Train Reservation System

Project II Report

11.21.2017

Team:

Vishnu Sripriya Akondi - Team Leader (va0017@uah.edu)

Jyothi Sairam Sankabathula (js0168@uah.edu)

Romanjali Ramwani (rr0070@uah.edu)

Vani Krishna Raghu Vamsha Kokkula (vk0018@uah.edu)

Prepared for:

UAH CS 687 – Database Systems

Instructor: Dr. Ramazan Aygun

Fall 2017

Table of Contents

Description:

1. Our project is to create a web application for Train reservation system which manages the reservation and cancellation of railway tickets for the passengers.
2. The interfaces are Main Page (consists of Train routes, Fare details, Train timings, Train type), Reservation Page, Make Payment Page, Ticket Confirmation Page, Modification Page, Cancellation Page and Admin Login page.
3. While booking the ticket, the database will store each Passenger's Name, Social Security Number, Gender, Age, Phone Number.
4. After issuing the ticket we provide the passenger with their Ticket Number, Train Name, Train ID, Date Reserved, Seat Number and Payment ID can be used for checking the status of train delays or cancelled or arrival times.
5. The passenger can cancel their Train reservation and change his/her date of travel by just using their Ticket Number.
6. A passenger can book their ticket with various modes of payment (VISA, MASTERCARD, or any other major credit card). Each payment has a unique number called payment ID, status of payment, Payment Date.
7. Insertion, deletion and updating of train routes and timings can be managed by 4 admins who has privileges of performing above mentioned actions.
8. A passenger can book tickets from the current date to the next 5 days.

Project Environment:

Operating System: Windows 10

Programming Environment: Java(JDBC), Java Script, Java Server Pages, HTML, Oracle 11g

Server: Apache Tomcat

Jars: ojdbc14.jar, classes111.jar

Project Plan:

- Submission of Group members and Project Title: September 5th
- Formal Description Submission: October 26th
- Create Database tables according to the schema: November 5th
- Implementation of front-end with described functionalities: November 7th
- Implementation of back-end with described functionalities: November 9th
- Report Submission: November 21th
- Demo (in-class): November 21st

Responsibilities of each Team Member:

Responsibilities	Owners
Requirement gathering and Analysis, Creation and Maintenance of DB (Backend), Server Connection(JSP)	Sri Priya, Jyothi Sai Ram
Creating tables, ER and Schema Diagrams, Database Connection (JDBC), Front-end (HTML and CSS)	Romanjali and Raghu Vamsha
Application Testing	Raghu Vamsha, Jyothi Sai Ram
Project Report	Raghu Vamsha, Sri Priya, Jyothi Sai Ram, Romanjali
Minutes of Meeting	Sri Priya

Formal Description of the Database:

Our project Train Reservation System has 5 entities and their attributes as described below:

TRAIN_ROUTE:

- Train_Name: Has name of the train of varchar datatype. It is unique attribute.
- Train_ID: Every train has a unique train_id which is primary key of integer datatype in the entity.
- Train_type: Describes type of the train. Example: Superfast, Express, etc. Datatype is varchar.
- Source: Location from where the train departs. Datatype is varchar.
- Destination: Location to where the train arrives. Datatype is varchar.
- Fare: Amount charged to the number of seats booked for the travel. Datatype is decimal.
- Arr_time: Time when the train arrives at the destination. Datatype is timestamp.
- Dep_time: Time when the train starts from the source. Datatype is timestamp.

PASSENGER:

- SSN: Unique identifier to every passenger. It is primary key with datatype of char(9).
- Ticket_No: Unique ticket number is generated for every travel booked by a passenger. Datatype is number(5).
- Fname: First name of the passenger. Datatype is varchar.
- Minit: Middle initial of the passenger. Datatype is character.
- Lname: Last name of the passenger. Datatype is varchar.
- Gender: Sex of the passenger with check constraint 'M' OR 'F' OR 'O'
- Age: Age of the passenger. Datatype is integer. There is a check constraint of the attribute having Age>0
- Phone_Number: Contact number details of the passenger. Datatype is char(10).
- P_train_id: It is foreign key referring to Train_id which is primary key in Train_Route entity.
- P_payment_id: It is foreign key referring to Payment_id which is primary key in Payment entity.

ADMIN:

- User_ID: Contains user identity for the admin. Datatype is varchar. It is a primary key.
- Password: Contains password to the admin's login. Datatype is varchar.

PAYMENT:

- Payment_ID: Unique payment id is generated on every successful payment done by the passenger. Datatype is number(5).
- Holder_name: Contains name of the card holder. Datatype is varchar.
- Credit_Num: Contains credit card number. Datatype is char(16).

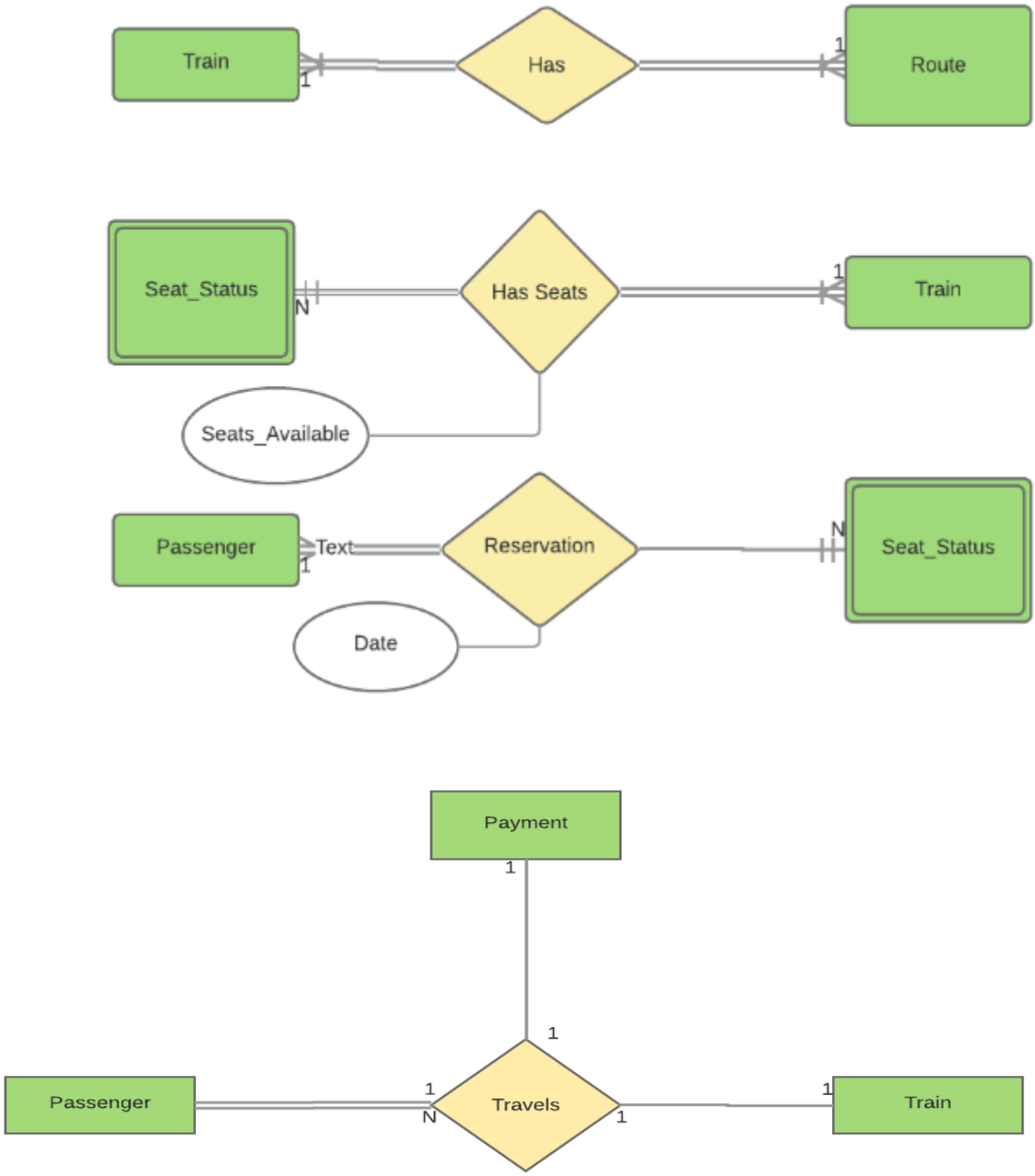
- Modes: It contains mode of payment options available for the passenger. Datatype is char. It has a check constraint 'V' OR 'C' OR 'M'. (V-Visa Card, C-Credit Card and M-Master Card)
- Payment_Date: Date on which the payment was made. Datatype is date.
- Status: It contains the status of the payment. Datatype is varchar.

SEAT_STATUS:

- Seats_Available: Number of seats available after booking the ticket. Datatype is integer. There is a check constraint Seats_Available > 0.
- Seat_No: Seat number allotted to the passenger while booking the ticket. Datatype is integer. There is a check constraint Seat_No > 0.
- Status: It contains status of the seat. If it is successful, the status of the seat is booked else the status is available. Datatype is varchar.
- Date: Date of travel. Datatype is date.
- S_train_id: It is foreign key referring to Train_id which is primary key in Train_Route entity.
- S_SSN: It is foreign key referring to SSN which is primary key in Passenger entity.

Mapping ER to Relational Database Schema

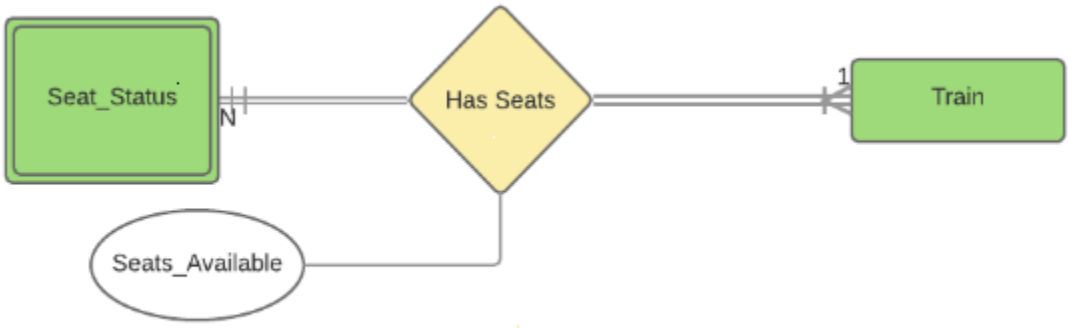
The following are the **mapping cardinalities** of ER to relational database schema:



A. One to Many (1 to N) relationship mapping into tables:

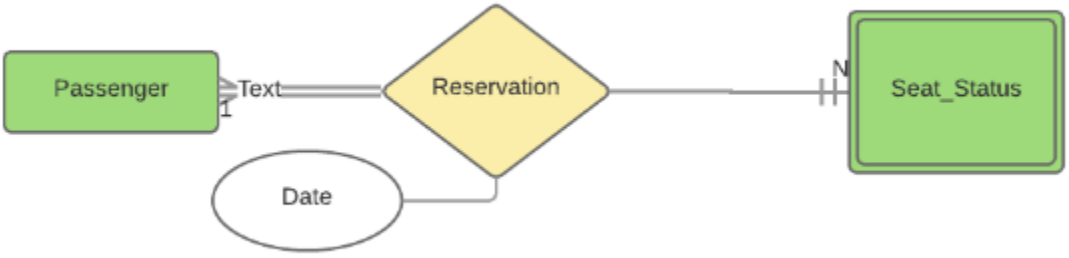
1. As per 1 to N mapping we added 'S_Train_id' attribute to Seat_Status as a Foreign Key that refers Primary key 'Train_id' of Train and added attribute Seats_Available of Relation to Seats_Status.

Seat_Status(Date, Status, Seat_no, S_Train_id, Seats_Available)
Train(Train_id, Train_type, Train_name)

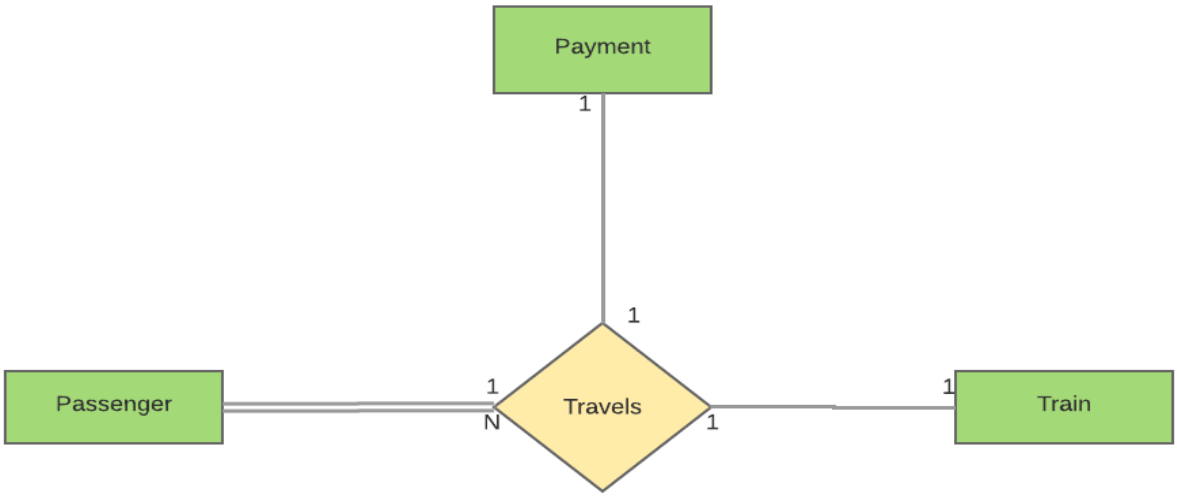


2. As per 1 to N mapping we added 'S_SSN' attribute to Seat_Status as a Foreign Key that refers Primary key 'SSN' of Passenger.

Passenger(SSN, Fname, Minit, Lname, Gender, Age, Phn_num)
Seat_Status(Date, Status, Seat_no, S_Train_id, S_SSN, Seats_Available)



3. For the following diagram:



- Consider Passenger to Train:
  As per 1 to N mapping we added 'P_Train_id' attribute to Passenger as a Foreign Key that refers Primary key 'Train_id' of Train and added attribute Ticket_no added to Passenger.

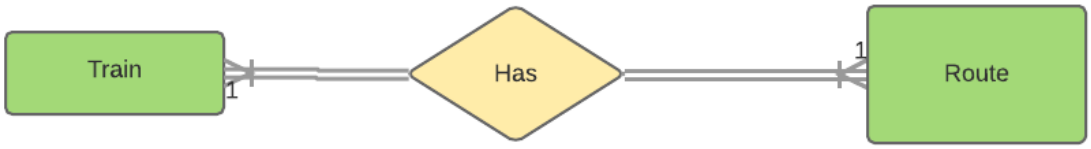
  Passenger(SSN, Fname, Minit, Lname, Gender, Age, Phn_num, P_Train_id, Ticket_no)
  Train(Train_id, Train_type, Train_name)

- Consider Passenger to Payment:
  As per 1 to 1 mapping we added 'P_payment_id' attribute to Passenger as a Foreign Key that refers Primary key 'Payment_id' of Payment.

  Passenger(SSN, Fname, Minit, Lname, Gender, Age, Phn_num, P_Train_id, Ticket_no, P_payment_id)
  Payment(Payment_id, Mode, Status, Payment_date)

B. One to One Relationship mapping:

1. As per 1 to 1 mapping we merged both the Train and Route as one entity and gave Train_id as a primary key.



Train_route(Train_id, Train_name, Train_type, Source ,Destination, Arr_time, Dep_time, Fare)

Entity Relationship (ER) Design
ER Design

The ER diagram below clearly outlines the entities and their attributes. The primary keys of each entity have been underlined. Cardinalities have been mentioned using their standard form (N-N and 1-N). Total participation and partial participation has also been shown in the figure.

ER (Entity Relation) diagram for our project gives a bit more clarity and depth of understanding of the database schema. Initially we have ADMIN entity with USER_ID and PASSWORD as attributes which has the privilege to maintain and perform operations on this application. It updates information related to train details that include

- insertion of (Train_id, Train_name, Train_type, Source, Destination, Arrival time, Departure time, Fare)
- updating of (Arrival time and departure time with respect to each train_id)
- deletion of train details by giving train_id as input.

PASSENGER entity has the information related to the passenger travel details with attributes SSN (unique to the traveller), Fname, Minit, Lname, Gender, Age, Phone_num. The entity is bound to TRAIN and PAYMENT entities with TRAVELS relationship which shows the passenger can travel by single train and the payment can be made by single passenger for the travel.

TRAIN entity has the information related to train details with attributes Train_id (unique), Train_name and Train_type. The entity is bound to ROUTE entity with HAS relationship which shows that a train can travel from source to destination directly without intermediate stops. The entity is also bound to SEAT_STATUS entity with HAS SEATS relationship which shows that a train can have N number of seats where Seats_Available is a derived attribute to this relation.

SEAT_STATUS entity has the information related to seat number and status where seat number is partial key and this entity is weak entity. This entity is bound to PASSENGER entity with RESERVATION relationship which shows that a passenger can book N seats.

ROUTE entity has attributes Source, Destination, Fare, Arr_time and Dep_time. In this entity Source, Destination, Arr_time and Dep_time together make the primary key.

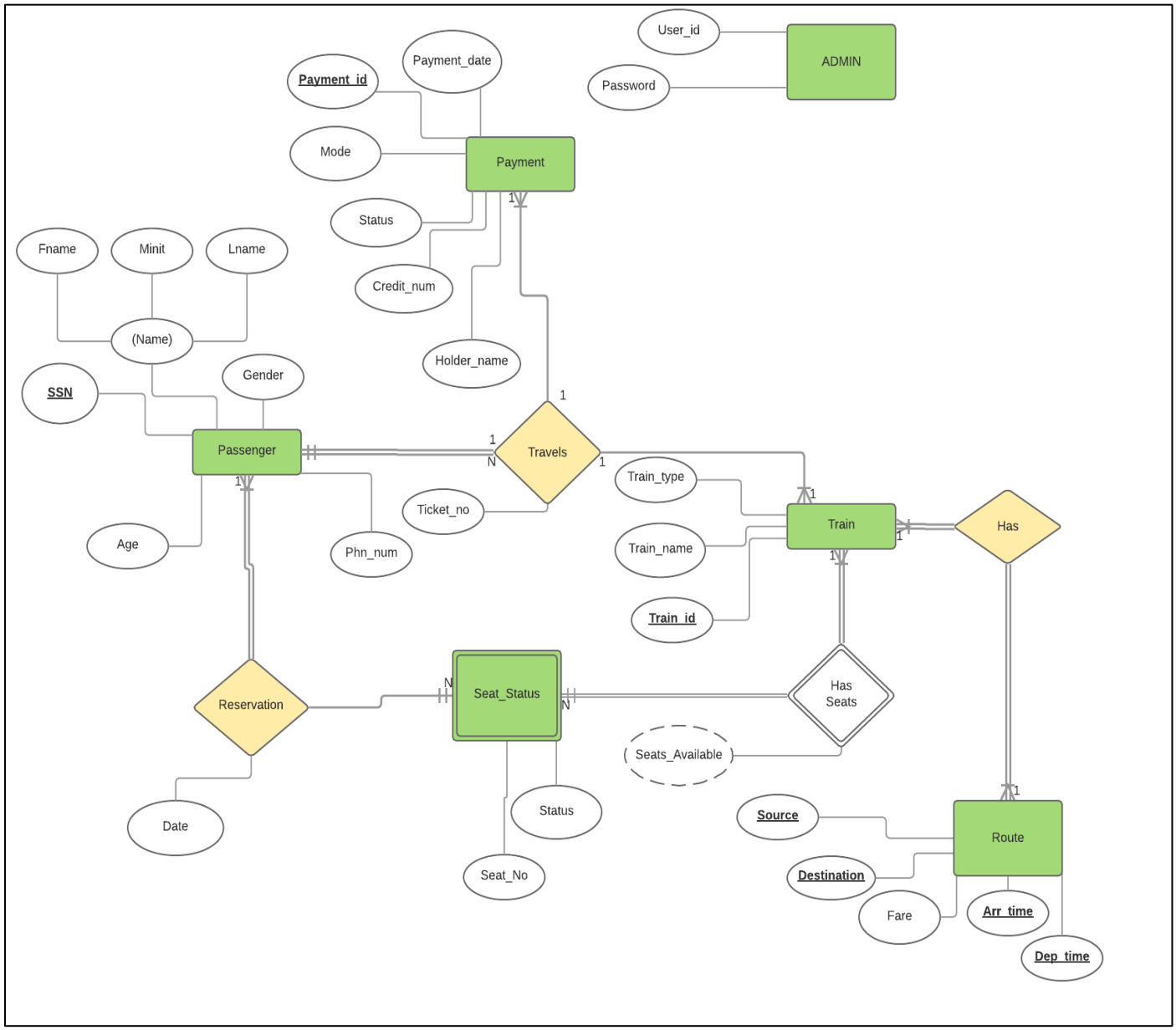
PAYMENT entity has information related to payment details with attributes Payment_id (unique), Payment_date, Holder_name, Credit_num, Mode and Status.

The TRAVELS relationship has attribute Ticket_No.

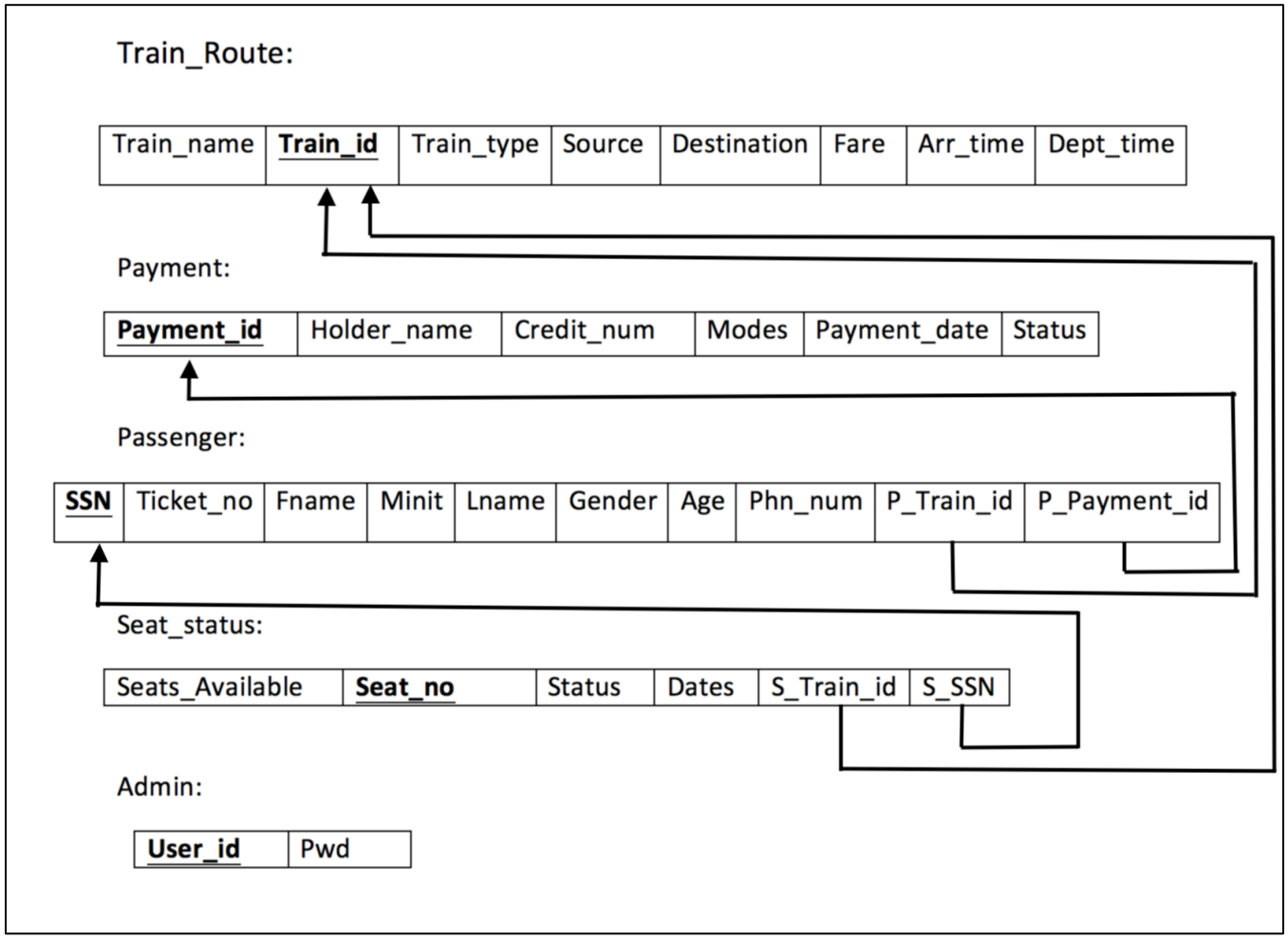
The RESERVATION relationship has attribute Date.

The PASSENGER entity has common composite attribute Name (Fname, Minit, Lname).

ER Diagram


User_id
ADMIN
Password
Payment_date
Payment_id
Mode
Payment
Status
Credit_num
Holder_name
Fname
Minit
Lname
(Name)
SSN
Gender
Passenger
Travels
Ticket_no
Age
Phn_num
Train_type
Train
Has
Train_name
Train_id
Reservation
Seat_Status
Has
Seats
Seats_Available
Date
Status
Seat_No
Source
Route
Destination
Fare
Arr_time
Dep_time
1
N

Relational Schema Diagram:



Create Table Statements:

Database Schema:

1. PASSENGER:

```
CREATE TABLE PASSENGER (
SSN CHAR(9) NOT NULL,
Ticket_no NUMBER(5) NOT NULL,
Fname VARCHAR(20) NOT NULL,
Minit CHAR,
Lname VARCHAR(20) NOT NULL,
Gender CHAR NOT NULL,
Age int NOT NULL,
Phn_num CHAR(10),
P_train_id NUMBER(5),
P_payment_id NUMBER(5),
CONSTRAINT ssn_pk PRIMARY KEY(SSN),
CONSTRAINT p_payment_id_fk FOREIGN KEY(P_payment_id) REFERENCES
PAYMENT(Payment_id),
CONSTRAINT p_train_id_fk FOREIGN KEY(P_train_id) REFERENCES
TRAIN_ROUTE(Train_id), CHECK(Gender='M' or Gender='F' or Gender='O'),
CHECK(Age>0));
```

2. TRAIN_ROUTE

```
CREATE TABLE TRAIN_ROUTE(
Train_name VARCHAR(20) NOT NULL,
Train_id NUMBER(5) NOT NULL,
Train_type VARCHAR(20) NOT NULL,
Source VARCHAR(20) NOT NULL,
Destination VARCHAR(20) NOT NULL,
Fare DECIMAL(5,2) NOT NULL,
Arr_time TIMESTAMP NOT NULL,
Dep_time TIMESTAMP NOT NULL,
CONSTRAINT train_id_pk PRIMARY KEY(Train_id), CHECK(Train_id > 0) ,
UNIQUE(Train_name));
```

3. SEAT_STATUS

```
CREATE TABLE SEAT_STATUS(
Seats_Available int NOT NULL,
Seat_no int NOT NULL,
Status VARCHAR(20) NOT NULL,
Dates DATE,
S_Train_id NUMBER(5),
S_SSN CHAR(9),
CONSTRAINT train_id_status_fk FOREIGN KEY(S_Train_id) REFERENCES
TRAIN_ROUTE(Train_id),
CONSTRAINT S_ssn_fk FOREIGN KEY(S_SSN) REFERENCES PASSENGER(SSN),
CHECK(Seat_no>0), CHECK(Seats_Available>0));
```

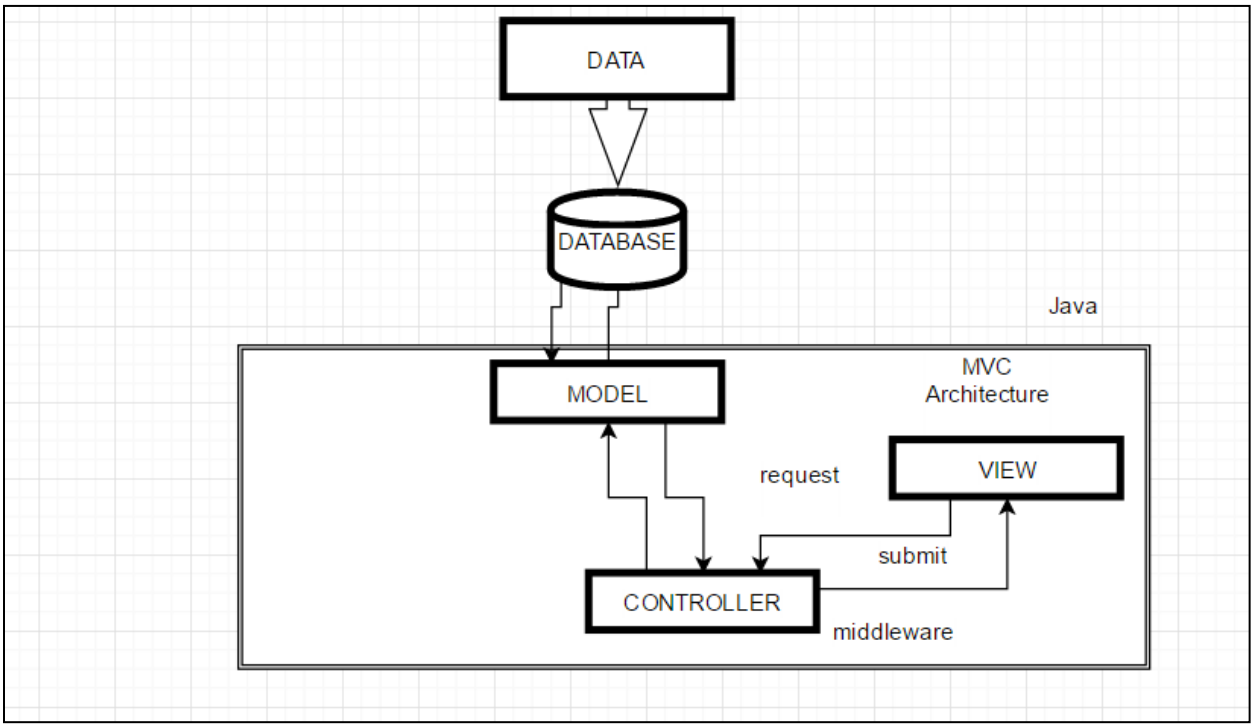
4. ADMIN

```
CREATE TABLE ADMIN(
User_id VARCHAR2(20) NOT NULL,
Pwd VARCHAR2(20) NOT NULL,
CONSTRAINT userid_pk PRIMARY KEY(User_id));
```

5. PAYMENT

```
CREATE TABLE PAYMENT(
Payment_id NUMBER(5) NOT NULL,
Holder_name VARCHAR(20) NOT NULL,
Credit_num CHAR(16) NOT NULL,
Modes CHAR,
Payment_date Date NOT NULL,
Status VARCHAR(15),
CONSTRAINT payment_id_pk PRIMARY KEY(Payment_id), CHECK(Payment_id>0),
CHECK(Modes='V' or Modes ='C' or Modes='M'));
```

Project Implementation
System Architecture



Model view Architecture (MVC) is followed for the application design

VIEW – View is a webpage used for display, programmed using JSP, HTML and Java Script. Passenger enters the Source, Destination, Number of seats and Date of travel on the home page to know the details of the train through the webpage. The request from the Passenger is sent to the controller.

CONTROLLER - Controller acts as the middleware, the request received from the view is checked and the correct method is executed in the model. while receiving the data from the model the controller again acts as the middleware and passes the results on to the view for display.

MODEL – Model connects with Oracle and the model, accessing the queries from the controller and retrieving data from oracle (database). After successful connection with the database the queries will be passed as parameter and the query will be executed and displayed on the Webpage.

Issues considered:

1. Is the application a single-user or multi-user application?
   ANSWER: This is a multi user application.

2. Is the application read-only (only retrieval) or read-write application?
   ANSWER: It is a read-write application. The user of this application will also be able to update or delete the tuples in database by using modify/cancel option. They can modify or cancel their ticket status.

3. How is concurrency handled? (i.e., are you aware of the problems that may occur if concurrency is not handled properly?)
   ANSWER: In this application concurrency is handled using sessions.

4. Is indexing or fast-retrieval an important part of your application?
   ANSWER: No.

5. What are the steps taken if the system or a transaction fails?
   ANSWER: We have included exception handling scenarios for every possible transaction failure. For example, if user does not enter any value, a message displays asking the user to correct format of values.

6. Is your application data stored on a cloud database?
   ANSWER: No.

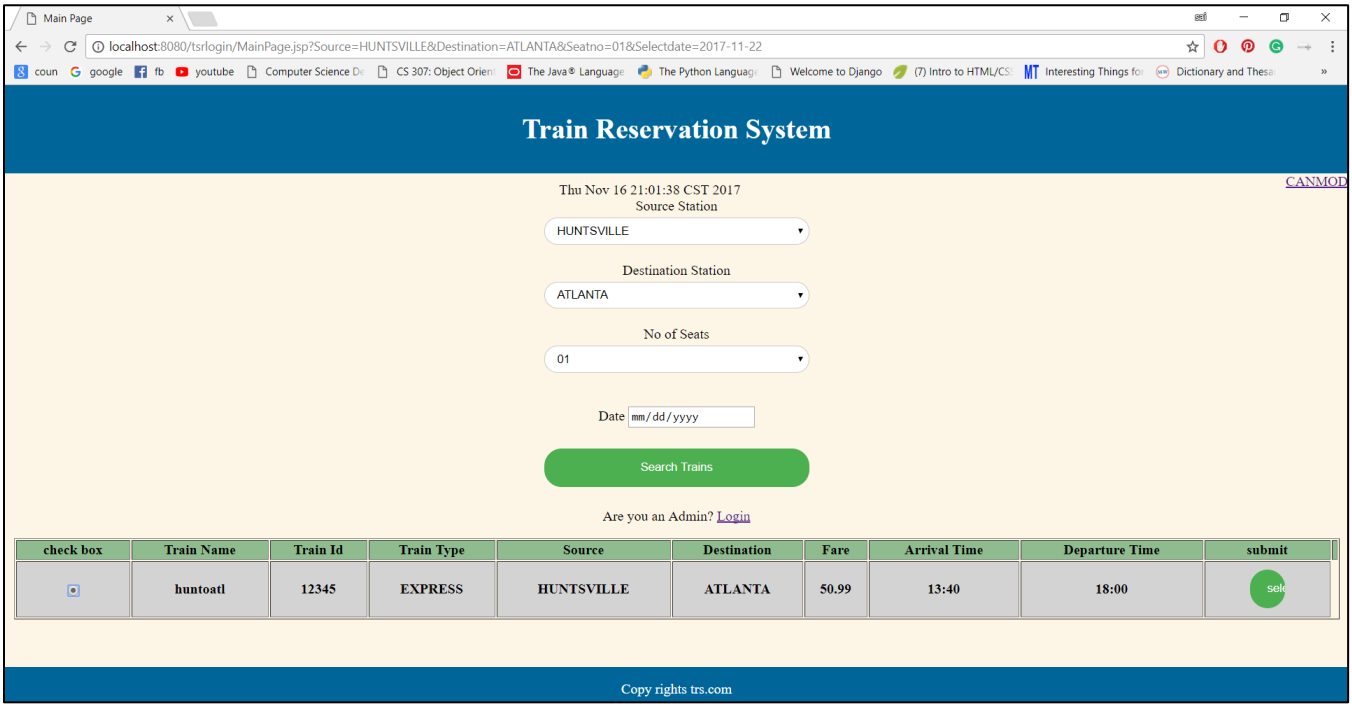
7. What is the size of your database? Are you able to populate your database with realistic data?
   ANSWER: Our database is relatively small around 500 records, we can insert more number of records as per need. We can support realistic data up-to certain extent, but we may have to improve or modify our code to support large database.

8. Do you get proper use of available APIs or SDKs made available by the companies?
   ANSWER: We did not use any API's or SDK's.

USER INTERFACE SNAPSHOTS

Main Page:

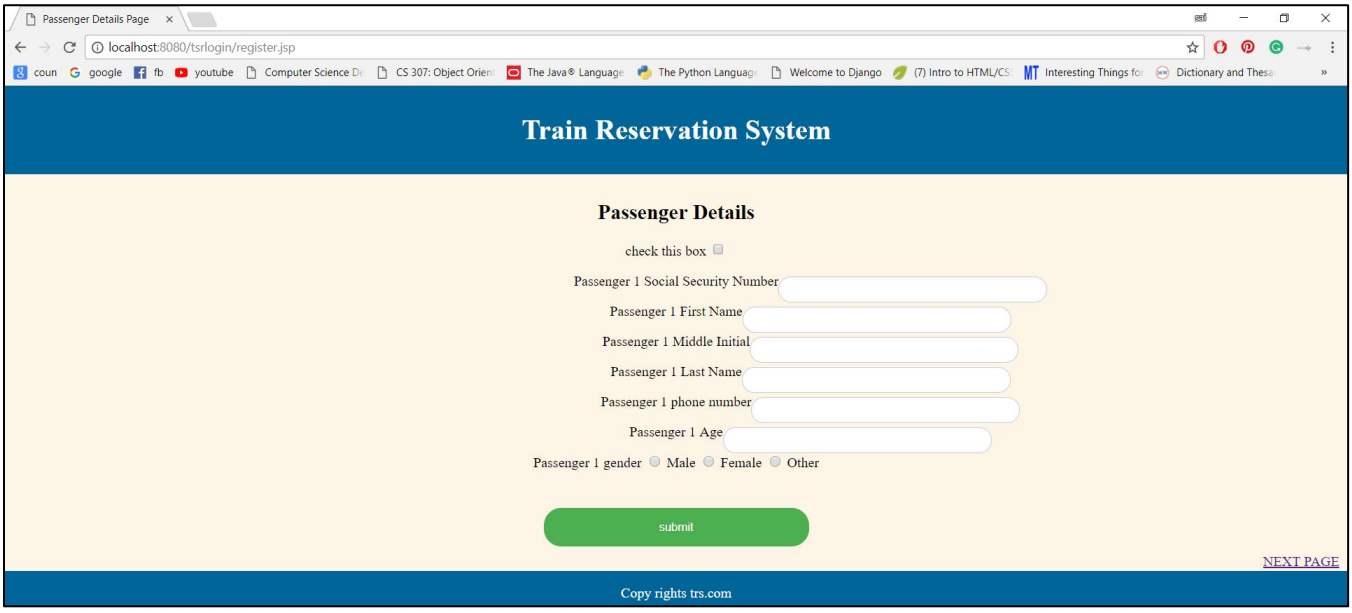
This is our main page with the user desired source and destination. We provide information about the trains available.

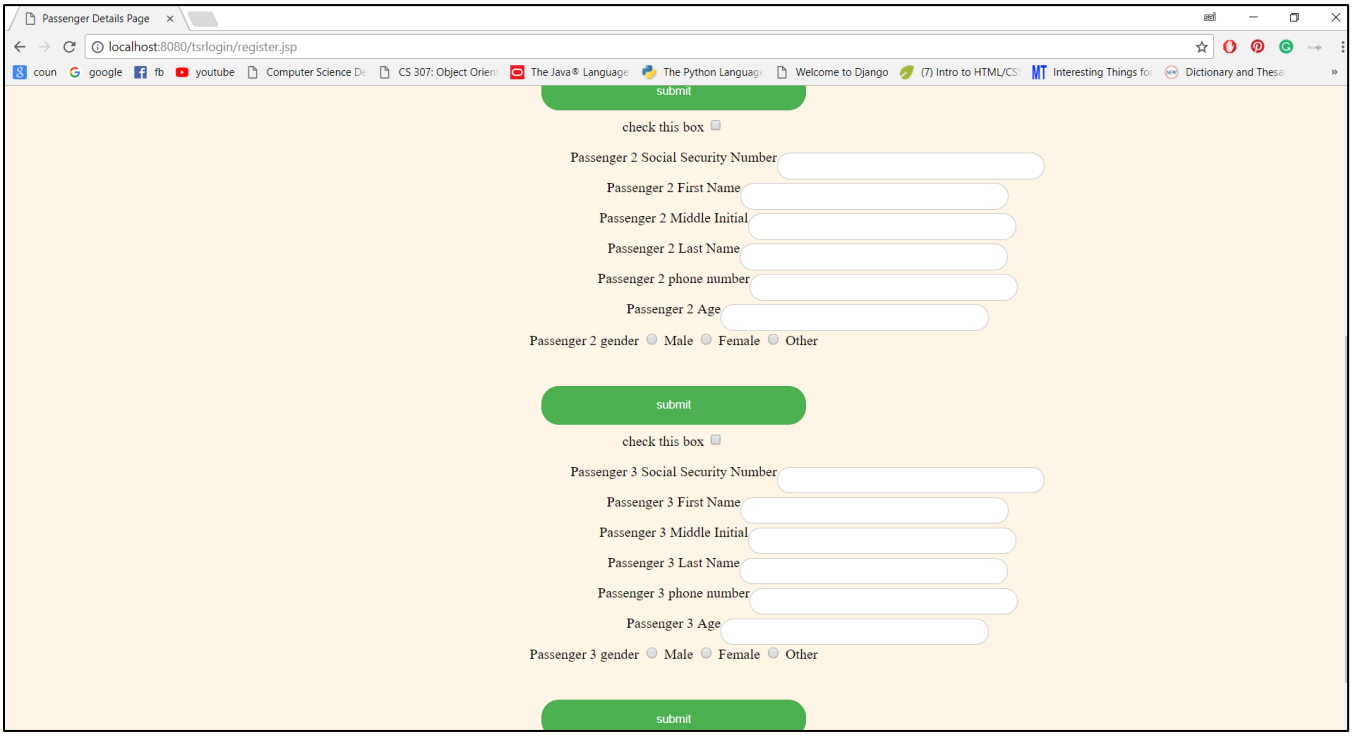


Passenger Details Page:

In this page, passenger can provide their details required for booking the train ticket.

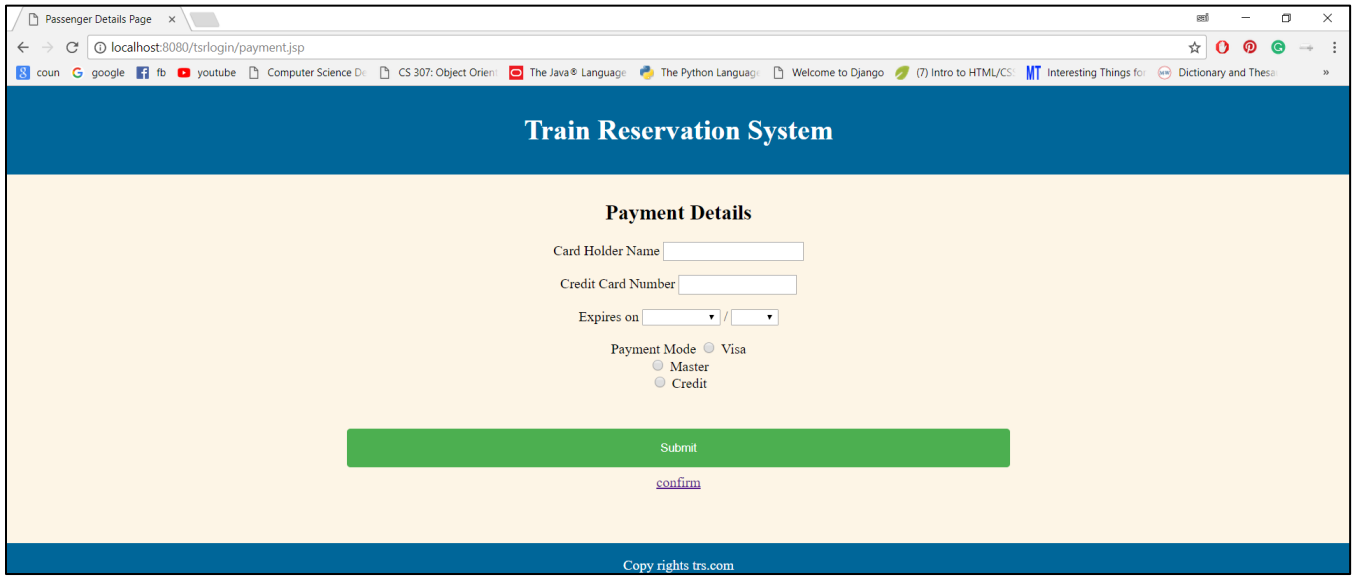
We also included screenshots, if passenger count is more than 1.





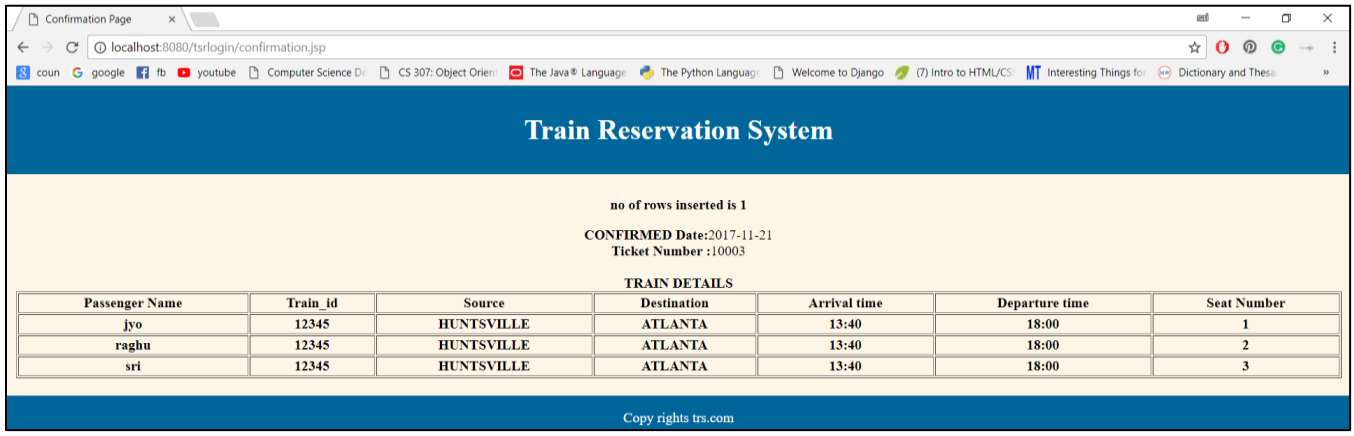
Payment confirmation Page:

In this page, passenger can provide their payment information in order to book the ticket.



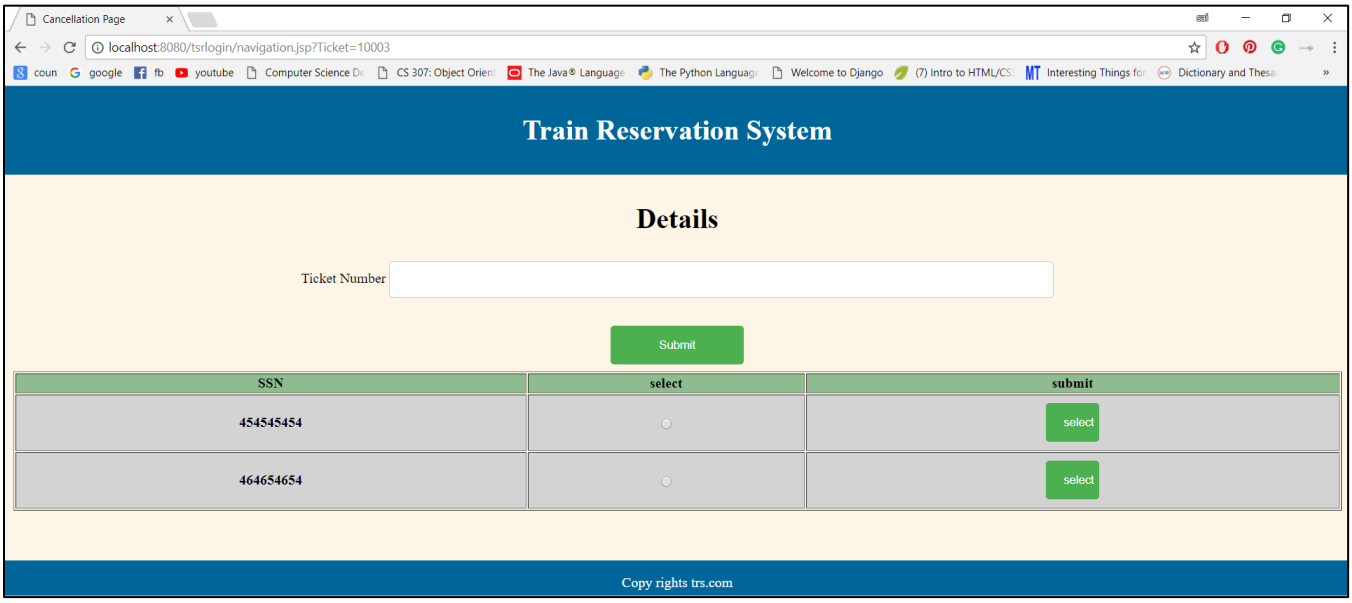
Ticket confirmation Page:

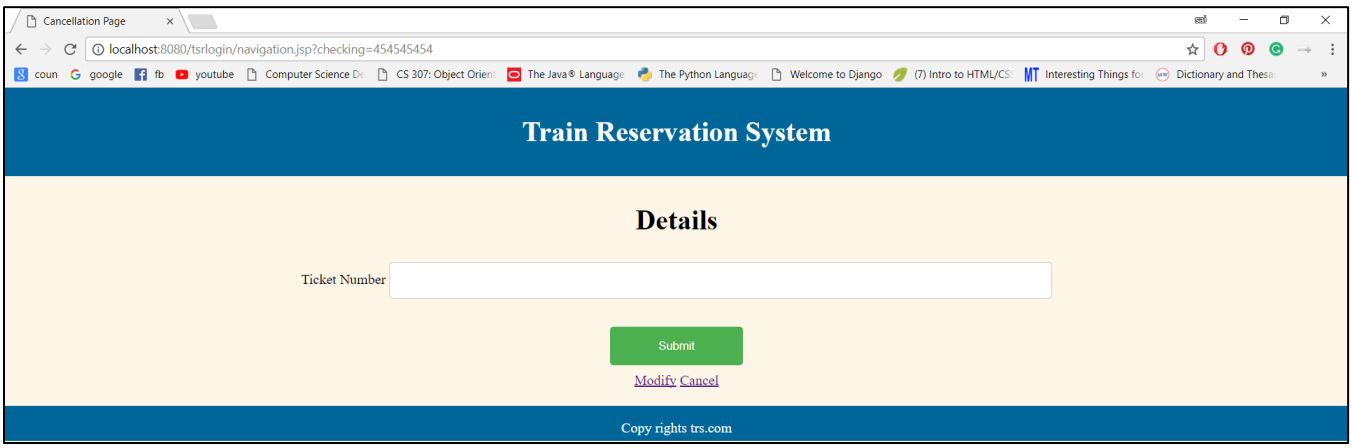
In this page, we provide final confirmation regarding the ticket booked by the passenger.



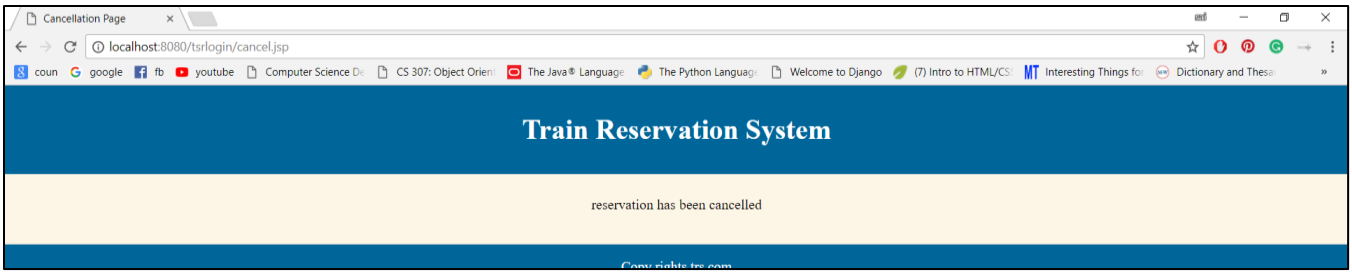
Reservation cancel/modify Page:

In this page, a passenger can cancel/modify their reservation by just providing the ticket number.

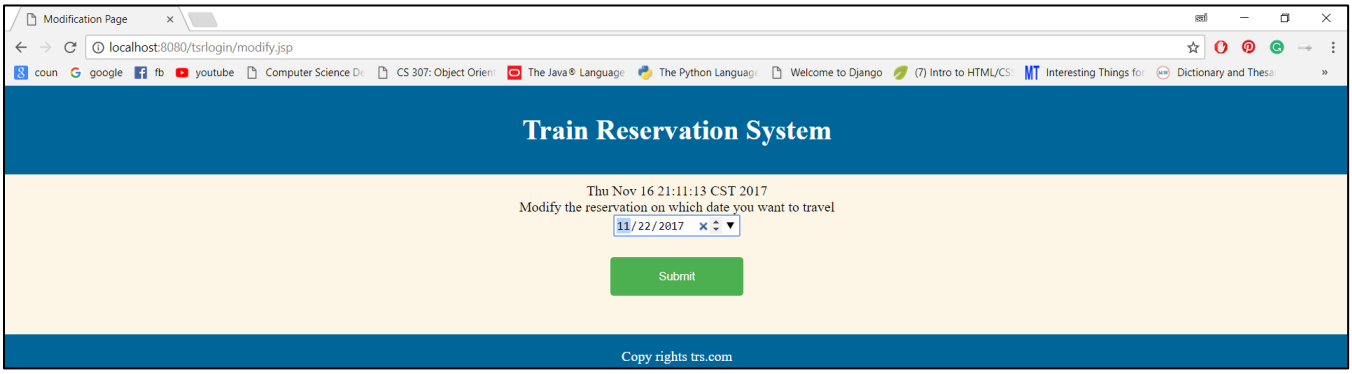




After entering the ticket number, if passenger selects to cancel then the passenger will navigated to cancel page shown below:

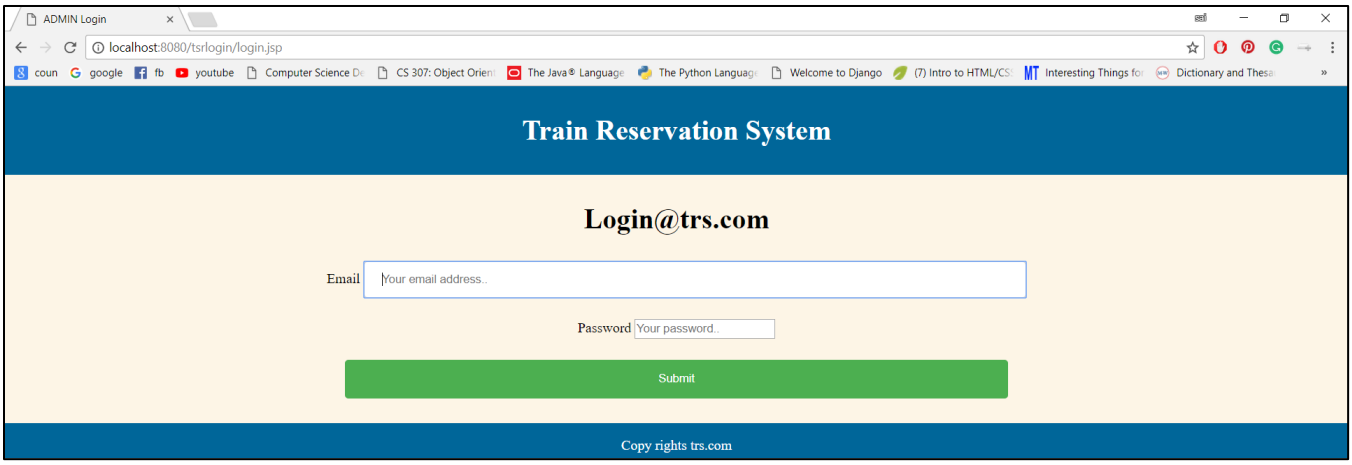


If passenger selects modify option, then the passenger will navigated to modify page shown below:

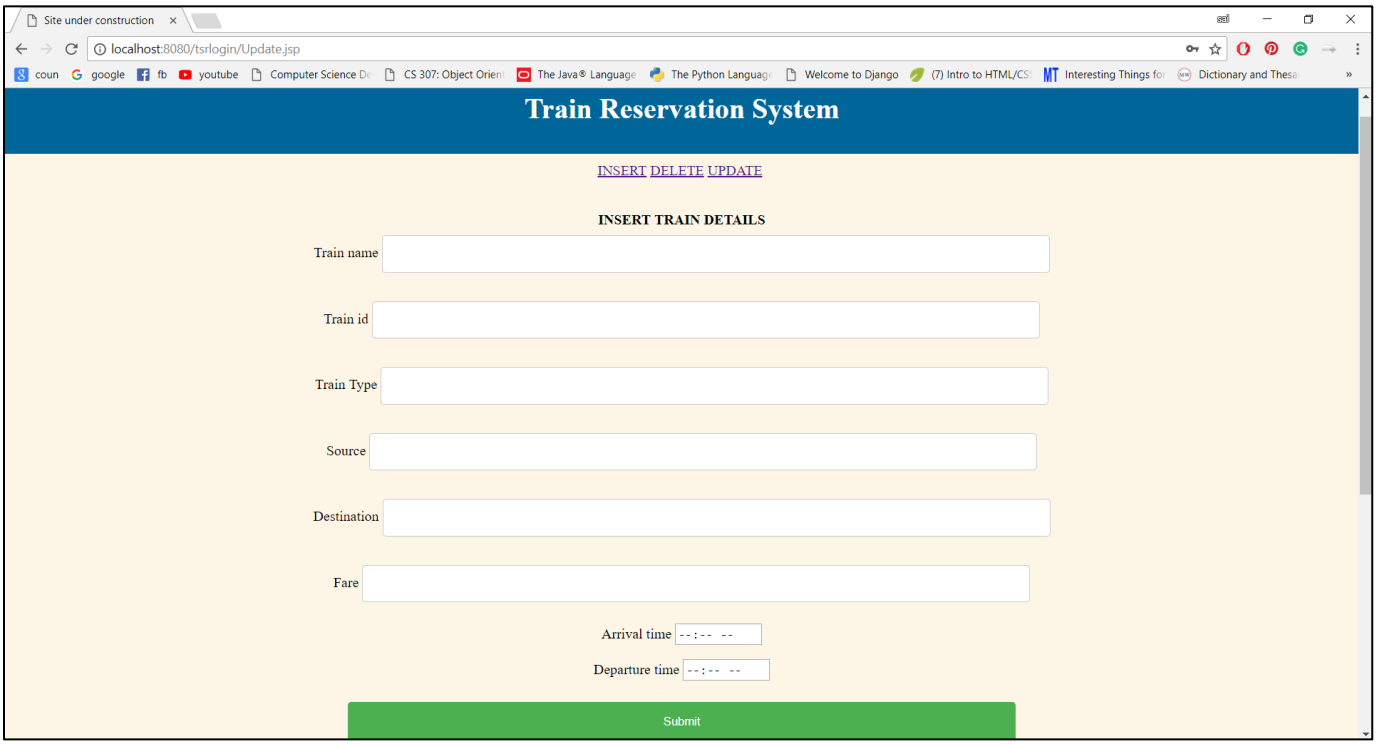


Admin Page:

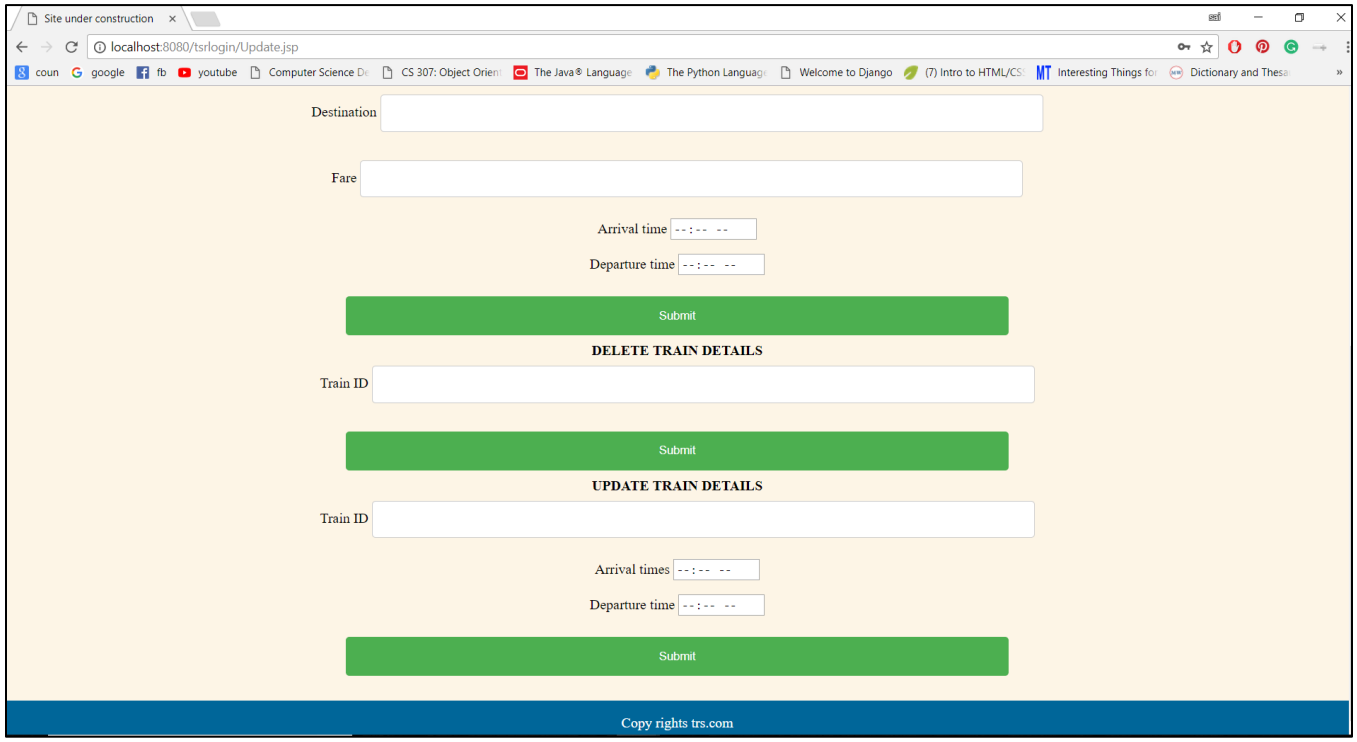
In this page, the admin will use their credentials to login to delete or insert or update train details.



In this page, admin can insert new train details.



In this page, admin can delete or update the existing train details.



Conclusion and Future Scope:

- The application allows the passenger to book a ticket, cancel and modify the reservation by changing the date of travel.
- After successful reservation a passenger will be notified with Ticket_no, Seat_no, Date of travel, Passenger name, Train_id, Source, Destination, Arrival time and Departure time.
- On a single ticket number, a maximum of three people can travel.
- The admin has the authority to update the train details (Insert, Update and Delete).
- Currently we have direct trains from source to destination i.e. single routes. Future scope of the application gives the users flexibility to choose his routes from source to destination. The passenger can update the seat allocation based on his needs and requirements. The passenger is allowed to cancel or update his reservation only 2 days before the travel date.

Minutes of Meeting:

Meeting 1:

Date & Time	**Location**	**Attendees**	**Duration**
August 20th, 4:00PM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	2 Hours
Topic	Selection of the project topic.		
Detailed Information:	• We considered various topics like Library Management, Book store and others, finally we decided to go with the Train Reservation System. • We had discussed on how to start the project and discussed a rough outline of how our application would look like.		

Meeting 2:

Date & Time	**Location**	**Attendees**	**Duration**
August 28th, 9:00AM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	2 Hours
Topic	Discussion regarding the project plan.		
Detailed Information:	• Discussion regarding the description of the database and project plan. • Assigned roles to each other. • Prepared a neat description and project plan which we had to submit before September 5th.		

Meeting 3:

Date & Time	**Location**	**Attendees**	**Duration**
Sept 4th, 2:00PM	Charger Union	SriPriya, Roma, Jyothi, Raghu	1hr
Topic	Review of the project plan before submission.		
Detailed Information:	• Reviewed the project name and plan before submission and each group member added valuable points which needs to be added. • We sent the Project Title and Project Plan.		

Meeting 4:

Date & Time	**Location**	**Attendees**	**Duration**
Sept 12th, 2:00PM	Tech Hall	Sripriya, Roma, Raghu, Jyothi	2 Hours
Topic	Discussion on how to build the project.		
Detailed Information:	• We referred some similar websites like priceline.com and other booking websites to get a general idea on how we should design and implement our project. • We finally came to a general idea on how our project design looks and decided on the design flow of our project.		

Meeting 5:

Date & Time	**Location**	**Attendees**	**Duration**
Sept 20th, 5:00PM	Overlook Apartments	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Project Design and Initialization of ER tables and Schema		
Detailed Information:	• We discussed how the front-end design looks and discussed how to code the front-end. • We came up with a ER Diagram Design • We started writing tables for our schema.		

Meeting 6:

Date & Time	Location	Attendees	Duration
Sept 28th, 5:00PM	Charger Union	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Complete discussion of Tables and Schema		
Detailed Information:	• We have discussed thoroughly and created all possible tables and tuples for our project. • Crosschecked any mistakes and saved our work.		

Meeting 7:

Date & Time	Location	Attendees	Duration
Oct 11th, 2:00PM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	2 Hours
Topic	Discussed about Front-end and Back-end.		
Detailed Information:	• We bounced off ideas regarding Front-end and back-end and discussed on how to connect front-end and back-end.		

Meeting 8:

Date & Time	Location	Attendees	Duration
Oct 23rd, 5:00PM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Initialize work for Formal description submission.		
Detailed Information:	• Started working on formal description. • We wrote Entities, Attributed for those entities, Relationships etc.		

Meeting 9:

Date & Time	Location	Attendees	Duration
Oct 24th, 2:00PM	Charger Union	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Drawing ER Diagram.		
Detailed Information:	• Discussed and drew ER Diagram for our application. • Raghu and Roma worked on creating and modifying the ER all other relationships.		

	• Jyothi and Sripriya worked on Database and creating schema. • Together we finalized ER and Schema.

Meeting 10:

Date & Time	**Location**	**Attendees**	**Duration**
Oct 25th, 5:00PM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Review of our Formal Description.		
Detailed Information:	• We reviewed our formal description and corrected mistakes if any, and mailed to the Dr. Aygun.		

Meeting 11:

Date & Time	**Location**	**Attendees**	**Duration**
Nov 3rd, 5:00PM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Implementing the Front-End Design.		
Detailed Information:	• Started implementing the front-end design. • Everyone contributed in designing the server pages and made it run on local host. • Front-end has been implemented.		

Meeting 12:

Date & Time	**Location**	**Attendees**	**Duration**
Nov 7th, 2:00PM	Tech Hall	SriPriya, Roma, Jyothi, Raghu	4 Hours
Topic	Implementing Back-end design.		
Detailed Information:	• We have used JDBC for the database connectivity		

	• We have implemented the insert, delete and update queries on java side and tested the application by running it on localhost. • Simultaneously we verified database outputs

Meeting 13:

Date & Time	Location	Attendees	Duration
	Overlook Apartments	SriPriya, Roma, Jyothi, Raghu	4 Hours
Topic	Testing the project.		
Detailed Information:	• We tried all the page functions and tuple insertion/deletion/updating and also tried the admin pages and his functions. • We added more tuples and tried the above functions, and got good results. • We started to work on final report and Demo.		

Meeting 14:

Date & Time	Location	Attendees	Duration
Nov 18th, 9:00 AM	Overlook Apartments	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Review of the Project.		
Detailed Information:	• Reviewed our project to see if any change is required. • Finally checked the tuple insertion/deletion/updating.		

Meeting 15:

Date & Time	Location	Attendees	Duration
Nov 20th, 5:00PM	Overlook Apartments	SriPriya, Roma, Jyothi, Raghu	3 Hours
Topic	Report and preparation of Demo.		
Detailed Information:	• Reviewed the project report. • Discussed on how to give the demo in less than 7 minutes.		

REFERENCES :

1. Database connectivity
   http://infolab.stanford.edu/~ullman/fcdb/oracle/or-jdbc.html#0.1_executeUpdate

2. Implementing JSP
   http://www.java2s.com/Tutorial/Java/0360__JSP/Insertdatatoatable.htm
3. Creating ER diagram, Creating Schema's
   Fundamentals of database systems 7th Edition Ramez Elmasri.