SLIIT

Discover Your Future

BSc (Hons) in Information Technology

Assignment Report

IT1090 – Information Systems and Data Modelling Year 1, Semester II, 2019

Online Train Reservation System

Group ID: MLB_11.1.1_04

Campus: Malabe Campus

	Student Registration Number	Name
1	IT19051130	L.S. Semasinghe
2	IT19031330	H.G.B. Shamendra
3	IT19176598	W.A. Nipun Sandeep
4	IT19054100	W. Malith Pinsara
5	IT19003542	B.A.I. Bhagya
6	IT19802404	Sophinia R.

Contents

Hypothetical Scenario

In an online train reservation system, passengers can reserve tickets using the train company's website. User must enter a valid email, their first name and last name. Only a single passenger can travel using one ticket. Users can be uniquely identified using their email.

Regular users can create a user account. Each user account has a username and a password for login authentication. Regular users can store their credit/debit card details on the system for making reservations faster and easier. They can add a phone number to the account for SMS notifications.

Payments can be done only by credit/ debit cards. One passenger can buy many tickets. For every transaction, a unique transaction ID will be generated. For every transaction, payment amount will be recorded.

Each ticket can uniquely identify by its Ticket Number. Ticket includes price of the ticket, passenger class, seat number and ticket type.

Trains can be uniquely identify using their train ID. Details of the available classes in the trains, train models and type of them are also stored in the system.

One train passes through many stations. Stations have station ID and station name. Many trains could arrive at a single station. Arrival times of the trains and their platform number could change from train to train.

Requirements Analysis Document

Purpose of this document is to describe the requirements for the proposed Online Train Reservation System (OTRS).

This system is for two types of users,

1. Administrator users – Staff of the company
2. Guest users – Customers of the company (Passengers & Ticket Agents)

Administrator users can add, remove and modify the contents in all the relations. Guest users can only add, remove or modify the content in the limited number of relations. Both types of users must access the system using their specific login interface.

Functional Requirements

Administrator users must be able to do the following actions using the proposed Online Train Reservation System.

- See Reports and Statistics
- Add, Update & Remove Train Information
- Add, Update & Remove Station Information

Guest users of the system must be able to do the following actions.

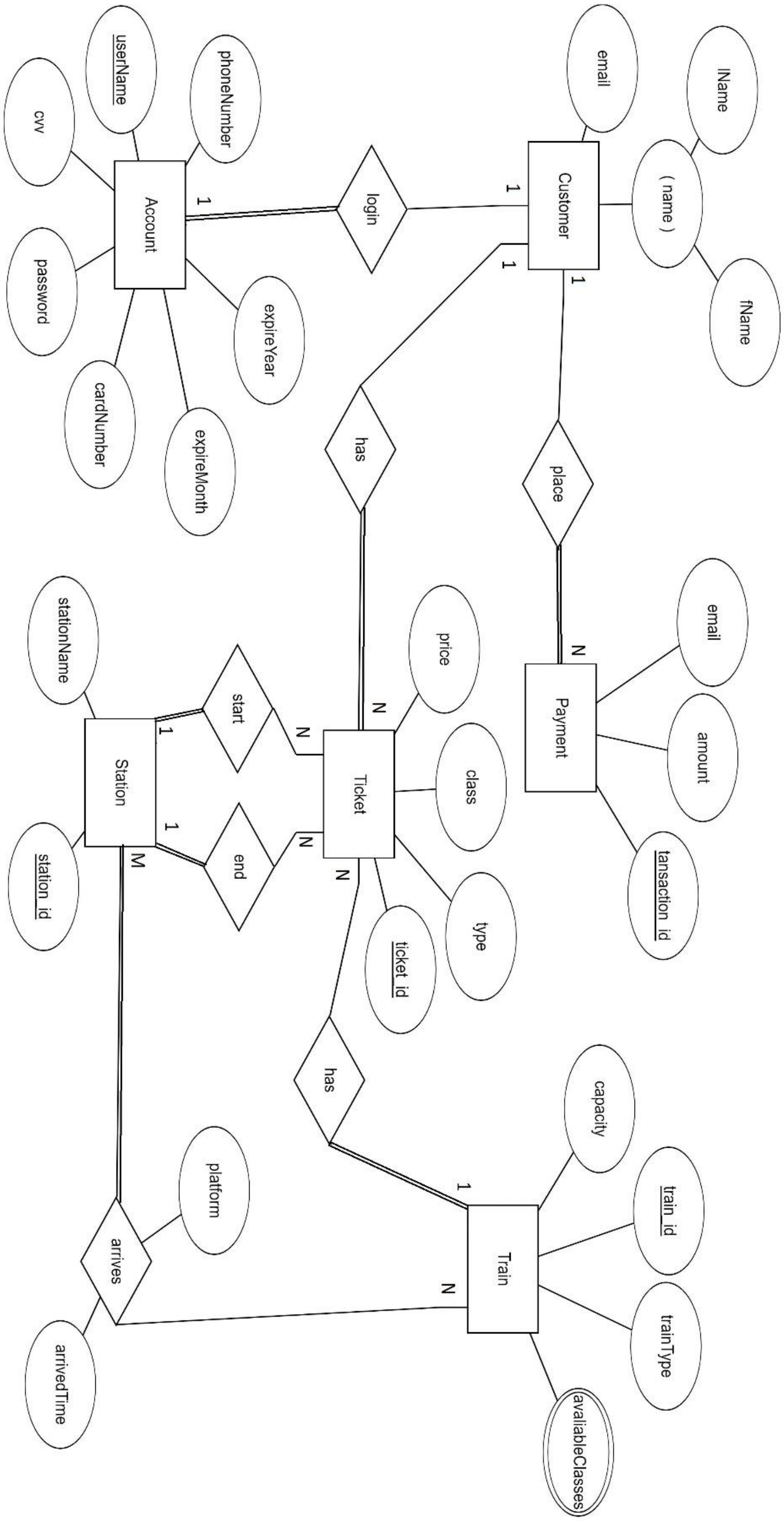
- Make, Update and Cancel Reservations
- See Train Schedule
- See Journey Information

Non-Functional Requirements

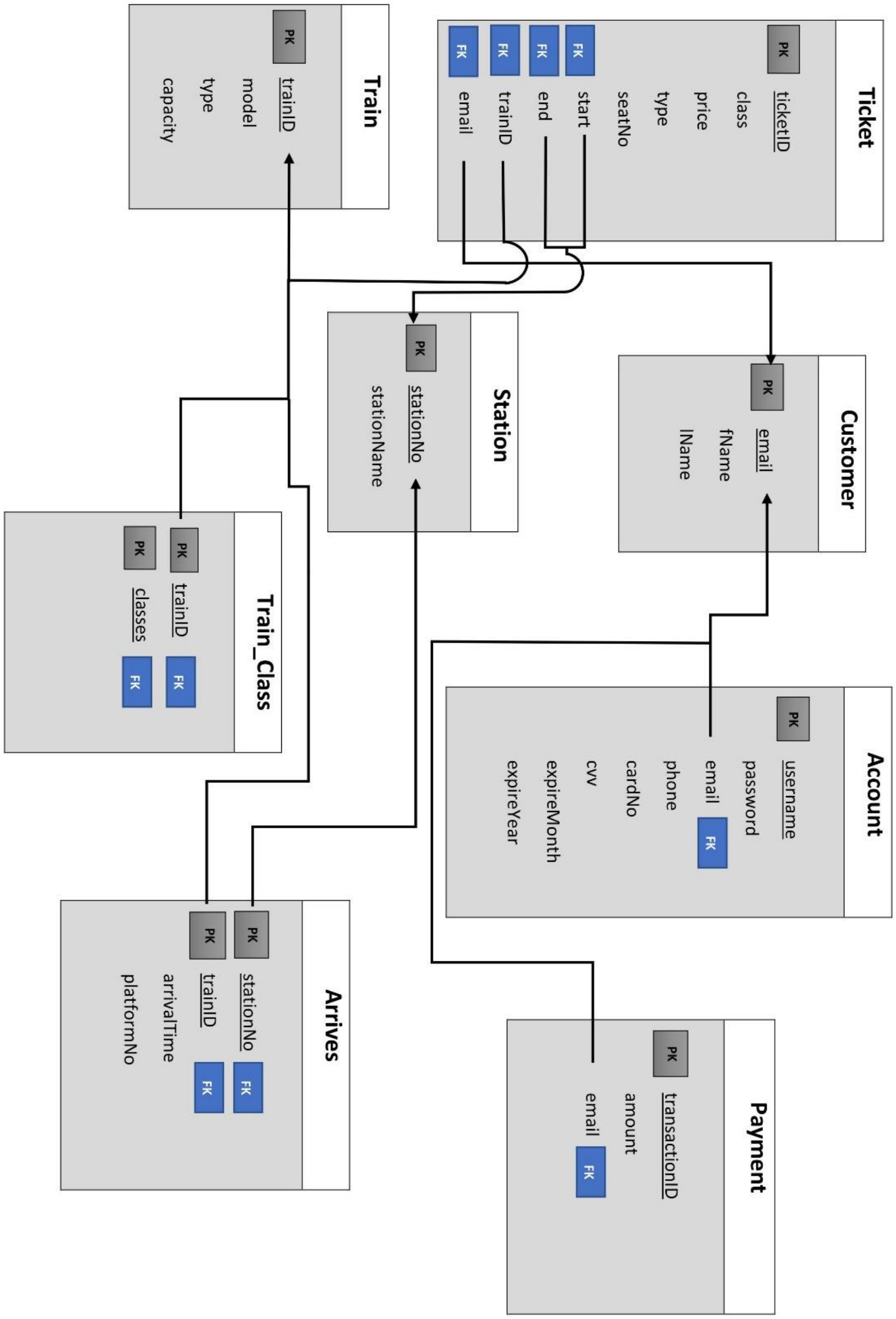
Following features will increase the value of the Online Train Reservation System.

- Availability
  - System should be available for public 24/7.
  - System should be accessible from all around the globe.
- User-friendliness
  - System's user interface (UI) should be user friendly. So, anyone could easily understand, navigate and make their needs accomplished.
- Maintainability
  - System should be easily maintainable without affecting its services provided.
- Reliability
  - System's database should be up to date.

ER Diagram


Account
userName
phoneNumber
cvv
password
cardNumber
expireMonth
expireYear
1
login
1
Customer
email
lName
(name)
fName
1
1
has
place
N
Payment
email
amount
tansaction_id
N
N
Ticket
price
class
type
ticket_id
N
N
start
end
1
1
Station
stationName
station_id
M
has
1
Train
capacity
train_id
trainType
avaliableClasses
N
arrives
platform
arrivedTime

Relational Schema



SQL Queries

Creating Tables

Customer Table

```
CREATE TABLE Customer
(
email VARCHAR(30),
fname VARCHAR(20) NOT NULL,
lname VARCHAR(20) NOT NULL,

CONSTRAINT Customer_PK PRIMARY KEY (email),
CONSTRAINT Email_Check CHECK (email LIKE '_%@_%._%')
);
```

Station Table

```
CREATE TABLE Station
(
stationID VARCHAR(5),
stationName VARCHAR(20) NOT NULL,

CONSTRAINT Station_PK PRIMARY KEY (stationID)
);
```

Train Table

```
CREATE TABLE Train
(
trainID VARCHAR(5),
model VARCHAR(10),
trainType CHAR(1),
capasity int NOT NULL,

CONSTRAINT Train_PK PRIMARY KEY (trainID),
CONSTRAINT Train_Type CHECK (trainType = 'E' OR trainType = 'S')
);
```

Arrives Table

```
CREATE TABLE Arrives
(
stationID VARCHAR(5),
trainID VARCHAR(5),
arrivalTime TIME NOT NULL,
platformNo INT,

CONSTRAINT Arrives_PK PRIMARY KEY (stationID, trainID),
CONSTRAINT Arrives_FK1 FOREIGN KEY (stationID) REFERENCES Station (stationID) ON
DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT Arrives_FK2 FOREIGN KEY (trainID) REFERENCES Train (trainID) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

Account Table

```
CREATE TABLE Account
(
username VARCHAR(20) NOT NULL,
password VARCHAR(16) NOT NULL,
cardNo CHAR(16) NOT NULL,
expMonth INT NOT NULL,
expYear INT NOT NULL,
CVV INT,
email VARCHAR(30),
phone CHAR(10),

CONSTRAINT Account_PK PRIMARY KEY (username),
CONSTRAINT Account_FK1 FOREIGN KEY (email) REFERENCES Customer(email) ON DELETE
CASCADE ON UPDATE CASCADE,
CONSTRAINT CardNo_Check CHECK (cardNo LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-
9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),
CONSTRAINT ExpMonth_Check CHECK (expMonth between 1 and 12),
CONSTRAINT ExpYear_Check CHECK (expYear between 2010 and 2030),
CONSTRAINT CVV_Check CHECK (CVV LIKE '[0-9][0-9][0-9]'),
CONSTRAINT Phone_Check CHECK (phone LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-
9][0-9][0-9]'),
CONSTRAINT Password_Check CHECK (len(password) >= 8),
CONSTRAINT Username_Check CHECK (username LIKE '____%')
);
```

Ticket Table

```
CREATE TABLE Ticket
(
ticketNo INT,
class CHAR(1) NOT NULL,
price MONEY NOT NULL,
ticketType CHAR(1) NOT NULL,
email VARCHAR(30),
startStationID VARCHAR(5),
endStationID VARCHAR(5),
trainID VARCHAR(5),
seatNo int,

CONSTRAINT Ticket_PK PRIMARY KEY (ticketNo),
CONSTRAINT Ticket_FK1 FOREIGN KEY (email) REFERENCES Customer (email) ON DELETE
CASCADE ON UPDATE CASCADE,
CONSTRAINT Ticket_FK2 FOREIGN KEY (startStationID) REFERENCES Station (stationID)
ON DELETE CASCADE ON UPDATE CASCADE,
CONSTRAINT Ticket_FK3 FOREIGN KEY (endStationID) REFERENCES Station (stationID) ON
DELETE NO ACTION ON UPDATE NO ACTION,
CONSTRAINT Ticket_FK4 FOREIGN KEY (trainID) REFERENCES Train (trainID) ON DELETE
CASCADE ON UPDATE CASCADE,
CONSTRAINT typeCheck CHECK (ticketType = 'F' OR ticketType = 'H'),
CONSTRAINT ClassCheck CHECK (class = '1' OR class = '2' OR class = '3')
);
```

Payment Table

```
CREATE TABLE Payment
(
transactionID INT,
amount MONEY NOT NULL,
email VARCHAR(30)

CONSTRAINT Payment_PK PRIMARY KEY (transactionID),
CONSTRAINT Payment_FK1 FOREIGN KEY (email) REFERENCES Customer(email) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

TrainClass Table

```
CREATE TABLE TrainClass
(
trainID VARCHAR(5),
classes int,

CONSTRAINT TrainClass_PK PRIMARY KEY (trainID, classes),
CONSTRAINT TrainClass_FK FOREIGN KEY (trainID) REFERENCES Train(trainID) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

Inserting Data to Tables

Customer Table

```
INSERT INTO Customer
VALUES ('sulochana@gmail.com', 'Sulochana', 'Lakmal');

INSERT INTO Customer
VALUES ('chamila@gmail.com', 'Chamila', 'Dilshan');

INSERT INTO Customer
VALUES ('ishinibagya@gmail.com', 'Ishini', 'Bagya');

INSERT INTO Customer
VALUES ('lakisuru@gmail.com', 'Lakisuru', 'Semasinghe');

INSERT INTO Customer
VALUES ('nipun@gmail.com', 'Nipun', 'Sandeep');

INSERT INTO Customer
VALUES ('buwaneka@yahoo.com', 'Buwaneka', 'Shamendra');
```

Station Table

```
    INSERT INTO Station
    VALUES ('M0','Colombo Fort');

    INSERT INTO Station
    VALUES ('M1','Polgahawela');

    INSERT INTO Station
    VALUES ('N1','Kurunegala');

    INSERT INTO Station
    VALUES ('N2','Anuradhapura');

    INSERT INTO Station
    VALUES ('N3','Jaffna');
```

Train Table

```
    INSERT INTO Train
    VALUES ('A1','s11','E',500);

    INSERT INTO Train
    VALUES ('A9','s10','S',600);

    INSERT INTO Train
    VALUES ('B6','s9','E',300);

    INSERT INTO Train
    VALUES ('B1','s13','E',500);

    INSERT INTO Train
    VALUES ('C5','s14','S',350);
```

Ticket Table

```
    INSERT INTO Ticket
    VALUES (1, 1, 1000, 'F', 'lakisuru@gmail.com', 'M0', 'M1', 'A1', 50);

    INSERT INTO Ticket
    VALUES (6, 2, 760, 'F', 'buwaneka@yahoo.com', 'N1', 'M1', 'C5', 69);

    INSERT INTO Ticket
    VALUES (2, 1, 1400, 'H', 'nipun@gmail.com', 'M0', 'N3', 'B6', 23);

    INSERT INTO Ticket
    VALUES (47, 3, 630, 'H', 'sulochana@gmail.com', 'N3', 'N1', 'A9', 163);

    INSERT INTO Ticket
    VALUES (23, 2, 860, 'F', 'ishinibagya@gmail.com', 'M0', 'N2', 'C5', 46);

    INSERT INTO Ticket
    VALUES (93, 3, 190, 'H', 'lakisuru@gmail.com', 'M0', 'N1', 'A1', 70);

    INSERT INTO Ticket
    VALUES (14, 3, 420, 'F', 'chamila@gmail.com', 'M0', 'N1', 'B1', 263);
```

Account Table

```sql
INSERT INTO Account
VALUES ('nipun_s97', 'wans1997', '4055349042415301', 02, 2021, 321,
'nipun@gmail.com', '0770889700');

INSERT INTO Account
VALUES ('lakisuru99', 'laki1918', '5412809764692012', 11, 2026, 765,
'lakisuru@gmail.com', '0712547890');

INSERT INTO Account
VALUES ('chamila.p1998', 'chamila9898', '2021398578906050', 07, 2025, 543,
'chamila@gmail.com', '0762239137');

INSERT INTO Account
VALUES ('sulochana2020', 'sulo1234', '1741098031248551', 04, 2023, 109,
'sulochana@gmail.com', '0762799510');

INSERT INTO Account
VALUES ('ishini_b98', 'ishini4321', '2020304065749856', 12, 2025, 890,
'ishinibagya@gmail.com', '0785589600');
```

Payment Table

```sql
INSERT INTO Payment
VALUES (1, 500, 'chamila@gmail.com');

INSERT INTO Payment
VALUES (2, 650, 'sulochana@gmail.com');

INSERT INTO Payment
VALUES (3, 1000, 'lakisuru@gmail.com');

INSERT INTO Payment
VALUES (4, 850, 'ishinibagya@gmail.com');

INSERT INTO Payment
VALUES (5, 700, 'nipun@gmail.com');

INSERT INTO Payment
VALUES (6, 890, 'buwaneka@yahoo.com');
```

Arrives Table

```
INSERT INTO Arrives
VALUES ('M0', 'A1', '18:05:00', 1);

INSERT INTO Arrives
VALUES ('M1', 'B6', '12:45:00', 2);

INSERT INTO Arrives
VALUES ('N1', 'B1', '09:49:00', 3);

INSERT INTO Arrives
VALUES ('N2', 'C5', '06:23:00', 6);

INSERT INTO Arrives
VALUES ('N3', 'A9', '14:40:00', 3);

INSERT INTO Arrives
VALUES ('M0', 'C5', '15:33:00', 4);

INSERT INTO Arrives
VALUES ('N1', 'B6', '19:20:00', 1);
```

TrainClass Table

```
INSERT INTO TrainClass
VALUES ('A1', 3);

INSERT INTO TrainClass
VALUES ('A9', 2);

INSERT INTO TrainClass
VALUES('B1', 1);

INSERT INTO TrainClass
VALUES ('B6', 2);

INSERT INTO TrainClass
VALUES ('C5', 3);

INSERT INTO TrainClass
VALUES('B1', 2);

INSERT INTO TrainClass
VALUES('B1', 3);

INSERT INTO TrainClass
VALUES ('A1', 2);

INSERT INTO TrainClass
VALUES ('B6', 1);
```

Performance Considerations

Following performances are expected from the system.

- System should allow both registered users as well as unregistered users to make reservations.
- Regular customers can create an account for faster and easier reservations.
- Registered users can store their credit/debit card number, CVV and expire date of the card. So, they do not need to enter those details over again and again.
- System should send an email notification to the customers’ email address for each reservation.
- Registered users can add their phone number for SMS notifications.

Security Requirements

Following security requirements are required to ensure the security of the system and its users.

- System should be able to protect customer information from outsiders.
- System should strictly protect the credit/debit card information of the registered customers.
- System's Administrator logins should be only limited for the respective staff members
- System should possess both data and service backup in case of system malfunction and data loss.
- System should validate and confirm the payment details when a customer is making a payment.
- To log in as a registered member, users must verify their user credentials (username and password).
- Passwords must be minimum 8 characters long.
- Card number must be minimum 16 digits long.
- CVV number must be a 3 digit number.
- Card's expire date must be between 2020 – 2030.
- Phone number must be number of 10 digits.
- Email address must be in form (<_________>@<_____>.<__>).