I. Project Background

1. Topic

Railway transportation has the advantage of carrying large numbers of passengers as well as handling huge amounts of commodities for transporting over long distances with, especially, affordable price options for customers compared with aircraft. However, the way to manage the railway system is what we are inspired by and look forward to exploring and learning the traffic system. In the project, the railway management system will be designed with an entity-relationship model, specifically with the relational model which is based on the dataset. Managing this system by creating a well-structured database might have several challenges and difficulties. Hence, The purpose of this project is endeavouring to control, record the data efficiently either for customer's information, the infrastructure such as tracks, tunnels, stations as well as keep track of the railway trip. Forming a well-design structure in the railway management system project will help us improve the capability of data management, creating a stable system, and smooth maintenance. Furthermore, we named our group as '4-Tech' which means 'For Technology' or can be understood as four members for technology.

2. Scope

IN SCOPE	**Priority**	**Features**	**Descriptions**
	Features For Staff		
	High	Ticket and information registration form	Staff can enter new passengers' information and register tickets that are name, national ID, contact information, etc.
		Ticket information page	Staff can search and edit information of tickets if needed
		Trip management section	To manage as well as displaying trains' information such as captains, departure or arrival time, destinations, etc.
	Medium	Application layout organisation	Organising the webpage into sections for more convenient and effective navigation.
		Staff information section	The section includes all of the staff background information, username and password, etc.
		Station information board	Display status, address, location, opening and closing time.
		Seat information section	To display, edit and add available seat information by carriages.
		Carriage information section	To display, edit or add carriage information by train.
		Statistical Report	This section shows statistical information for the

			managers that are tables, charts and summary of monthly performance. Information can be search, filter and sort
	Low	Login/Logout page	Restrict access for staff only
		Account authorisation	Admin can control access of staff by providing authorisation to certain individuals.
OUT SCOPE		Report system	For passengers to report or complain about unpleasant experience so that the railway service can improve
		Maintenance screen	Screen to inform the maintenance event id for staff and maintenance message for passengers.
		Route Management	To manage information of routes
		Staff department, position, and job history	To manage more detailed information of staff
		Features for passengers	For users to navigate the platform as well as interact with the railway service

After many weeks we have updated the scope by moving a few functions into the out scope section as we realised the features are unfeasible and would result in redundancy. Such functions are route management and some information regarding staff. For route management, we recognised that it is unnecessary to keep track of the components since the route's existence stays constant and their operations are controlled by a different system. Secondly, we have managed to trim down the bulky staff data schema as it dawned on us that certain information is managed by other systems, redundant if included in a railway management system.

Furthermore, the group has decided to focus solely on the functions for staff, moving passengers' to out-scope. This action is to simplify the project within the time limitation of the course and to improve the overall quality of the application.

Outcome

After a period of 2 months, the project was concluded with most of the features stated in the scope having completed. The team has designed and developed successfully a database with multiple tables, so that data of different subjects can be efficiently and effectively stored as well as displayed in tables in relevant sections. Besides that, the application is also able to perform CRUD tasks on the subjects, enabling the users to manage the data in the database conveniently. Aside from data management, the application can generate a report page and a dashboard with informative visualisations to help users keeping track of the changes in data and development trends of the railway system. Last but not least, authentication and authorisation have also been implemented to restrict access to certain individuals and protect the data.

3. Progress

There are some modifications related to the Project's timeline due to the adjust in making plan during our studying time. Referring to the previous timeline, we are in the stage of designing and constructing the main sections. Parts of our work are completely progress, however, for the final advanced products, some sections have been replaced or dropped. This schedule will illustrate more detail.

	TIMETABLE			
WEEK	**Duration**	**Start**	**Finish**	**Note**
Building the ***Ticket and information registration***	4 weeks	30/11/2020	21/12/2020	
Design the ***Ticket + Passengers information page + Ticket booking section***	4 weeks	30/11/2020	21/12/2020	
Building ***Train and Trip management section.***	4 weeks	30/11/2020	21/12/2020	
Building the ***Seats + Carriages information navigation***	2 weeks	22/12/2020	5/1/2021	
Building Scope: Medium				
Insert data for all these tables	2 weeks	22/12/2020	5/1/2021	
Design the ***Dashboard, Query section***	2 weeks	22/12/2020	5/1/2021	Display in table
Design the ***Staff information section + Statistical Report***	2 weeks	22/12/2020	5/1/2021	
Feedback page	2 weeks	22/12/2020	5/1/2021	
Building Scope: Low				
Sign up/Sign in/Sign out page	1 weeks	6/1/2021	13/1/2021	

We attempt to retain the timeline in scope from the most essential to the least important as well as keep the deadline and people performing each task.

What has changed:

- Instead of building a homepage for customers to book tickets, we focus more on upgrading the information management system for tickets, customers, trains for Staff.
- We replace the Route table with the Trip table because it is more convenient to query.
- Carriage is also new data for information systems.
- Data Insert also takes a large amount of time compared to the old plan.

4. Future directions

In the field of information technology, there is no perfect system including a database system. In the project of designing a database for the railway management system, our whole team had a specific plan for each stage for each specific feature of the railway management system. During the project's implementation, we cannot avoid the limitations, which are the challenges we need to improve and develop a highly realistic railway system. In the later stages of the project, in the future we have many directions to improve the project with a high quality and full range of features required for an effective railway management system including technical and non-technical.

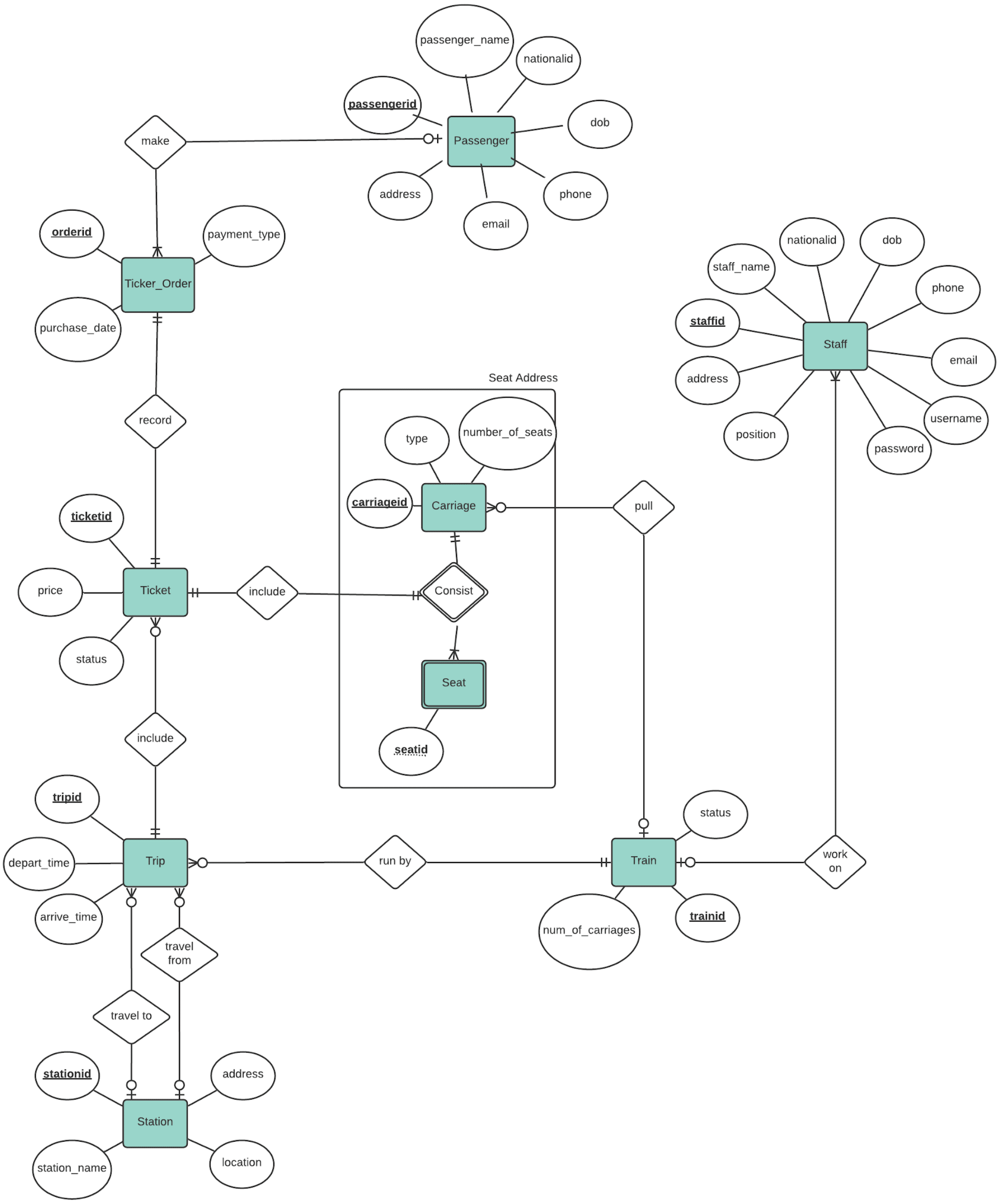
- One of the directions that our team has set for the next phase of the project is to research and come up with solutions to implement out of scope of the project. Extensive ticketing features and database statistics are required for users including railroad operators and passengers, who will book train tickets through the online booking system. This section is considered to develop and upgrade the system of user interface more efficiently and convenient.

- For the technical future direction of the railway management system project, our team will continue to develop databases including report systems, maintenance screen, route management, staff department, position and job history, features for passengers sections on the Oracle Application Express (Oracle APEX) platform. These system features that our team plan to implement in the future require database programming techniques through the Oracle programming language and some user interface programming techniques on the Oracle APEX. It can be said that these technical requirements are not too difficult and complex, but they require the whole team to spend time researching and coming up with suitable solutions for each function of the system so that it brings a significant effect.

- In the future, our team not only has technical orientation to programming, but also plans to develop work capabilities using special diagrams, ER Diagrams. It can be seen that the relationship between the database tables is very important to help users easily query related data. Not only the railway management system project but also all projects that work directly to the database, a diagram is indispensable if one wants to create a useful and logical system. We will focus on developing the system's activity database outlines through the method of using the ER diagram to represent relational schema relationships. Our team will continue to study what is a highly logical diagram from which to easily connect to the existing system, edit and release a useful railway system management database.

II. Application Design and Implementation

1. ERD



Entity Relationship Diagram of Railway Management System

As visualised in the entity-relationship diagram, the entity Passenger will maintain personal information of the customer, which are distinct ID, name, national ID, date of birth, phone number, email and address. Furthermore, each customer when booking one or multiple train tickets, at the same will make one or many Ticket_Order.

Ticket_Order will record and archive information of the purchase and transaction that are the dissimilar order ID, day of purchase and payment method. Furthermore, each order will also include the information regarding a Ticket that passengers intend to purchase.

Every Ticket will have a unique ID, price and status. Status is to determine and distinguish between the tickets that have been purchased and those that remain available. Besides that, in a ticket, information of Trips and Seat Address are also displayed.

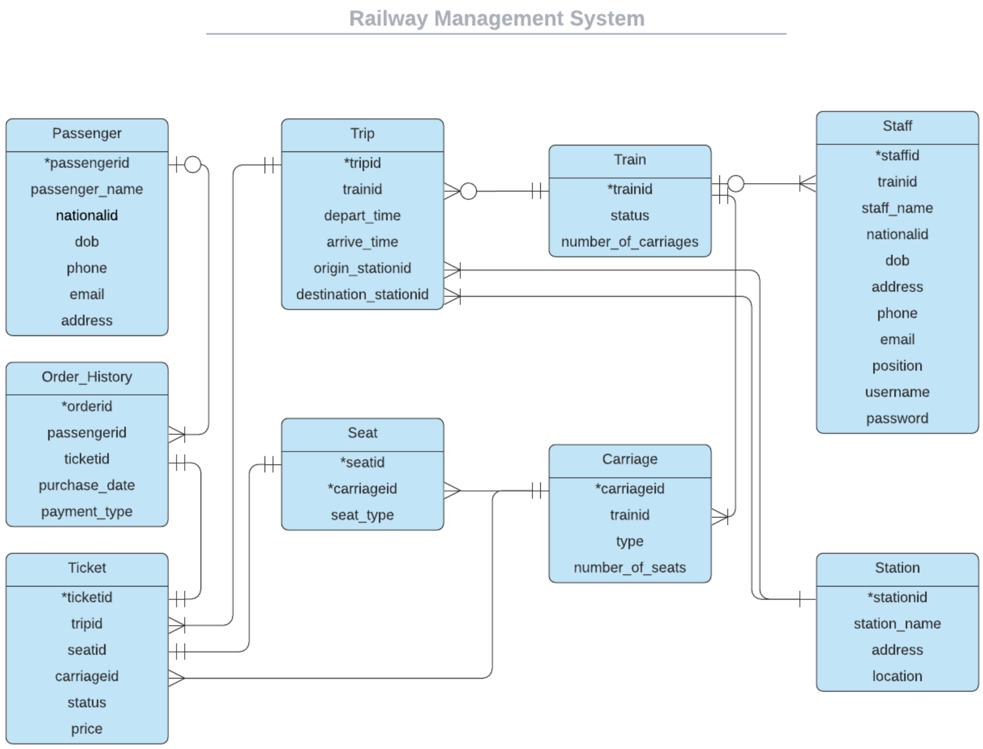
Each trip will have its own ID for convenience of management, depart time and arrival time. Moreover, a trip will also require information of Station in order to register the station of origin and destination. The information of each station such as distinct ID, station names, location and addresses are correspondingly logged and maintained.

Regarding the Seat Address, it is an aggregation of carriage and seat as seat is a weak entity and is inseparable from a carriage. Henceforth, a seat ID requires a carriage ID to form a complete address of Seat Address. A carriage, in addition, maintains other attributes being type and number of seats it consists of.

One or multiple carriages can be attached to and pulled by a train, though not always. Each train can be used for many trips and are distinguished by their ID, along train status and numbers of carriages.

Finally, each train will have numerous staff assigned to work on. Staff personal information is registered and recorded which are ID as primary key, name, national ID, date of birth, email, phone number and address, plus with position, username and password.

2. Relational schema

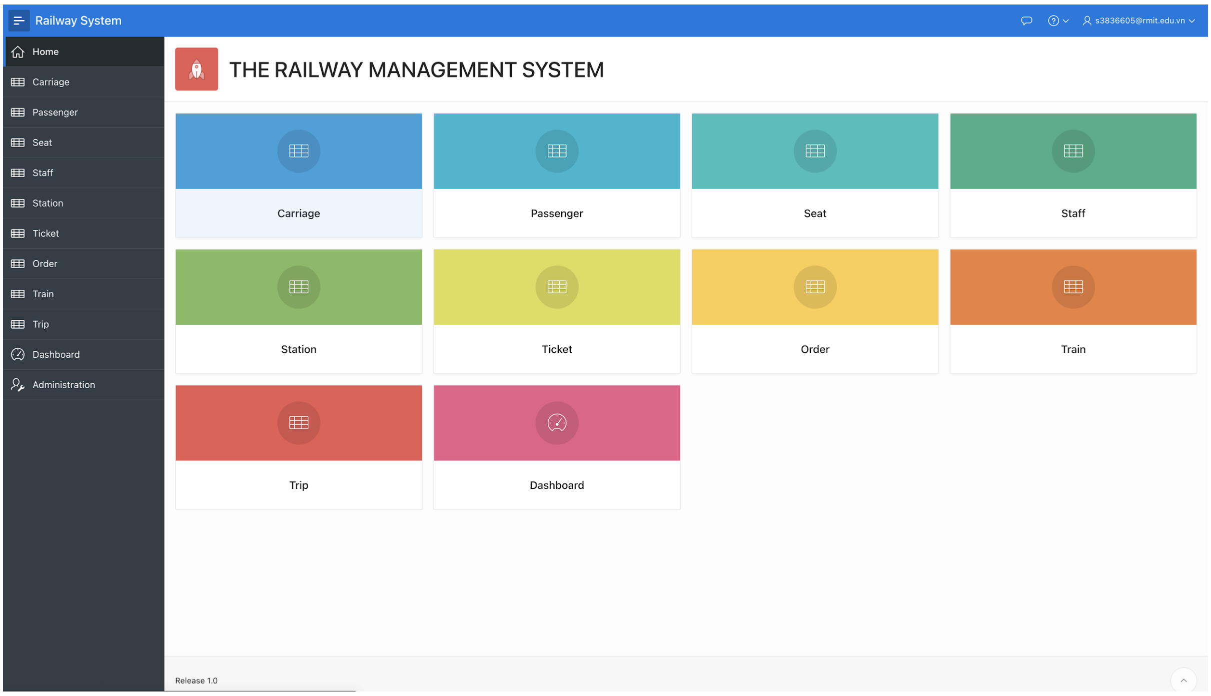


- Passenger(passengerid, passenger_name, nationwide, dob, phone, email, address)
- Order_History(orderid, **Passenger.passengerid** passengerid, **Ticket.ticketid** ticketid, purchase_date, payment_type)
- Ticket(ticketid,**Trip.tripid** tripid, **Seat.seatid** seatid, **Carriage.carriageid** carriageid, status, price)
- Trip(tripid, **Train.trainid** trainid, depart_time, arrive_time, **Station.stationid** origin_stationid, **Station.stationid** destination_stationid)
- Seat(seatid, **Carriage.carriageid** carriageid, seat_type)
- Carriage(carriageid, **Train.trainid** trainid, type, number_of_seats)
- Train(trainid, status, number_of_carriages)
- Staff(staffid, **Train.trainid** trainid, staff_name, nationalid, dob, address, phone, email, position, username, password)
- Station(stationid, station_name, address, location)

3. Database creation

4. Application features

The application is a prototype and an interface to manage a railway system. With this application, information of subjects involved in the system will be listed out. The subjects included are: Passenger, Ticket, Order, Staff, Seat, Carriage, Train, Station and Trip. Additionally, a dashboard is available to visualise and report insights to users. In each section, various actions can be done to manage data in the database and are described in detail below.



P.1: Homepage of the application

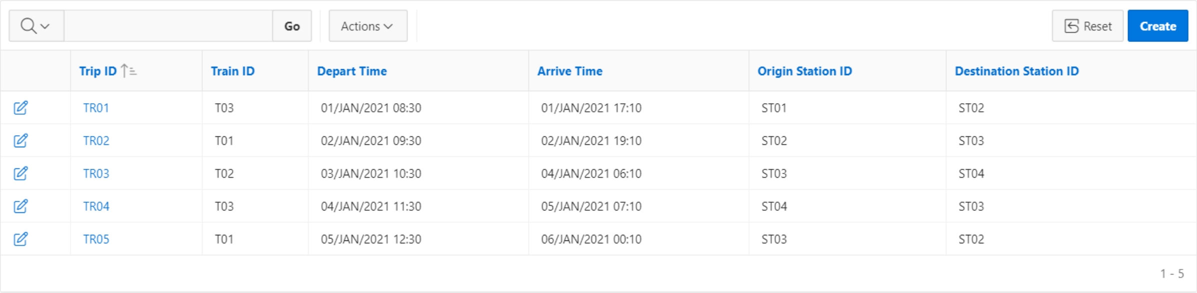
1. Viewing and managing data of Railway Management Application

By using this application, staff are able to view all information of each entity of the system. The information of the ticket can include: the entity's ID, relevant information, and other IDs of relating entities. The IDs will serve as relationships for query and create complete queries and views, comprehensible for a non-technical user. Accessing this section is straightforward and can be accessed as following:

- From Homepage (as seen in picture P.1 above), users can navigate directly to the section by clicking on the according icon.
- Another more global method to access the sections is to navigate the side menu (As shown on the far left side in picture P.1).

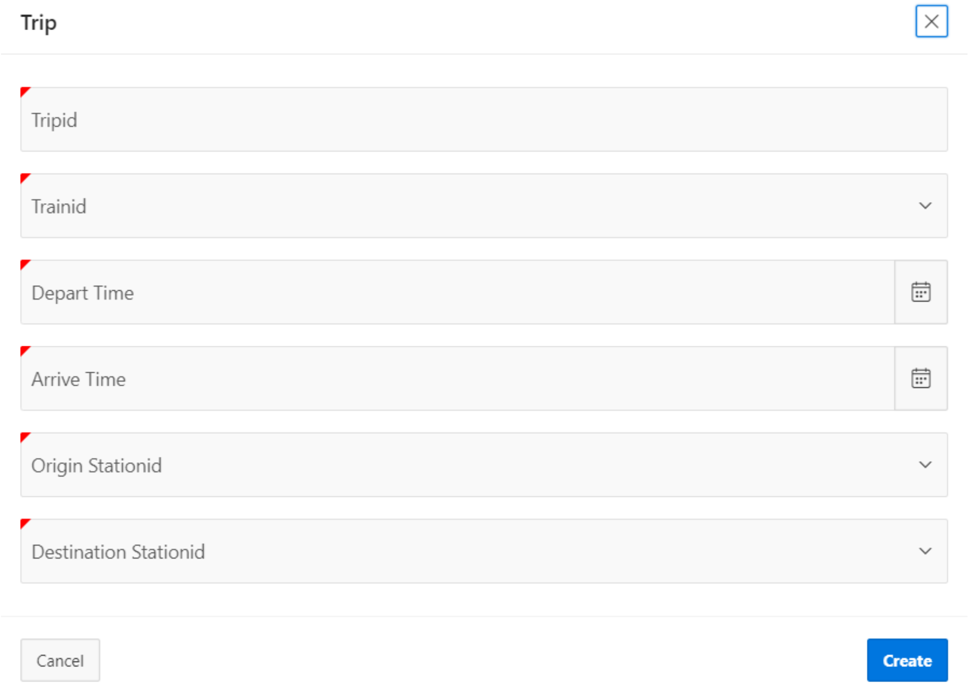
In each section, various actions can be done to manage information about the subject such as: Creating new instances, searching for existing instances, editing information and deleting instances.

- Create new instances:



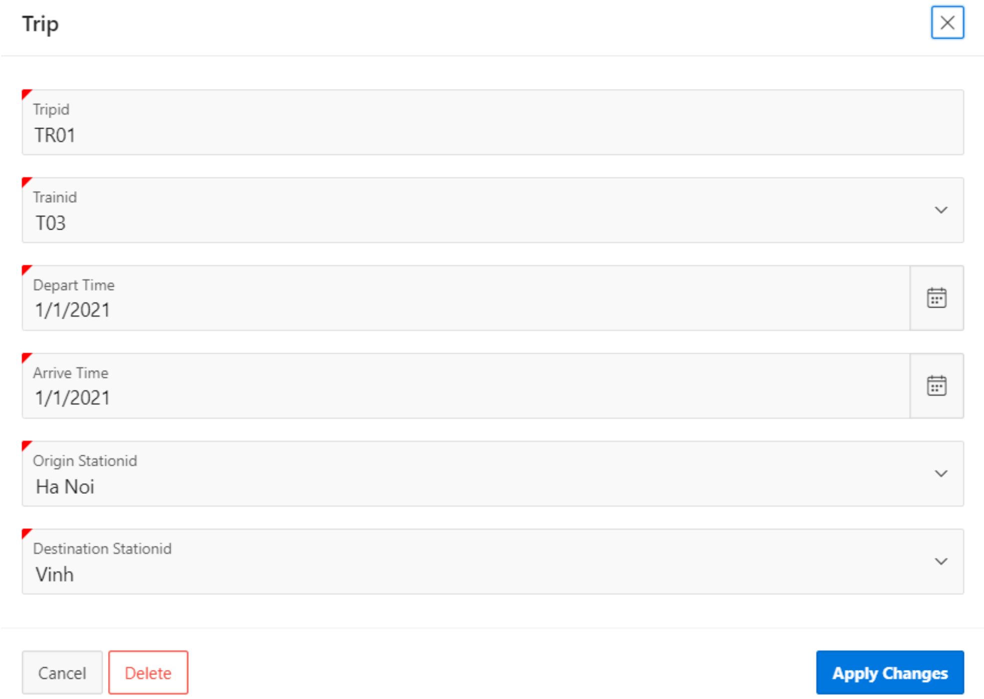
	Trip ID	Train ID	Depart Time	Arrive Time	Origin Station ID	Destination Station ID
✎	TR01	T03	01/JAN/2021 08:30	01/JAN/2021 17:10	ST01	ST02
✎	TR02	T01	02/JAN/2021 09:30	02/JAN/2021 19:10	ST02	ST03
✎	TR03	T02	03/JAN/2021 10:30	04/JAN/2021 06:10	ST03	ST04
✎	TR04	T03	04/JAN/2021 11:30	05/JAN/2021 07:10	ST04	ST03
✎	TR05	T01	05/JAN/2021 12:30	06/JAN/2021 00:10	ST03	ST02

Step 1: To create a new instance, users can click the Create button located on the top right corner of the table. Once clicked, a form will appear acquiring necessary information for an instance.



Step 2: Users can start filling in necessary information. After all information is filled in, the new subject can be created by clicking Create, or discard by clicking Cancel or the X icon on the top right corner of the form.

- Edit and deleting instances:



Trip

Tripid
TR01

Trainid
T03

Depart Time
1/1/2021

Arrive Time
1/1/2021

Origin Stationid
Ha Noi

Destination Stationid
Vinh

Step 1: Both tasks can be performed by navigating to the subject that users intend to make change.

Step 2: Then click on the Edit icon to open the panel where information about an existing instance is displayed.

Step 3: Users can then make changes to the data by replacing current information.

Step 4: Confirm by clicking Apply Changes located at the bottom right. Besides that, to delete the data, users can simply click the red Delete button at the bottom left.

As new data are entered and current data changed by the users, constraints are set up to ensure the integrity and consistency of the database. For instance, national ID and phone numbers must comply with national format or certain fields must contain completely unique data. If the input by the users violates the constraints of the application, the form will not be submitted and, instead, errors will be highlighted to notify the users.

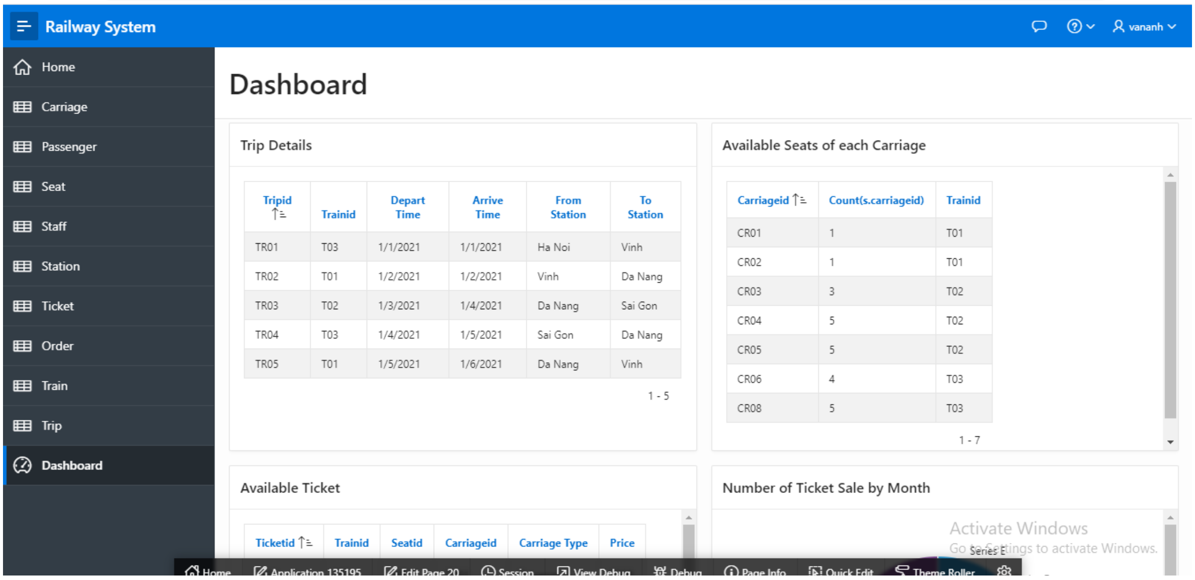
In addition, tasks regarding displaying the data are available, which are filtering, sorting, grouping and generating visualisations. These features are to create a convenient environment for the users to quickly extract insights and manage the data. To access the functions simply:

Step 1: navigate to the data table and click on the Action button located to the right of the search bar.

Step 2: A menu will drop down and users can choose the intended action.

2. Report

The report is the aggregate report and summarises the tables' data values in the railway management system illustrated by detailed tables. The purpose is for the manager to have an overview of the data so that it can undoubtedly be summarized as well as implement strategic directions in the future.



Dashboard

Trip Details

Tripid ↑	Trainid	Depart Time	Arrive Time	From Station	To Station
TR01	T03	1/1/2021	1/1/2021	Ha Noi	Vinh
TR02	T01	1/2/2021	1/2/2021	Vinh	Da Nang
TR03	T02	1/3/2021	1/4/2021	Da Nang	Sai Gon
TR04	T03	1/4/2021	1/5/2021	Sai Gon	Da Nang
TR05	T01	1/5/2021	1/6/2021	Da Nang	Vinh

1 - 5

Available Seats of each Carriage

Carriageid ↑	Count(s.carriageid)	Trainid
CR01	1	T01
CR02	1	T01
CR03	3	T02
CR04	5	T02
CR05	5	T02
CR06	4	T03
CR08	5	T03

1 - 7

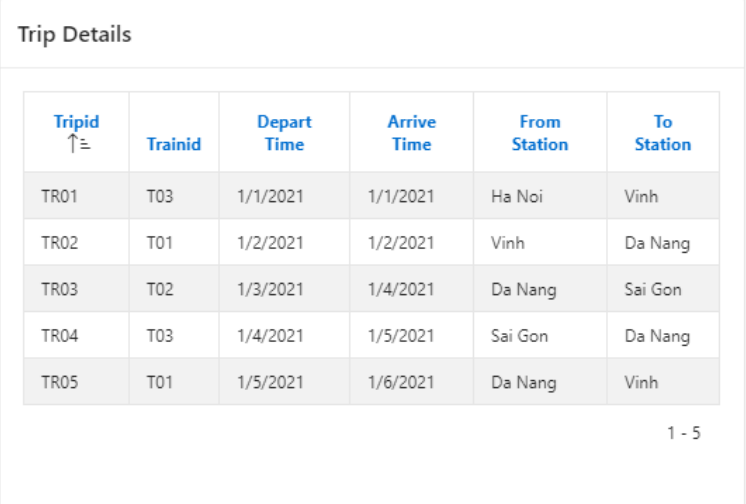
Available Ticket

Ticketid ↑	Trainid	Seatid	Carriageid	Carriage Type	Price

Number of Ticket Sale by Month

- **Trip Details**

Display all trips available corresponded with Depart Time (start) and Arrive Tim (end), name of two Stations for each Trip.



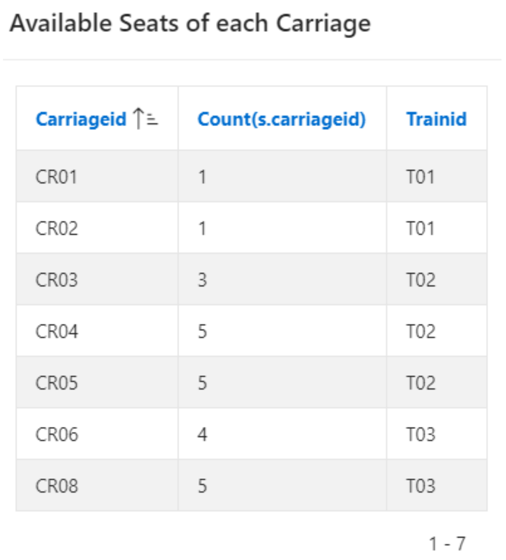
Trip Details

Tripid ↑	Trainid	Depart Time	Arrive Time	From Station	To Station
TR01	T03	1/1/2021	1/1/2021	Ha Noi	Vinh
TR02	T01	1/2/2021	1/2/2021	Vinh	Da Nang
TR03	T02	1/3/2021	1/4/2021	Da Nang	Sai Gon
TR04	T03	1/4/2021	1/5/2021	Sai Gon	Da Nang
TR05	T01	1/5/2021	1/6/2021	Da Nang	Vinh

1 - 5

- **Available Seats of Each Carriage**

The table displays the seats that are currently still available for purchase in each carriage. The number of seats are counted and grouped by carriage ID and train ID for viewers to identify the number of carriages.



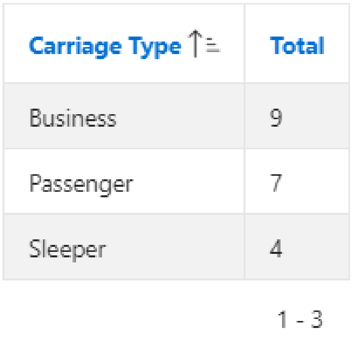
Available Seats of each Carriage

Carriageid	Count(s.carriageid)	Trainid
CR01	1	T01
CR02	1	T01
CR03	3	T02
CR04	5	T02
CR05	5	T02
CR06	4	T03
CR08	5	T03

1 - 7

- **Available Tickets**

The Available Tickets present list of tickets have not sold yet with Trainid, Seatid, Carriage Id, CarriageType and Price. It supports our Staff to check which tickets are available for Passenger in the process of booking tickets.

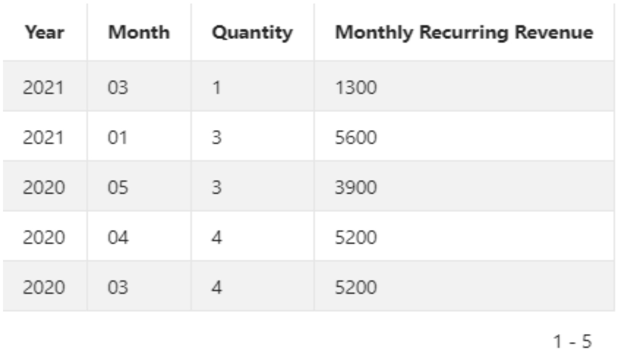


Carriage Type	Total
Business	9
Passenger	7
Sleeper	4

1 - 3

- **Number of tickets sold by month**

The number of tickets sold by month tables assist the administrator to observe aggregation of their Railway business.



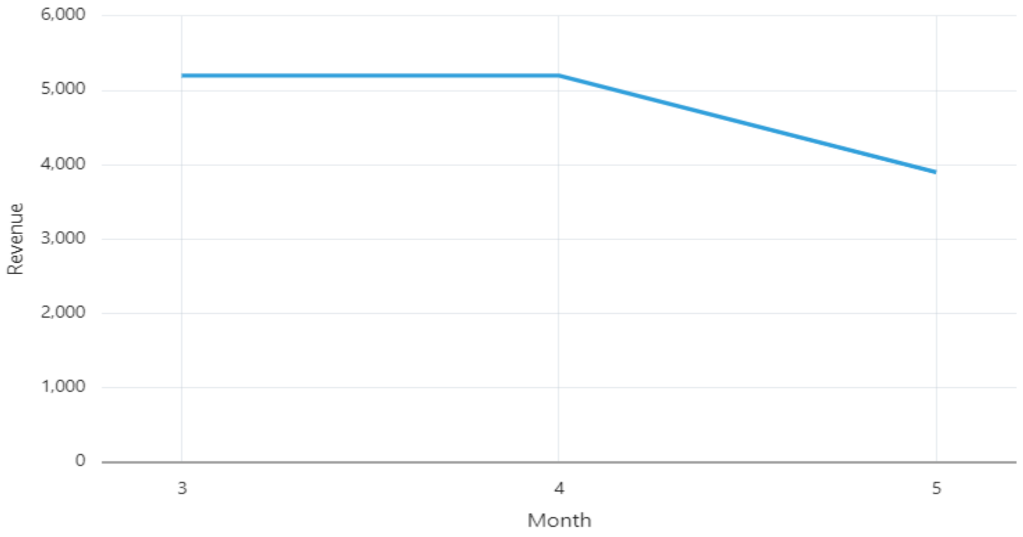
Year	Month	Quantity	Monthly Recurring Revenue
2021	03	1	1300
2021	01	3	5600
2020	05	3	3900
2020	04	4	5200
2020	03	4	5200

1 - 5

3. Dashboard

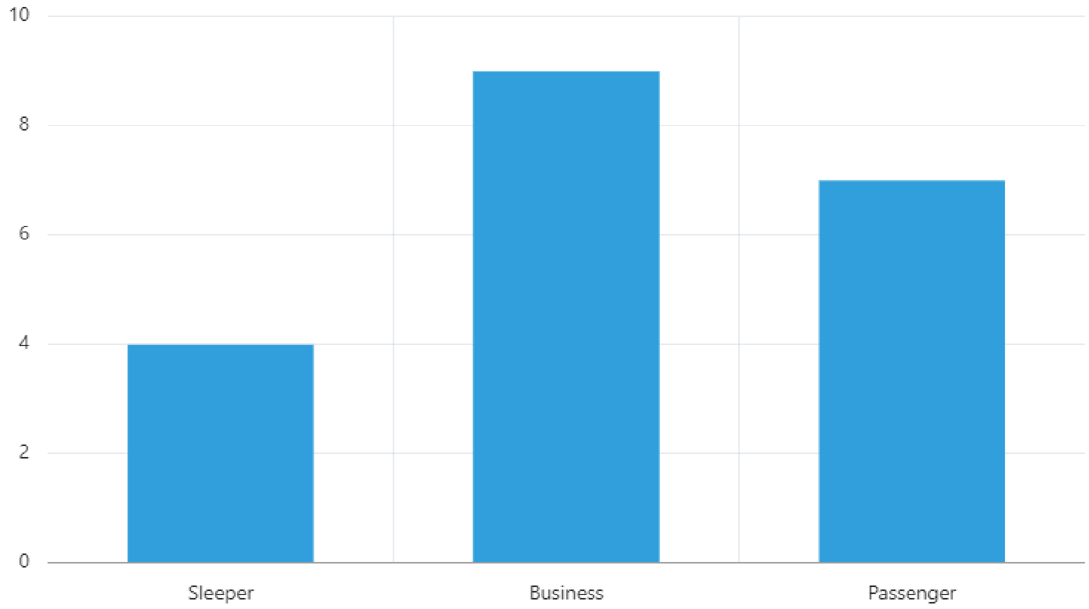
- **Monthly Recurring Revenue - 2020**

The line chart below shows the revenue every month in the year 2020. Statistical graphics visualise the data change by month. Currently, the chart only contains 3 month for demonstrating the data.



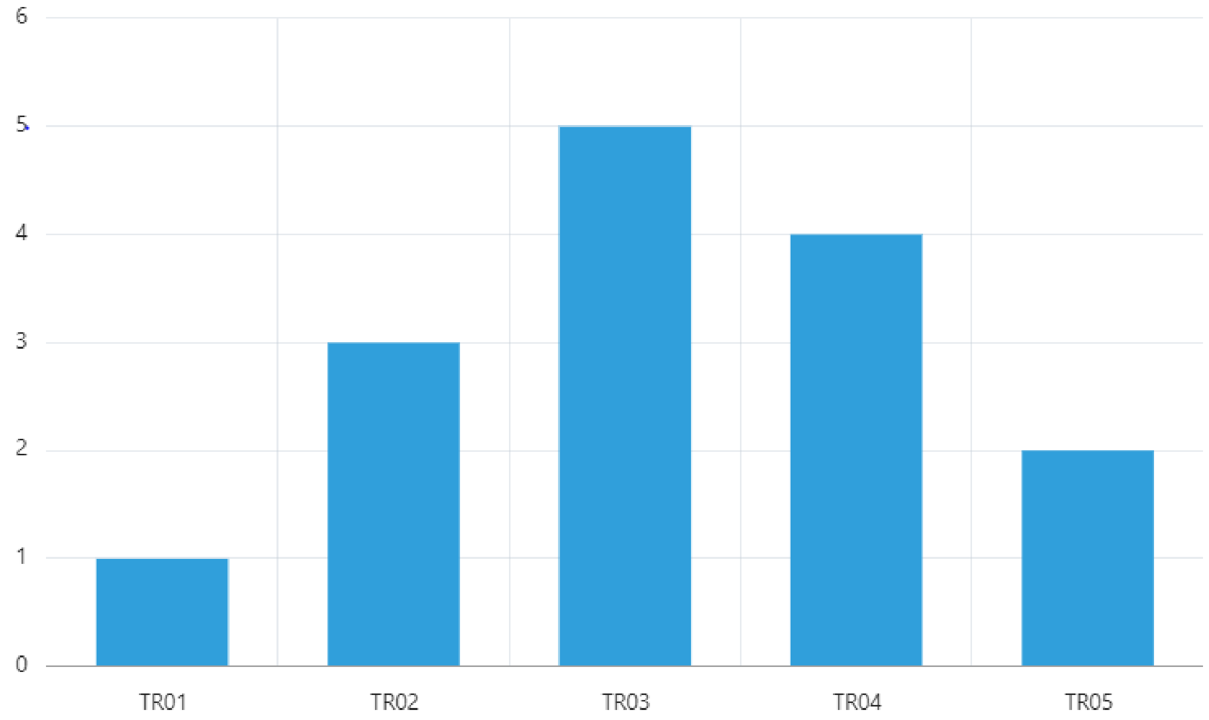
- **Available Seats by Carriage Type**

The bar chart represents the type of carriage bought by the customers. The staff as well as the manager can see the highest and lowest type of the carriage with this type.



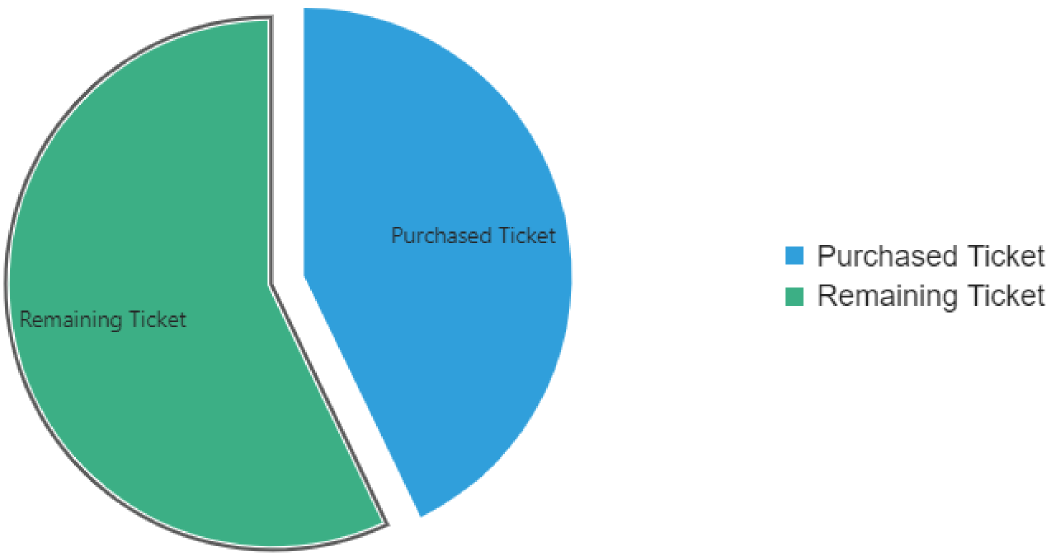
- **Number of Passengers by Trip**

The bar chart illustrates the quantity of the customers for each trip. From the chart, the manager can easily know the highest quantity of customers for the trip as well as the one.



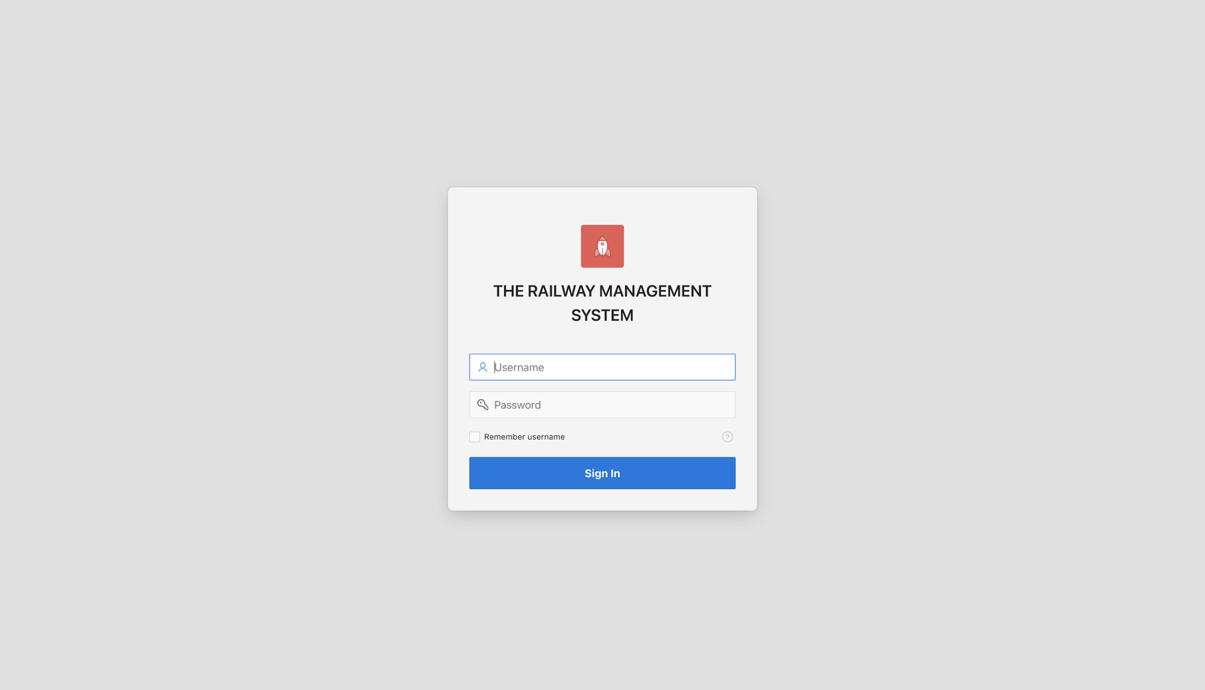
- **Ticket Status**

The pie chart shows the quantity of the purchased ticket and the remaining ticket of all the current systems. This kind of chart can illustrate the quantity of the data more efficiently.



4. Log in/Log out the Railway Management Application

Users who have the authorization to access our Railway Application are able to log-in and log-out whenever entrance. The Railway Management Application automatically prompts users to execute the log-in process again when they exit the application.



THE RAILWAY MANAGEMENT SYSTEM

5. Authorization for Reader, Contributor and Administrator

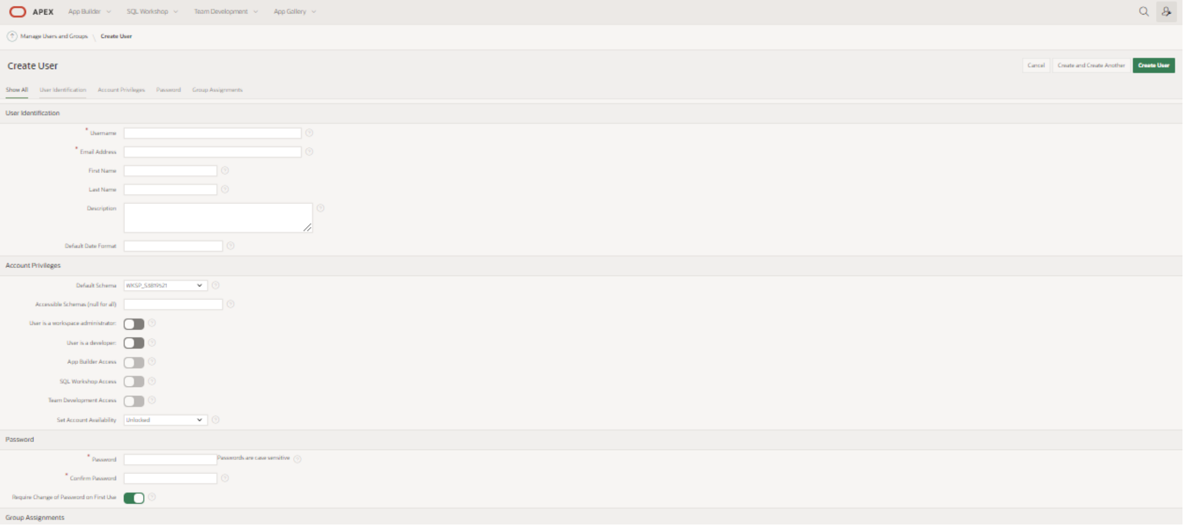
Administration can authorite for Reader, Contributor and Administrator the Railway Application:

- Readers can only view, they cannot add as well as modify the information.
- Contributors can add data, modify the information.
- Administrators can design, share and adjust everything of the Application.

To execute the Scope, the Administrator have to follow these steps:

Step1: The administrator have to log-in the Application then go to the **"Manage Users and Groups" Page**

For example: Create an account have a **username: VANANH** ; **password: 1234567z.**



Create User

User Identification

* Username	
* Email Address	
First Name	
Last Name	
Description	
Default Date Format	

Account Privileges

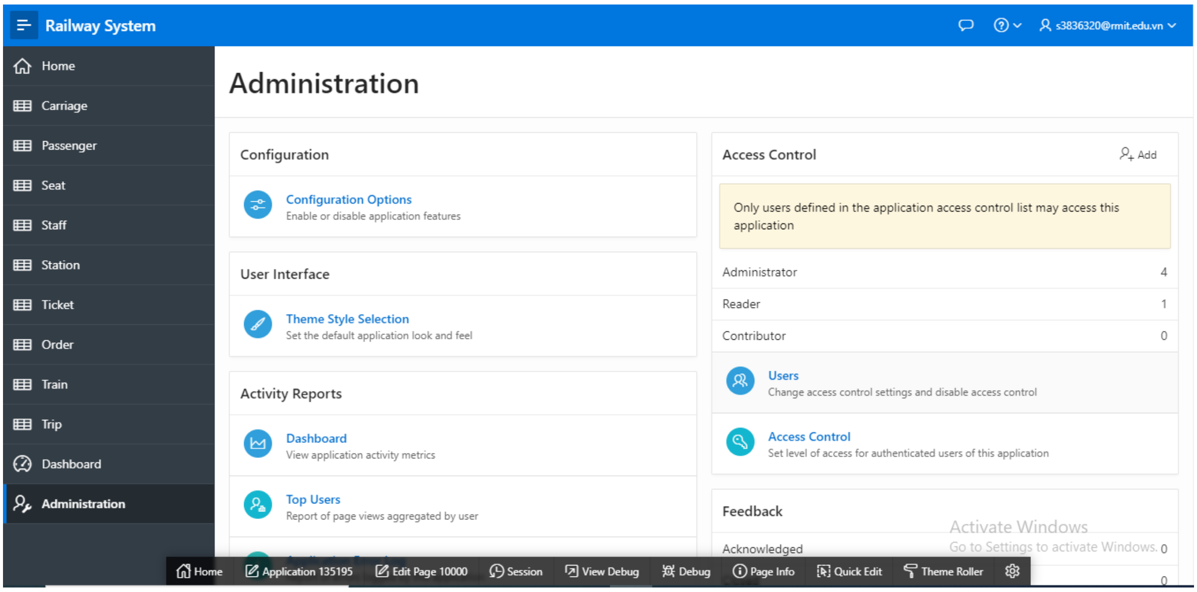
Default Schema	WKSP_S3819521
Accessible Schemas (null for all)	
User is a workspace administrator:	○
User is a developer:	○
App Builder Access	○
SQL Workshop Access	○
Team Development Access	○
Set Account Availability	Unlocked

Password

* Password	Passwords are case sensitive
* Confirm Password	
Require Change of Password on First Use	●

Group Assignments

Step2: After filling all the information needed, the administrator logs in to the Railway Management Application for Authorization step. The administrator selects the **User icon** on the right side.



Administration

Configuration

Configuration Options
Enable or disable application features

User Interface

Theme Style Selection
Set the default application look and feel

Activity Reports

Dashboard
View application activity metrics

Top Users
Report of page views aggregated by user

Access Control

Only users defined in the application access control list may access this application

Administrator	4
Reader	1
Contributor	0

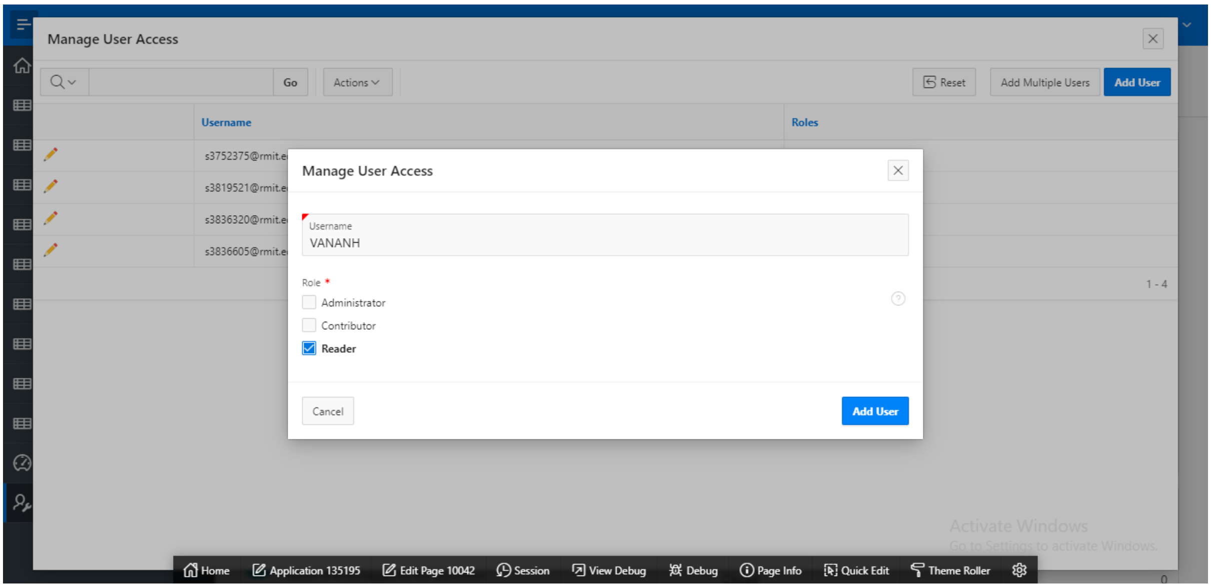
Users
Change access control settings and disable access control

Access Control
Set level of access for authenticated users of this application

Feedback

Acknowledged	0
	0

Step 3: After the User icon is selected, the screen displays the Table below. Administrator selects the **"Add User" button**.



Username	Roles
s3752375@rmit.e	
s3819521@rmit.e	
s3836320@rmit.e	
s3836605@rmit.e	

Manage User Access

Username
VANANH

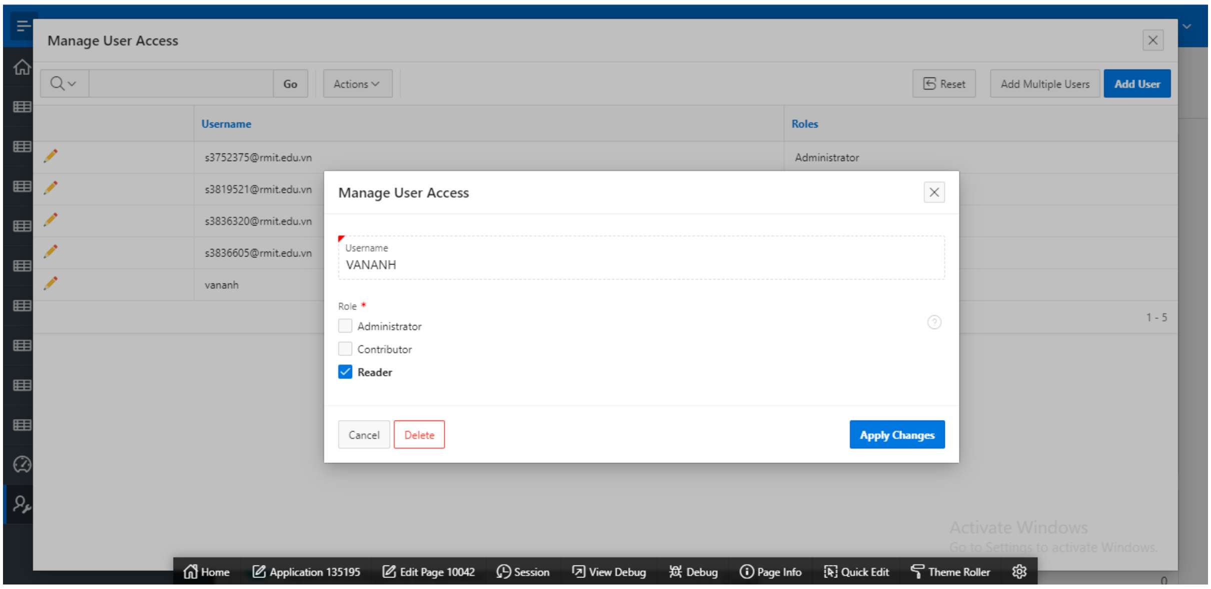
Role *
□ Administrator
□ Contributor
☑ Reader

Administrators fill the Username and then choose the Role they want to share.

Eg: In here, I fill username : VANANH and select the Role is Reader.

Optional Step:

Administrator can also modify all accounts they have shared by selecting ✎ the before the username.



Username	Roles
s3752375@rmit.edu.vn	Administrator
s3819521@rmit.edu.vn	
s3836320@rmit.edu.vn	
s3836605@rmit.edu.vn	
vananh	

Manage User Access

Username
VANANH

Role *
□ Administrator
□ Contributor
☑ Reader

6. Queries:

This is one of the queries we did against the Oracle database platform. Most of these queries our team used to represent graphically on the Dashboard page, the data tables on the Reports page of the railway management system application. These queries can provide a railway system manager with an overview of relevant information about the system and can be readily added and edited to user, train, and trip, number of tickets sold or purchased data:

```
Trip Details
-- SELECT T.tripid, T.trainid, T.depart_time, T.arrive_time,
S1.station_name from_station, S2.station_name to_station
-- FROM Trip T
--    LEFT JOIN  Station S1 ON S1.stationid = T.origin_stationid
--    LEFT JOIN  Station S2 ON S2.stationid =
T.destination_stationid;


 Available Seats of each Carriage
-- select cr.carriageid, count(s.carriageid),cr.trainid
-- from Seat s, Ticket tk, Carriage cr
-- where (tk.seatid = s.seatid and tk.carriageid = s.carriageid
and cr.carriageid = tk.carriageid
-- and (s.seatid, s.carriageid) not in ( select tk.seatid,
tk.carriageid
--                                                 from
Ticket_order t, Ticket tk
--                                                 where
tk.ticketid = t.ticketid))
-- group by cr.carriageid, cr.trainid;


 Number of Ticket Sale by Month
-- select to_char(T.purchase_date, 'YYYY') Year,
to_char(T.purchase_date, 'MM') Month, count(TK.ticketid)
Quantity, sum(Tk.price) Monthly_Recurring_Revenue
-- from ticket_order T, ticket Tk
-- where Tk.ticketid = T.ticketid
-- group by to_char(T.purchase_date, 'YYYY'),
to_char(T.purchase_date, 'MM')
-- order by to_number(to_char(T.purchase_date, 'YYYY')) desc,
to_number(to_char(T.purchase_date, 'MM')) desc;



Available Seats by Carriage Type
-- select  c.carriage_type, count(c.carriage_type ) as Total
-- from seat s, ticket tk1, carriage c
-- where tk1.seatid = s.seatid and tk1.carriageid = s.carriageid
and c.carriageid = tk1.carriageid
-- and (s.seatid, s.carriageid) not in ( select tk.seatid,
tk.carriageid
--                                                 from
ticket_order t, ticket tk
--                                                 where
tk.ticketid = t.ticketid)
-- group by c.carriage_type;
```

Monthly Recurring Renevue (Line Chart)

```
-- select to_number(to_char(T.purchase_date, 'YYYY')) Year,
to_number(to_char(T.purchase_date, 'MM')) Month,
count(TK.ticketid) Quantity, sum(Tk.price)
Monthly_Recurring_Revenue
-- from ticket_order T, ticket Tk
-- where Tk.ticketid = T.ticketid
-- and to_char(T.purchase_date, 'YYYY') = '2020'
-- group by to_number(to_char(T.purchase_date, 'YYYY')),
to_number(to_char(T.purchase_date, 'MM'))
-- order by to_number(to_char(T.purchase_date, 'MM'));
```

Available Seats by Carriage Type (Bar Chart)

```
-- select  c.carriage_type, count(c.carriage_type ) as Total
-- from seat s, ticket tk1, carriage c
-- where tk1.seatid = s.seatid and tk1.carriageid = s.carriageid
and c.carriageid = tk1.carriageid
-- and (s.seatid, s.carriageid) not in ( select tk.seatid,
tk.carriageid
--                                                 from
ticket_order t, ticket tk
--                                                 where
tk.ticketid = t.ticketid)
-- group by c.carriage_type;
```

Number of Passenger by Trip (Bar chart)

```
-- select Tr.tripid Trip, count(Tko.orderid) "Number of
Passengers"
-- from Trip Tr, Ticket_order Tko, Ticket Tk
-- where Tko.ticketid = Tk.ticketid and Tk.tripid = Tr.tripid
-- group by Tr.tripid;
```

Ticket Status (Pie Chart)

```
-- select 'Purchased Ticket',count(*) Total
-- from ticket_order T, ticket Tk
-- where Tk.ticketid = T.ticketid
-- union
-- select 'Remaining Ticket',count(*) as Name
-- from ticket Tk
-- where Tk.ticketid not in (select T1.ticketid from ticket_order
T1)
```

7. Application Link :

Link: https://apex.oracle.com/pls/apex/s3819521/r/project45/login
Username: long_nguyen
Password: 12345678z.   (notice that remember the dot - “.” character)