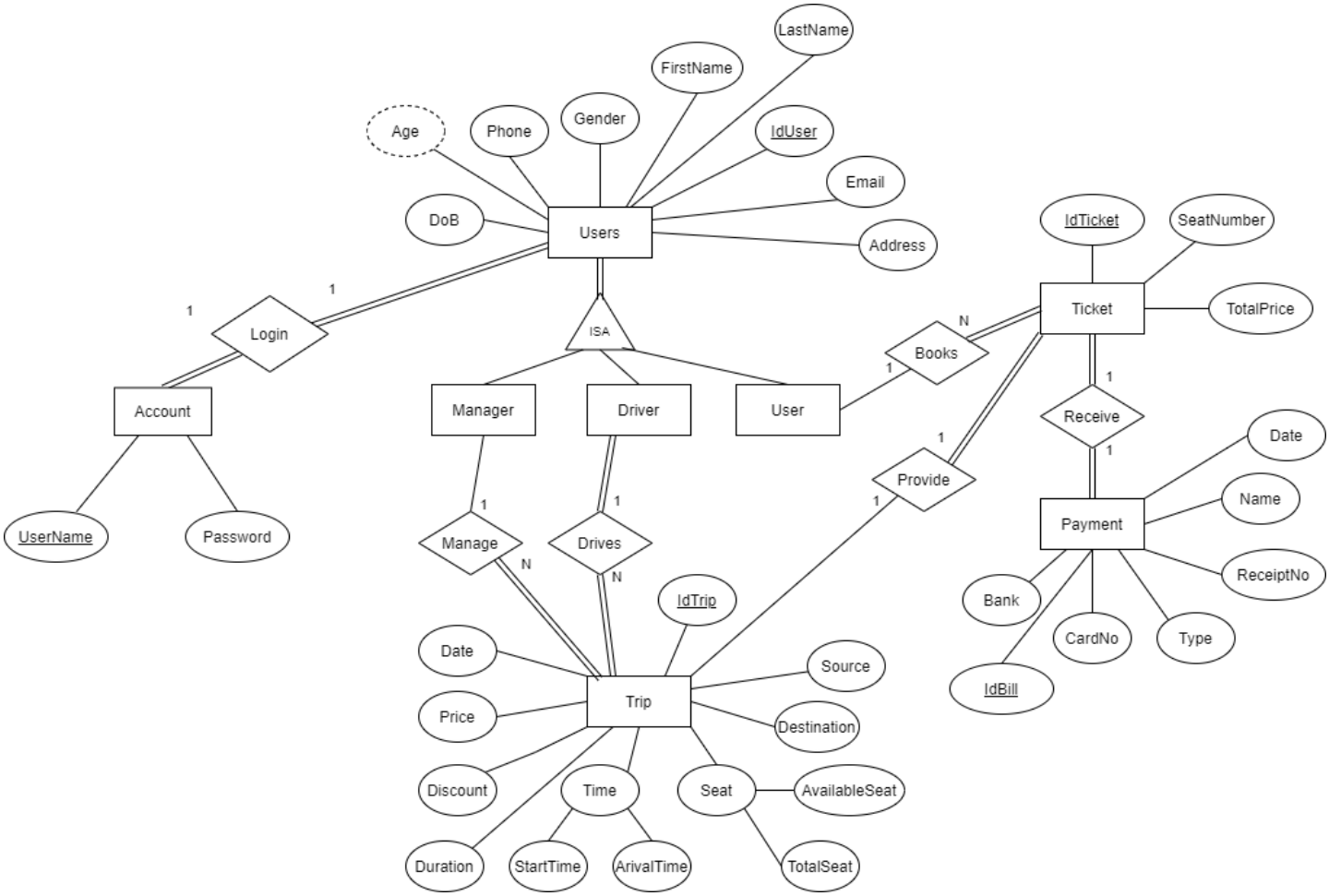
The User can book the ticket that is provided by trip information. And the Driver can view his trip in a period of time.

Each trip has its own ID trip, source, destination, seat's information, time, date, price, discount. Each ticket has its number, seat number, and total price of the trip it is provided.

The ticket will be received the payment from the users when they pay it by card. Each payment has a receipt number, name, bank, card number, type includes a debit card and credit card and date

.

2. Entity Relationship Diagram (ERD)



ER Model of Coach Ticket Selling database system

Advantages:

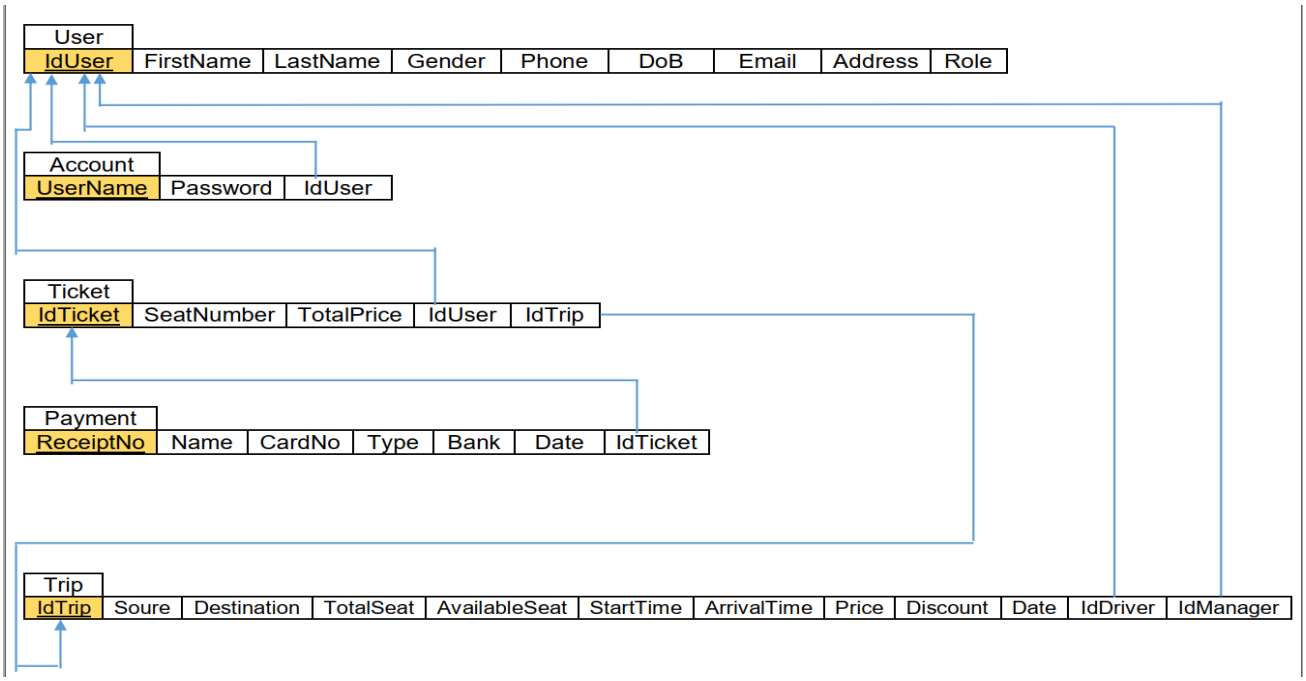
- Easy to visualize the relationship among entities and relationships.
- It is an effective communication tool for database designer
- It is highly integrated with the relational model

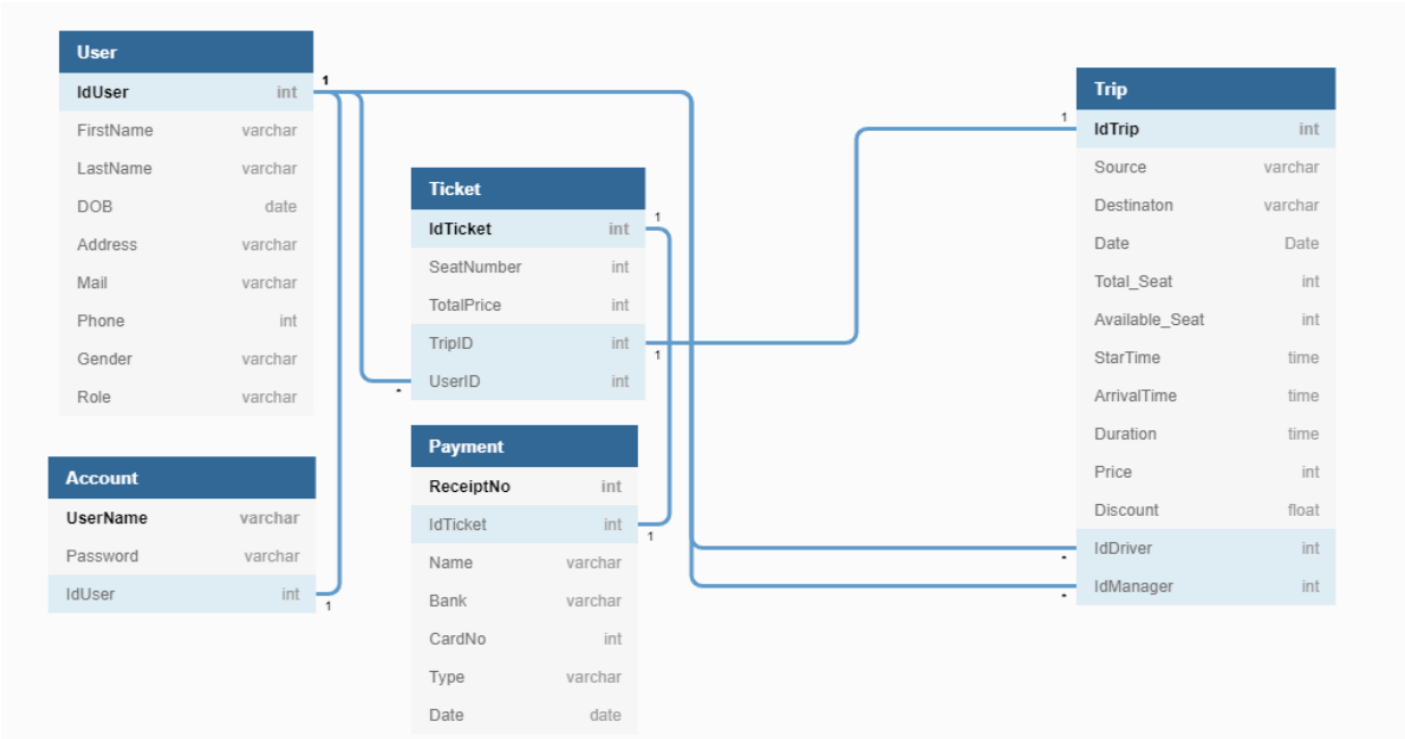
Disadvantages:

- Some information could be hidden in ER model
- Limited relationship representation
- No representation of data manipulation
- Popular for high level design

III. Relational Model

1. Relational Model





2. Explanation

a. For the entity:

We have five entities in total (Users, Account, Trip, Ticket, Payment). Thus, changing from ER diagram to relational model gives out five schemas.

Each has the primary keys as given:

Users(IdUser, FirstName, LastName, Gender, Phone, DoB, Email, Address, Role)

Account(UserName, Password)

Trip(IdTrip, Soure, Destination, TotalSeat, AvailableSeat, StartTime, ArrivalTime, Price, Discount, Date)

Ticket(IdTicket, SeatNumber, TotalPrice)

Payment(ReceiptNo, Name, CardNo, Type, Bank, Date)

b. For the relationship

Login relationship (between Users and Account): It is a 1-1 relationship. Therefore, we will place the primary key of Users in the schema of Account as foreign key or we can also do the opposite.

Adds relationship and Drives relationship (between Users and Trip): They are 1:N relationship. So, we will only have a way to present which is placing the primary key of Users, in detail, ID_Manager, ID_ Driver, in the schema of Trip as foreign key.

Books relationship (between Users and Ticket): It is a 1-N relationship. So, we will only have a way to present which is placing the primary key of Users, in detail, ID_User in the schema of Ticket as foreign key.

Provide relationship (between Trip and Ticket): It is a 1-1 relationship. Therefore, we will place the primary key of Trip in the schema of Ticket as foreign key or we can also do the opposite.

Receive relationship (between Ticket and Payment): It is a 1-1 relationship. Therefore, we will place the primary key of Ticket in the schema of Payment as foreign key or we can also do the opposite.

Combining (1) and (2), the relation schema is:

Users (IdUser, FirstName, LastName, Gender, Phone, DoB, Email, Address, Role)

Account (**UserName**, Password, *IdUser*)

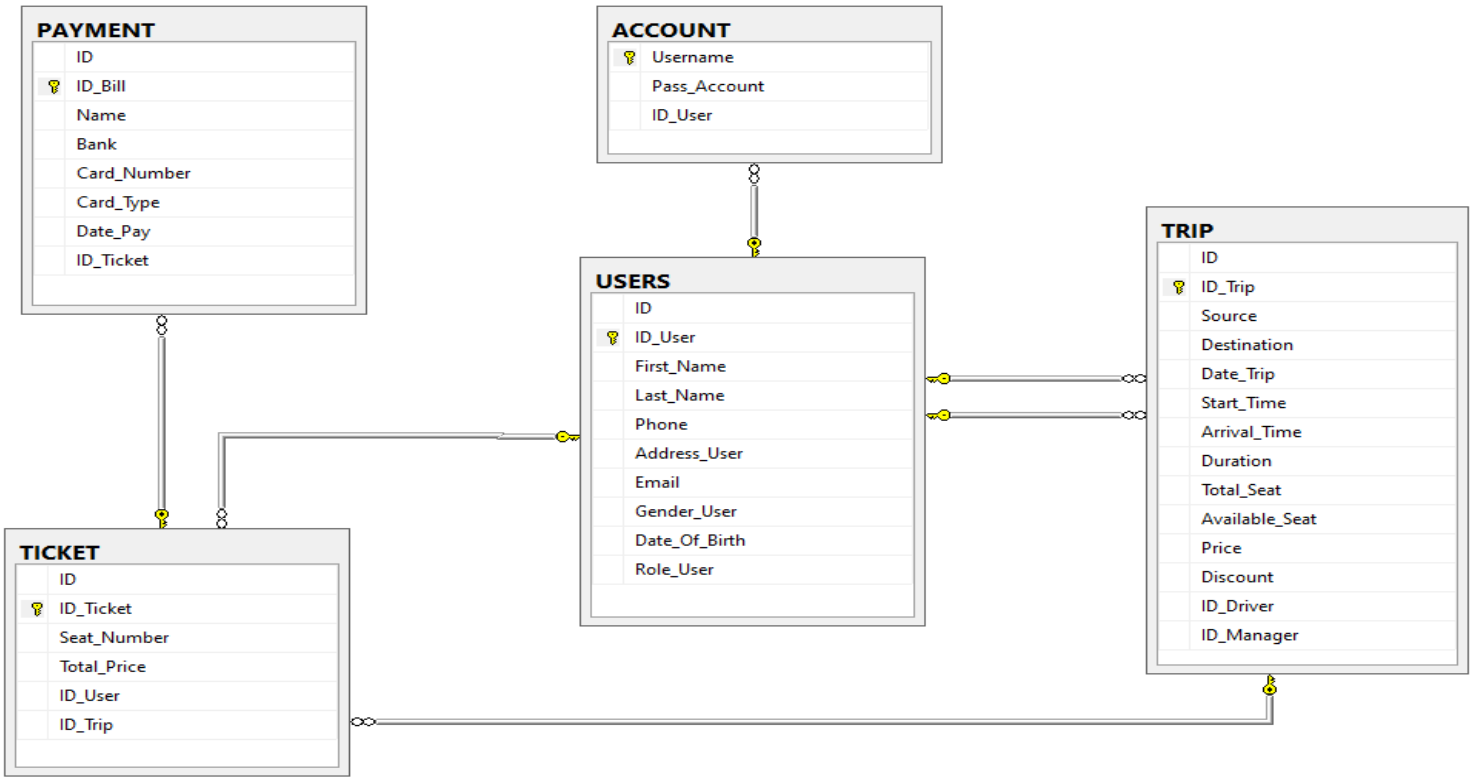
Trip (**IdTrip**, Source, Destination, TotalSeat, AvailableSeat, StartTime, ArrivalTime, Price, Discount, Date, *IdDriver*, *IdManager*)

Ticket (**IdTicket**, SeatNumber, TotalPrice**,** *IdUser*, *IdTrip*)

Payment (**ReceiptNo**, Name, Card_No, Type, Bank, Date, *IdTicket*)

IV. DATABASE STRUCTURE

1. Database Diagram



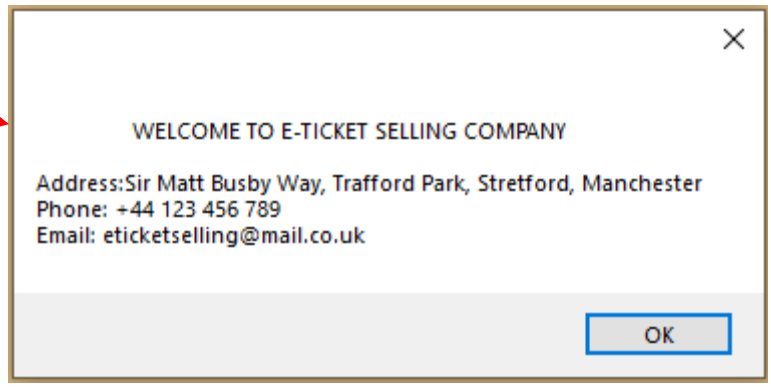
Database Diagram of Coach Ticket Selling

2. Explanation

TABLE	FUNCTION
ACCOUNT	It save all account of the system includes username, password and ID User reference to User information.
USERS	It contains all personal information of User with distinguish by ID User. Each user has its own role in using different functions of the system.
TRIP	It contains all information of Trip with distinguish by unique ID Trip. It has 2 foreign key ID Driver to manage who drive this trip and ID Manager to manage who modify this trip and both key reference to User.
TICKET	It contains information of User, Trip, Total price and Seat number. There are 2 foreign key: ID User to get information of User who bought this ticket and ID Trip to get information of the Trip that was booked by User.
PAYMENT	Save the transaction information and reference to Ticket by using ID Ticket. That will get the Ticket information that was paid by User.

V. EXECUTION

1) Main Frame



When open the application, the Main Frame GUI will appear interact with user.

There are 3 buttons corresponding 3 functions that user has to be done before starting buying ticket.

Login: Login to access this program.

Sign up: If the users do not have any account. Sign up to before login the app.

Contact: Show the message box to announce the information of this company.

2) Login

Input user's username

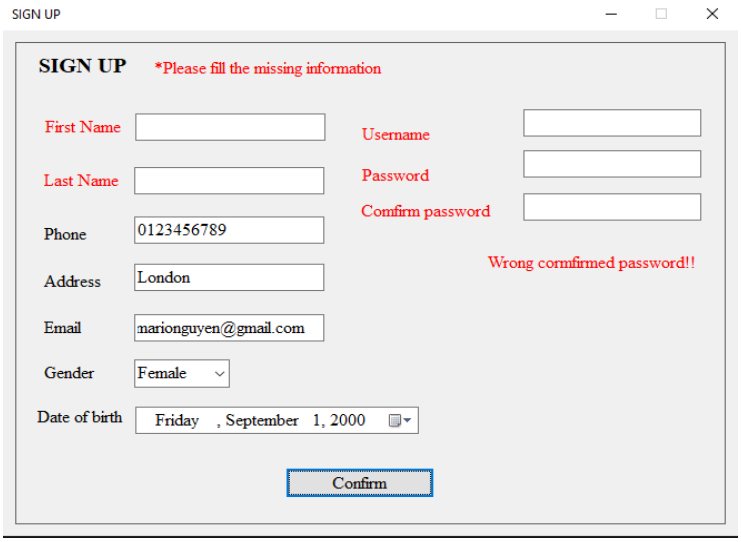
Input user's password

When click Log in button in Main Frame, the Sign in UI will appear and the users input their account to access.

Then push Login button to login

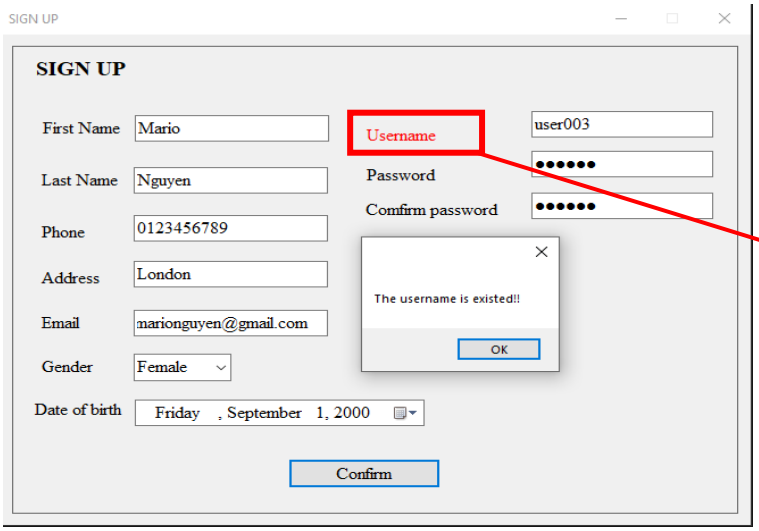
The system will check account is valid or invalid. In order to prevent SQL Injection, we query user name and password by using Procedure and transmit parameter @pass and @username

3) Sign Up



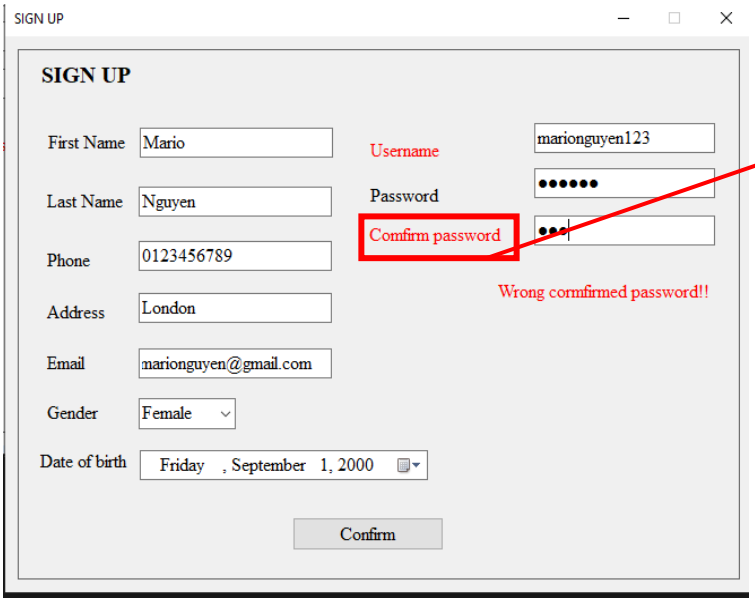
When Users do not have an account. They have to sign up. The UI will appear and requires the users to input their information includes the username and password of the account.

If they confirm without filling all information. An announcement will appear and the labels of missing information will change color to warn.



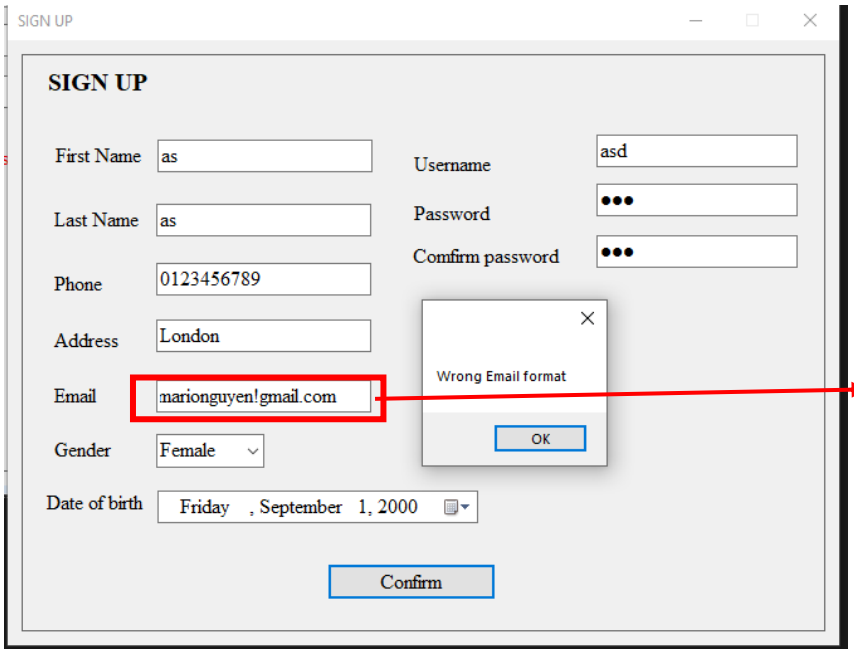
After pushing the confirm button, the system will check all conditions.

In this case, the announcement will appear when the username had been registered by another user.



A special feature in this confirm password is the system will check each character be input by users. It is easy for users to recognizes the wrong character in their confirm password.

If there is a wrong character is inputted. The announcement to notify users that they have input a wrong character. After deleting this wrong character, the announcement will disappear. Confirm password is correct when the label “Confirm password” is black

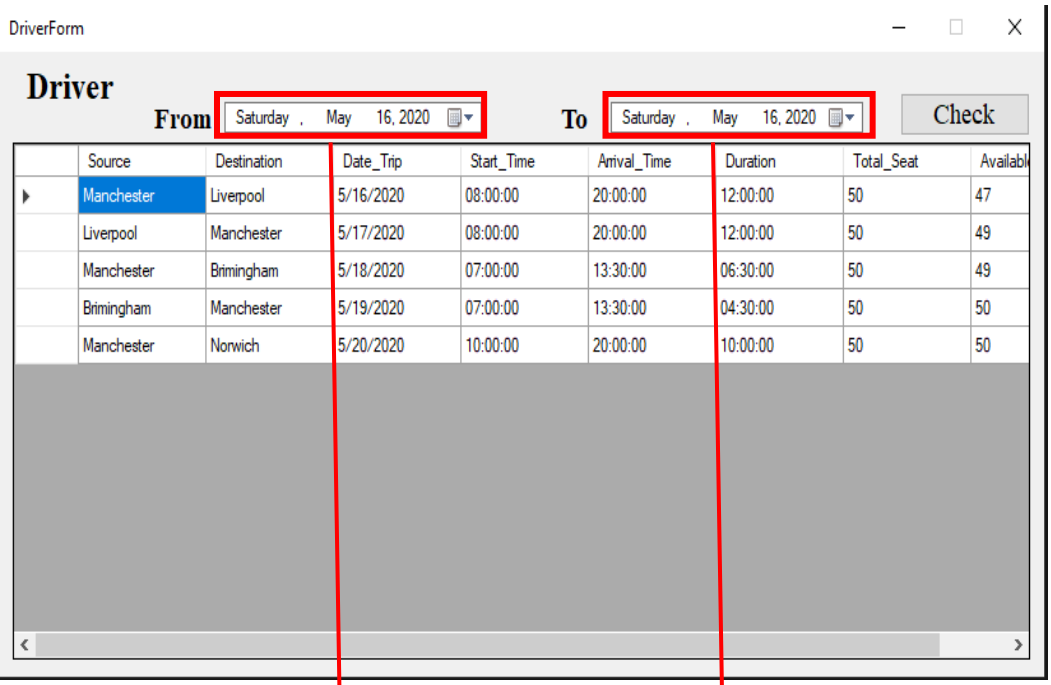


One more thing, the system also check the format of the email like character valid or invalid when users input their email.

Invalid email: there is not '!' in the email. Its has be '@' character.

If all check case pass, sign up successfully, the system will insert their information into database to store.

4) Driver



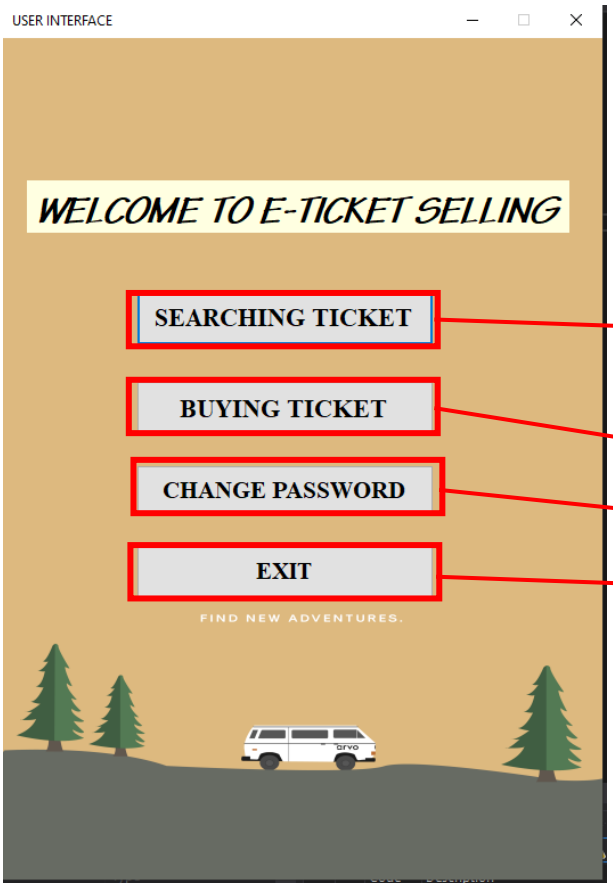
Choose start date

Choose end date

After login successfully, the system will check the role of the account. If the role is Driver, this UI will appear to search the trip that has to drive in from a date in "From" to date in "To"

Then push Check button, a satisfied list trip has appeared.

5) User



This is the main GUI of users.

There are 4 main actions for users to interact with application.

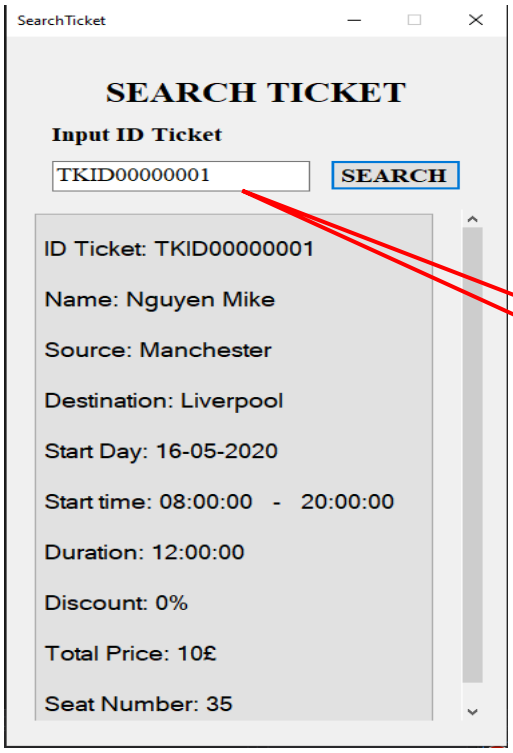
Searching Ticket: The users can find their ticket information by using their ticket's ID.

Buying Ticket: Buying ticket by app.

Change Password: Change password

Exit: Exit the program.

a. Searching Ticket:

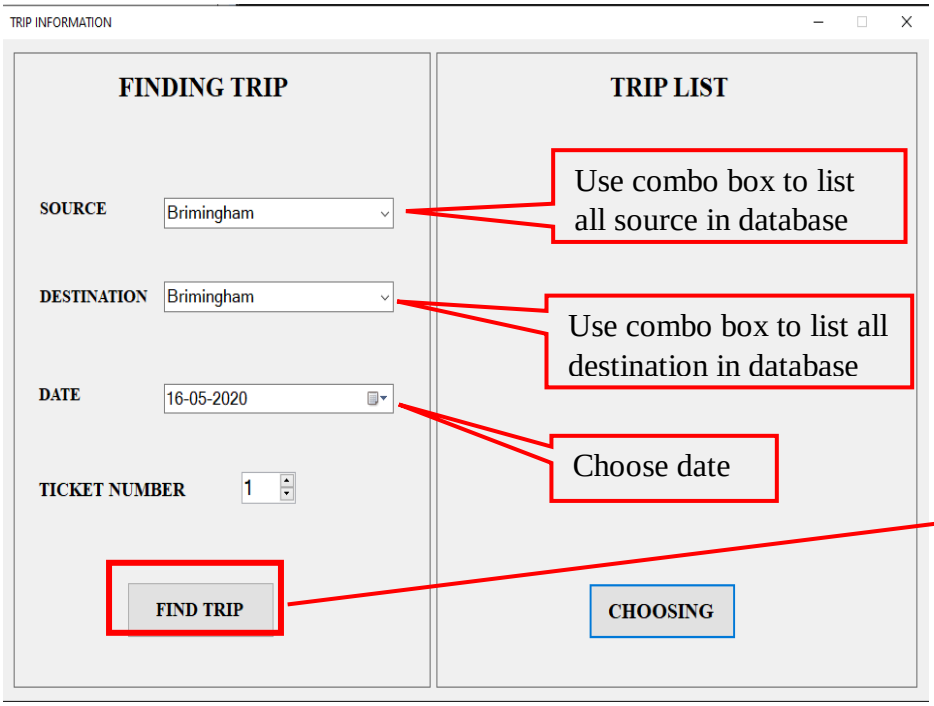


Search Ticket UI will appear when users click on SEARCHING TICKET button.

They input their ID Ticket to view ticket information when they miss some information.

Input ID Ticket

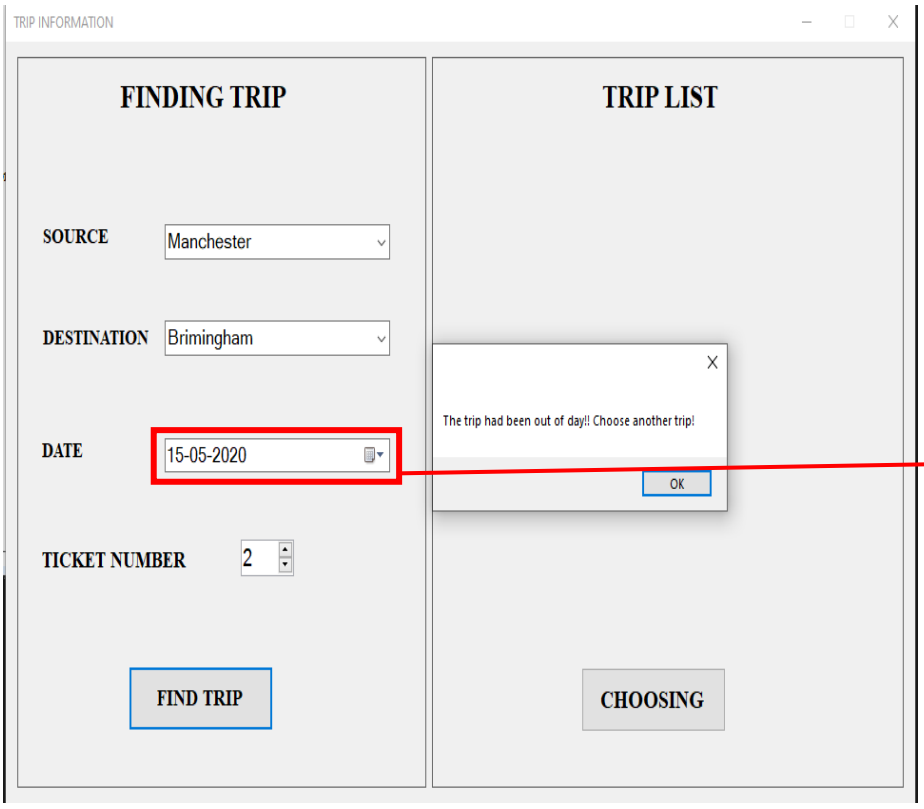
b. Buying Ticket



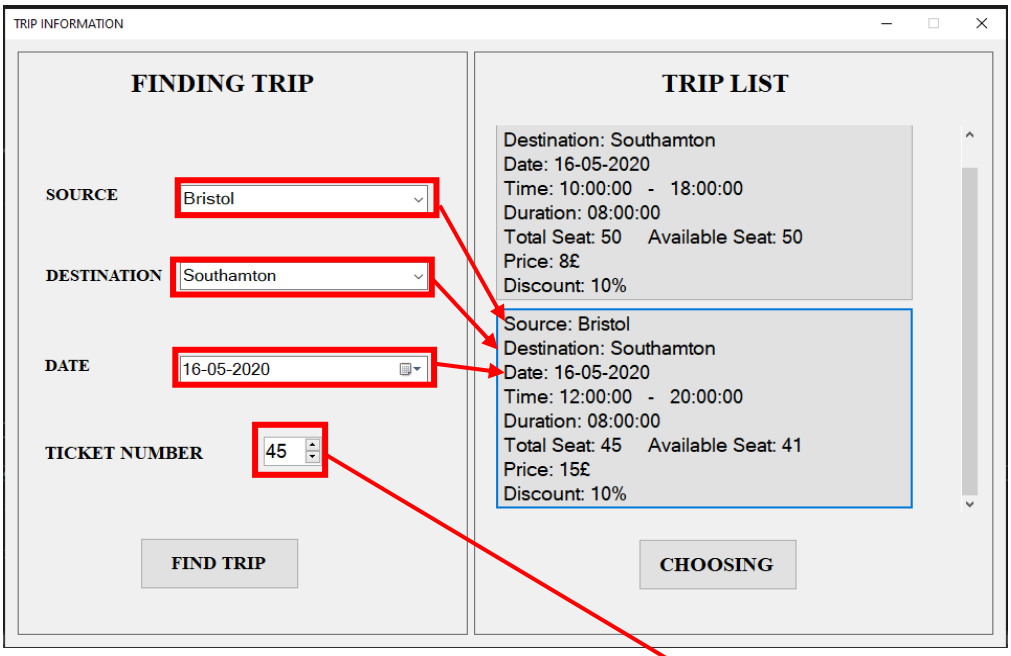
This is the Buying ticket GUI for buying ticket.

Users have to select the source, destination, date to find the trip.

The system will list all trip that is satisfied with source, destination and date.



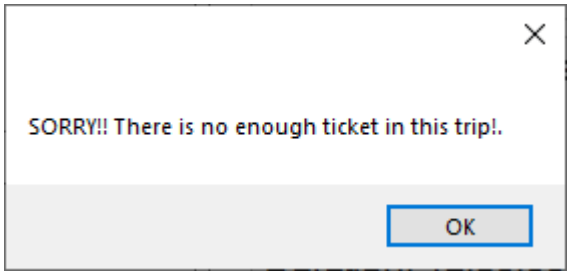
Besides the system check the trip from source to destination on a particular date to output the list, it also checks the date of the trip is valid or not. If the user would like to buy a ticket on the date before today or it means the trip has been done. An announcement will appear to notice that.



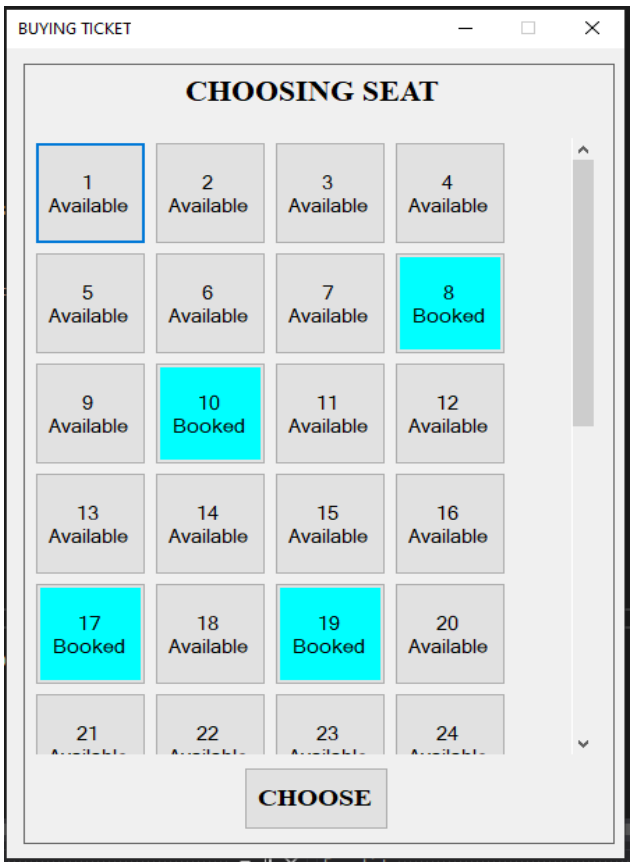
If all information passes the check cases. A list of trips will be shown on the right side.

The sorted trip information will appear for users to choose from.

Then they have to select a number of the ticket that they would like to buy before pushing the "CHOOSING" button. If not, the number of tickets is 1.



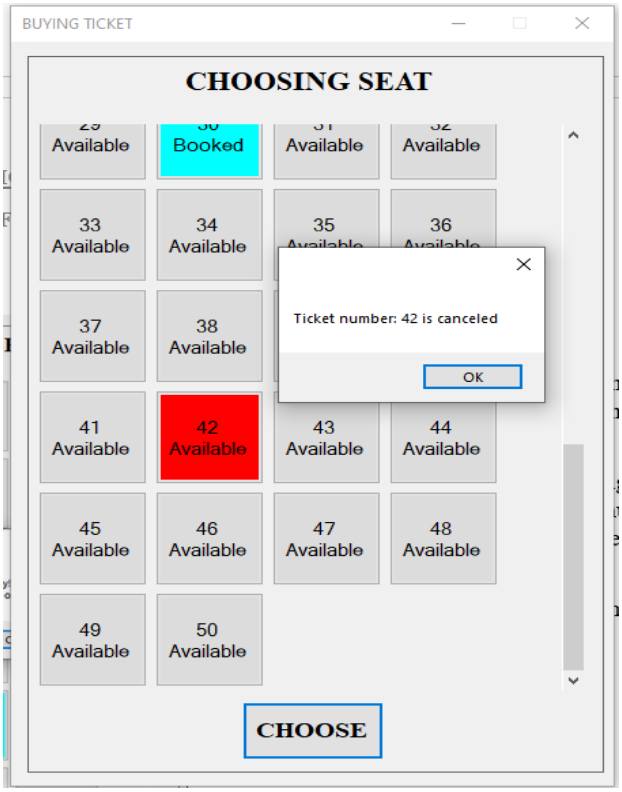
In this case, if the users choose the number of ticket (45) is larger than available seat (41). The announcement will appear and they can not choose this trip.



After the system check, the number of tickets, date, source, the destination for choosing the trip is satisfied. A list of the seat appears.

The number of appeared seats is the same as the number of seats on this trip. And all booked seats are marked by another color with status "Booked" which means that the users can not choose this seat.

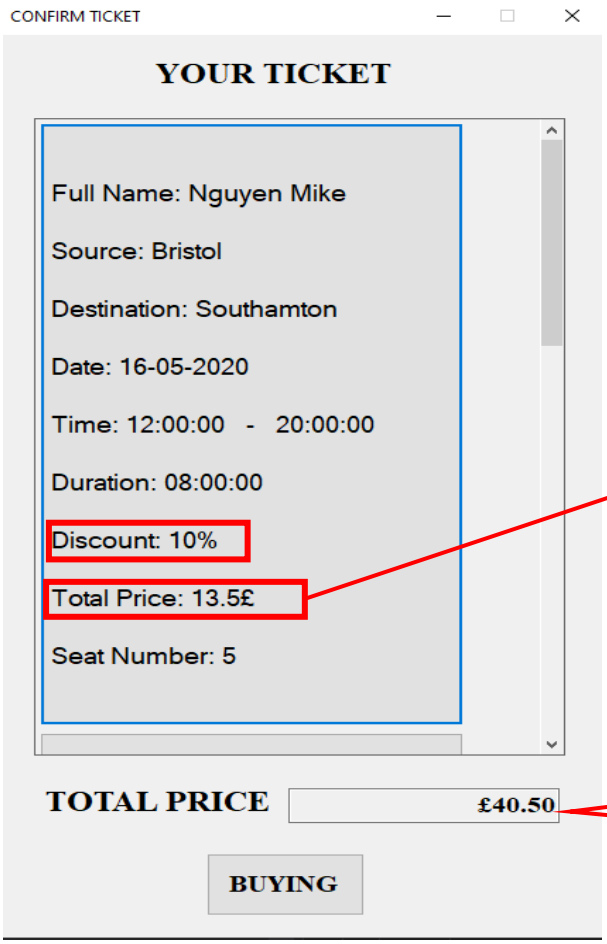
They must choose the "Available" seats.



The users have to choose the number of seats corresponding to the number of tickets.

When a seat is chosen, it changes the color to help users know the number of their seat that they have been chosen.

If the users choose this seat again, this seat is canceled out of their list seat and back to original status.



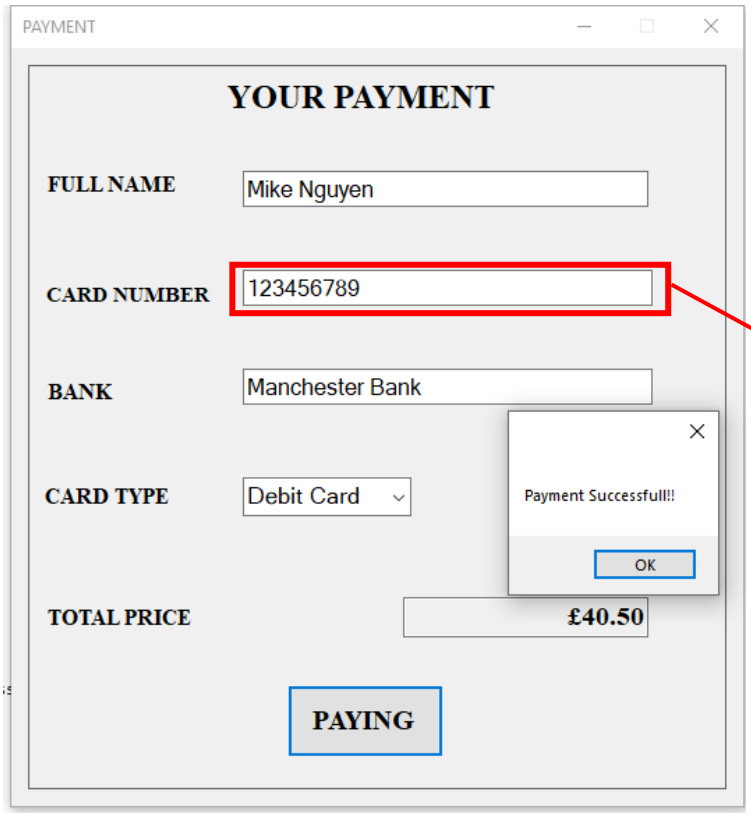
After completing choosing seats for their tickets. A list of tickets will appear with important information for users could check it again.

Specially, the total price in ticket is not a price is appeared in trip. This is the finally price includes discount (10%)

The **total price** of all tickets that they have already bought.

This is the sum of each total price in list tickets.

Users can not write anything in textbox "Total price"



Then the users have to fill all information to pay for this ticket. Similar withy Sign Up features, if there is exist a blank is not filled yet, the announcement will appear and cannot pay.

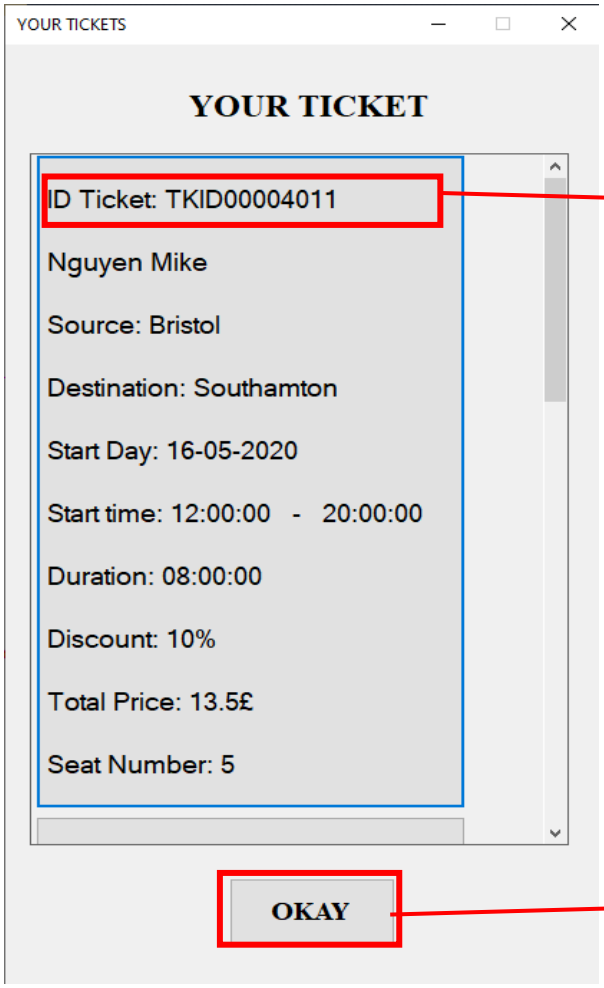
In Card Number, the system set up the feature that users cannot input any character except number.

Although buying more than 1 ticket, the users just pay **only one time for all of them** but the data will be stored in the database by each bill for each ticket. It means there are (3) bills corresponding to (3) tickets.

TKID00004011	5	13.5	UID00000001	TID00001021
TKID00004012	34	13.5	UID00000001	TID00001021
TKID00004013	40	13.5	UID00000001	TID00001021

PID00005002	Mike Nguyen	Manchester Bank	123456789	Debit Card	2020-05-16	TKID00004011
PID00005003	Mike Nguyen	Manchester Bank	123456789	Debit Card	2020-05-16	TKID00004012
PID00005004	Mike Nguyen	Manchester Bank	123456789	Debit Card	2020-05-16	TKID00004013

- In the databases, the data will be stored like this.
- There are 3 tickets with different ID, seat number.
- That lead there are 3 bills corresponding to 3 tickets with different ID and ID Ticket.



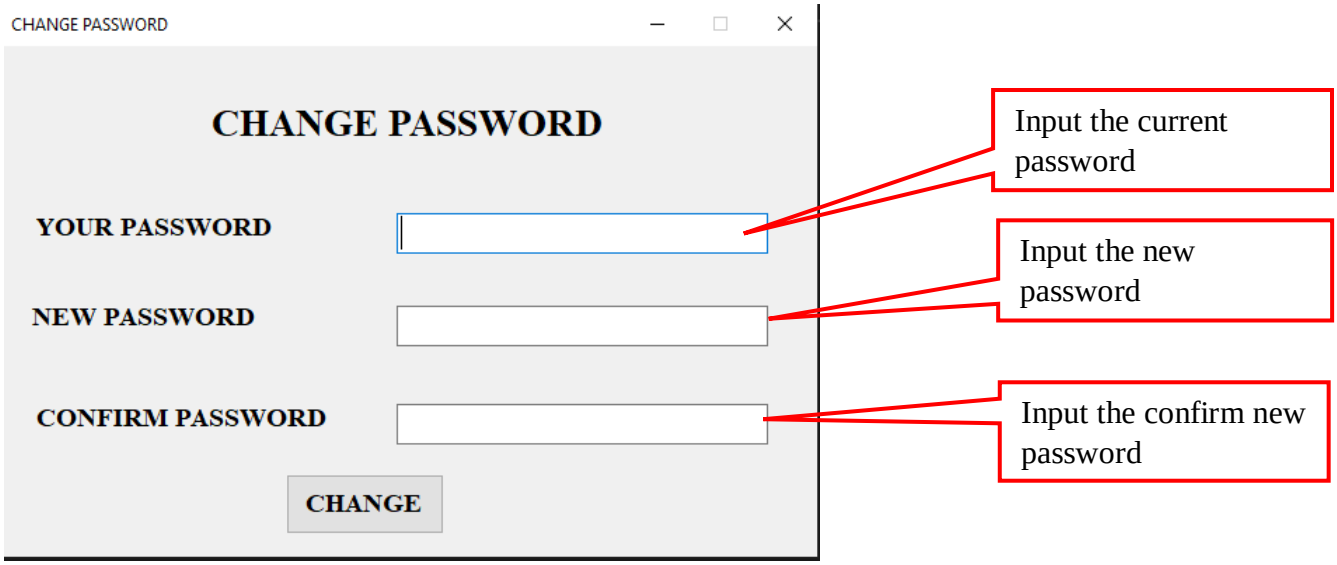
Finally, after finish payment, each ticket will have **own ID**.

A list of user's ticket with ID will appear again to remainder the users.

They can use this ID to find their ticket on the system if they forget.

Push OKAY button to finish the process and back to main User' GUI.

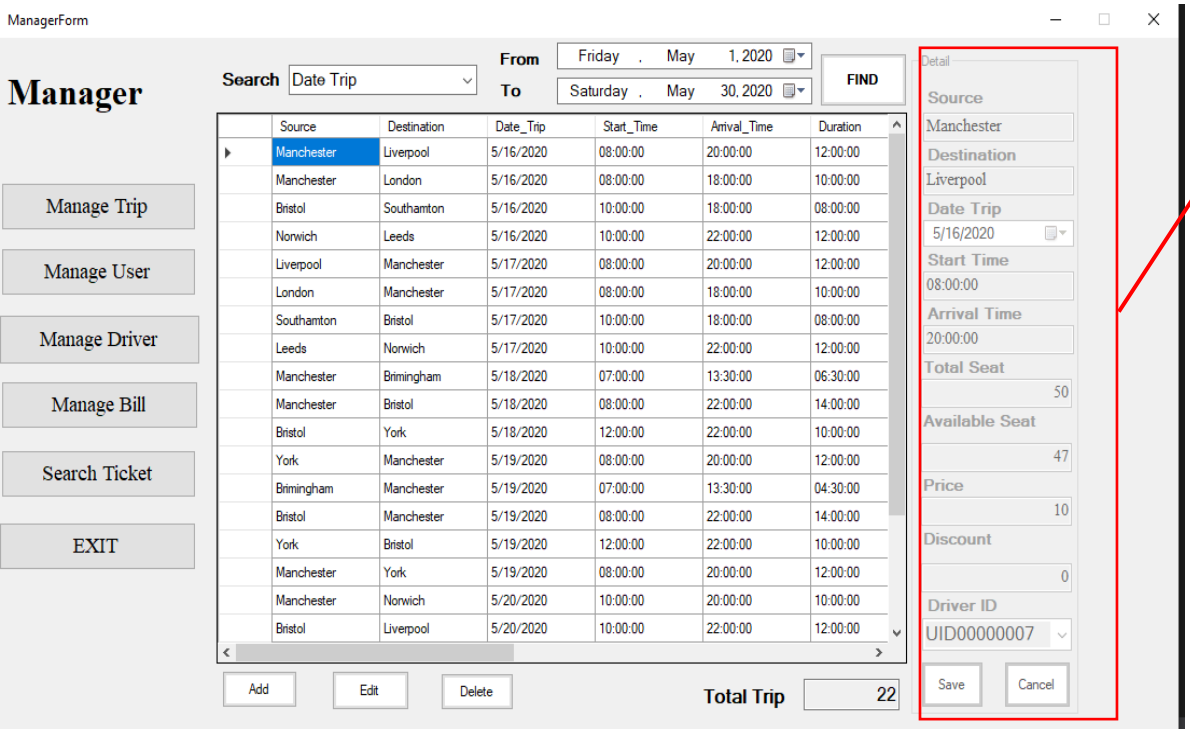
c. Change Password



d. Exit

Exit to close the application.

6. Manager



Detail information area.

This helps managers easier to track the data they are choosing to view. After touching a cell in the table, all the data in the row will be filled into the detail group box.

This is the windows when manager logged in. They have 6 options:

Manage Trip: View trip, add, edit, search trip by many conditions and sum up the total trips by each of conditions.

Manage User: View and search the list of users by particular condition.

Manager Driver: View driver, add, edit, search driver by many conditions and sum up the total driver by each of conditions.

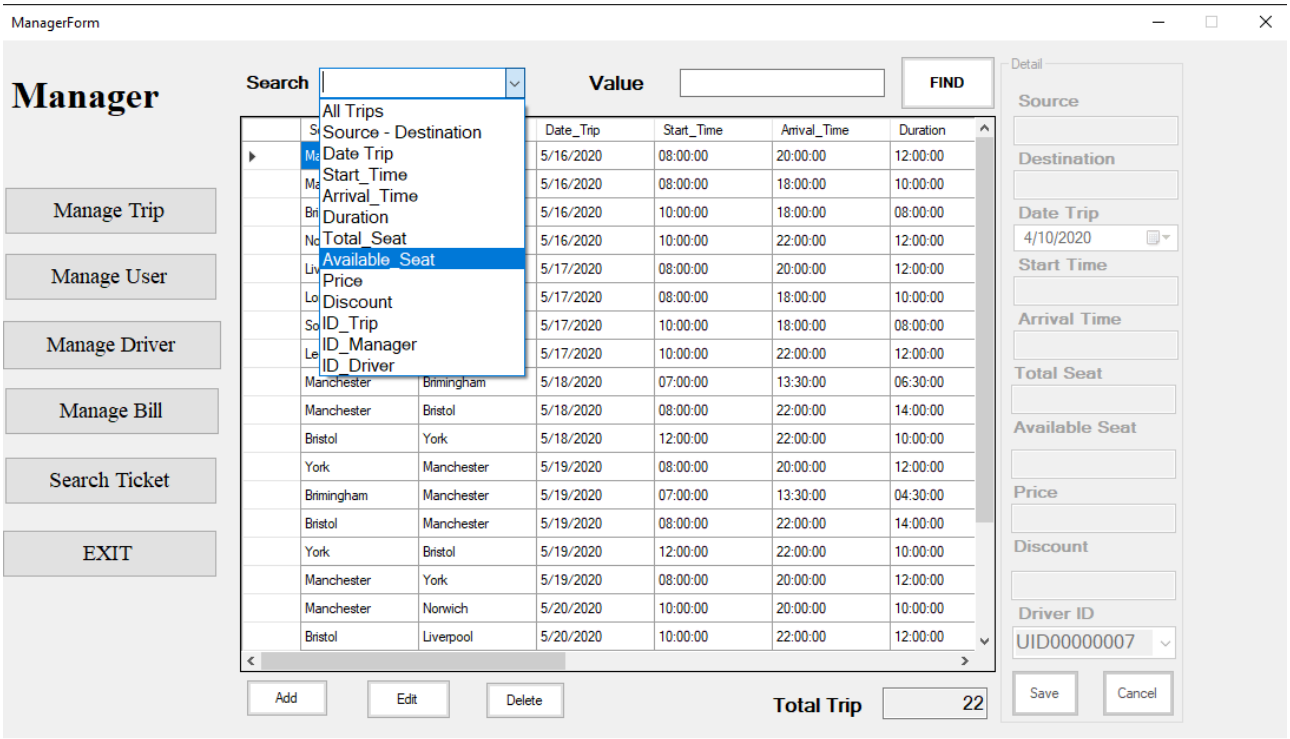
Manage Bill: Manage revenue/amount of bills of some of trip in a period time. (a month, a year,..)

Search Ticket: Searching ticket by ID Ticket to get the ticket of users.

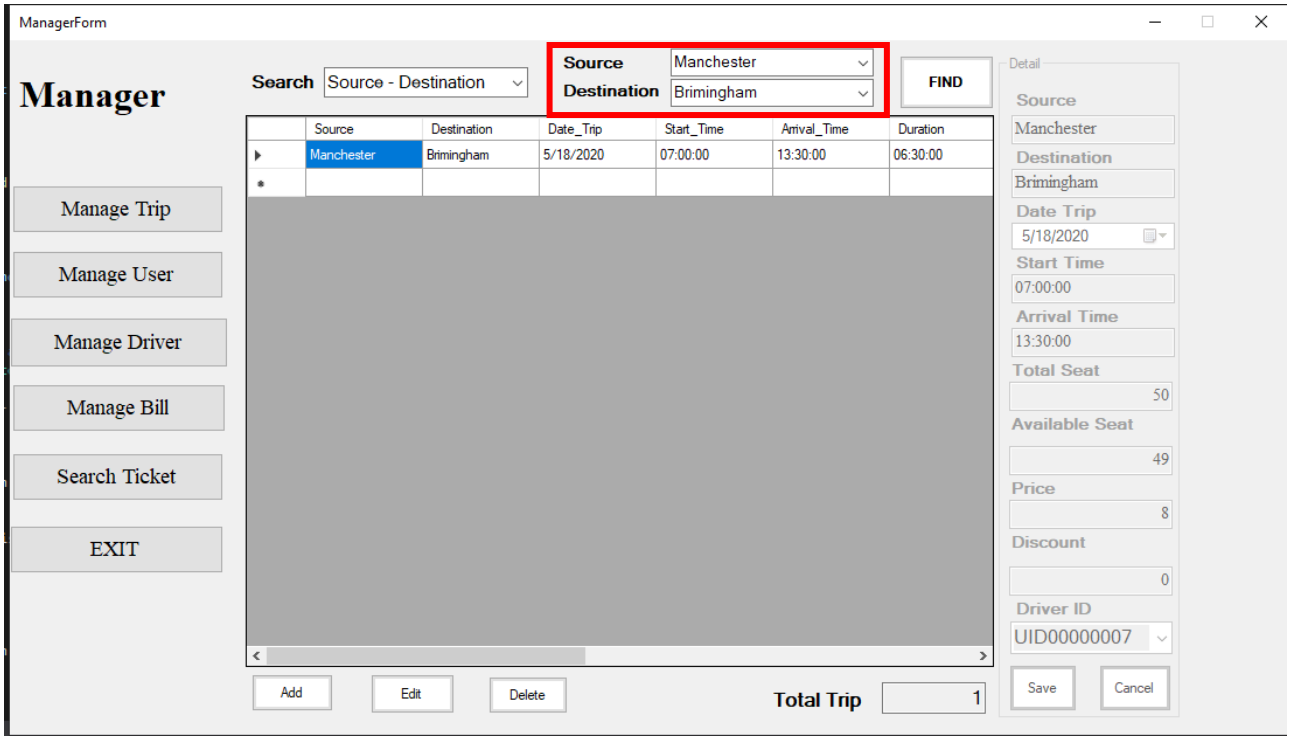
Exit: Close the program.

A right panel will appear and allow manager to manage it.

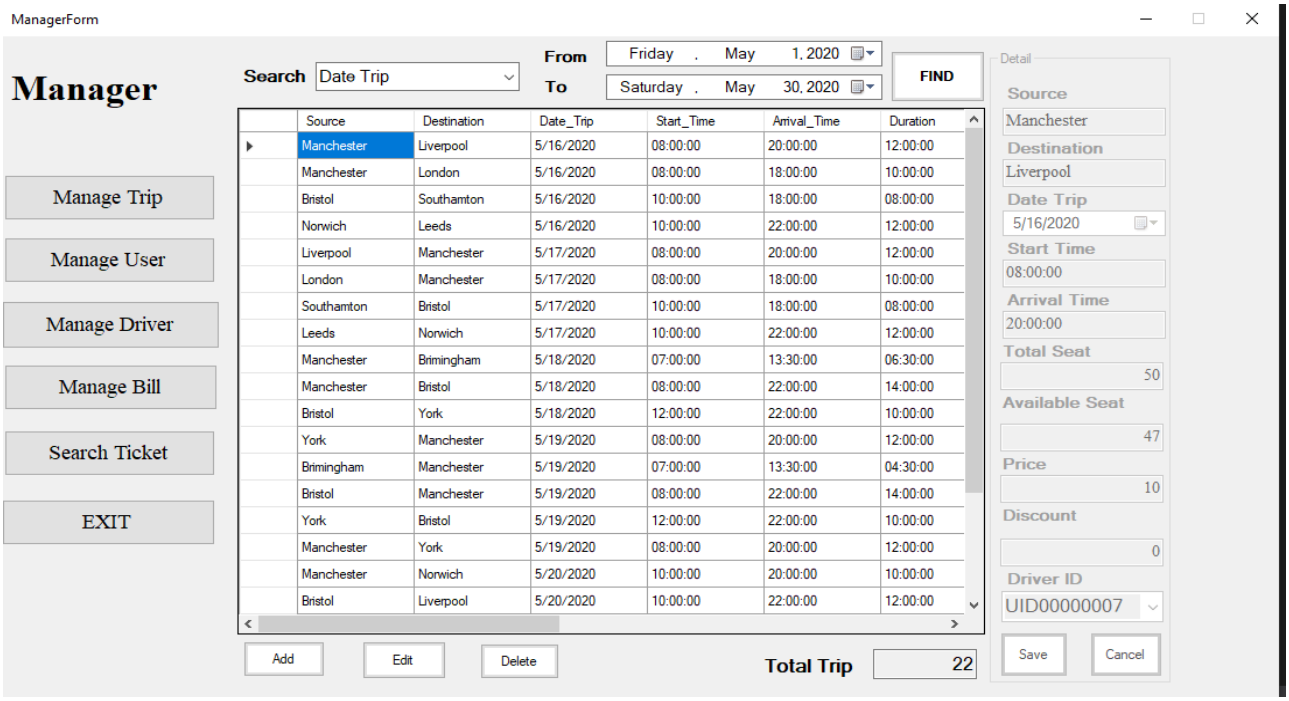
a. Manage Trip



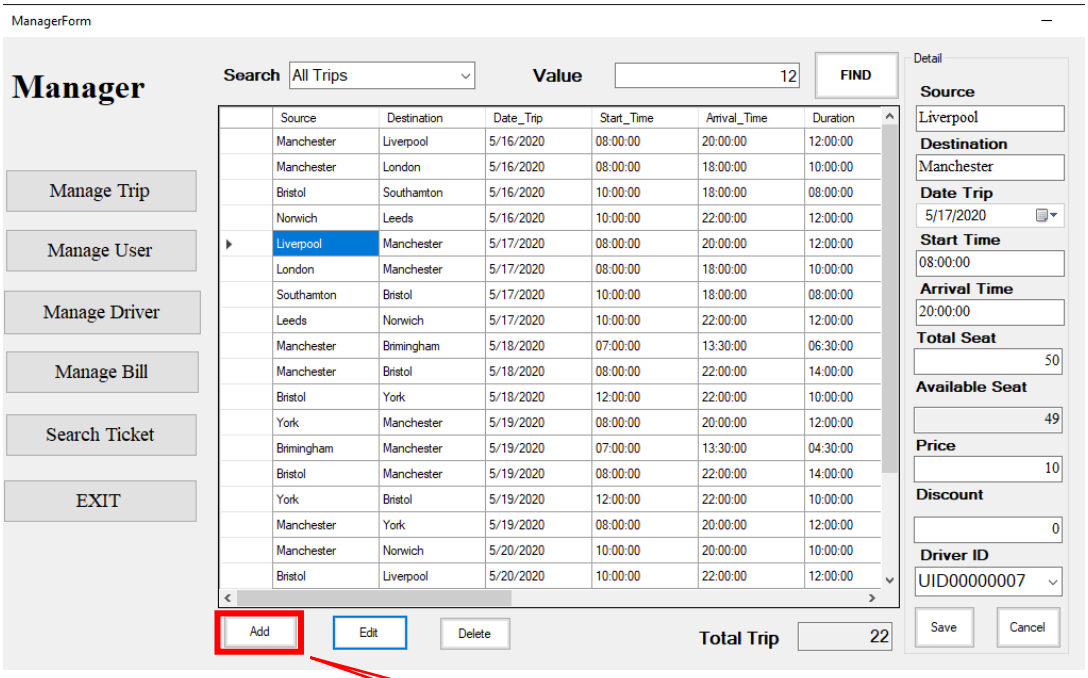
Manager can search anything in the combo box with at least one keyword. If they are blank, the form will refresh the database



Searching by Source and Destination do not use key value, a pair of combo box Source and Destination will appear for Manager to select.



Manager can search Trip in a period time by using Search bay Date Trip form this date to that date.

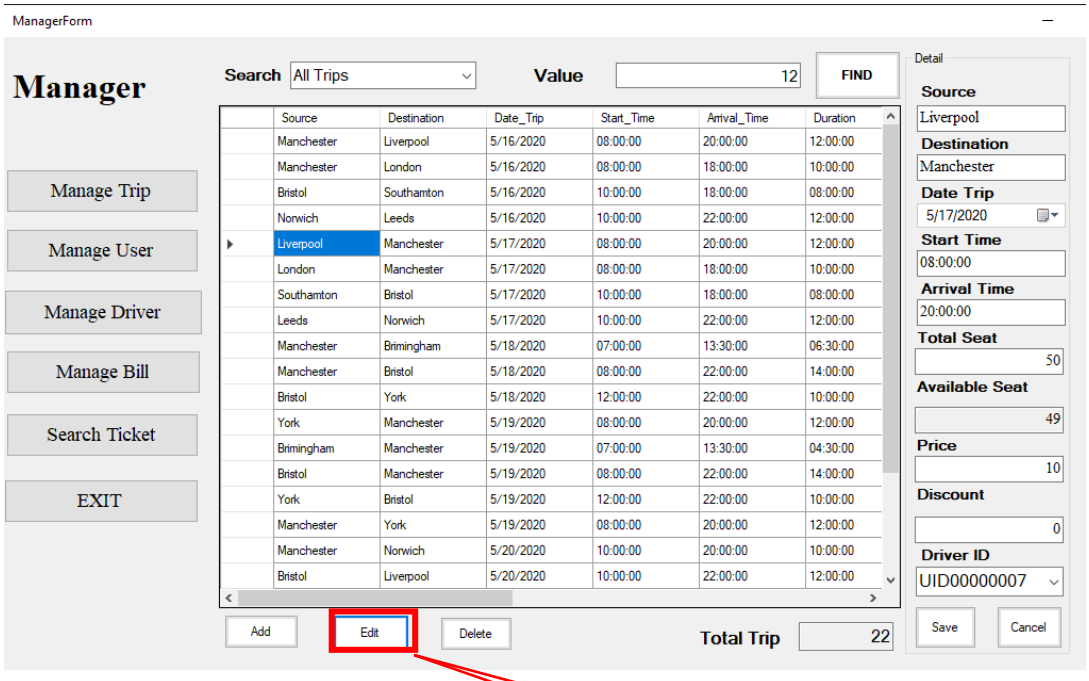


Add button

By clicking Add, the right panel will be enabled.

Manager will be required to input Details of the adding trip.

Detail can be referred to the table. After filling data, the system will check if the data is blank or there is any duplicated data.



Edit button

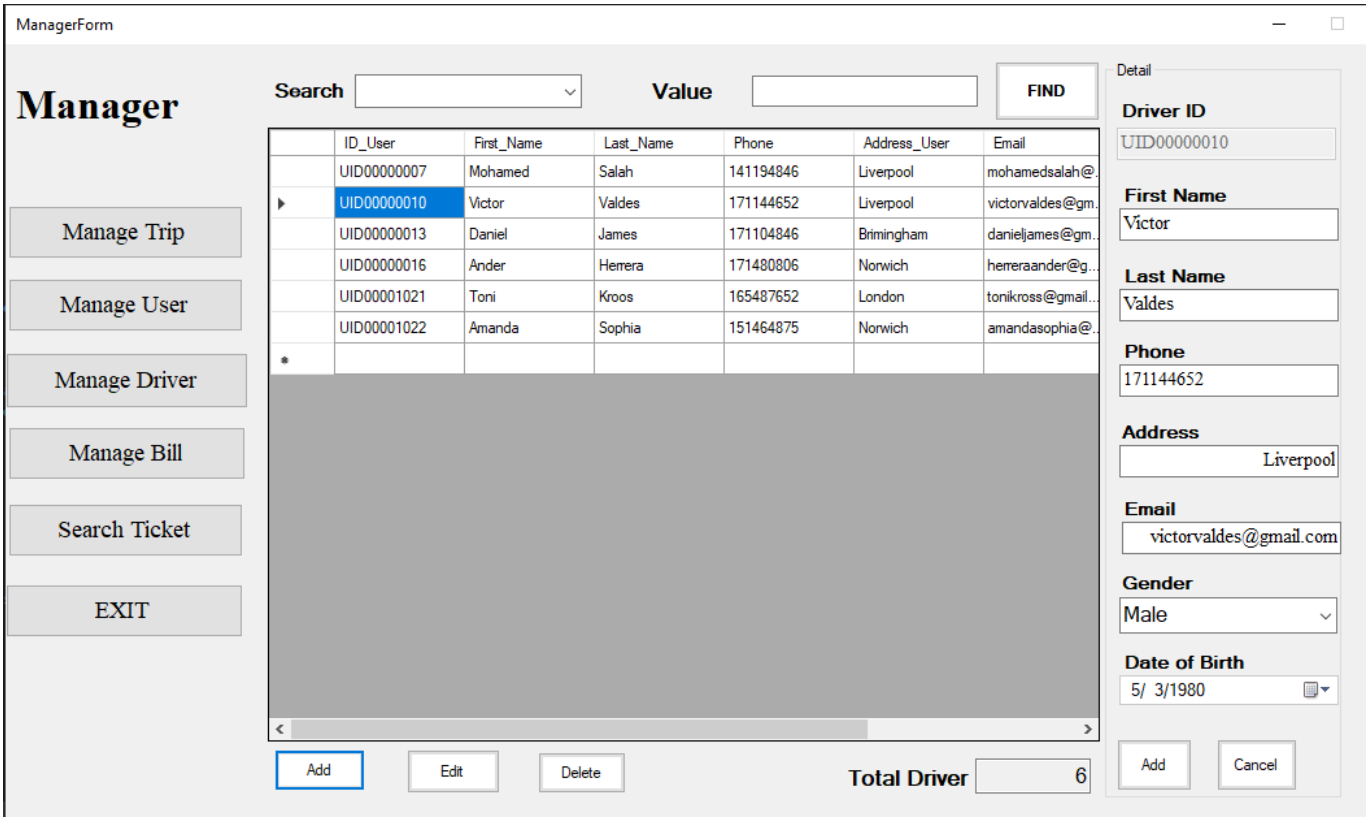
This feature only available when manager choose a cell to edit in the table.

Managers are able to edit any information in the right panel.

When managers change the driver ID or the time, the system will check if it is valid or not after they hit the save button. If not, the system will show an error. If yes, the data will be saved, and the table will be refreshed. The system also allows managers to input number only in price and discount text box.

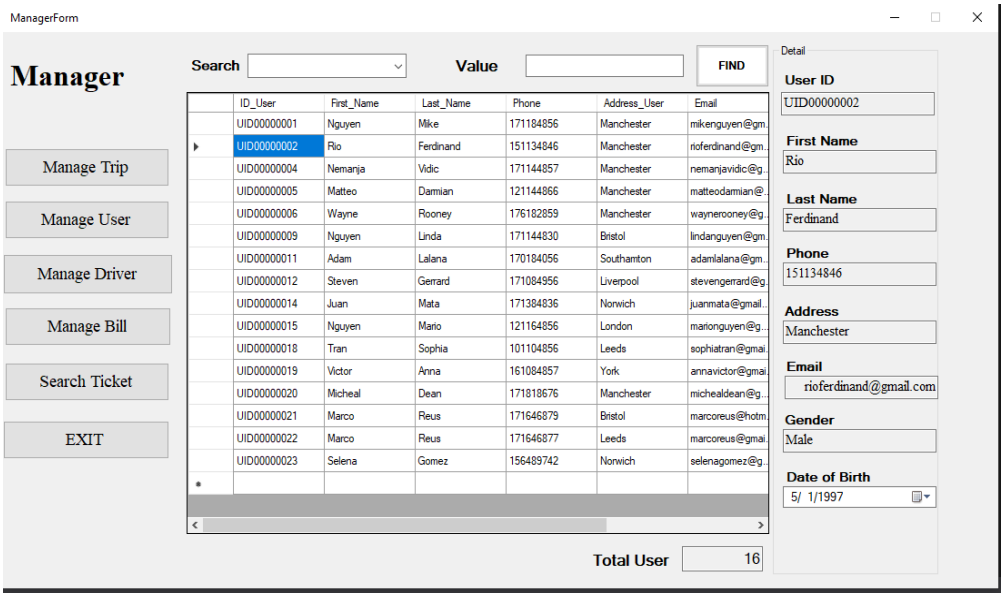
Extra Feature: In add trip, edit trip, choosing Driver for the trip is also important. It is checked carefully. In fact, with the long – time trip, the driver has to drive at most 2 trips per day and usually 1 trip. And 2 adjacent trips must be logical. It means if the first trip is Manchester to London, the second trip must-have source is from London to another. It is impossible when after the first trip destination is London and the Source of the second trip is Norwich like this. It is inconvenient for Driver to move to Norwich to start the trip. It is important to check this driver's position and they have the trip on this day or not before assigning.

b. Manage User



Allow manager to search for Drivers' information. After managers click the Manage Driver button, a form will appear on the right panel. Other functions are the same to the Trip form.

c. Manage Driver

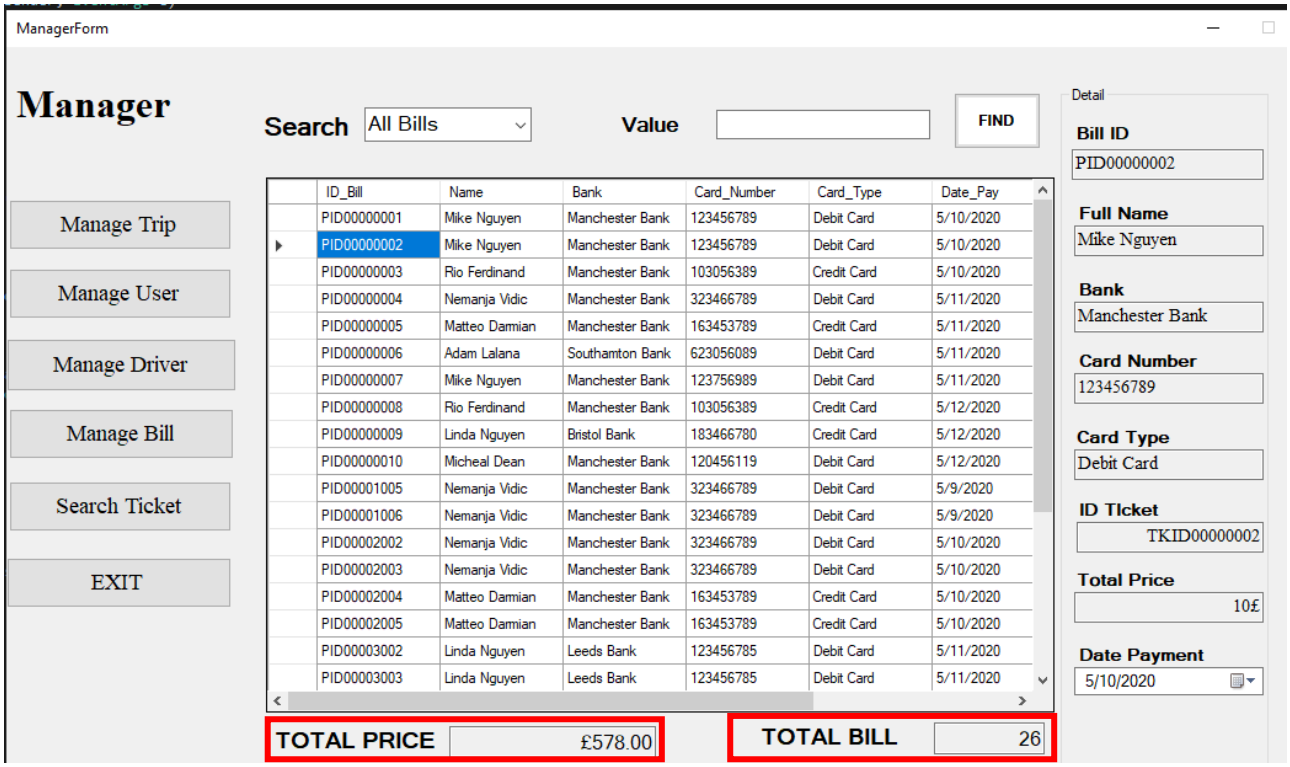


Allow manager to search for Users' information.

After managers click the Manager User button, a form will appear on the right panel.

Manager just can view user information, not add, edit,..

d. Manage Bill



Allow manager to search for Bill' information.

After managers click the Manager Bill button, a form will appear on the right panel.

Manager just can view Bill information, not add, edit,..

This is similar with Manage User form and in Bill Form, there are 2 new things is total price and total bill corresponding to the result of searching.

Extra feature✨: The system manages to save each form after manager load another form. This reduce the time changing between forms, and also help manager to edit information easier since all of the data in the detail panel is saved.

VI. QUERY COMMAND

a. Count the number of tickets that were sold in a particular month in a particular year

```
DECLARE @Month INT
DECLARE @Year INT
SET @Month = 5
SET @Year = 2020
```

$\pi_{COUNT(*)}(\sigma_{MONTH(Date\_Pay)=@Month\ AND\ YEAR(Date\_Pay)=@Year} PAYMENT)$

	Number of Tickets
1	26

b. Average revenue in each month of a particular year

```
DECLARE @Year INT
SET @Year = 2020
```

$MONTH(PAYMENT.Date\_Pay)\ \mathcal{G}_{AVG(TICKET.Total\_Price)}(\sigma_{YEAR(Date\_Pay)=@Year\ AND\ TICKET.ID\_Ticket = PAYMENT.ID\_Ticket}(TICKET \times PAYMENT))$

	Month	Avarage Revenue
1	5	22.135

c. Find all the trips have booked in each month of a particular year and order to descending times of booking

```
DECLARE @Year INT
SET @Year = 2020
```

$MONTH(PAYMENT.Date\_Pay), TRIP.ID\_Trip\ \mathcal{G}_{COUNT(*)}(\sigma_{YEAR(Date\_Pay)=@Year\ AND\ TICKET.ID\_Ticket = PAYMENT.ID\_Ticket\ AND\ TRIP.ID\_Trip=TICKET.ID\_Trip}(TICKET \times PAYMENT \times TRIP))$

(ORDER BY (COUNT(*)) DESC)

	Month	ID_Trip	Times
1	5	TID00001021	7
2	5	TID00000007	4
3	5	TID00000001	3
4	5	TID00000020	2
5	5	TID00000022	2
6	5	TID00000008	2
7	5	TID00000009	1
8	5	TID00000011	1
9	5	TID00000012	1
10	5	TID00000019	1
11	5	TID00000002	1
12	5	TID00000005	1

d. Find ID of users that booked at least 10 trips in a month

$\pi_{TICKET.ID\_User,datename(MONTH,TRIP.Date\_Trip),COUNT(*)}(TRIP \bowtie TICKET)$(GROUB BY TICKET.ID_User, TRIP.Source, HAVING COUNT(*) > =10)

e. Find ID of users, name of source place that they have booked most recently, number of them, and the same for the destination.

$\pi_{TICKET.ID\_User,TRIP.Source,COUNT(*)}(TRIP \bowtie TICKET)$ (GROUB BY TICKET.ID_User, TRIP.Source)
$\pi_{TICKET.ID\_User,TRIP.Destination,COUNT(*)}(TRIP \bowtie TICKET)$ (GROUB BY TICKET.ID_User, TRIP.Destination)

	ID_User	Source	(No column name)
1	UID00000001	Brimingham	1
2	UID00000001	Bristol	5
3	UID00000001	Manchester	2
4	UID00000002	Liverpool	1
5	UID00000002	Manchester	1
6	UID00000004	Bristol	2
7	UID00000004	Manchester	3
8	UID00000004	Southamton	2
9	UID00000005	Manchester	1
10	UID00000005	Southamton	2
11	UID00000005	York	1
12	UID00000009	Leeds	2
13	UID00000009	York	1
14	UID00000011	Bristol	1
15	UID00000020	Brimingham	1

	ID_User	Destination	(No column name)
1	UID00000001	Liverpool	2
2	UID00000001	Southamton	5
3	UID00000001	York	1
4	UID00000002	Brimingham	1
5	UID00000002	Manchester	1
6	UID00000004	Bristol	2
7	UID00000004	Liverpool	3
8	UID00000004	Southamton	2
9	UID00000005	Bristol	2
10	UID00000005	London	1
11	UID00000005	Manchester	1
12	UID00000009	Brimingham	1
13	UID00000009	Norwich	2
14	UID00000011	York	1
15	UID00000020	York	1

f. Find all trips from one place to another on a date with a total duration of less than 12 hours. Search only for trips that have one-stop. Needed information will be given by the customers.

$$\rho_{T1(Destination,Date\_trip,duration)}(\pi_{Destination,Date\_trip,\text{DATEPART(HOUR,Duration)}*60+\text{DATEPART(MINUTE,Duration)}}(TRIP))$$

$$\rho_{T2(Source,Date\_trip,duration)}(\pi_{Source,Date\_trip,\text{DATEPART(HOUR,Duration)}*60+\text{DATEPART(MINUTE,Duration)}}(TRIP))$$

$$\pi_{\text{T1.DestinationT2.Source,T1.duration+T2.duration}}(\sigma_{\text{T1.Date\_Trip = T2.Date\_Trip AND T1.Destination = T2.Source AND T1.duration+T2.duration<12}*60}(T1 \cup T2))$$

g. Find the driver who drives the most trips in a month.

$\pi_{\text{TOP 1 USER.ID\_User, USER.First\_Name, USER.Last\_Name, COUNT(*)}}(TRIP \bowtie USER)$ (GROUB BY USER.ID_User, USER.First_Name, USER.Last_Name, ORDER BY COUNT(*) DESC)

	ID_User	First_Name	Last_Name	(No column name)
1	UID00000016	Ander	Herrera	8

h. Change the Password

```
CREATE PROC USP_UpdateAccount

@idUser NVARCHAR (100), @passWord NVARCHAR (100), @newPassWord NVARCHAR(100)

AS

BEGIN

    DECLARE @isRightPass INT

        π@isRightPass=COUNT(*)(σID_User=@idUser AND Pass_Account=@passWordACCOUNT)

    IF (@isRightPass = 1)

    BEGIN
        UPDATE dbo.ACCOUNT SET Pass_Account = @newPassWord WHERE ID_User =
    @idUser

    END

END
```

VII. PLAN WORIKING

TIMELINE OF DONG PROJECT (15/3 - 16/5) TOPIC 8: COACH TICKET SELLING					
WEEK	**WORKING**		**CODING**	**DEADLINE**	**CHECK**
15/3 - 22/3	Doing Proposal		**Thanh, Cong, Man**		**DONE**
23/3 - 29/3	Create Databases	User, Account, Trip	**Thanh**	**28/3**	**Man**
		Payment, Ticket	**Cong**		**Thanh**
		Check linked in whole database	**Man**		**Cong**
		Optimize database			
	Meeting, discussion and decided about general background of the application. How many functions in this, included functions of user and manager		**Thanh, Cong, Man**		
30/3 - 5/4		Design the main display of **User, and Driver, Customer's Ticket Form and Payment Form**	**Thanh**	**4/4**	**Man**
		Design the main display of **Trip Information Form. UI of Manager**	**Cong**		**Man**
		Design the main display of **Login Form, Sign Up Form, Main Form of Manager**	**Man**		**Thanh**
		Function of **Login Form, Sign Up Form**	**Man**		**Thanh (Check**

6/4 - 12/4	Querying the database using C# Database Connectivity			**11/4**	**and test)**
		Function of **Main UI, User UI, Driver Form**	**Thanh**		**Cong**
		Basic function of Manager Form: UI of Manager	**Cong**		**Man (Check and fix bug)**
13/4 - 19/4		Function of Driver and **Trip: Find Trip, Check valid trip and choose trip.**	**Thanh**	**18/4**	**Man**
		Manager Form (cont): Manage Trip function: Find Trip	**Cong**		**Man**
		Add more feature in Sign Up Form and continue to correct Sign In and Sign Up Form bug.	**Man**		**Thanh**
20/4 - 26/4		Function of Trip: choose seat, check valid seat, check ticket information, pay trip confirm ticket, search ticket.	**Thanh**	**25/4**	**Man (Check and discuss feature)**
		Manager Form (cont): Manage Trip function: Add Trip, Output detail each trip.	**Cong**		**Man (Check and fix bug)**
		Function Manager Form: Optimize manage trip function.	**Man**		**Thanh (Check and test)**
27/4 - 3/5		Function of Manager Form: Manage Bill, combine other function together in Manager	**Thanh**		**Man (Check and test)**

		Form.		**2/5**	
		Function of Manager Form: Manage Driver, Manage User.	**Man**		**Thanh**
		Connect Each Form Together, check to completely applications	**Thanh, Man**		**Cong** **(Run and Test to find bug)**
4/5 - 10/5	Meeting, discussion and decided about the structure, quality of the application, improve the code (if needed). Developed the application with more new features if have new ideas.		**Thanh, Cong, Man**	**9/5**	
11/5 - 16/5	Write the report		**Thanh, Cong, Man**	**16/5**	
16/5 – 22/5 (Extend Deadline)	Add more feature and optimize algorithm in this project.		**Thanh, Cong, Man**		

VIII. CONCLUSION

E-Ticket Selling project is combined with many important skills for the student to improve their knowledge, coding skill, thinking logically, develop problems, and solve them.

While making the project, we made our progress by solving problems. This provided us experiences that will be useful in the future. We came to know that how we can use C# Winform to make an app, create a logical database that is suitable for the project and link it with the programming language. Doing project is known as one of the best ways to learn more algorithms and optimize them.

Some important things that we learned include designing a good program architecture and analyst customer requirements, converting real-life situations into efficient code.

Therefore, besides having a deeper knowledge into Database structure, this project also helps us improve programming ability.