INDIAN RAILWAY RESERVATION SYSTEM

INTRODUCTION

Indian railway reservation system facilitates the passengers to enquire about the Trains available on the basis of source and destination, booking and cancellation of tickets, enquire about the status of the booked tickets etc.

The aim of case study is to design and develop a database maintaining The records of different trains, train status, and passengers .The record of train includes its number, name, source, destination, and days on which it is available, whereas record of train status includes dates for which tickets can be booked, total number of seats available, and number of seats already booked.

Description:

Passengers can book their tickets for the train in which seats are available. For this, passenger has to provide the desired train number and the data for which ticket is to be booked. Before booking a ticket for passenger, the validity of train number and booking date is checked. Once the train number and booking date are validated, it is checked whether the seat is available. If yes, the ticket is booked with confirm status and corresponding tickets are booked, certain number of tickets are booked and a message of non-available of seats is displayed.

The ticket once booked can be cancelled at any time. For this, the passenger has to provide the ticked ID .The ticket ID is searched and the corresponding record is deleted. With this, the first ticket with waiting status also gets confirmed.

Tools used:-

1.Android studio

2.sublime text editor

Database:-

1.Oracle database 2.Jdbc drivers for connection

Entity-Relationship Diagram

An *entity-relationship (ER) diagram*, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems.

Entity:

An entity is a piece of data-an object or concept about which data is stored.

Attribute:

Attributes are the properties which define the entity

Types:-

Key attribute:-

The attribute which uniquely identifies each entity in the entity set is called key attribute

Composite Attribute:-

An attribute composed of many other attribute is called as composite attribute

Multivalued Attribute:-

An attribute consisting more than one value for a given entity

Derived Attribute:-

An attribute which can be derived from other attributes of the entity type is known as derived attribute

Relationship:-

Relationship is the association between entities

Types:-

Unary Relationship:-

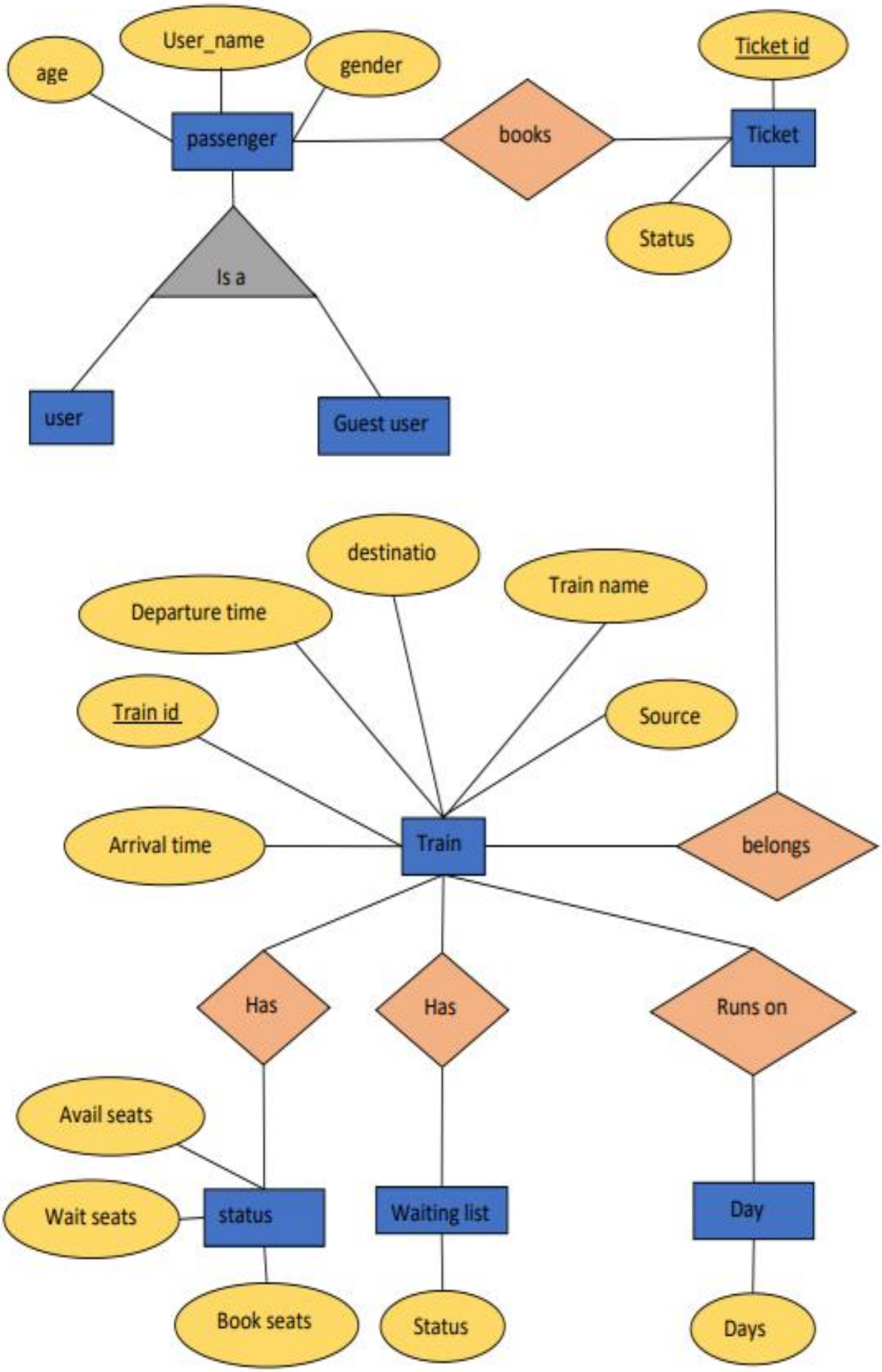
When there is only ONE entity set participating in a relation, the relationship is called as unary relationship

Binary Relationship:-

When there are TWO entities set participating in a relation, the relationship is called as binary relationship

n-ary Relationship:-

When there are n entities set participating in a relation, the relationship is called as n-ary relationship.


User_name
age
gender
passenger
books
Ticket id
Ticket
Status
Is a
user
Guest user
destinatio
Train name
Departure time
Train id
Source
Arrival time
Train
belongs
Has
Has
Runs on
Avail seats
Wait seats
status
Waiting list
Day
Book seats
Status
Days

Database structure

The tables used are as follows:

tday: contains available days of train which can be extracted using TID

TID	TDAY

Train1: contains train details such as train id, source, destination, train name , departure, arrival

TID	SRC	TNAME	DEST	DEPARTURE	ARRAIVAL

Tickets1: contains ticket id which is generated via triggers, and the ticket status

TC_ID	TID	STATUS

Passeng: contains basic information of the user such as username, age ,gender and the information regarding ticket Id and train id.

U_NAME	AGE	GENDER	TC_ID	TID

T_status : contains information regarding the train its available seats, booked seats, waiting seats which can be identified by train id.

AVALABLE_SEATS	BOOKED_SEATS	WAITING_SEATS	TID

Waiting_list : contains information about ticked id of the waiting list and the train id.

TC_ID	TID

Relational Tables

The tables are as follows:

create table tday(

```
tid integer,
tday varchar(10),
foreign key(tid) references Train1(tid)
);

create table Train1(
tid integer primary key,
src varchar(20),
dst varchar(20) ,
tname varchar(20),
departure varchar(20),
arraival varchar(20)
)

create table tickets1(
tc_id integer,
tid integer ,
status char(10),
foreign key(tid) references Train1(tid) ,
foreign key(tc_id) references passeng(tc_id)
on delete cascade
)

create table passeng(
u_name varchar(20),
tid integer,
age integer,
gender varchar(10),
tc_id integer unique,
foreign key(tid) references Train1(tid)
)
```

```
create table T_status(
avalable_seats integer,
booked_seats integer,
waiting_seats integer,
tid integer ,
foreign key(tid) references Train1(tid)
)

create table waiting_list(
tc_id integer,
tid integer ,
foreign key(tid) references Train1(tid) ,
foreign key(tc_id) references passeng(tc_id)    )
```

Train1

	TID	SRC	TNAME	DEST	DEPARTURE	ARRAIVAL
1	12864	tirupathi	Ypr-howra...	visakhapatnam...	02:30	15:55
2	17482	tirupathi	Tpty-bsp exp	visakhapatnam...	10:35	04:45
3	17480	tirupathi	Tpty-puri...	visakhapatnam...	10:35	04:45
4	22872	tirupathi	Tpty-bbs ...	visakhapatnam...	12:15	02:00
5	22880	tirupathi	Tpty-bbs ...	visakhapatnam...	12:15	02:00
6	12846	tirupathi	Bhubanesh...	visakhapatnam...	14:35	04:10
7	12890	tirupathi	Ypr-tata exp	visakhapatnam...	14:35	04:12
8	22856	tirupathi	Tpty-vskp...	visakhapatnam...	14:35	04:10
9	12739	visakhapa...	garib rat...	secunderabad ...	20:30	20:30
10	12842	chennai	Coramandal	visakapatnam ...	08:45	21:50
11	12721	visakhapa...	godavari ...	hyderabad ...	16:30	07:30

Tickets1

	TC_ID	TID	STATUS
1	1	12864	c

T_status

	AVALABLE_SEATS	BOOKED_SEATS	WAITING_SEATS	TID
1	19	1	0	12864
2	20	0	0	17482
3	20	0	0	17480
4	20	0	0	22872
5	20	0	0	22880
6	20	0	0	12846
7	20	0	0	12890
8	20	0	0	22856
9	20	0	0	12739
10	20	0	0	12842
11	1	0	0	12721

Passeng

	U_NAME	TID	AGE	GENDER	TC_ID
1	sunny	12864	20	M	1

Waiting list

TC_ID	TID

Tday

	TID	TDAY
1	12864	mon
2	12864	tue
3	12864	wed
4	12864	thu
5	12864	fri
6	12864	sat
7	12864	sun
8	17482	sun
9	17480	mon

Triggers

Hear various triggers are used in order to perform automation for various tables after alteration to tables.

First trigger:

```
create trigger booking
after insert on passeng
for each row
enable
declare
t_id integer;
tcid integer;
waiting integer;
begin
select waiting_seats into waiting from T_status where tid=:new.tid;
if(waiting>0)
then
insert into tickets1(tc_id,tid,status)
values(:new.tc_id,:new.tid,'w');
else
insert into tickets1(tc_id,tid,status)
values(:new.tc_id,:new.tid,'c');
end if;
end;
```

Second trigger:

```
create trigger updatestatus
after insert or delete on passing
```

```
for each row
enable
declare
avail integer;
waiting integer;
begin
if inserting then
select avalable_seats into avail from T_status where tid=:new.tid;
if(avail>0) then
update T_status
set avalable_seats=avalable_seats-1,booked_seats=booked_seats+1
where tid=:new.tid;
else
update T_status
set waiting_seats=waiting_seats+1
where tid=:new.tid;
end if;
end if;
if deleting then
update T_status
set avalable_seats=avalable_seats+1,booked_seats=booked_seats-1
where tid=:old.tid;
end if;
end;
```

Third trigger:

```
create trigger something
after update on tickets1
for each row
enable
declare
```

```
begin
update T_status
set
waiting_seats=waiting_seats-1,booked_seats=booked_seats+1,avalable_seats=avala
ble_seats-1
where tid=:new.tid;
end;
```

Connectivity:

In the connectivity of the database to the application we have created a class "Connectdb" which returns an database connection object. Here we used JDBC drivers.

C:\Users\UMAKANTH RAYIPUDI\Desktop\connect.java - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

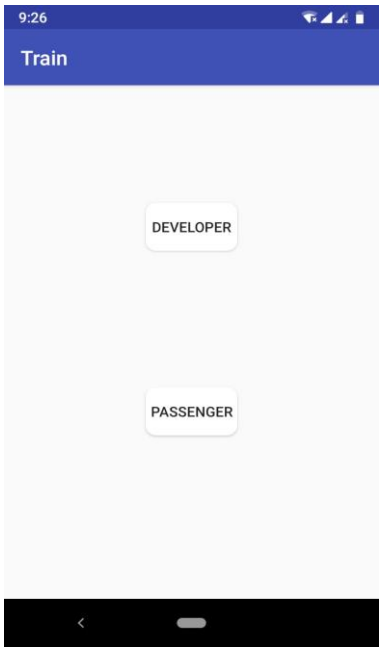
connect.java | Booking.java

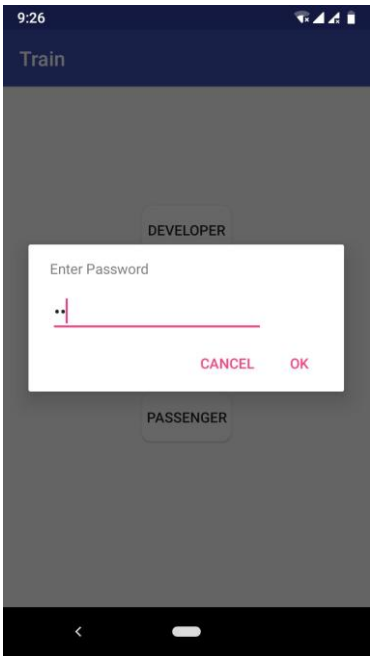
```java
import android.content.Context;
import android.os.StrictMode;
import android.widget.Toast;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Connectdb {
    public Connection connection;
    public static final String DEFAULT_DRIVER = "oracle.jdbc.driver.OracleDriver";
    public static final String DEFAULT_URL = "jdbc:oracle:thin:@192.168.23.144:1521/ORCL";
    public static final String DEFAULT_USERNAME = "gh";
    public  String DEFAULT_PASSWORD = "gh";

    public Connection connect(String DEFAULT_PASSWORD) {
        if(DEFAULT_PASSWORD!=null)
        this.DEFAULT_PASSWORD =DEFAULT_PASSWORD;
        if (android.os.Build.VERSION.SDK_INT > 9) {
            StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);
        }
        try {
            this.connection = createConnection();
            return connection;
        } catch (Exception e) {
            return null;
        }

    }
    public Connection createConnection(String driver, String url, String username, String password) throws ClassNotFoundException, SQLException {

        Class.forName(driver);
        return DriverManager.getConnection(url, username, password);
    }

    public  Connection createConnection() throws ClassNotFoundException, SQLException {
        return createConnection(DEFAULT_DRIVER, DEFAULT_URL, DEFAULT_USERNAME, DEFAULT_PASSWORD);
    }
}
```

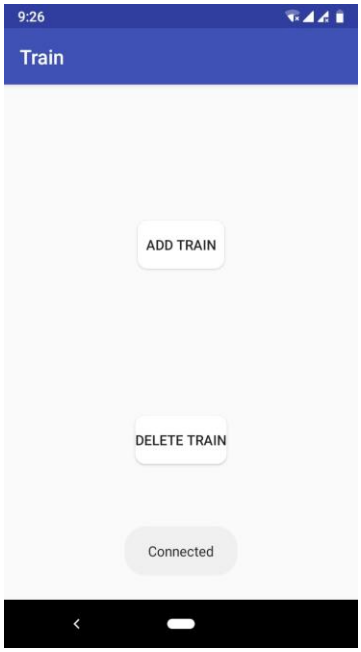
USER INTERFACE

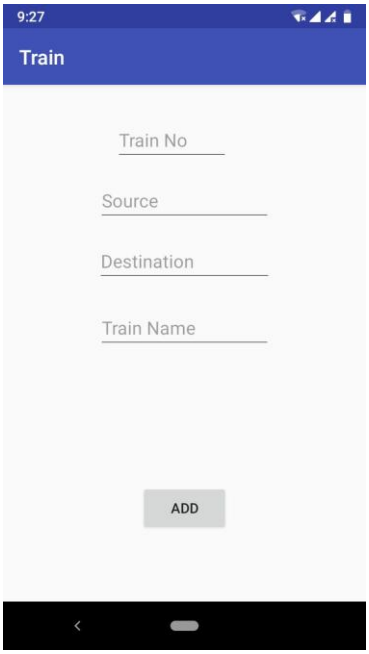
Here we created an Android application.

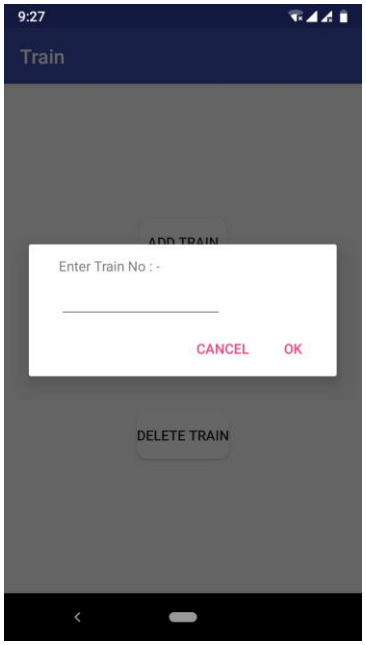
It has Xml file for front end and java code for backend.

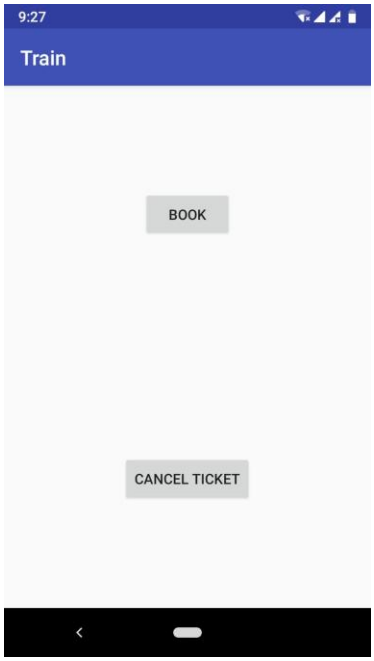


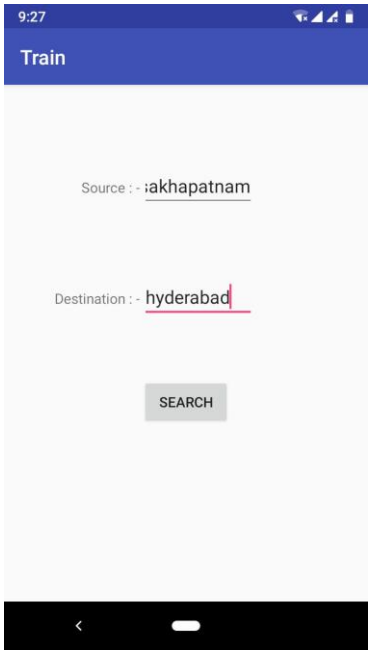


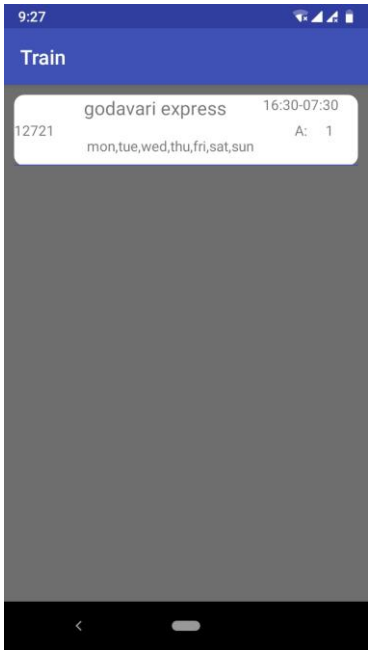




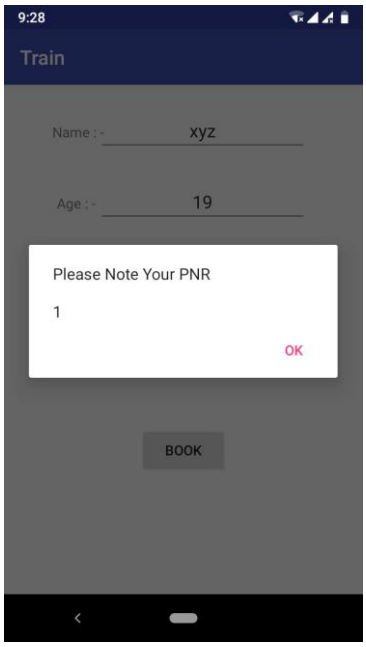

9:27
Train
ADD TRAIN
Enter Train No : -
CANCEL
OK
DELETE TRAIN

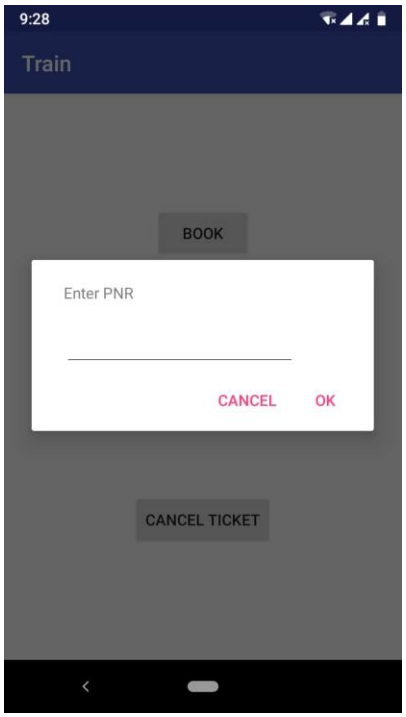

9:27
Train
BOOK
CANCEL TICKET


9:27
Train
Source : - ;akhapatnam
Destination : - hyderabad
SEARCH


9:27
Train
godavari express
16:30-07:30
12721
A: 1
mon,tue,wed,thu,fri,sat,sun


9:28
Train
Name : -
Age : -
Gender (M/F): -
BOOK


9:28
Train
Name : -
xyz
Age : -
19
Please Note Your PNR
1
OK
BOOK

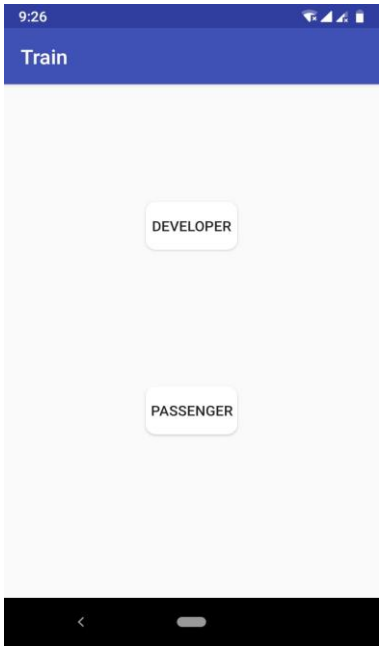

9:28
Train
BOOK
Enter PNR
CANCEL
OK
CANCEL TICKET

Working

Our application provides two modes

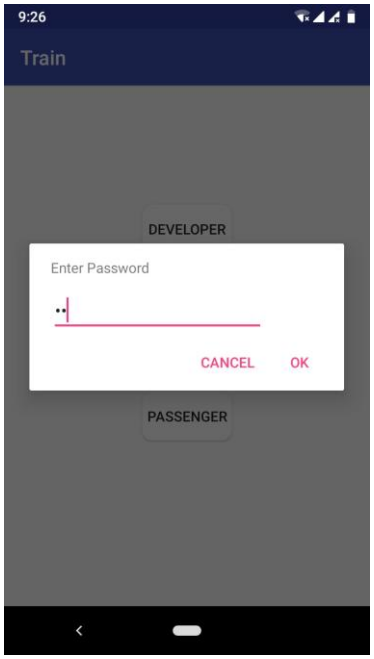
1.developer mode :- he can add or delete a train

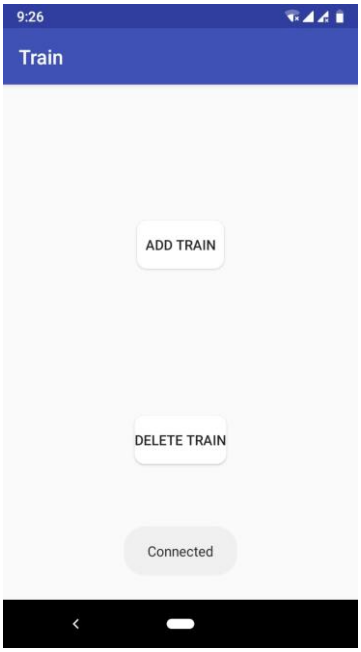
2.passenger mode :- he can book or cancel ticket



Developer mode:-

Developer has to enter password to connect to data base

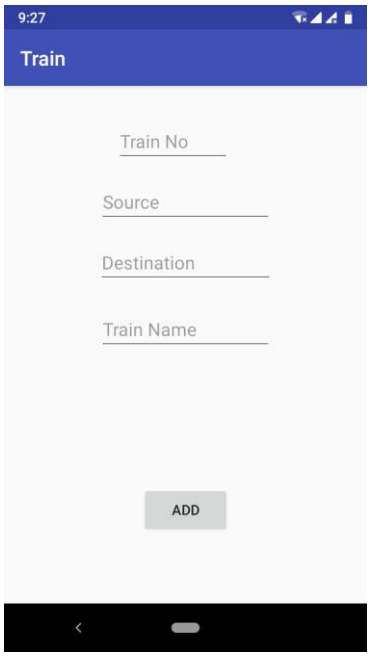


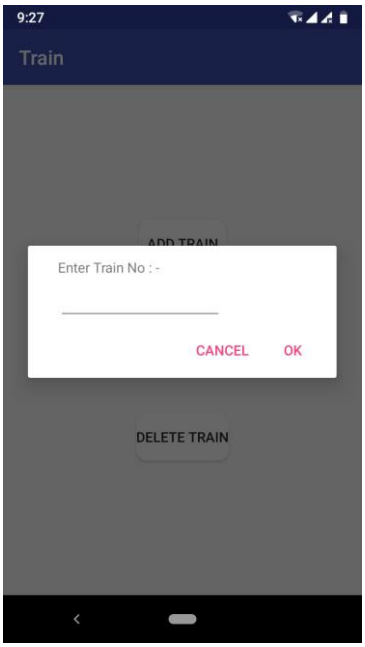


now he has two

1.add train

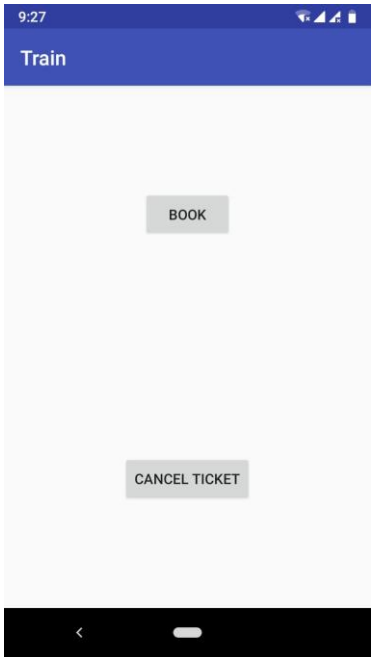
2.delete train





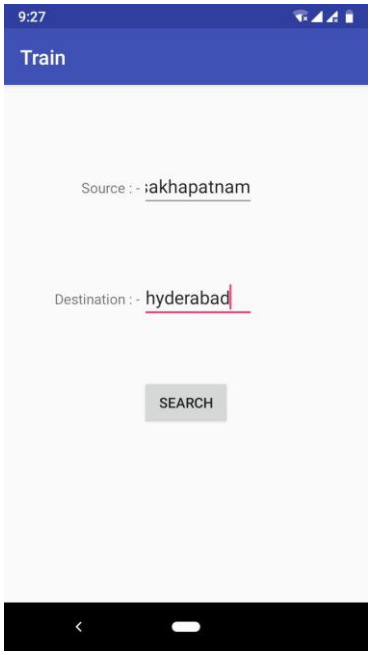
User mode:-

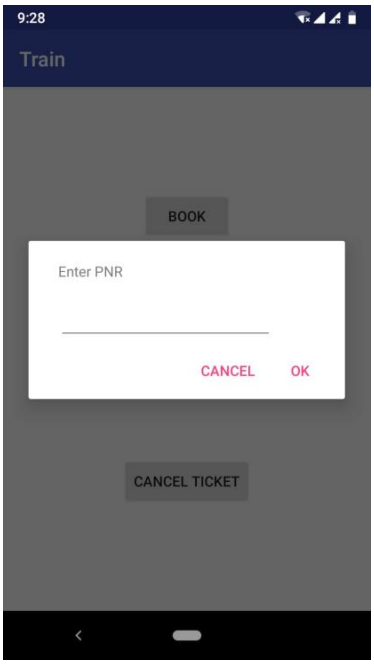
He has two options



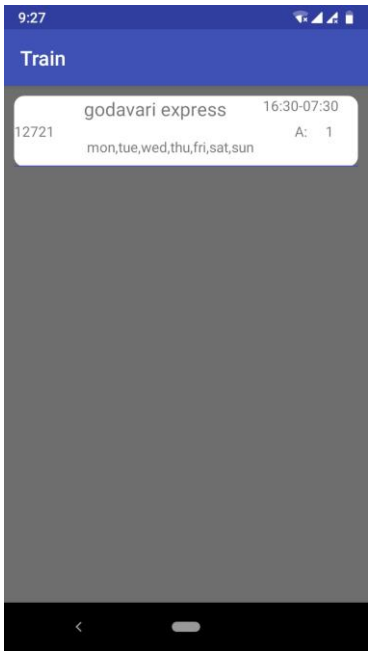
1.book tickect

2.cancel ticket



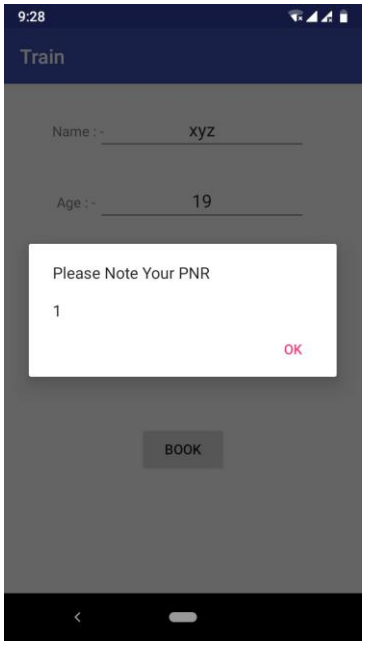


Search train and seat availability



enter PNR number to cancel

Customer enters details and book ticket



Finally PNR number is generated.

Conclusion:-

The main aim of developing Indian Railway Reservation system is to provide all information that is required by the users.

User friendliness is a must that is the user must get the details without complicated searching process.

Other important requirements of software are data security, extensibility and maintainability.

Acknowledgement:-

I would like to express my special thanks of gratitude to Himaja mam for giving an opportunity to learn and develop new things. I would also like to express my thanks to AP government for conducting android workshop under the supervision of Thammi reddy sir, HOD of CSE department.

References:-

https://android-developers.googleblog.com/

https://www.javatpoint.com/example-to-connect-to-the-mysql-database

https://www.w3schools.com/sql/