Spring + React Assessment

Q5) Railway Reservation System

The Railway Reservation System will provide the available Train-list, and Seat availability, via-details. To book a ticket passengers can pay through online/offline mode. After successful payment of the ticket fare the System will generate the ticket and PNR no. will be given to the passenger.

By ,

Abishek K – 12120

Railway Reservation System – W3H Analysis

What?	How?
What is the implementation required to be implemented? **USER:** **<u>1) Question:</u>** **Do you want the user to register in your webpage?** Ans: yes, user want to register the page. **<u>2) Question:</u>** **Do you want the user to book the train?** Ans: yes, user want to book the train. 3**<u>) Question:</u>** **Do to you want the guest to add/update/delete/view their details?** Ans: yes, user want to do this operation. **<u>4) Question:</u>** **How user pay fare for the train?** Ans: yes, user want to pay the fare for train, by payment method. **<u>5) Question:</u>** **User able to give reviews ?** Ans: yes, user make an reviews. **<u>6) Question:</u>** **User able to cancel the ticket?** Ans: yes, user can able to cancel the ticket	**<u>1)Question:</u>** **Method1:** Through the mobile number. **Method2:** Through the email-id. **Method3:** Through the both number or email-id. **<u>2)Question:</u>** Method1: By note the starting point of the user Method2: By note the ending point of the user. Method3: By note both the starting point and ending point. **<u>3)Question:</u>** **Method1:** User able to add/update details **Method2:** User able to delete/ view details **Method3:** User able to add/update/delete/view their details. **<u>4)Question:</u>** **Method1:** Through the bank transfer **Method2:** Through the upi, cards method **Method3:** Through the both bank and card,upi **<u>5)Question:</u>** **Method1:** Review through the Text message. **Method2:** Review through the Audio. **Method3:** Through the both Text and Audio

<u>ADMIN:</u>	**<u>6)Question:</u>**

7) Question: **Do you want the admin to add/update/delete the room details?** Ans: yes, admin able to do this operations. **8) Question:** **Do you want the admin can able to monitor the all activities?** Ans: yes, admin able to monitor. **9) Question:** **The admin can able to cancel the ticket status?** Ans: yes, admin can able to cancel. **10) Question:** **Admin can able to view the user details?** Ans: yes, admin can able to view.	**Method1:** Through train number. **Method2:** Through the ticket id **Method3:** Through the both ticket-id and train number **Admin:** **7)Question:** **Method1:** Admin able to add/update details train details. **Method2:** Admin able to delete/ view details **Method3:** Admin able to add/update/delete/view details of train. **8)Question:** **Method1:** Entering the admin name. **Method2:** Entering the admin-id. **Method3:** Entering the admin login credentials. **9)Question:** **Method1:** Through the specific train number to cancel.. **Method2:** Through the train name to cancel the room. **Method3:** Through the specific ticket-id.

	10)Question: **Method1:** Enter the user name. **Method2:** Enter the user Mobile number. **Method3:** Enter the user login credentials to view the details of guest.

Why?	**Why Not?**
1)Question: **Method3:** Through the both number or email-id. (Through the both number and	**1)Question:**

email-id user can easily register and login into the page) **2)Question:** **Method3:** By note both the starting point and ending point. (By note the both starting and ending point only we can easily book the ticket) **3)Question:** **Method3:** user able to add/update/delete/view their details. (If giving an access to guest can to do all operation and it is user friendly for the guest)	Some of them are not having the email-id, if provide the mobile number easily to register and login. **2)Question:** By note the both starting point and ending point of the user is important. **3)Question:** If restrict the some operation to the user it make difficult to the user to make some operation in the page. **4)Question:** By making the many payment method it is very easier for the guest to pay and avoid the failures. **5)Question:** By giving the review in both text and audio other user can easily understand about the hotel facilities

4)Question: **Method3:** Through the both bank and card methods. (By many payment methods to avoid the transaction failure) **5)Question**: **Method3:** Through the both Text and Audio (By giving the review in both text and audio other user can easily understand about the hotel facilities) **6)Question:** **Method3:** Through the both ticket-id and train number. (It is easy for to cancel the ticket user friendly) **Admin:** **7)Question:**	**6)Question:** If the user cannot able to cancel the ticket it is not an user friendly to the user or passenger. **7)Question:** We cannot the restrict the operation to the admin. **8)Question:** By entering the admin name and id it is not safe for the logging other than the login credentials. **9)Question:** Through the specific train name and number make difficult to cancel the train. **10)Question:**

Method3: Admin able to add/update/delete/view details of room. (It is the main operation of the admin.) **<u>8)Question:</u>** **Method3:** Entering the admin login credentials. (By entering the login credentials of admin is the only way to safe to login in the admin portal) **9)Question:** **Method3:** .Through the specific room-id. (Through the specific room-id only easily cancel the room and avoid the confusion about the room details) **<u>10)Question:</u>** **Method3:** Enter the guest login credentials to view the details of guest (It is the safest way to login, to perform the operation in the guest profile admin can able to login)	By entering the user mobile number and name is not safe and make difficult to view the user portal.

Algorithm for the User:

Step-1: START

Step-2: open the browser.

Step-3: open the Railway Reservation Page (Login page).

Step-4: Enter the login credential of the user, if the credentials are correct it will redirect into the reservation page. If the credentials are wrong, it again redirects to the main page.

Step-5: Reservation page will open.

Step-6: Enter the starting point and ending point of the user.

Step-7: Available trains on the route will be shown.

Step-8: choose the train, check the availability of the seat in the train.

Step-9: If the seat is available you will move to the booking process, or else the seat is not available you will redirect to choose another train.

Step-10: If the seat is available and the train is chosen, you will redirect to the booking process.

Step-11: Enter the passenger details like name,age,mobile number, no.of. Passengers.

Step-12: Select the anyone of the payment method to pay the fare.

Step-13: If the payment is made you will receive the confirmation email and e-ticket.

Step-14: END

Algorithm for the Admin:

Step1: START

Step2: open the browser

Step3: open the Railway Reservation Page (Admin-Login page)

Step4: Enter the login credentials

Step5: The Login credentials are correct; you move to main page. Otherwise, Re-Enter the Login credentials.

Step6: Add the details of the Train/User.

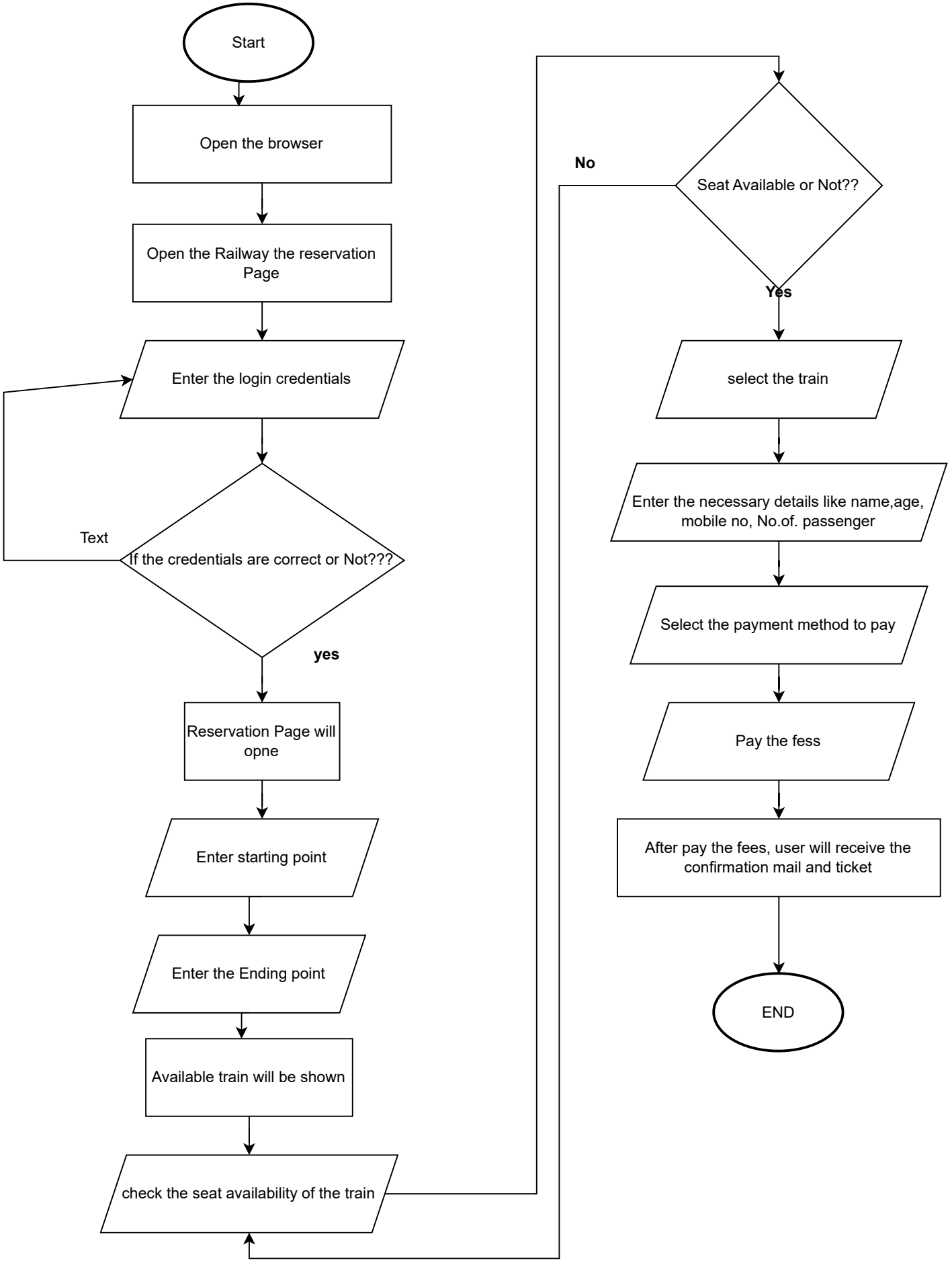
Step7: Update the details of the Train/User.

Step8: View the details of the Train/User.

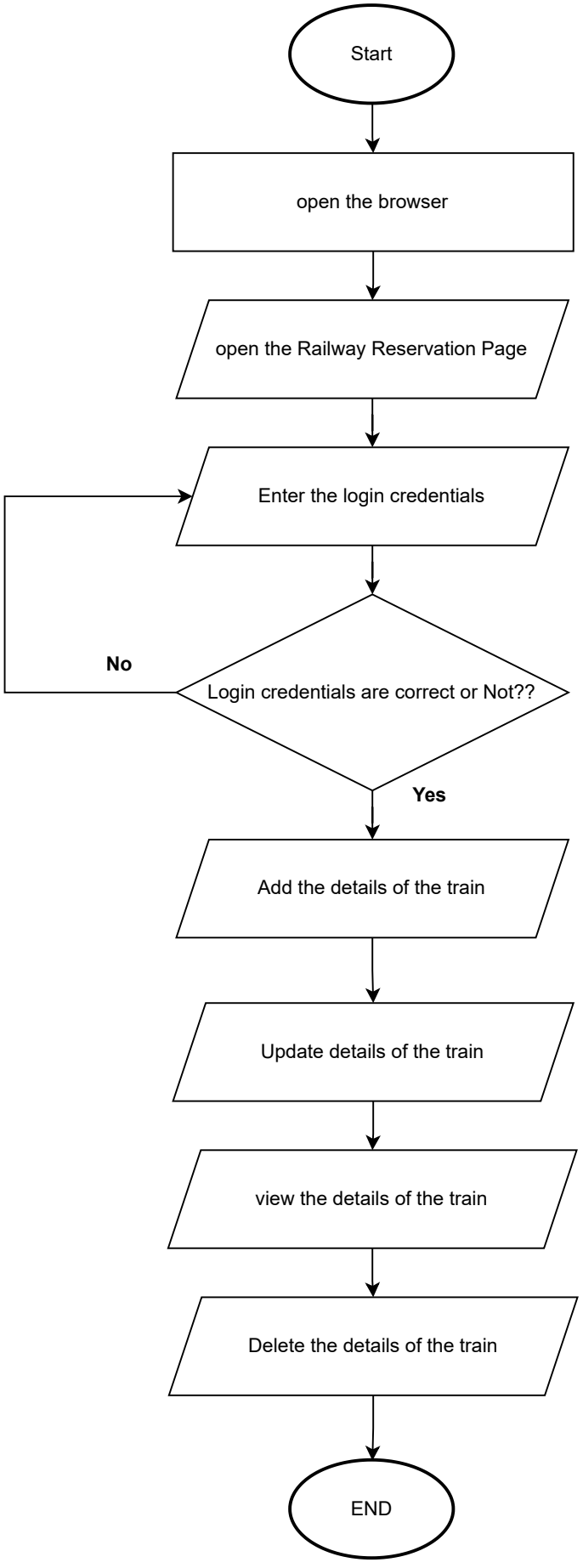
Step9: Delete the details of the Train/User.

Step10: END

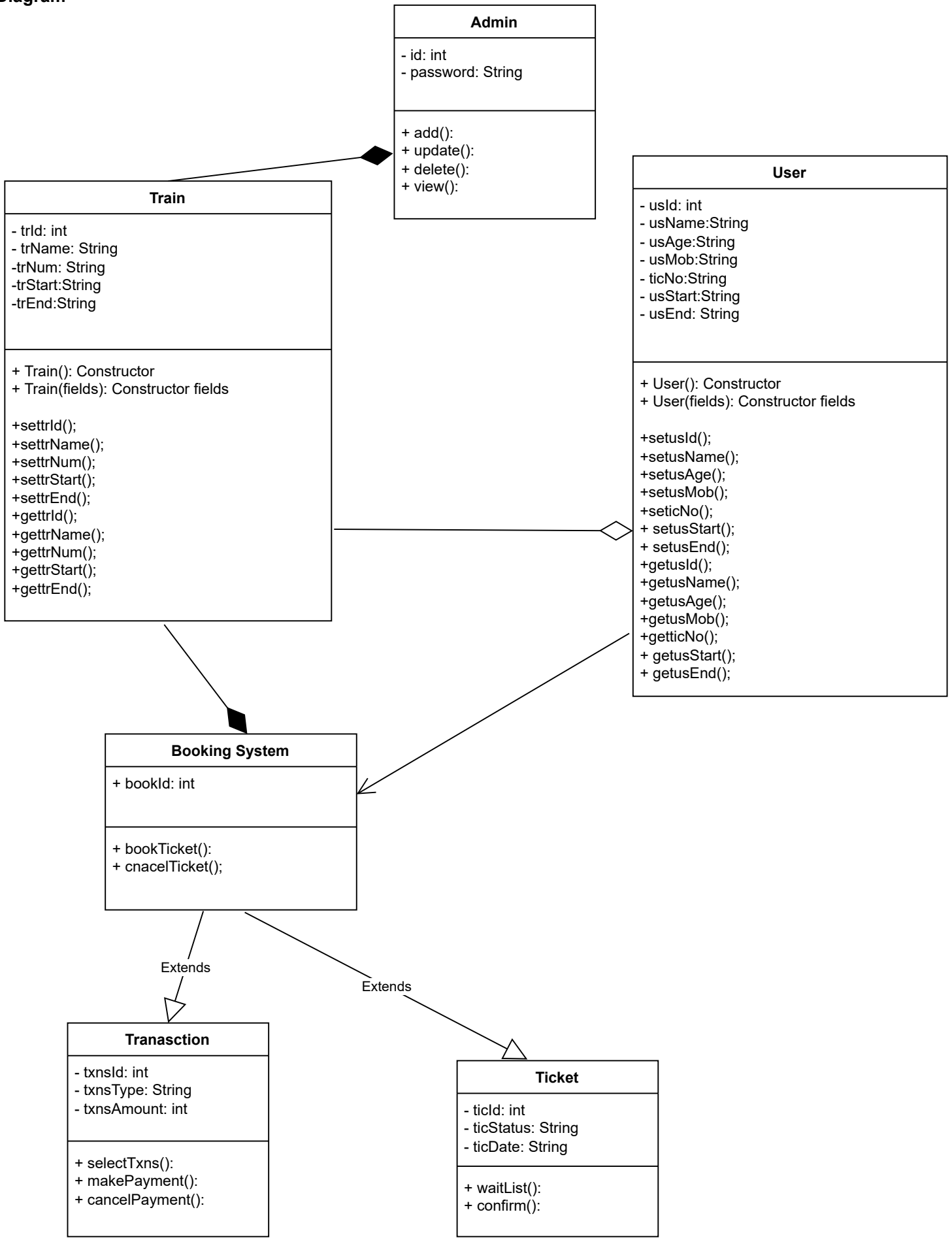
Flowchart - User



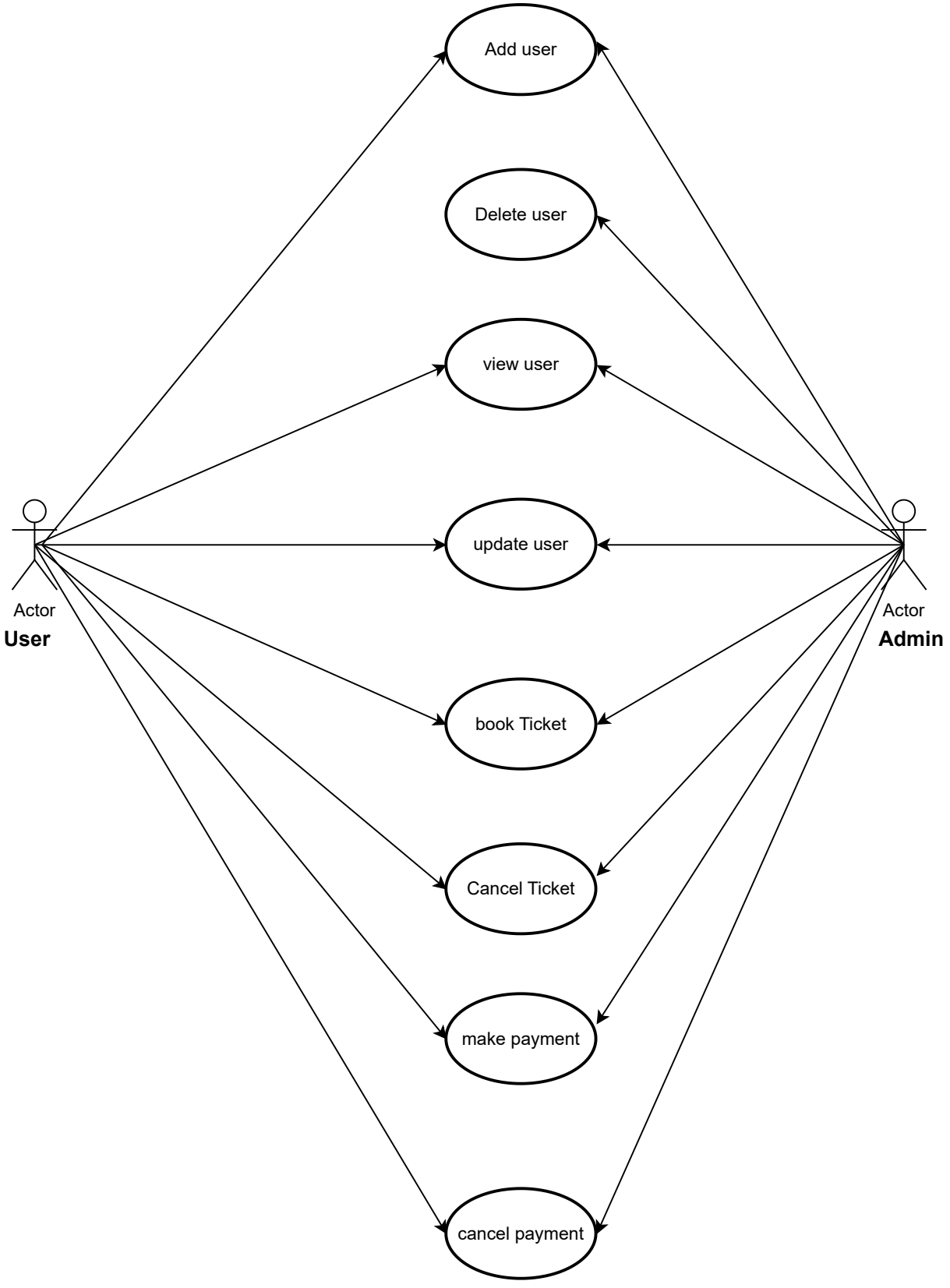
Flowchart Admin



Class Diagram

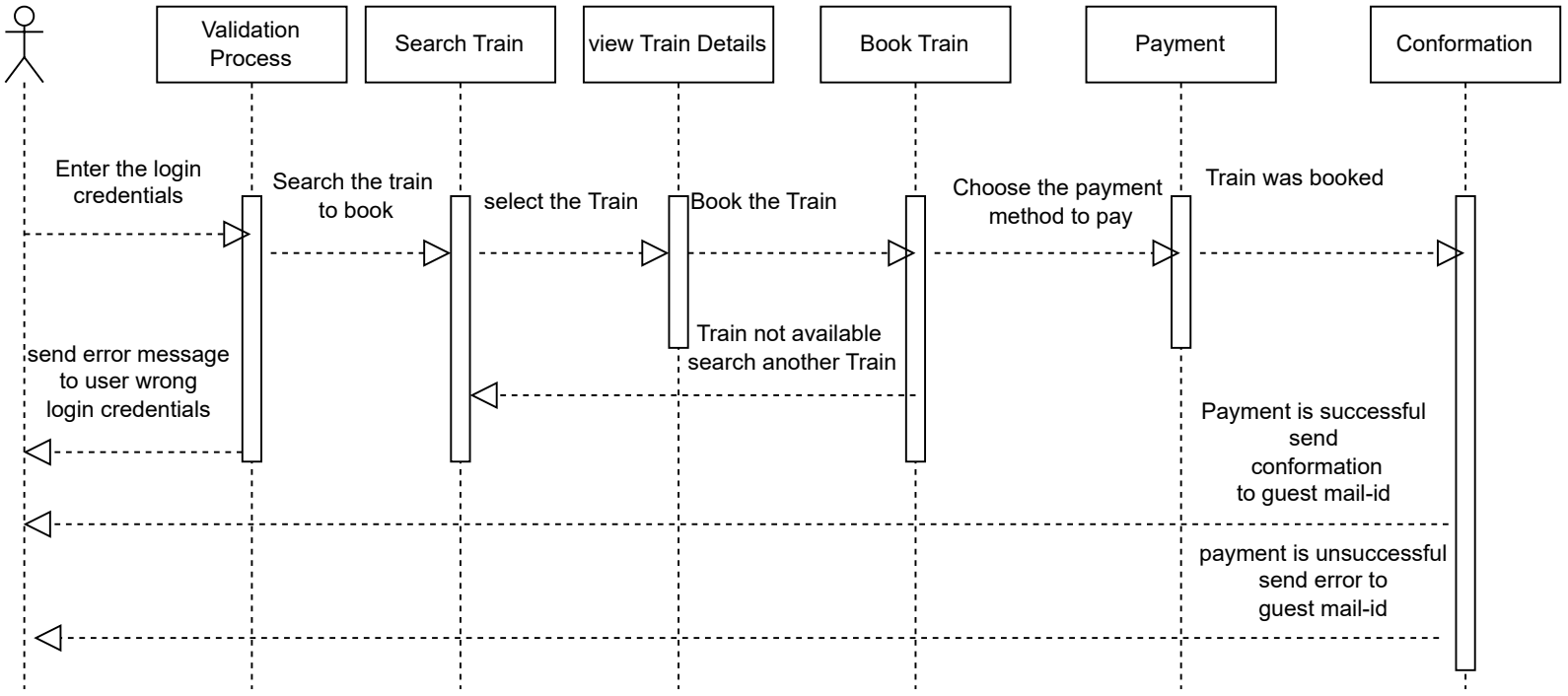


UseCase Diagram

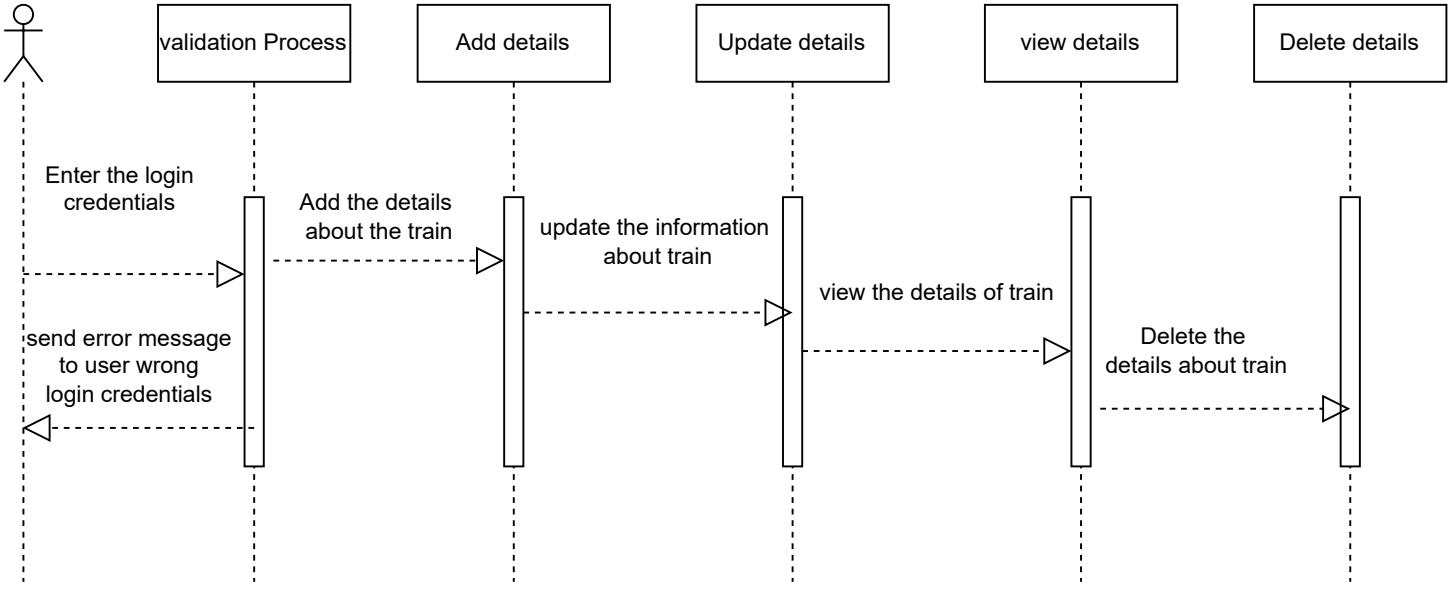


Sequence Diagram

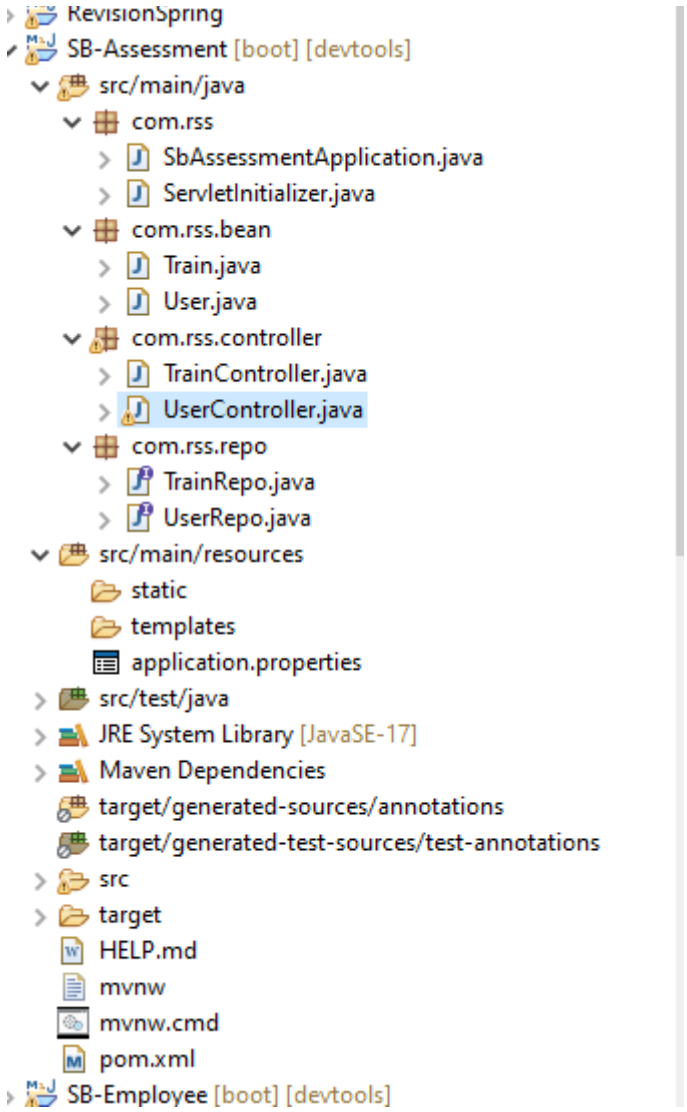
USER



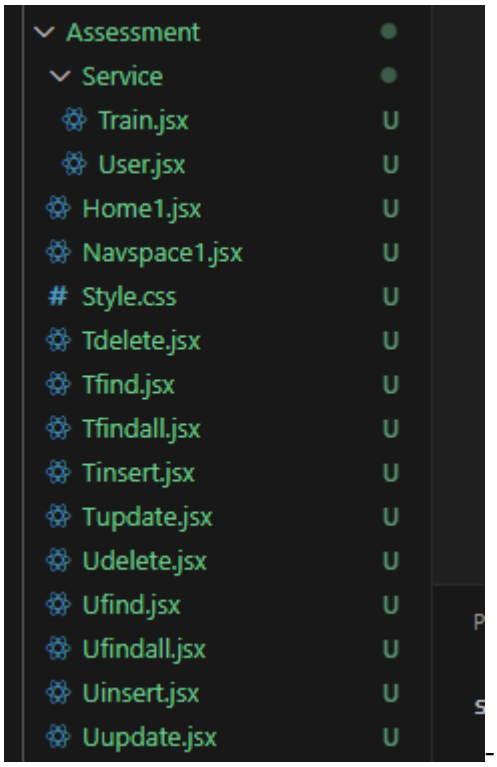
Admin



Folder Structure in SpingBoot:

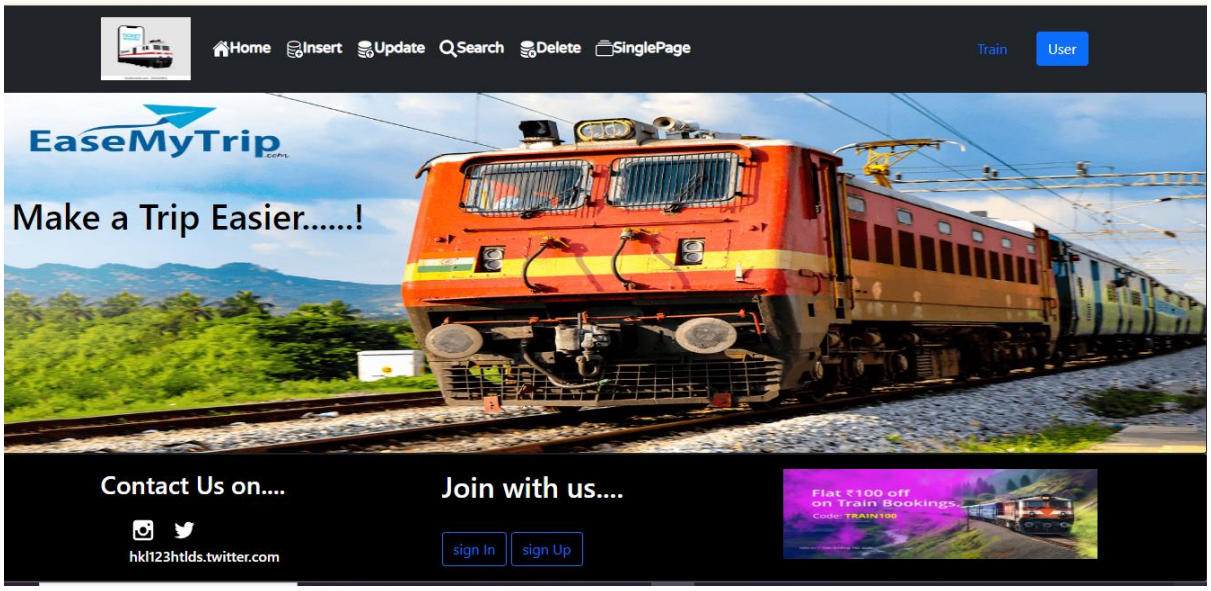


Folder Structure in React:



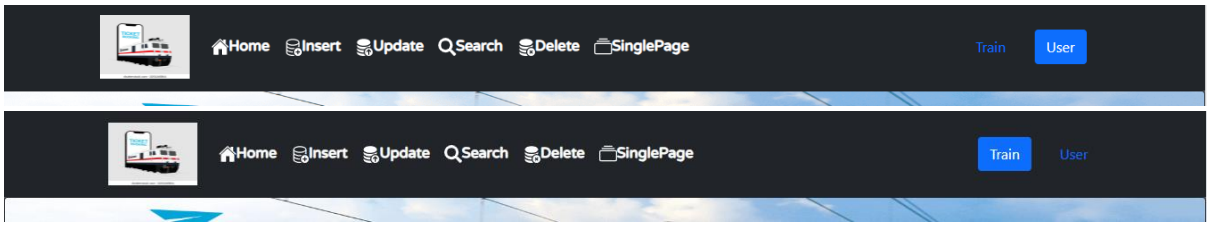
Crud-Operations:

Landing-Page:



Nav-Bar:

In the nav-bar, i had used two components like Train and User, Bu the Nav-link are common for the both components,because i had done a conditinal rendering in the nav-bar.Like if i click the Train-component the nav-link will switched to the Train link pages. If i click the User-component the nav-link will switched to the User link pages. We can easily find out the active component by having backgroung-blue in it.If i click the Train component the it will become active, and it's background color will be blue and it will occur for the user also.So that we can easily findout the active component or we are in the which component or page.

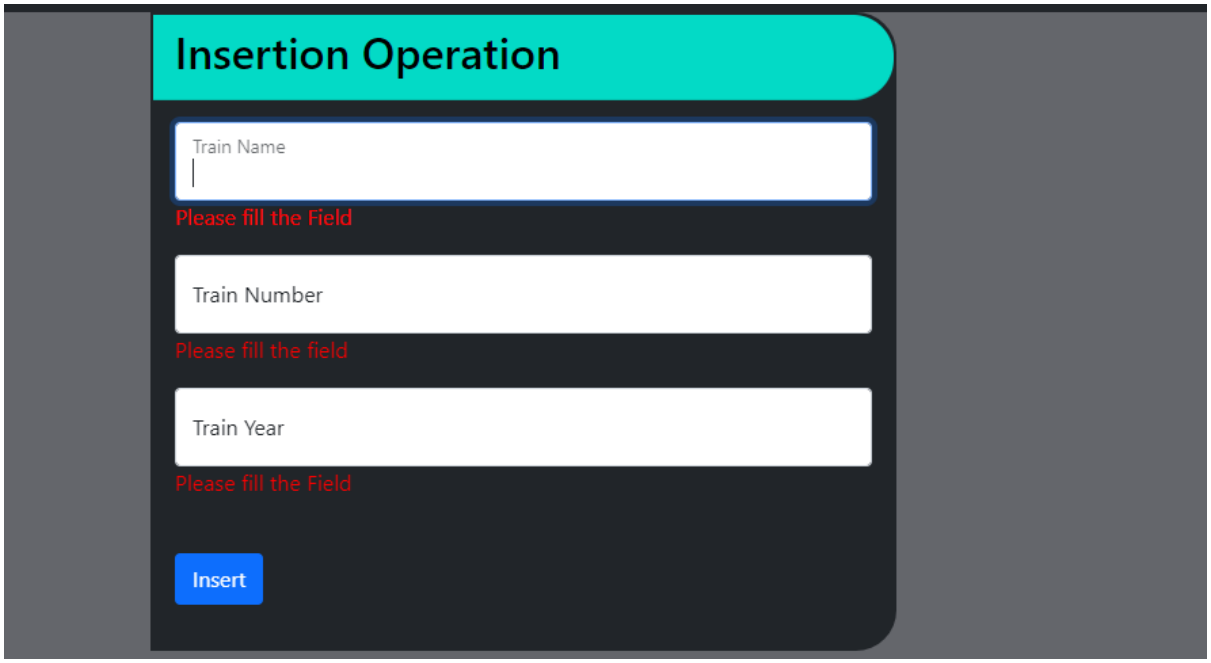


Insert Operations:

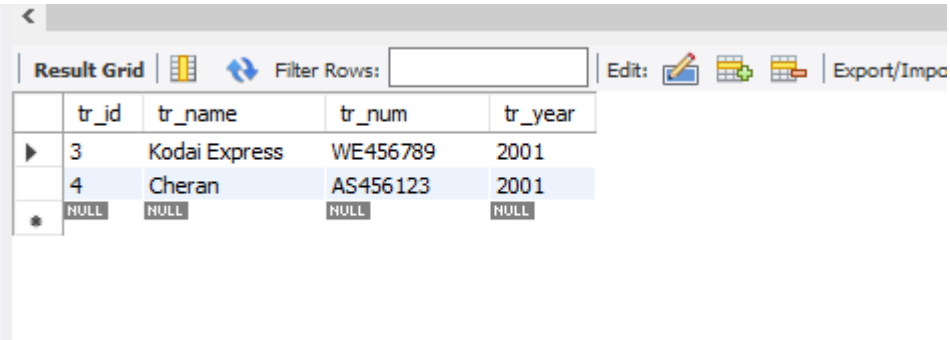
In the case study I used the one Tomany relationship. Train is one and the User will be many.

Train:

In that validation is occurs:



Before Database:



tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	Cheran	AS456123	2001
NULL	NULL	NULL	NULL

After Inserting Data:

After Database:

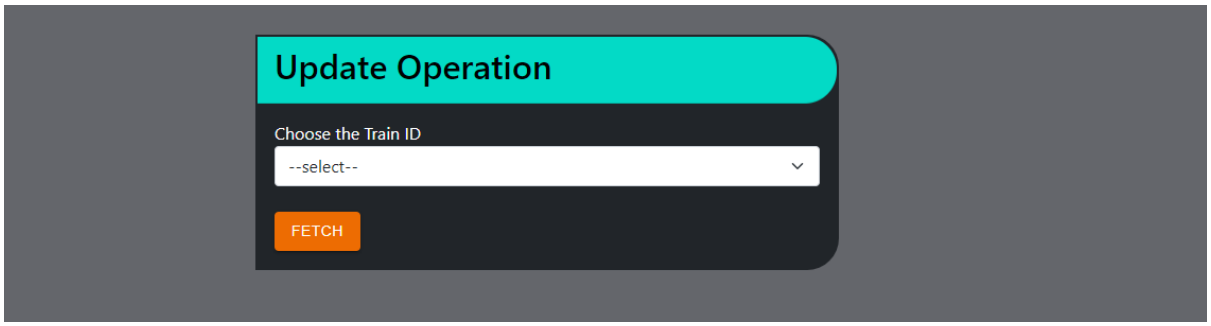
Result Grid | Filter Rows: | Edit: | Export/I

tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	Cheran	AS456123	2001
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

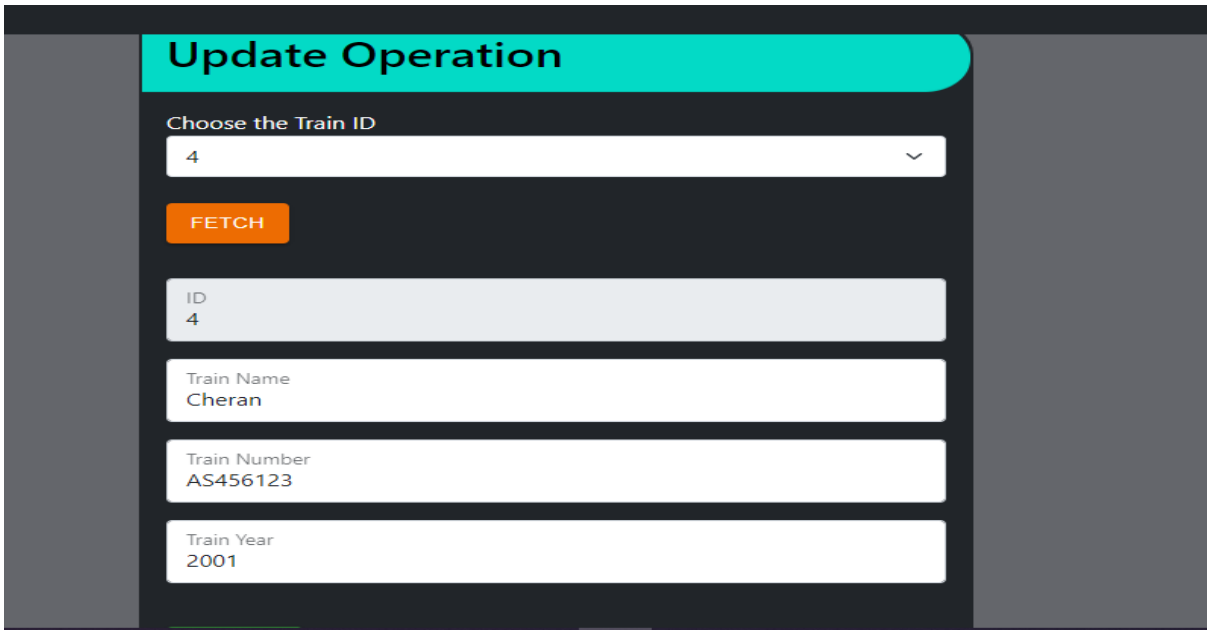
Updating Operations:

I used the fetch operation to fetch the old details in the details in the database.

Before Fetch:



After Giving the Train Id:



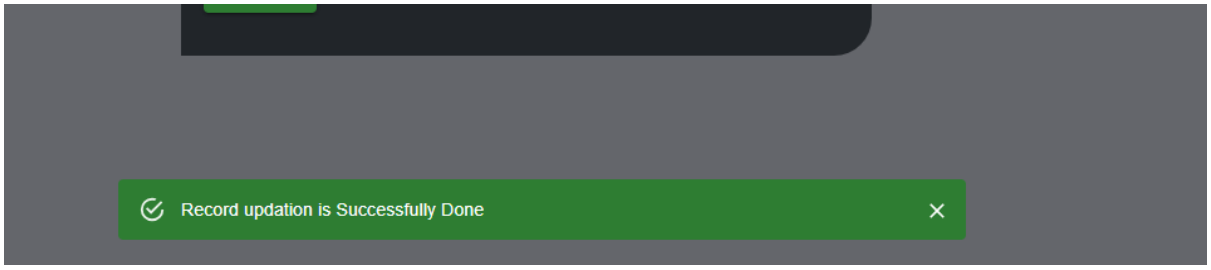
Before Database:

Result Grid | Filter Rows: | Edit: | Export/Import:

tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	Cheran	AS456123	2001
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

Updating Details:

I changed the 4th row details,



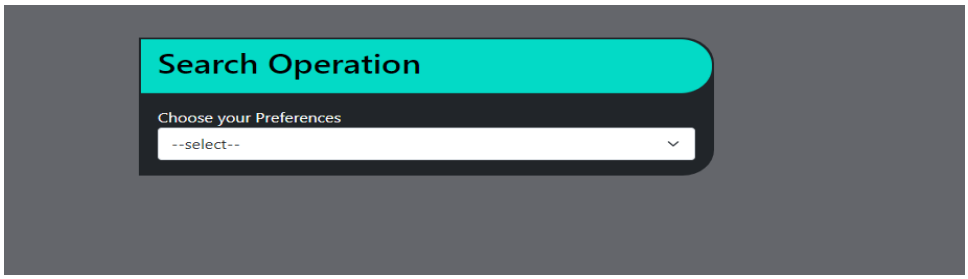
After Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wr

tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	VadheBharat	AS456123	2022
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

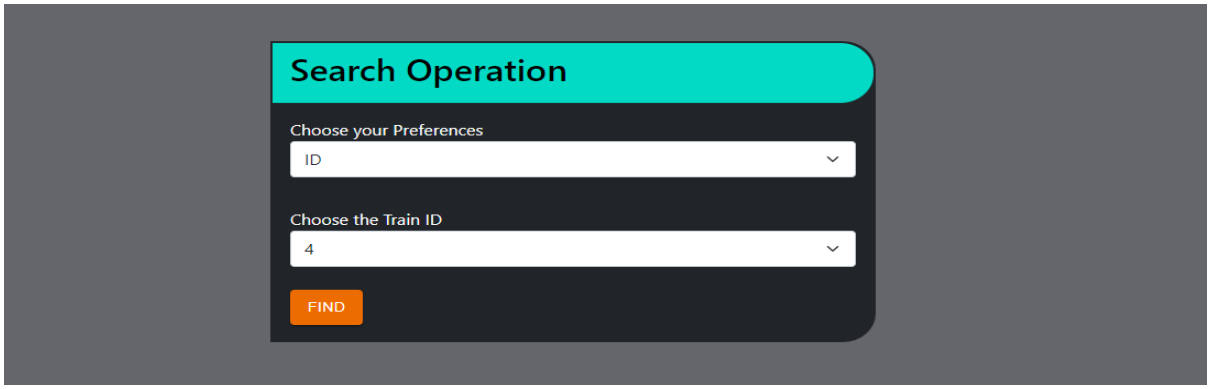
Find-Operation:

Find by done by both the Id and Name:

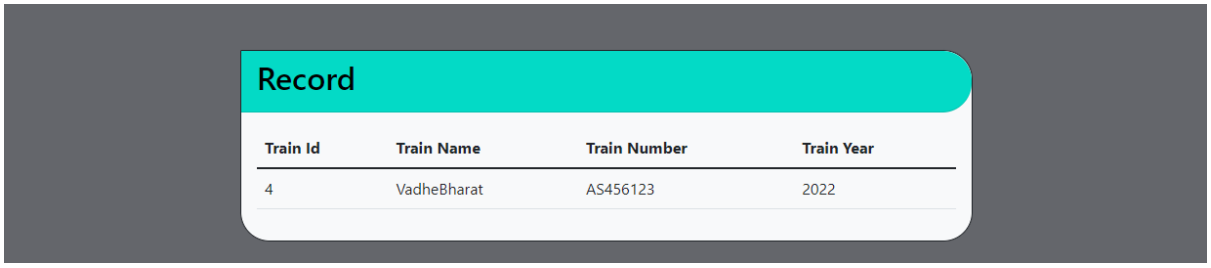


We have to choose the preference like Id or Name,

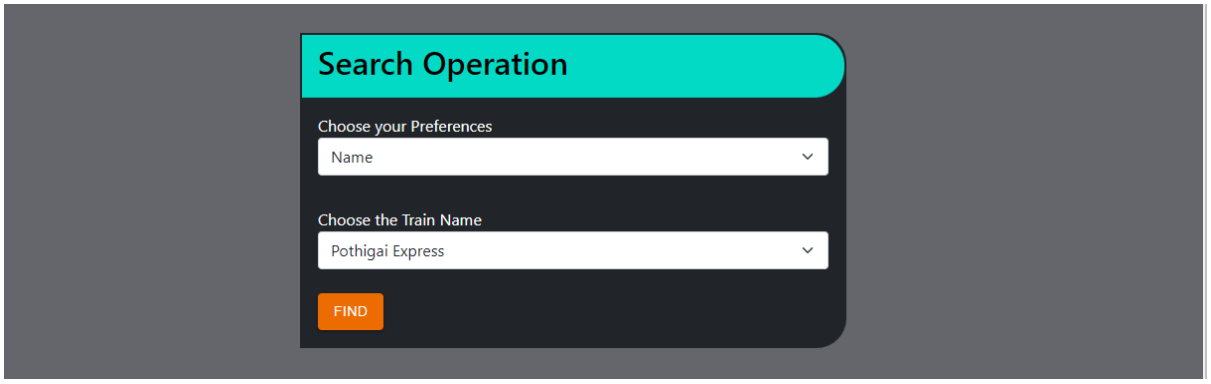
By Id,



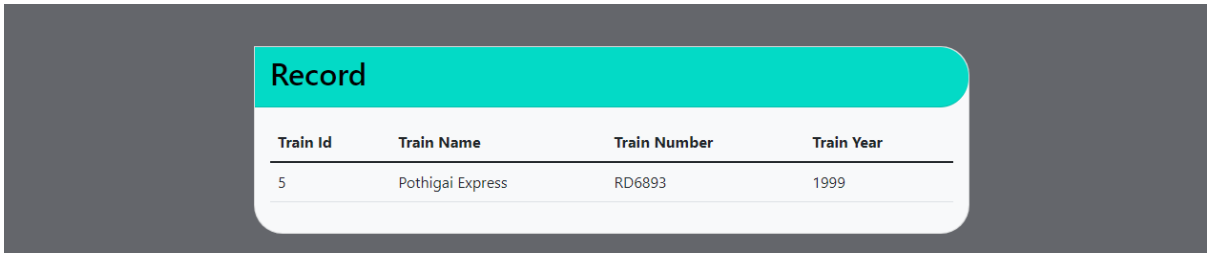
Output:



ByName:



Output:



Delete-Operations:

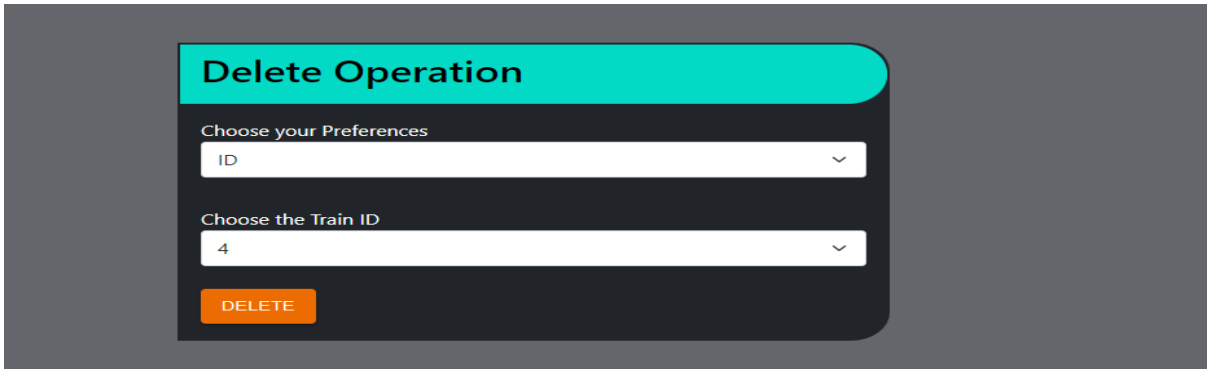
Delete done by both Id and Name:

Before Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap

tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	VadheBharat	AS456123	2022
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

After Deleting By Id,



I am able to delete the 4th train id, i can't able delete it because it map with User entity. It primary key is act as an foreign key in user table.

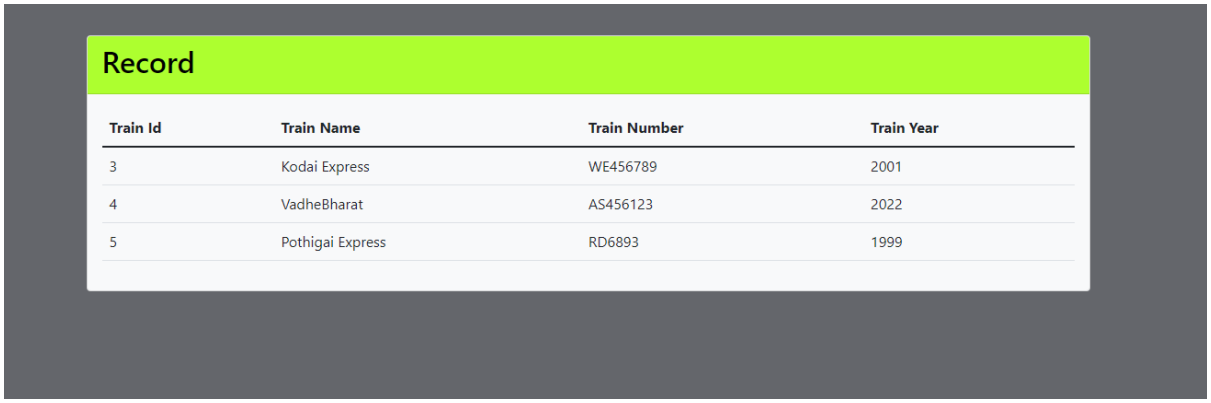
After Database:

Result Grid | Filter Rows: | Edit: | Export/Import: |

tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	VadheBharat	AS456123	2022
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

When I delete the particular user who are mapped with train id 4, then only i can able to delete it.

Find All-Operation:



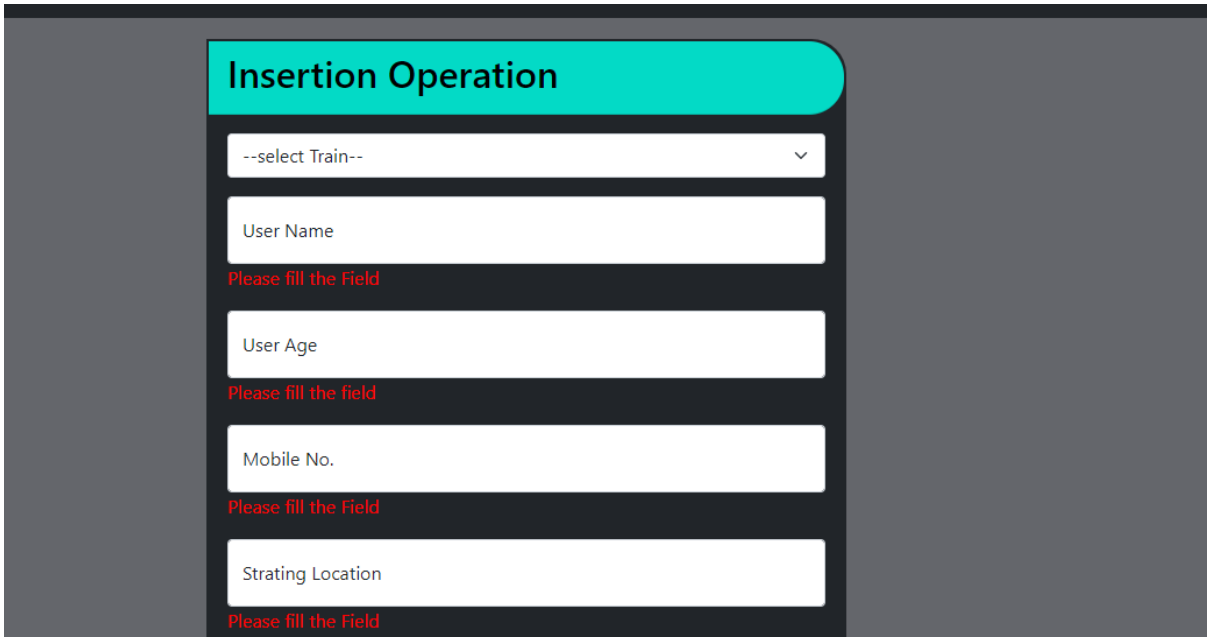
Record

Train Id	Train Name	Train Number	Train Year
3	Kodai Express	WE456789	2001
4	VadheBharat	AS456123	2022
5	Pothigai Express	RD6893	1999

User- Class

Insertion operations:

Validation occurs:



Insertion Operation

--select Train--

User Name

Please fill the Field

User Age

Please fill the field

Mobile No.

Please fill the Field

Strating Location

Please fill the Field

During insertion operation I had choose the train to go for journey, because both are in association mapping.

Before Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
4	24	13-04-2024	Madurai	7418529630	Lokii	23-04-2024	Theni	4
5	21	26-01-2024	Virudhunagar	74178965	Vicky	25-01-2024	Karur	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

In the user table the train Id will act as an foreign key.

After inserting details:

After Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
4	24	13-04-2024	Madurai	7418529630	Lokii	23-04-2024	Theni	4
5	21	26-01-2024	Virudhunagar	74178965	Vicky	25-01-2024	Karur	3
6	23	14-05-2024	Chennai	7418529630	Vishnu	12-05-2024	Tiruppur	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The new user is mapped with the train id 5.

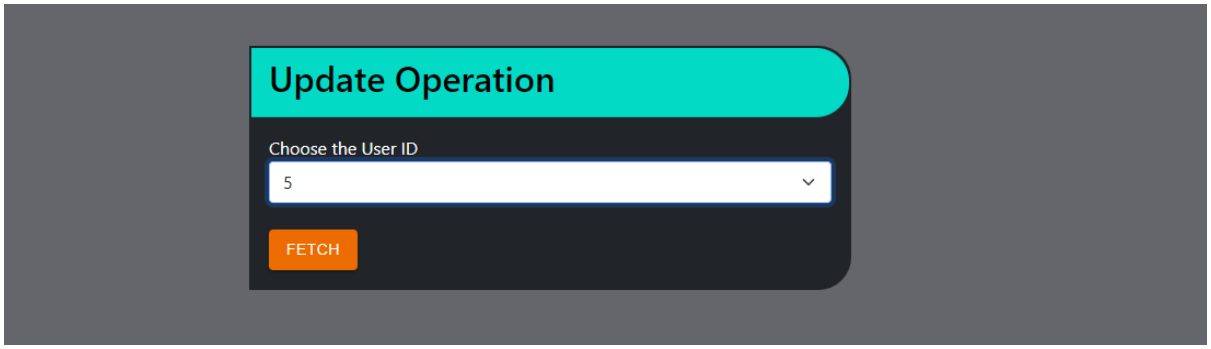
My Train Table:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

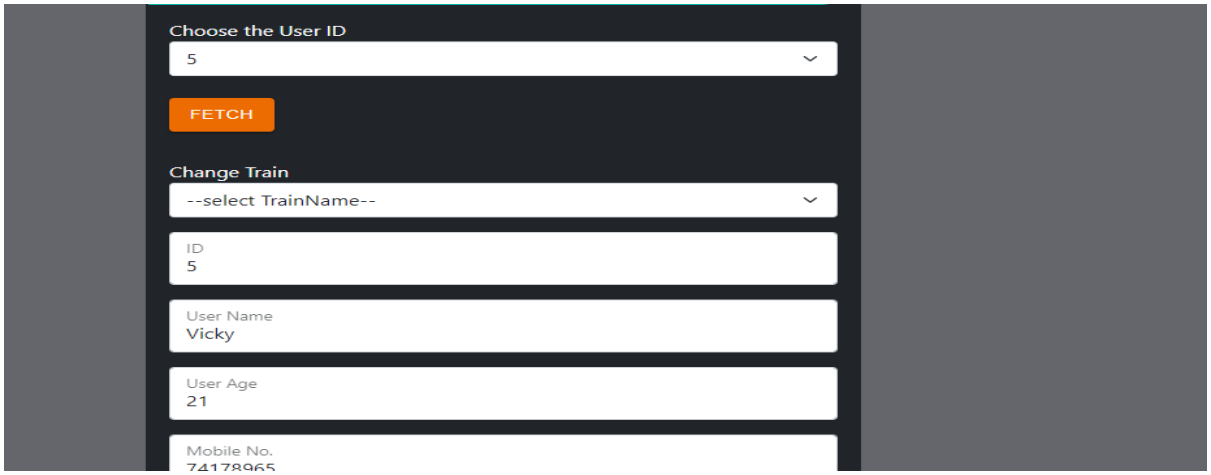
tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	VadheBharat	AS456123	2022
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

Updation Operations:

Here also i am fetch method.



After Fetching:

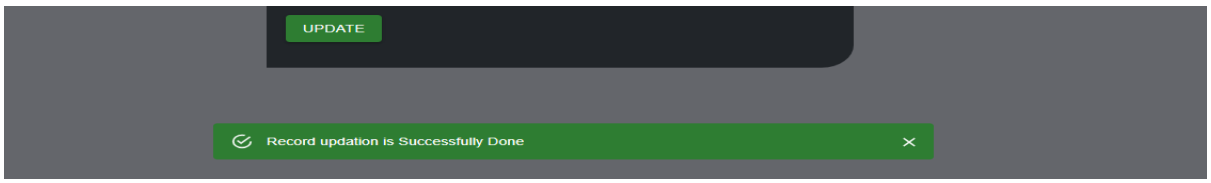


Before Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
4	24	13-04-2024	Madurai	7418529630	Lokii	23-04-2024	Theni	4
5	21	26-01-2024	Virudhunagar	74178965	Vicky	25-01-2024	Karur	3
6	23	14-05-2024	Chennai	7418529630	Vishnu	12-05-2024	Tiruppur	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

After Updating details:



After Database:

I update the details in the 5th row i changed the train id also..

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

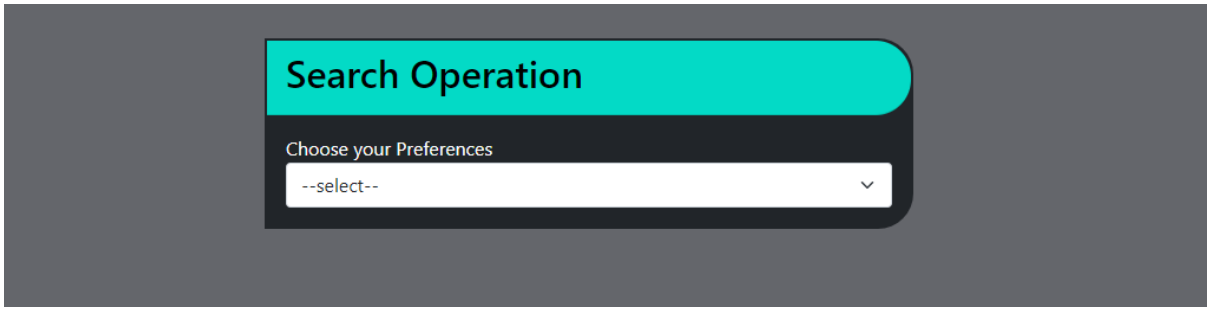
us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
4	24	13-04-2024	Madurai	7418529630	Lokii	23-04-2024	Theni	4
5	21	26-01-2024	Tuticorin	74178965	Kumar	25-01-2024	Namakkal	5
6	23	14-05-2024	Chennai	7418529630	Vishnu	12-05-2024	Tiruppur	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

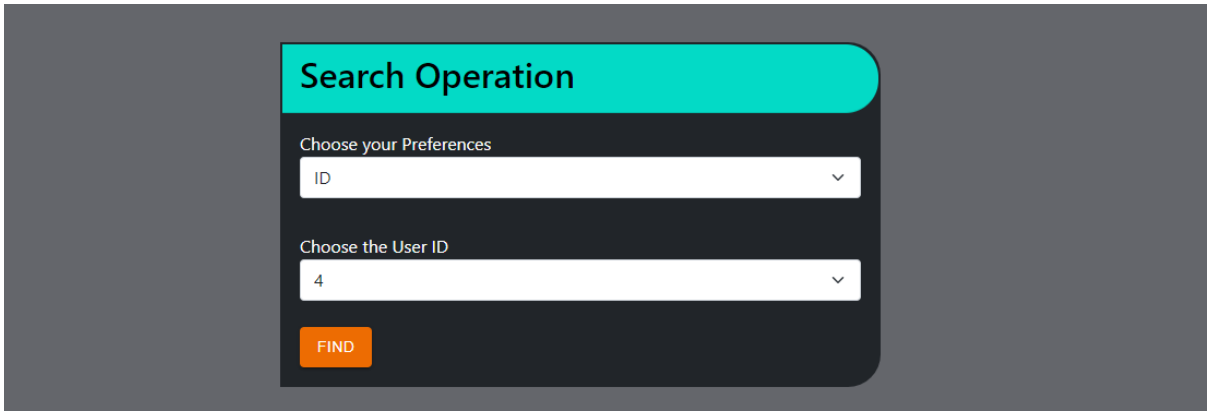
Find-Operations:

Find-operation done in both the way like name and Id,

In the find operation the the association train table will also get details of it,

By Id,





Output :

Record

User Id	User Name	User Age	Mobile No.	Arrival Location	Depature Location	Arrival Date	Depature Date	Train Id	Train Name	Train Number	Train Year
4	Lokii	24	7418529630	Theni	Madurai	23-04-2024	13-04-2024	4	VadheBharat	AS456123	2022

By Name:

Search Operation

Choose your Preferences

Name

Choose the User Name

Abishek

FIND

Record

User Id	User Name	User Age	Mobile No.	Arrival Location	Depature Location	Arrival Date	Depature Date	Train Id	Train Name	Train Number	Train Year
3	Abishek	21	7418529630	Mohanur	Virudhunagar	12-04-2024	25-04-2024	3	Kodai Express	WE456789	2001

Delete-Operation:

Deletion done by both the Id and Name:

Before Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
5	21	26-01-2024	Tuticorin	74178965	Kumar	25-01-2024	Namakkal	5
6	23	14-05-2024	Chennai	7418529630	Vishnu	12-05-2024	Tiruppur	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

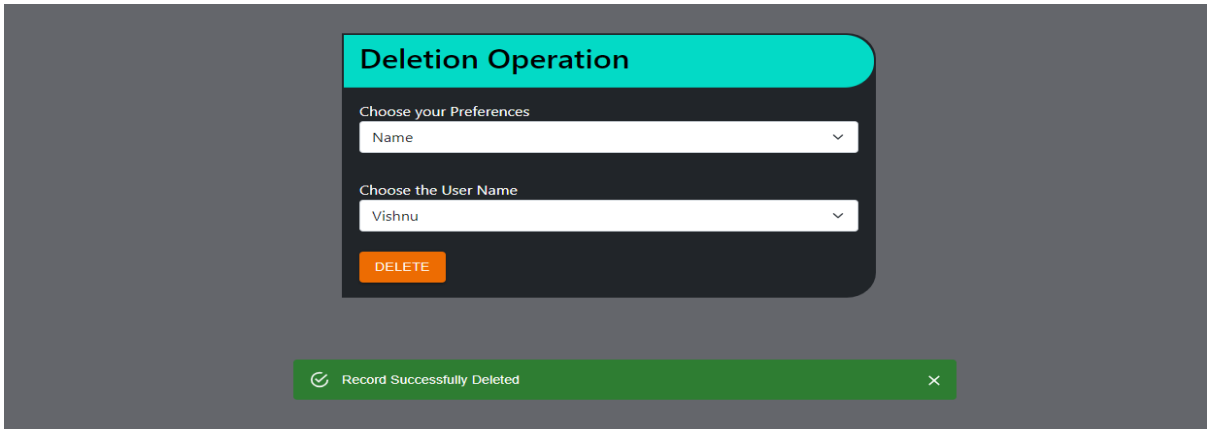
By Id:

ByName:

Before Database:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
5	21	26-01-2024	Tuticorin	74178965	Kumar	25-01-2024	Namakkal	5
6	23	14-05-2024	Chennai	7418529630	Vishnu	12-05-2024	Tiruppur	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL



After Database:

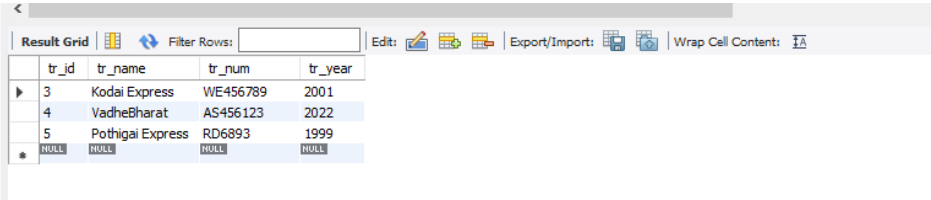
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
5	21	26-01-2024	Tuticorin	74178965	Kumar	25-01-2024	Namakkal	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

55)
55)
55)
55)
55)
55)

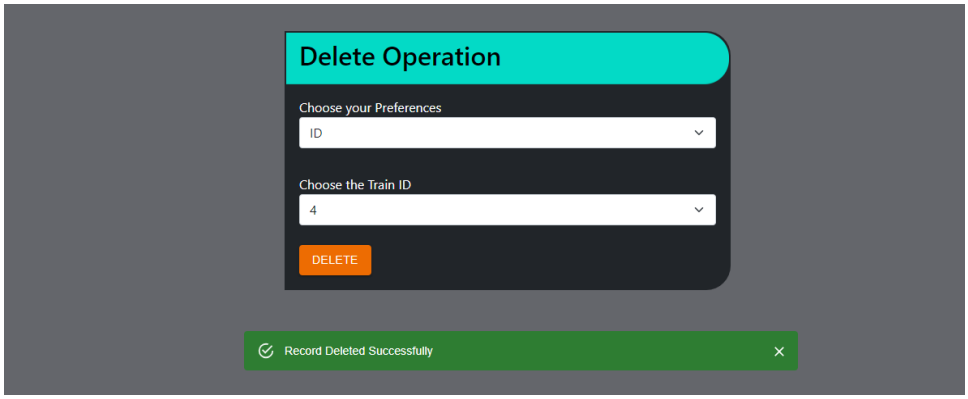
In that operation I had deleted the train Id 4th association mapped user. So that train id 4 can be deleted.

Before Database of Train table:

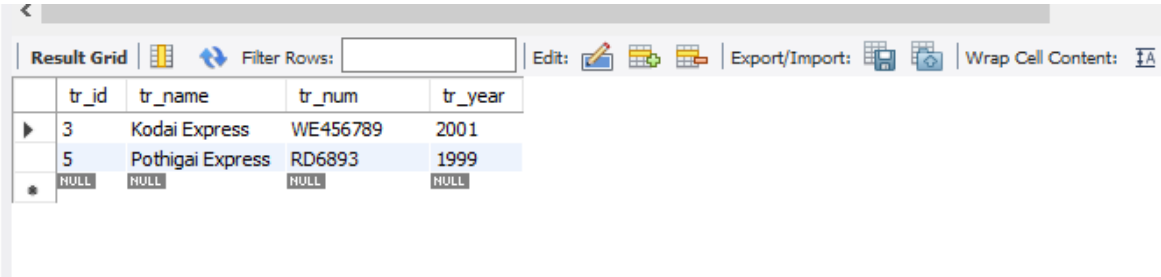


tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
4	VadheBharat	AS456123	2022
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

After deleting the train id4:



After database of Train-table:



tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

Find All Operation:

Record

User Id	User Name	User Age	Mobile No.	Arrival Location	Depature Location	Arrival Date	Depature Date	Train Id	Train Name	Train Number	Train Year
3	Abishek	21	7418529630	Mohanur	Virudhunagar	12-04-2024	25-04-2024	3	Kodai Express	WE456789	2001
5	Kumar	21	74178965	Namakkal	Tuticorin	25-01-2024	26-01-2024	5	Pothigai Express	RD6893	1999
7	Lokii	21	74178965	Theni	Virudhunagar	25-01-2024	26-01-2024	3	Kodai Express	WE456789	2001
8	Vicky	22	74178965	Karur	Madurai	23-04-2024	25-04-2024	5	Pothigai Express	RD6893	1999

Database Tabels:

- tbl_train
- tbl_user
- Views

Train Table:

Result Grid | Filter Rows: | Edit: | Export/Import:

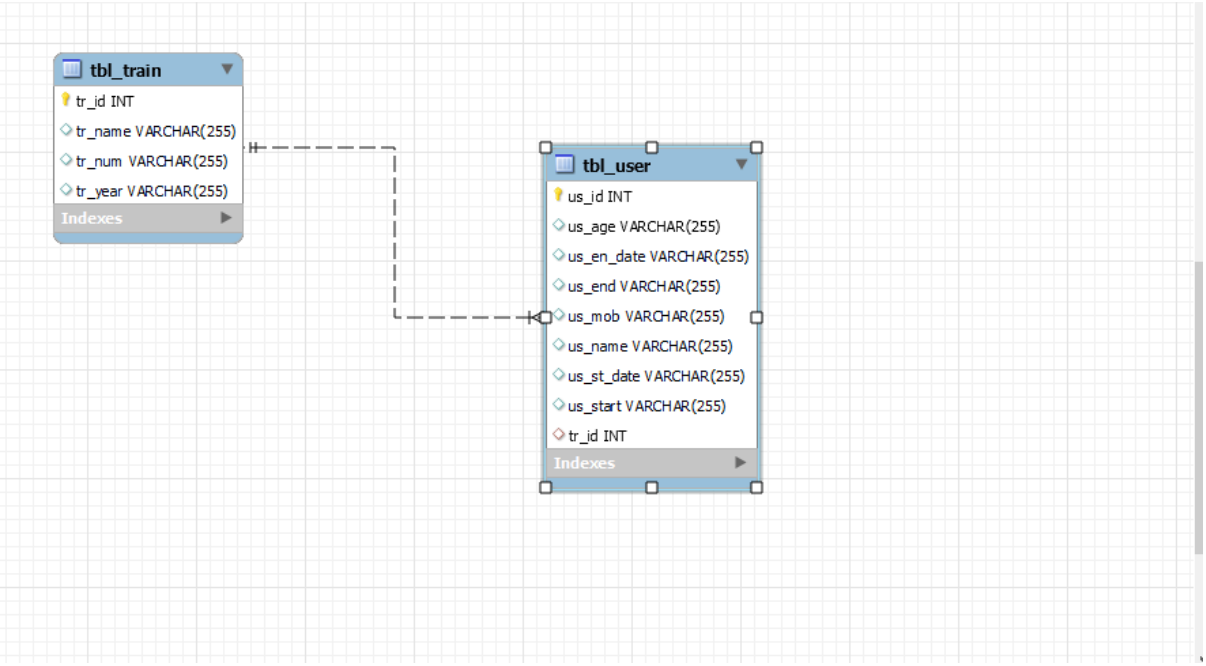
tr_id	tr_name	tr_num	tr_year
3	Kodai Express	WE456789	2001
5	Pothigai Express	RD6893	1999
NULL	NULL	NULL	NULL

User Table:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

us_id	us_age	us_en_date	us_end	us_mob	us_name	us_st_date	us_start	tr_id
3	21	25-04-2024	Virudhunagar	7418529630	Abishek	12-04-2024	Mohanur	3
5	21	26-01-2024	Tuticorin	74178965	Kumar	25-01-2024	Namakkal	5
7	21	26-01-2024	Virudhunagar	74178965	Lokii	25-01-2024	Theni	3
8	22	25-04-2024	Madurai	74178965	Vicky	23-04-2024	Karur	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

ER-Diagram:



Code of the Project:

Beans – Train

```
package com.rss.bean;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.Table;

@Entity
@Table(name="tbl_Train")
public class Train {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int trId;
private String trName;
private String trNum;
private String trYear;
public Train() {
super();
// TODO Auto-generated constructor stub
}
public Train(int trId, String trName, String trNum, String trYear) {
super();
this.trId = trId;
this.trName = trName;
this.trNum = trNum;
this.trYear = trYear;
}
public int getTrId() {
return trId;
}
public void setTrId(int trId) {
this.trId = trId;
}
public String getTrName() {
return trName;
}
public void setTrName(String trName) {
this.trName = trName;
}
```

```java
public String getTrNum() {
return trNum;
}
public void setTrNum(String trNum) {
this.trNum = trNum;
}
public String getTrYear() {
return trYear;
}
public void setTrYear(String trYear) {
this.trYear = trYear;
}



}
```

Bean-User:

```java
package com.rss.bean;

import jakarta.persistence.CascadeType;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.JoinColumn;
import jakarta.persistence.ManyToOne;
import jakarta.persistence.Table;

@Entity
@Table(name="tbl_User")
public class User {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private int usId;
private String usName;
private String usAge;
private String usMob;

private String usStart;
```

```java
private String usEnd;
private String usStDate;
private String usEnDate;

@ManyToOne(targetEntity = Train.class, cascade = CascadeType.DETACH)
@JoinColumn(name="trId")
private Train trn;

public User() {
super();
// TODO Auto-generated constructor stub
}

public User(int usId, String usName, String usAge, String usMob,
String usStart, String usEnd, String usStDate,
String usEnDate, Train trn) {
super();
this.usId = usId;
this.usName = usName;
this.usAge = usAge;
this.usMob = usMob;
this.usStart = usStart;
this.usEnd = usEnd;
this.usStDate = usStDate;
this.usEnDate = usEnDate;
this.trn = trn;
}

public int getUsId() {
return usId;
}

public void setUsId(int usId) {
this.usId = usId;
}

public String getUsName() {
return usName;
}

public void setUsName(String usName) {
this.usName = usName;
```

```
}

public String getUsAge() {
return usAge;
}

public void setUsAge(String usAge) {
this.usAge = usAge;
}

public String getUsMob() {
return usMob;
}

public void setUsMob(String usMob) {
this.usMob = usMob;
}

public String getUsStart() {
return usStart;
}

public void setUsStart(String usStart) {
this.usStart = usStart;
}

public String getUsEnd() {
return usEnd;
}

public void setUsEnd(String usEnd) {
this.usEnd = usEnd;
}

public String getUsStDate() {
return usStDate;
}

public void setUsStDate(String usStDate) {
this.usStDate = usStDate;
}
```

```
public String getUsEnDate() {
return usEnDate;
}

public void setUsEnDate(String usEnDate) {
this.usEnDate = usEnDate;
}

public Train getTrn() {
return trn;
}

public void setTrn(Train trn) {
this.trn = trn;
}




}
```

Controller-Train:

```
package com.rss.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.rss.bean.Train;
import com.rss.repo.TrainRepo;

@RestController
```

```java
@CrossOrigin(origins = "http://localhost:3000/")
@RequestMapping("/api")
public class TrainController {

@Autowired
TrainRepo repo;

@PostMapping("/insertTrain")
public String doInsert(@RequestBody Train tr) {
String msg="";
try {
repo.save(tr);
msg="Insertion successfull";
}catch(Exception e) {
msg="Insertion Failure";
}
return msg;
}

@PutMapping("/updateTrain")
public String doUpdate(@RequestBody Train tr) {
String msg="";
try {
repo.save(tr);
msg="Updation successfull";
}catch(Exception e) {
msg="Updation Failure";
}
return msg;
}

@GetMapping("/findTrainId/{trId}")
public Train doFindId(@PathVariable("trId")int trId) {
Train tr = repo.findById(trId).get();
return tr;
}

@GetMapping("/findTrainName/{trName}")
public List<Train> doFindId(@PathVariable("trName")String trName) {
List<Train> trlist = repo.findByTrName(trName);
return trlist;
}
```

```
@DeleteMapping("/deleteTrainId/{trId}")
public String doDeleteId(@PathVariable("trId")int trId) {
String msg="";
try {
repo.deleteById(trId);
msg="Deletion successfull";
}catch(Exception e) {
msg="Deletion Failure";
}
return msg;
}

@DeleteMapping("/deleteTrainName/{trName}")
public String doDeleteId(@PathVariable("trName")String trName) {
String msg="";
try {
repo.deleteByTrName(trName);
msg="Deletion successfull";
}catch(Exception e) {
msg="Deletion Failure";
}
return msg;
}

@GetMapping("/findallTrain")
public List<Train> doFindAll(){
List<Train> trList = (List<Train>) repo.findAll();
return trList;
}

}
```

Controller-User:

```
package com.rss.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
```

```java
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.rss.bean.Train;
import com.rss.bean.User;
import com.rss.repo.UserRepo;

@RestController
@CrossOrigin(origins = "http://localhost:3000/")
@RequestMapping("/api")
public class UserController {

@Autowired
UserRepo repo;

@PostMapping("/insertUser")
public String doInsert(@RequestBody User usr) {
String msg="";
try {
repo.save(usr);
msg="Insertion Successfull";
}catch(Exception e) {
msg="Inserion Failure";
}
return msg;
}

@PutMapping("/updateUser")
public String doUpdate(@RequestBody User usr) {
String msg="";
try {
repo.save(usr);
msg="Updation successfull";
}catch(Exception e) {
msg="Updation Failure";
}
return msg;
```

```
}

@GetMapping("/findUserId/{usId}")
public User doFindId(@PathVariable("usId")int usId) {
User usr = repo.findById(usId).get();
return usr;
}

@GetMapping("/findUserName/{usName}")
public List<User> doFindId(@PathVariable("usName")String usName) {
List<User> usList = repo.findByUsName(usName);
return usList;
}

@DeleteMapping("/deleteUserId/{usId}")
public String doDeleteId(@PathVariable("usId")int usId) {
String msg="";
try {
repo.deleteById(usId);
msg="Deletion successfull";
}catch(Exception e) {
msg="Deletion Failure";
}
return msg;
}

@DeleteMapping("/deleteuserName/{usName}")
public String doDeleteId(@PathVariable("usName")String usName) {
String msg="";
try {
repo.deleteByUsName(usName);
msg="Deletion successfull";
}catch(Exception e) {
msg="Deletion Failure";
}
return msg;
}

@GetMapping("/findallUser")
public List<User> doFindAll(){
List<User> usList = (List<User>) repo.findAll();
return usList;
```

```
}

}
```

Repo –Train:

```
package com.rss.repo;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import com.rss.bean.Train;

import jakarta.transaction.Transactional;

public interface TrainRepo extends CrudRepository<Train, Integer> {

@Transactional
public List<Train> findByTrName(String trName);

@Transactional
public void deleteByTrName(String trName);

}
```

Repo-User:

```
package com.rss.repo;

import java.util.List;

import org.springframework.data.repository.CrudRepository;

import com.rss.bean.User;

import jakarta.transaction.Transactional;

public interface UserRepo extends CrudRepository<User, Integer> {

@Transactional
public List<User> findByUsName(String usName);
```

```java
@Transactional
public void deleteByUsName(String usName);

}
```

Front-EndPage:
Service Layers:
Trains:

```java
import axios from "axios";
import React, { Component } from "react";

const insert = "http://localhost:2024/api/insertTrain";
const getAll = "http://localhost:2024/api/findallTrain";
const update = "http://localhost:2024/api/updateTrain";
const findId = "http://localhost:2024/api/findTrainId/";
const findName = "http://localhost:2024/api/findTrainName/";
const delId = "http://localhost:2024/api/deleteTrainId/";
const delName = "http://localhost:2024/api/deleteTrainName/";

class Train extends Component {
  doInsert(data) {
    return axios.post(insert, data);
  }

  doUpdate(data) {
    return axios.put(update, data);
  }

  dofindId(data) {
    return axios.get(findId + data);
  }

  dofindName(data) {
    return axios.get(findName + data);
  }

  deleteId(data) {
    return axios.delete(delId + data);
  }

  deleteName(data) {
    return axios.delete(delName + data);
  }
  getAll() {
    return axios.get(getAll);
  }
```

```
}

export default new Train();
```

User:

```
import axios from "axios";
import React, { Component } from "react";

const insert = "http://localhost:2024/api/insertUser";
const getAll = "http://localhost:2024/api/findallUser";
const update = "http://localhost:2024/api/updateUser";
const findId = "http://localhost:2024/api/findUserId/";
const findName = "http://localhost:2024/api/findUserName/";
const delId = "http://localhost:2024/api/deleteUserId/";
const delName = "http://localhost:2024/api/deleteuserName/";

class User extends Component {
  doInsert(data) {
    return axios.post(insert, data);
  }

  doUpdate(data) {
    return axios.put(update, data);
  }

  dofindId(data) {
    return axios.get(findId + data);
  }

  dofindName(data) {
    return axios.get(findName + data);
  }

  deleteId(data) {
    return axios.delete(delId + data);
  }

  deleteName(data) {
    return axios.delete(delName + data);
  }
  getAll() {
    return axios.get(getAll);
  }
}

export default new User();
```

Trains:

Insert Page:

```
import React, { useState } from "react";
import "./Style.css";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "react-bootstrap/Button";
import Alert from "@mui/material/Alert";
import Stack from "@mui/material/Stack";
import { useForm } from "react-hook-form";
import Train from "./Service/Train";

const Tinsert = () => {
  const [alert, setalert] = useState(false);
  const {
    register,
    handleSubmit,
    formState: { errors },
  } = useForm();

  const getFormData = (data) => {
    console.log(data);
    Train.doInsert(data).then(() => {
      setalert(true);
    });
  };
  return (
    <>
      <Card bg="dark" border="dark" id="Mform">
        <Card.Header id="header">
          <h2>Insertion Operation</h2>
        </Card.Header>
        <Card.Body>
          <form onSubmit={handleSubmit(getFormData)}>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Train Name"
                {...register("trName", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Train Name</label>
              {errors.trName && <p className="error">Please fill the
Field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
```

```
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Train Number"
                {...register("trNum", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Train Number</label>
              {errors.trNum && <p className="error"> Please fill the
field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Train Year"
                {...register("trYear", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Train Year</label>
              {errors.trYear && <p className="error">Please fill the
Field</p>}
            </Form.Floating>

            <br></br>
            <Form.Floating className="mb-3">
              <Button as="input" type="submit" value="Insert" size="md" />{"
"}
            </Form.Floating>
          </form>
        </Card.Body>
      </Card>
      <br />
      {alert && (
        <>
          <Stack
            sx={{
              width: "50%",
              marginLeft: "390px",
              marginBottom: "50px",
              color: "yellowgreen",
            }}
            spacing={2}
          >
            <Alert
              variant="filled"
              onClose={() => {
                setalert(false);
                window.location.reload();
              }}
```

```
            >
              Records Inserted Successfully
            </Alert>
          </Stack>
        </>
      )}
    </>
  );
};

export default Tinsert;
```

Update-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "@mui/material/Button";
import Alert from "@mui/material/Alert";
import Stack from "@mui/material/Stack";
import Train from "./Service/Train";

const Tupdate = () => {
  const [Adata, setAdata] = useState([]);
  const [alert, setalert] = useState(false);
  const [Idopt, setIdopt] = useState(0);
  const [formDisp, setformDisp] = useState(false);

  const [trId, changetrId] = useState("");
  const [trName, changetrName] = useState("");
  const [trNum, changetrNum] = useState("");
  const [trYear, changetrYear] = useState("");

  const UpData = {
    trId,
    trName,
    trNum,
    trYear,
  };

  useEffect(() => {
    Train.getAll().then((response) => {
      setAdata(response.data);
    });
  }, []);

  const openForm = () => {
    Train.dofindId(Idopt).then((response) => {
```

```
    changetrId(response.data.trId);
    changetrName(response.data.trName);
    changetrNum(response.data.trNum);
    changetrYear(response.data.trYear);
  });

  setformDisp(true);
};

const doUpdate = () => {
  console.log(UpData);
  Train.doUpdate(UpData).then(() => {
    setalert(true);
  });
};

return (
  <div>
    <Card bg="dark" border="dark" id="Mform">
      <>
        <Card.Header id="header">
          <h2>Update Operation</h2>
        </Card.Header>
        <Card.Body>
          <label style={{ color: "white" }}>Choose the Train ID</label>
          <Form.Select
            aria-label="Default select example"
            onChange={(e) => {
              setIdopt(e.target.value);
            }}
          >
            <option>--select--</option>
            {Adata.map((pdt) => (
              <option value={pdt.trId}>{pdt.trId}</option>
            ))}
          </Form.Select>
          <br></br>
          <Stack direction="row" spacing={2}>
            <Button
              variant="contained"
              color="warning"
              onClick={() => openForm()}
            >
              Fetch
            </Button>
          </Stack>
        </Card.Body>
      </>
```

```
{formDisp && (
  <Card.Body>
    <form>
      <Form.Floating className="mb-3">
        <Form.Control
          id="floatingInputCustom"
          type="text"
          placeholder="ID"
          disabled
          defaultValue={trId}
        />
        <label htmlFor="floatingInputCustom">ID</label>
      </Form.Floating>
      <Form.Floating className="mb-3">
        <Form.Control
          id="name"
          type="text"
          placeholder="Train Name"
          name="pName"
          defaultValue={trName}
          onChange={(e) => {
            changetrName(e.target.value);
          }}
        />
        <label htmlFor="floatingPasswordCustom">Train Name</label>
      </Form.Floating>
      <Form.Floating className="mb-3">
        <Form.Control
          id="floatingPasswordCustom"
          type="text"
          placeholder="Product Price"
          defaultValue={trNum}
          onChange={(e) => {
            changetrNum(e.target.value);
          }}
        />
        <label htmlFor="floatingPasswordCustom">Train Number</label>
      </Form.Floating>
      <Form.Floating className="mb-3">
        <Form.Control
          id="floatingPasswordCustom"
          type="text"
          placeholder="Product seller"
          defaultValue={trYear}
          onChange={(e) => {
            changetrYear(e.target.value);
          }}
```

```
              />
              <label htmlFor="floatingPasswordCustom">Train Year</label>
            </Form.Floating>

            <br></br>
            <Form.Floating className="mb-3">
              <Stack direction="row" spacing={2}>
                <Button
                  variant="contained"
                  color="success"
                  onClick={doUpdate}
                >
                  Update
                </Button>
              </Stack>
            </Form.Floating>
          </form>
        </Card.Body>
      )}
    </Card>
    <br />
    {alert && (
      <>
        <br></br>
        <Stack
          sx={{
            width: "50%",
            marginLeft: "390px",
            marginBottom: "50px",
            fontWeight: "bold",
            color: "yellowgreen",
          }}
          spacing={2}
        >
          <Alert
            variant="filled"
            onClose={() => {
              setalert(false);
              window.location.reload();
            }}
          >
            Record updation is Successfully Done
          </Alert>
        </Stack>
      </>
    )}
  </div>
);
```

```
};

export default Tupdate;
```

Find-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "@mui/material/Button";
import Stack from "@mui/material/Stack";
import Train from "./Service/Train";

const Tfind = () => {
  const [Adata, setAdata] = useState([]);
  const [Fdata, setFdata] = useState([]);
  const [choose, setchoose] = useState("");
  const [Idopt, setIdopt] = useState(0);
  const [Nameopt, setNameopt] = useState("");
  const [TblId, setTblId] = useState(false);
  const [TblName, setTblName] = useState(false);

  useEffect(() => {
    Train.getAll().then((response) => {
      setAdata(response.data);
    });
  }, []);
  console.log(Idopt);
  console.log(Nameopt);

  const doFind = (data) => {
    data === "Name"
      ? Train.dofindName(Nameopt).then((response) => {
          setFdata(response.data);
          console.log(response.data);
          setTblId(false);
          setTblName(true);
        })
      : Train.dofindId(Idopt).then((response) => {
          setFdata(response.data);
          setTblName(false);
          setTblId(true);
        });
  };

  return (
    <>
      <Card bg="dark" border="dark" id="Mform">
```

```
        <Card.Header id="header">
          <h2>Search Operation</h2>
        </Card.Header>
        <Card.Body>
          <label style={{ color: "white" }}>Choose your Preferences</label>
          <Form.Select
            aria-label="Default select example"
            onChange={(e) => {
              setchoose(e.target.value);
            }}
          >
            <option>--select--</option>
            <option value="ID">ID</option>
            <option value="Name">Name</option>
          </Form.Select>
        </Card.Body>

        {choose === "ID" && (
          <Card.Body>
            <label style={{ color: "white" }}>Choose the Train ID</label>
            <Form.Select
              aria-label="Default select example"
              onChange={(e) => {
                setIdopt(e.target.value);
              }}
            >
              <option>--select--</option>
              {Adata.map((pdt) => (
                <option value={pdt.trId}>{pdt.trId}</option>
              ))}
            </Form.Select>
            <br></br>
            <Stack direction="row" spacing={2}>
              <Button
                variant="contained"
                color="warning"
                onClick={() => doFind(1)}
              >
                Find
              </Button>
            </Stack>
          </Card.Body>
        )}
        {choose === "Name" && (
          <Card.Body>
            <label style={{ color: "white" }}>Choose the Train Name</label>
            <Form.Select
              aria-label="Default select example"
```

```
            onChange={(e) => {
              setNameopt(e.target.value);
            }}
          >
            <option>--select--</option>
            {Adata.map((pdt) => (
              <option value={pdt.trName}>{pdt.trName}</option>
            ))}
          </Form.Select>
          <br></br>
          <Stack direction="row" spacing={2}>
            <Button
              variant="contained"
              color="warning"
              onClick={() => doFind("Name")}
            >
              Find
            </Button>
          </Stack>
        </Card.Body>
      )}
    </Card>
    <br></br>
    {TblId && (
      <Card bg="light" border="dark" className="tbl">
        <Card.Header id="header">
          <h2>Record</h2>
        </Card.Header>
        <Card.Body>
          <table class="table">
            <thead>
              <tr>
                <th scope="col">Train Id</th>
                <th scope="col">Train Name</th>
                <th scope="col">Train Number</th>
                <th scope="col">Train Year</th>
              </tr>
            </thead>
            <tbody>
              <tr>
                <td>{Fdata.trId}</td>
                <td>{Fdata.trName}</td>
                <td>{Fdata.trNum}</td>
                <td>{Fdata.trYear}</td>
              </tr>
            </tbody>
          </table>
        </Card.Body>
```

```
        </Card>
      )}
      {TblName && (
        <Card bg="light" className="tbl">
          <Card.Header id="header">
            <h2>Record</h2>
          </Card.Header>
          <Card.Body>
            <table class="table">
              <thead>
                <tr>
                  <th scope="col">Train Id</th>
                  <th scope="col">Train Name</th>
                  <th scope="col">Train Number</th>
                  <th scope="col">Train Year</th>
                </tr>
              </thead>
              <tbody>
                {Fdata.map((item) => (
                  <tr>
                    <td>{item.trId}</td>
                    <td>{item.trName}</td>
                    <td>{item.trNum}</td>
                    <td>{item.trYear}</td>
                  </tr>
                ))}
              </tbody>
            </table>
          </Card.Body>
        </Card>
      )}
    </>
  );
};

export default Tfind;
```

Delete-Page:

```
import React from "react";
import { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "@mui/material/Button";
import Stack from "@mui/material/Stack";
import Alert from "@mui/material/Alert";
import Train from "./Service/Train";
```

```
const Tdelete = () => {
  const [alert, setalert] = useState(false);
  const [Adata, setAdata] = useState([]);
  const [choose, setchoose] = useState("");
  const [Idopt, setIdopt] = useState(0);
  const [Nameopt, setNameopt] = useState("");

  useEffect(() => {
    Train.getAll().then((response) => {
      setAdata(response.data);
    });
  }, []);

  const doDelete = (data) => {
    data === "Name"
      ? Train.deleteName(Nameopt).then(() => {
          setalert(true);
        })
      : Train.deleteId(Idopt).then(() => {
          setalert(true);
        });
  };
  return (
    <>
      <Card bg="dark" border="dark" id="Mform">
        <Card.Header id="header">
          <h2>Delete Operation</h2>
        </Card.Header>
        <Card.Body>
          <label style={{ color: "white" }}>Choose your Preferences</label>
          <Form.Select
            aria-label="Default select example"
            onChange={(e) => {
              setchoose(e.target.value);
            }}
          >
            <option>--select--</option>
            <option value="ID">ID</option>
            <option value="Name">Name</option>
          </Form.Select>
        </Card.Body>

        {choose === "ID" && (
          <Card.Body>
            <label style={{ color: "white" }}>Choose the Train ID</label>
            <Form.Select
              aria-label="Default select example"
              onChange={(e) => {
```

```
            setIdopt(e.target.value);
          }}
        >
          <option>--select--</option>
          {Adata.map((pdt) => (
            <option value={pdt.trId}>{pdt.trId}</option>
          ))}
        </Form.Select>
        <br></br>
        <Stack direction="row" spacing={2}>
          <Button
            variant="contained"
            color="warning"
            onClick={() => doDelete(1)}
          >
            Delete
          </Button>
        </Stack>
      </Card.Body>
    )}
    {choose === "Name" && (
      <Card.Body>
        <label style={{ color: "white" }}>Choose the Train Name</label>
        <Form.Select
          aria-label="Default select example"
          onChange={(e) => {
            setNameopt(e.target.value);
          }}
        >
          <option>--select--</option>
          {Adata.map((pdt) => (
            <option value={pdt.trName}>{pdt.trName}</option>
          ))}
        </Form.Select>
        <br></br>
        <Stack direction="row" spacing={2}>
          <Button
            variant="contained"
            color="warning"
            onClick={() => doDelete("Name")}
          >
            Delete
          </Button>
        </Stack>
      </Card.Body>
    )}
  </Card>
  {alert && (
```

```
        <>
          <Stack
            sx={{
              width: "50%",
              marginLeft: "390px",
              marginBottom: "50px",
              color: "yellowgreen",
            }}
            spacing={2}
          >
            <Alert
              variant="filled"
              onClose={() => {
                setalert(false);
                window.location.reload();
              }}
            >
              Record Deleted Successfully
            </Alert>
          </Stack>
        </>
      )}
    </>
  );
};

export default Tdelete;
```

FindAll-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Train from "./Service/Train";

const Tfindall = () => {
  const [Adata, setAdata] = useState([]);
  useEffect(() => {
    Train.getAll().then((response) => {
      setAdata(response.data);
    });
  }, []);
  return (
    <div>
      <Card bg="light" id="findall" class="table table-hover">
        <Card.Header id="fheader">
          <h2>Record</h2>
        </Card.Header>
        <Card.Body>
```

```
          <table class="table">
            <thead>
              <tr>
                <th scope="col">Train Id</th>
                <th scope="col">Train Name</th>
                <th scope="col">Train Number</th>
                <th scope="col">Train Year</th>
              </tr>
            </thead>
            <tbody>
              {Adata.map((item) => (
                <tr>
                  <td>{item.trId}</td>
                  <td>{item.trName}</td>
                  <td>{item.trNum}</td>
                  <td>{item.trYear}</td>
                </tr>
              ))}
            </tbody>
          </table>
        </Card.Body>
      </Card>
    </div>
  );
};

export default Tfindall;
```

User:

Insert Page:

```
import React, { useEffect, useState } from "react";
```

```
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "react-bootstrap/Button";
import Alert from "@mui/material/Alert";
import Stack from "@mui/material/Stack";
import { useForm } from "react-hook-form";
import Train from "./Service/Train";
import User from "./Service/User";

const Uinsert = () => {
  const [alert, setalert] = useState(false);
  const {
    register,
    handleSubmit,
    formState: { errors },
  } = useForm();
  const [Adata, setAdata] = useState([]);
  const [trId, changetrId] = useState(0);

  useEffect(() => {
    Train.getAll().then((response) => {
      console.log(response.data.data);
      setAdata(response.data);
    });
  }, []);

  console.log(trId);

  const getFormData = (data) => {
    var { usName, usAge, usMob, usStart, usEnd, usStDate, usEnDate } = data;
    var indata = {
      usName,
      usAge,
      usMob,
      usStart,
      usEnd,
      usStDate,
      usEnDate,
      trn: {
        trId,
      },
    };
    console.log(indata);
    User.doInsert(indata).then(() => {
      setalert(true);
    });
  };
  return (
```

```
    <>
      <Card bg="dark" border="dark" id="Mform">
        <Card.Header id="header">
          <h2>Insertion Operation</h2>
        </Card.Header>
        <Card.Body>
          <Form.Select
            className="mb-3"
            aria-label="Default select example"
            onChange={(e) => {
              changetrId(e.target.value);
            }}
          >
            <option>--select Train--</option>
            {Adata.map((idlst) => (
              <option value={idlst.trId}>{idlst.trId}</option>
            ))}
          </Form.Select>
          <form onSubmit={handleSubmit(getFormData)}>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="User Name"
                {...register("usName", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">User Name</label>
              {errors.usName && <p className="error">Please fill the
Field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="number"
                placeholder="User Age"
                {...register("usAge", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">User Age</label>
              {errors.usAge && <p className="error"> Please fill the
field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Mobile No."
                {...register("usMob", { required: true })}
              />
```

```
              <label htmlFor="floatingPasswordCustom">Mobile No.</label>
              {errors.usMob && <p className="error">Please fill the Field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Strating location"
                {...register("usStart", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Strating
Location</label>
              {errors.usStart && <p className="error">Please fill the
Field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Destination"
                {...register("usEnd", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Destination</label>
              {errors.usEnd && <p className="error">Please fill the Field</p>}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Date of Arrival"
                {...register("usStDate", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Date of Arrival</label>
              {errors.usStDate && (
                <p className="error">Please fill the Field</p>
              )}
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Date of Depature"
                {...register("usEnDate", { required: true })}
              />
              <label htmlFor="floatingPasswordCustom">Date of Depature</label>
              {errors.usEnDate && (
                <p className="error">Please fill the Field</p>
              )}
```

```
            </Form.Floating>

            <br></br>
            <Form.Floating className="mb-3">
              <Button as="input" type="submit" value="Insert" size="md" />{"
"}
            </Form.Floating>
          </form>
        </Card.Body>
      </Card>
      <br />
      {alert && (
        <>
          <Stack
            sx={{
              width: "50%",
              marginLeft: "390px",
              marginBottom: "50px",
              color: "yellowgreen",
            }}
            spacing={2}
          >
            <Alert
              variant="filled"
              onClose={() => {
                setalert(false);
                window.location.reload();
              }}
            >
              Records Inserted Successfully
            </Alert>
          </Stack>
        </>
      )}
    </>
  );
};

export default Uinsert;
```

Update-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "@mui/material/Button";
```

```
import Alert from "@mui/material/Alert";
import Stack from "@mui/material/Stack";
import User from "./Service/User";
import Train from "./Service/Train";

const Uupdate = () => {
  const [Adata, setAdata] = useState([]);

  const [Pdata, setPdata] = useState([]);
  const [alert, setalert] = useState(false);
  const [Idopt, setIdopt] = useState(0);
  const [formDisp, setformDisp] = useState(false);
  const [trId, changetrId] = useState(0);

  var [usId, changeusId] = useState(0);
  const [usName, changeusName] = useState("");
  const [usAge, changeusAge] = useState("");
  const [usMob, changeusMob] = useState("");
  const [usStart, changeusStart] = useState("");
  const [usEnd, changeusEnd] = useState("");
  const [usStDate, changeusStDate] = useState("");
  const [usEnDate, changeusEnDate] = useState("");

  useEffect(() => {
    User.getAll().then((response) => {
      setAdata(response.data);
      console.log(response.data);
    });
  }, []);

  useEffect(() => {
    Train.getAll().then((response) => {
      console.log(response.data);
      setPdata(response.data);
    });
  }, []);

  const openForm = () => {
    console.log(Idopt);

    User.dofindId(Idopt).then((response) => {
      console.log(response.data);
      changeusId(response.data.usId);
      changeusName(response.data.usName);
      changeusAge(response.data.usAge);
      changeusMob(response.data.usMob);
      changeusStart(response.data.usStart);
      changeusEnd(response.data.usEnd);
```

```
    changeusStDate(response.data.usStDate);
    changeusEnDate(response.data.usEnDate);
  });

  setformDisp(true);
};

const doUpdate = () => {
  const inData = {
    usId,
    usName,
    usAge,
    usMob,
    usStart,
    usEnd,
    usStDate,
    usEnDate,
    trn: { trId },
  };
  console.log(inData);
  User.doUpdate(inData).then(() => {
    setalert(true);
  });
};
return (
  <>
    <Card bg="dark" border="dark" id="Mform">
      <Card.Header id="header">
        <h2>Update Operation</h2>
      </Card.Header>
      <Card.Body>
        <label style={{ color: "white" }}>Choose the User ID</label>
        <Form.Select
          aria-label="Default select example"
          onChange={(e) => {
            setIdopt(e.target.value);
          }}
        >
          <option>--select--</option>
          {Adata.map((idlst) => (
            <option value={idlst.usId}>{idlst.usId}</option>
          ))}
        </Form.Select>
        <br></br>
        <Stack direction="row" spacing={2}>
          <Button
            variant="contained"
            color="warning"
```

```
            onClick={() => openForm()}
          >
            Fetch
          </Button>
        </Stack>
      </Card.Body>
      {formDisp && (
        <Card.Body>
          <label style={{ color: "white" }}>Change Train</label>
          <Form.Select
            className="mb-3"
            aria-label="Default select example"
            onChange={(e) => {
              changetrId(e.target.value);
            }}
          >
            <option>--select TrainName--</option>
            {Pdata.map((idlst) => (
              <option value={idlst.trId}>{idlst.trName}</option>
            ))}
          </Form.Select>
          <Form.Floating className="mb-3">
            <Form.Control
              id="floatingInputCustom"
              type="text"
              placeholder="ID"
              value={usId}
            />
            <label htmlFor="floatingInputCustom">ID</label>
          </Form.Floating>
          <Form.Floating className="mb-3">
            <Form.Control
              id="name"
              type="text"
              placeholder="Name"
              defaultValue={usName}
              onChange={(e) => {
                changeusName(e.target.value);
              }}
            />
            <label htmlFor="floatingPasswordCustom">User Name</label>
          </Form.Floating>
          <Form.Floating className="mb-3">
            <Form.Control
              id="floatingPasswordCustom"
              type="text"
              placeholder="Age"
              defaultValue={usAge}
```

```
            onChange={(e) => {
              changeusAge(e.target.value);
            }}
          />
          <label htmlFor="floatingPasswordCustom">User Age</label>
        </Form.Floating>
        <Form.Floating className="mb-3">
          <Form.Control
            id="floatingPasswordCustom"
            type="text"
            placeholder="Mobile"
            defaultValue={usMob}
            onChange={(e) => {
              changeusMob(e.target.value);
            }}
          />
          <label htmlFor="floatingPasswordCustom">Mobile No.</label>
        </Form.Floating>
        <Form.Floating className="mb-3">
          <Form.Control
            id="floatingPasswordCustom"
            type="text"
            placeholder="Arrival"
            defaultValue={usStart}
            onChange={(e) => {
              changeusStart(e.target.value);
            }}
          />
          <label htmlFor="floatingPasswordCustom">Arrival location</label>
        </Form.Floating>
        <Form.Floating className="mb-3">
          <Form.Control
            id="floatingPasswordCustom"
            type="text"
            placeholder="Depature"
            defaultValue={usEnd}
            onChange={(e) => {
              changeusEnd(e.target.value);
            }}
          />
          <label htmlFor="floatingPasswordCustom">Depature</label>
        </Form.Floating>
        <Form.Floating className="mb-3">
          <Form.Control
            id="floatingPasswordCustom"
            type="text"
            placeholder="Arrival Date"
            defaultValue={usStDate}
```

```
                onChange={(e) => {
                  changeusStDate(e.target.value);
                }}
              />
              <label htmlFor="floatingPasswordCustom">Arrival Date</label>
            </Form.Floating>
            <Form.Floating className="mb-3">
              <Form.Control
                id="floatingPasswordCustom"
                type="text"
                placeholder="Arrival Date"
                defaultValue={usEnDate}
                onChange={(e) => {
                  changeusEnDate(e.target.value);
                }}
              />
              <label htmlFor="floatingPasswordCustom">Depature Date</label>
            </Form.Floating>

            <br></br>
            <Form.Floating className="mb-3">
              <Stack direction="row" spacing={2}>
                <Button variant="contained" color="success"
onClick={doUpdate}>
                  Update
                </Button>
              </Stack>
            </Form.Floating>
          </Card.Body>
        )}
      </Card>
      {alert && (
        <>
          <br></br>
          <Stack
            sx={{
              width: "50%",
              marginLeft: "390px",
              marginBottom: "50px",
              fontWeight: "bold",
              color: "yellowgreen",
            }}
            spacing={2}
          >
            <Alert
              variant="filled"
              onClose={() => {
                setalert(false);
```

```
                window.location.reload();
              }}
            >
              Record updation is Successfully Done
            </Alert>
          </Stack>
        </>
      )}
    </>
  );
};

export default Uupdate;
```

Find-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "@mui/material/Button";
import Stack from "@mui/material/Stack";
import User from "./Service/User";
import Style from "./Style.css";

const Ufind = () => {
  const [Adata, setAdata] = useState([]);
  const [Fdata, setFdata] = useState([]);
  const [choose, setchoose] = useState("");
  const [Idopt, setIdopt] = useState(0);
  const [Nameopt, setNameopt] = useState("");
  const [Tbl, setTbl] = useState(false);

  useEffect(() => {
    User.getAll().then((response) => {
      setAdata(response.data);
      console.log(response.data);
    });
  }, []);

  const doFind = () => {
    User.dofindName(Nameopt).then((response) => {
      setFdata(response.data);
      setTbl(true);
    });
  };
  return (
    <>
      <Card bg="dark" border="dark" id="Mform">
```

```
        <Card.Header id="header">
          <h2>Search Operation</h2>
        </Card.Header>
        <Card.Body>
          <label style={{ color: "white" }}>Choose your Preferences</label>
          <Form.Select
            aria-label="Default select example"
            onChange={(e) => {
              setchoose(e.target.value);
            }}
          >
            <option>--select--</option>
            <option value="ID">ID</option>
            <option value="Name">Name</option>
          </Form.Select>
        </Card.Body>

        {choose === "ID" && (
          <Card.Body>
            <label style={{ color: "white" }}>Choose the User ID</label>
            <Form.Select
              aria-label="Default select example"
              onChange={(e) => {
                setNameopt(e.target.value);
              }}
            >
              <option>--select--</option>
              {Adata.map((item) => (
                <option value={item.usName}>{item.usId}</option>
              ))}
            </Form.Select>
            <br></br>
            <Stack direction="row" spacing={2}>
              <Button
                variant="contained"
                color="warning"
                onClick={() => doFind()}
              >
                Find
              </Button>
            </Stack>
          </Card.Body>
        )}
        {choose === "Name" && (
          <Card.Body>
            <label style={{ color: "white" }}>Choose the User Name</label>
            <Form.Select
              aria-label="Default select example"
```

```
            onChange={(e) => {
              setNameopt(e.target.value);
            }}
          >
            <option>--select--</option>
            {Adata.map((item) => (
              <option value={item.usName}>{item.usName}</option>
            ))}
          </Form.Select>
          <br></br>
          <Stack direction="row" spacing={2}>
            <Button
              variant="contained"
              color="warning"
              onClick={() => doFind()}
            >
              Find
            </Button>
          </Stack>
        </Card.Body>
      )}
    </Card>
    {Tbl && (
      <Card bg="light" className="tbl" id="ubtl">
        <Card.Header id="header">
          <h2>Record</h2>
        </Card.Header>
        <Card.Body>
          <table class="table">
            <thead>
              <tr>
                <th scope="col">User Id</th>
                <th scope="col">User Name</th>
                <th scope="col">User Age</th>
                <th scope="col">Mobile No.</th>
                <th scope="col">Arrival Location</th>
                <th scope="col">Depature Location</th>
                <th scope="col">Arrival Date</th>
                <th scope="col">Depature Date</th>
                <th scope="col">Train Id</th>
                <th scope="col">Train Name</th>
                <th scope="col">Train Number</th>
                <th scope="col">Train Year</th>
              </tr>
            </thead>
            <tbody>
              {Fdata.map((item) => (
                <tr>
```

```
                    <td>{item.usId}</td>
                    <td>{item.usName}</td>
                    <td>{item.usAge}</td>
                    <td>{item.usMob}</td>
                    <td>{item.usStart}</td>
                    <td>{item.usEnd}</td>
                    <td>{item.usStDate}</td>
                    <td>{item.usEnDate}</td>
                    <td>{item.trn.trId}</td>
                    <td>{item.trn.trName}</td>
                    <td>{item.trn.trNum}</td>
                    <td>{item.trn.trYear}</td>
                  </tr>
                ))}
              </tbody>
            </table>
          </Card.Body>
        </Card>
      )}
    </>
  );
};

export default Ufind;
```

Delete-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import Form from "react-bootstrap/Form";
import Button from "@mui/material/Button";
import Stack from "@mui/material/Stack";
import Alert from "@mui/material/Alert";
import User from "./Service/User";

const Udelete = () => {
  const [alert, setalert] = useState(false);
  const [Adata, setAdata] = useState([]);
  const [choose, setchoose] = useState("");
  const [Idopt, setIdopt] = useState(0);
  const [Nameopt, setNameopt] = useState("");

  useEffect(() => {
    User.getAll().then((response) => {
      setAdata(response.data);
      console.log(response.data);
    });
  }, []);
```

```
const doDelete = (data) => {
  data === "Name"
    ? User.deleteName(Nameopt).then(() => {
        setalert(true);
      })
    : User.deleteId(Idopt).then(() => {
        setalert(true);
      });
};
return (
  <>
    <Card bg="dark" border="dark" id="Mform">
      <Card.Header id="header">
        <h2>Deletion Operation</h2>
      </Card.Header>
      <Card.Body>
        <label style={{ color: "white" }}>Choose your Preferences</label>
        <Form.Select
          aria-label="Default select example"
          onChange={(e) => {
            setchoose(e.target.value);
          }}
        >
          <option>--select--</option>
          <option value="ID">ID</option>
          <option value="Name">Name</option>
        </Form.Select>
      </Card.Body>

      {choose === "ID" && (
        <Card.Body>
          <label style={{ color: "white" }}>Choose the User ID</label>
          <Form.Select
            aria-label="Default select example"
            onChange={(e) => {
              setIdopt(e.target.value);
            }}
          >
            <option>--select--</option>
            {Adata.map((item) => (
              <option value={item.usId}>{item.usId}</option>
            ))}
          </Form.Select>
          <br></br>
          <Stack direction="row" spacing={2}>
            <Button
              variant="contained"
```

```
            color="warning"
            onClick={() => doDelete(1)}
          >
            Delete
          </Button>
        </Stack>
      </Card.Body>
    )}
    {choose === "Name" && (
      <Card.Body>
        <label style={{ color: "white" }}>Choose the User Name</label>
        <Form.Select
          aria-label="Default select example"
          onChange={(e) => {
            setNameopt(e.target.value);
          }}
        >
          <option>--select--</option>
          {Adata.map((item) => (
            <option value={item.usName}>{item.usName}</option>
          ))}
        </Form.Select>
        <br></br>
        <Stack direction="row" spacing={2}>
          <Button
            variant="contained"
            color="warning"
            onClick={() => doDelete("Name")}
          >
            Delete
          </Button>
        </Stack>
      </Card.Body>
    )}
  </Card>
  {alert && (
    <>
      <br></br>
      <Stack
        sx={{
          width: "50%",
          marginLeft: "390px",
          marginBottom: "50px",
          fontWeight: "bold",
          color: "yellowgreen",
        }}
        spacing={2}
      >
```

```
            <Alert
              variant="filled"
              onClose={() => {
                setalert(false);
                window.location.reload();
              }}
            >
              Record Successfully Deleted
            </Alert>
          </Stack>
        </>
      )}
    </>
  );
};

export default Udelete;
```

FindAll-Page:

```
import React, { useEffect, useState } from "react";
import { Card } from "react-bootstrap";
import User from "./Service/User";

const Ufindall = () => {
  const [Adata, setAdata] = useState([]);

  useEffect(() => {
    User.getAll().then((response) => {
      setAdata(response.data);
      console.log(response.data);
    });
  }, []);
  return (
    <>
      <Card bg="light" id="findall" class="table table-hover">
        <Card.Header id="fheader">
          <h2>Record</h2>
        </Card.Header>
        <Card.Body>
          <table class="table">
            <thead>
              <tr>
                <th scope="col">User Id</th>
                <th scope="col">User Name</th>
                <th scope="col">User Age</th>
                <th scope="col">Mobile No.</th>
                <th scope="col">Arrival Location</th>
```

```
                <th scope="col">Depature Location</th>
                <th scope="col">Arrival Date</th>
                <th scope="col">Depature Date</th>
                <th scope="col">Train Id</th>
                <th scope="col">Train Name</th>
                <th scope="col">Train Number</th>
                <th scope="col">Train Year</th>
              </tr>
            </thead>
            <tbody>
              {Adata.map((item) => (
                <tr>
                  <td>{item.usId}</td>
                  <td>{item.usName}</td>
                  <td>{item.usAge}</td>
                  <td>{item.usMob}</td>
                  <td>{item.usStart}</td>
                  <td>{item.usEnd}</td>
                  <td>{item.usStDate}</td>
                  <td>{item.usEnDate}</td>
                  <td>{item.trn.trId}</td>
                  <td>{item.trn.trName}</td>
                  <td>{item.trn.trNum}</td>
                  <td>{item.trn.trYear}</td>
                </tr>
              ))}
            </tbody>
          </table>
        </Card.Body>
      </Card>
    </>
  );
};

export default Ufindall;
```