# Airline Booking Portal

Developed By: Shaik Akhila

GitHub : https://github.com/Instantgaming2356/JAVAFSD-Project02

The prototype of the application starts from the frontend, and it can also directly start from the project folder. This portal allows us to do flight management across administrator and provide flight booking facilities across client side which will at the end page goes to the payment portal(dummy). This prototype is built through various webpages (mainly .jsp file) which are interconnected with backend (servlets, database, models).

The implementation is done with the help of Hibernate, maven, Servlet, Java EE, Apache tomcat v8.5 and Eclipse.

## Sprint Planning

The Implementation is done in two sprints which are mentioned below:

Sprint 1:

- Clarify the specification and requirements.
- Implement a framework of certain pages such as admin login, homepage.
- Implement a blueprint of Controller, Models part at backend and its core functionality.
- Identifying the various association for mapping of Passenger with Flight database along with its attributes containing primary key in both the tables.

Sprint 2:

- Building a platform for the prototype with hibernate, maven (webapp archetype) integration along with MySQL as a database which will run on local server (Tomcat v8.5) and required dependencies.
- Creating JSP Page as a starting point containing a hyperlink that will take us to the Admin Login page and contains a html table containing booking details such as source, destination, date of boarding and number of persons.
- Afterward, as this part is broadly categorized into two parts: admin and passenger section.
- Introducing a single controller (Servlet) as the data will be share not just within admin or passenger section but also with each other, packages consisting of models, hibernate configuration, data transfer objects for database connectivity.

Sprint 3:

- Implement functionality in controller for validation in admin login page consisting of email and password which is stated as email (admin@test.com) and password (admin).

- Implement functionality for changing password in admin section which consist of new password and confirmation password (same as new password for recheck).
- Implementing a web page for admin login page and after successful login will jump to the admin main page.
- Implementing another JSP page for changing password which is not connected currently.

Sprint 4:

- Developing the main page displaying flight details along with add, change password and logout button.
- Adding functionality for adding flights acting as hyperlink that will take us to next page asking for Flight Details such as flight number, airline, origin, destination, flight date.
- Once the admin submits the required details, it will store all the valid data into the database through servlet along with auto increment primary key not null values. And it will display in the main page along with a “edit” and “delete” button on status column.

Sprint 5:

- From the passenger side, once a user registered the details for their travel, a filter operation is performed in backend by retrieving the flight details from database and filter it according to source, destination, and boarding date.
- Along each flight details, there will be a hyperlink button name “ Book Now”.
- From the “Book Now” button, it will jump to register page where user have to put his/her details according to the number of persons.

Sprint 6:

- In the registration page, there will be a option to select add passenger and a view table where user can view all the passenger details and modifications can also be done through status column option.
- Once the registration is done, it will show all the flight summary such as source, destination, date of boarding and total flight price.
- If a user registered passenger details more than number of persons, it will go back to the homepage again.

Sprint 7:

- From the passenger side, once a user registered the details for their travel, a filter operation is performed in backend by retrieving the flight details from database and filter it according to source, destination, and boarding date.
- Afterward, a dummy payment page will be displayed having an option to go back to home page as a hyperlink.
- Ensuring all the operations are tight and working well.
- Documentation.

**Technologies used in Airline Booking applications**

1. **Servlets:** Servlet technology is used to create a web application (resides at server side and generates a dynamic web page). Servlet technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language
2. **JDBC :** JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. It is a part of JavaSE (Java Standard Edition). JDBC API uses JDBC drivers to connect with the database.

A list of popular interfaces of JDBC API are given below:

3. **SQL:** SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database.
4. **JSP:** It stands for Java Server Pages. It is a server side technology. It is used for creating web application. It is used to create dynamic web content.

## Algorithm

Step 1> Start.

Step 2> Two options in the home page:

Case 1: If user select “admin” section then go to step 3.

Case 2: If user goes to passenger side through registering all booking details, then go step 7.

Step 3> Once a user select admin, it will prompt for admin email and admin password.

Step 4> An admin main page will be displayed containing three options and showing list of flights that admin has entered:

Case 1: Add Flights -> step 5.

Case 2: Change Password -> step 6.

Case 3: Logout and go back to step 2.

Step 5> A new window will be shown where admin can enter flight details and go back step 4.

Step 6> For changing password, it will ask for new and confirmation password from admin. Once it is validated correctly it will go back to home page.

Step 7> This will show a window having flights details that are filtered out through booking details. And the user has to select the flight for further action.

Step 8> Once, the user has selected the flight, user has to register his/her details and should be less than or equal to number of persons that user has specified.

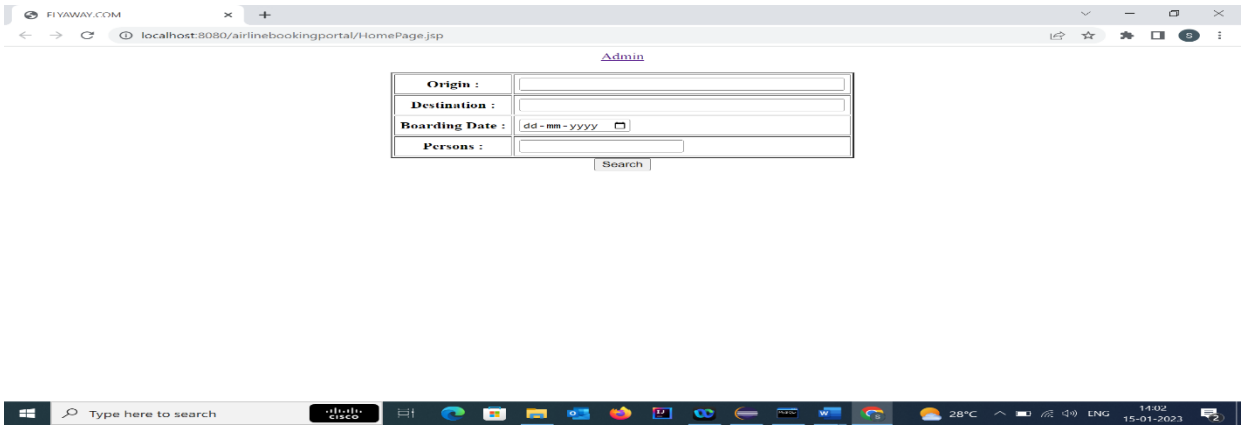
Step 9> After continuation, it will show the summary of user's flight and will prompt the user for payment (Dummy Payment Gateway).

Step 10> Once payment is successful, it will take the user back to step 2.

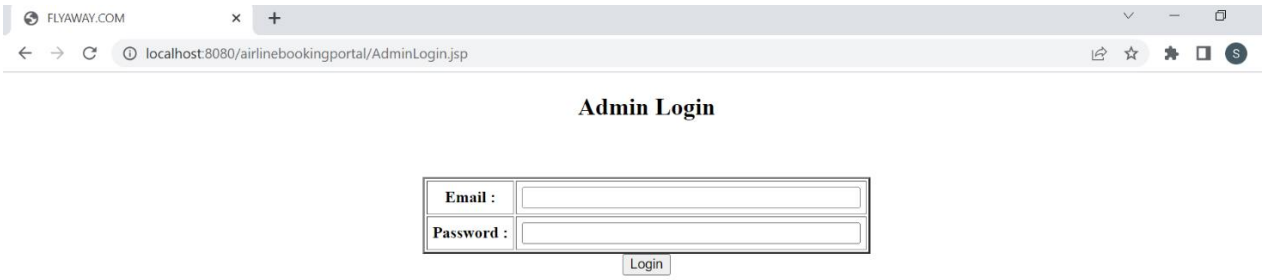
**Output Screens of the project:**

Here are the various output of Java Servlet Pages screens of the airline booking portal.

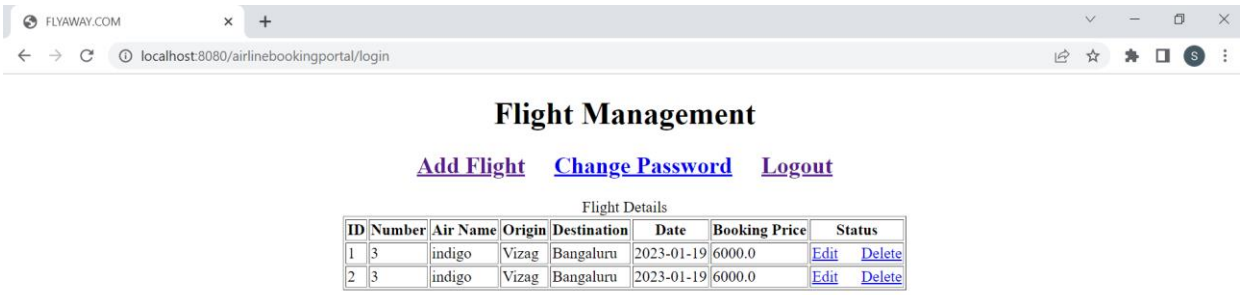
1: Home Page



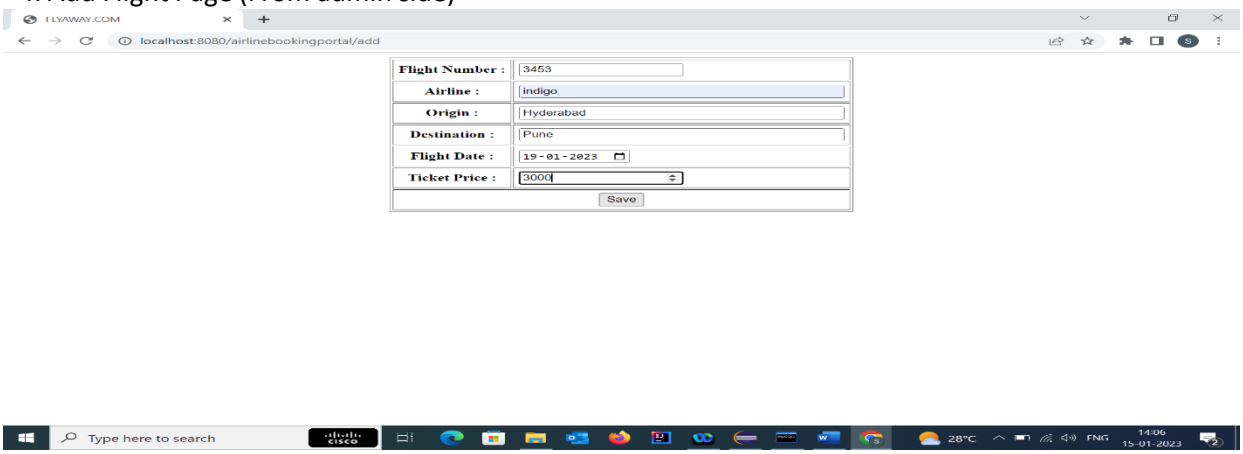

2: Admin Login Page (From admin side)



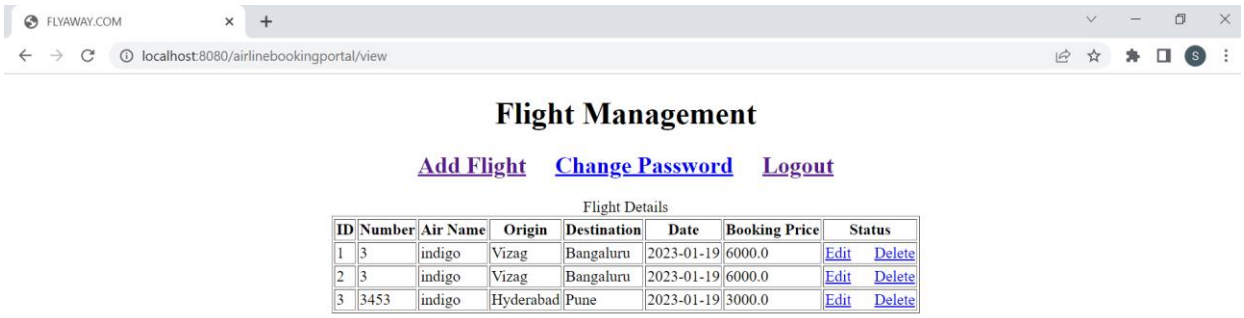

3: Admin main page (From admin side)




4: Add Flight Page (From admin side)

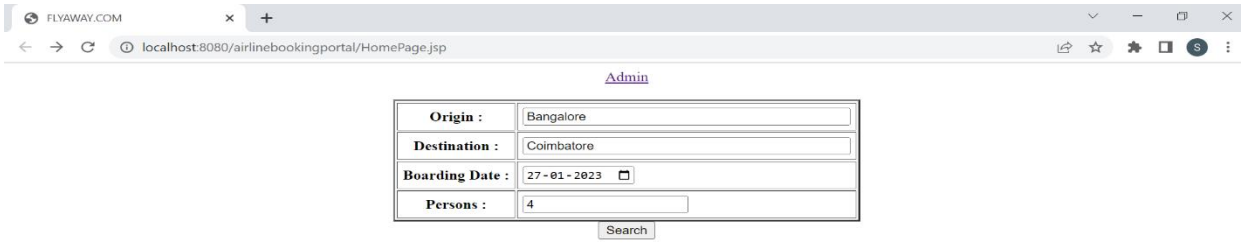



*Note: Once the user save it after inserting details, it will be inserted into the database.

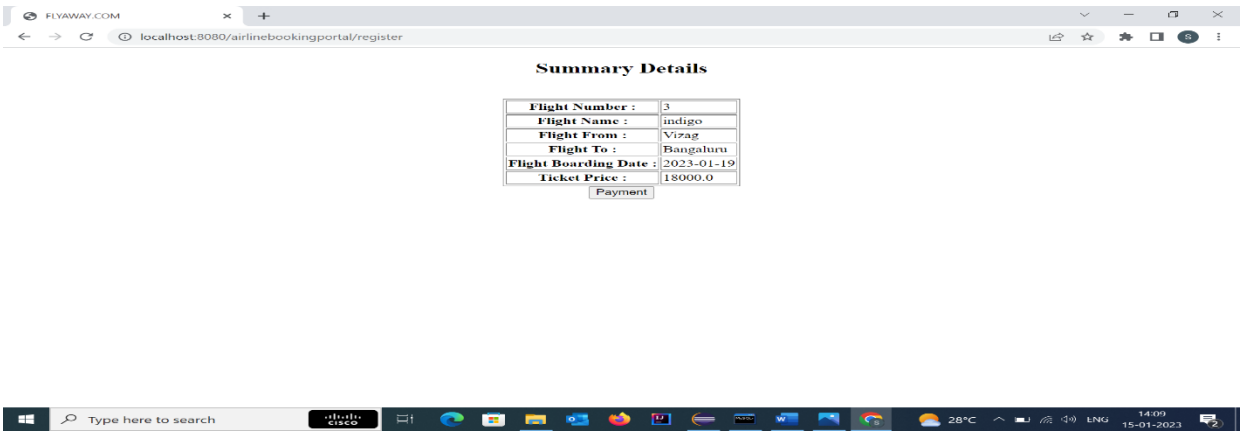



After, Admin can logout back to the homepage.

5: Insert the booking details from client side.

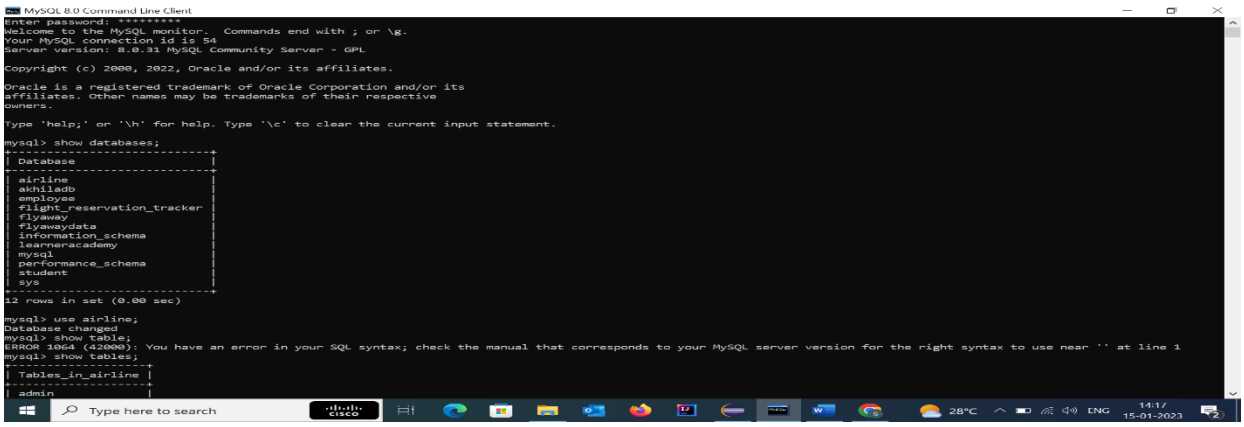


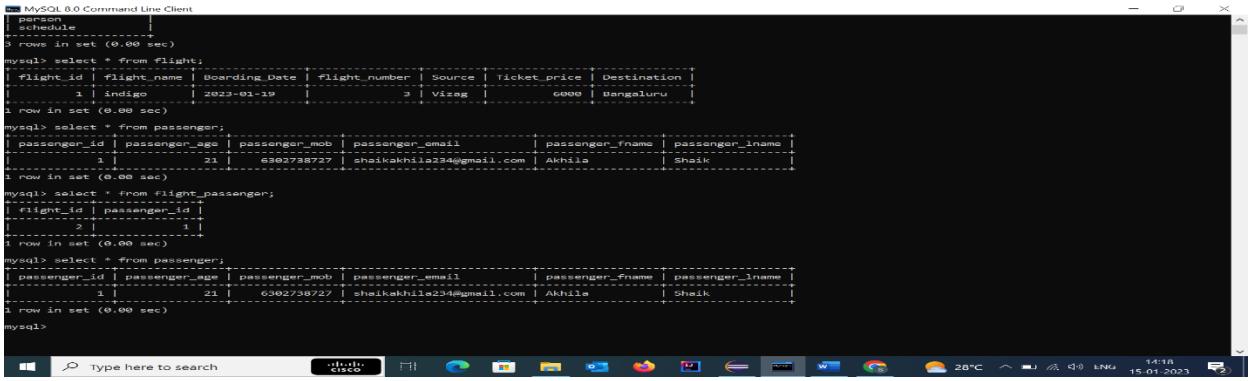

9: Summary Page




10: Payment Page

11: In MySqL Database, we have implemented @ManyToMany associations, here is the two tables flight and passengers table with its attributes and primary key.



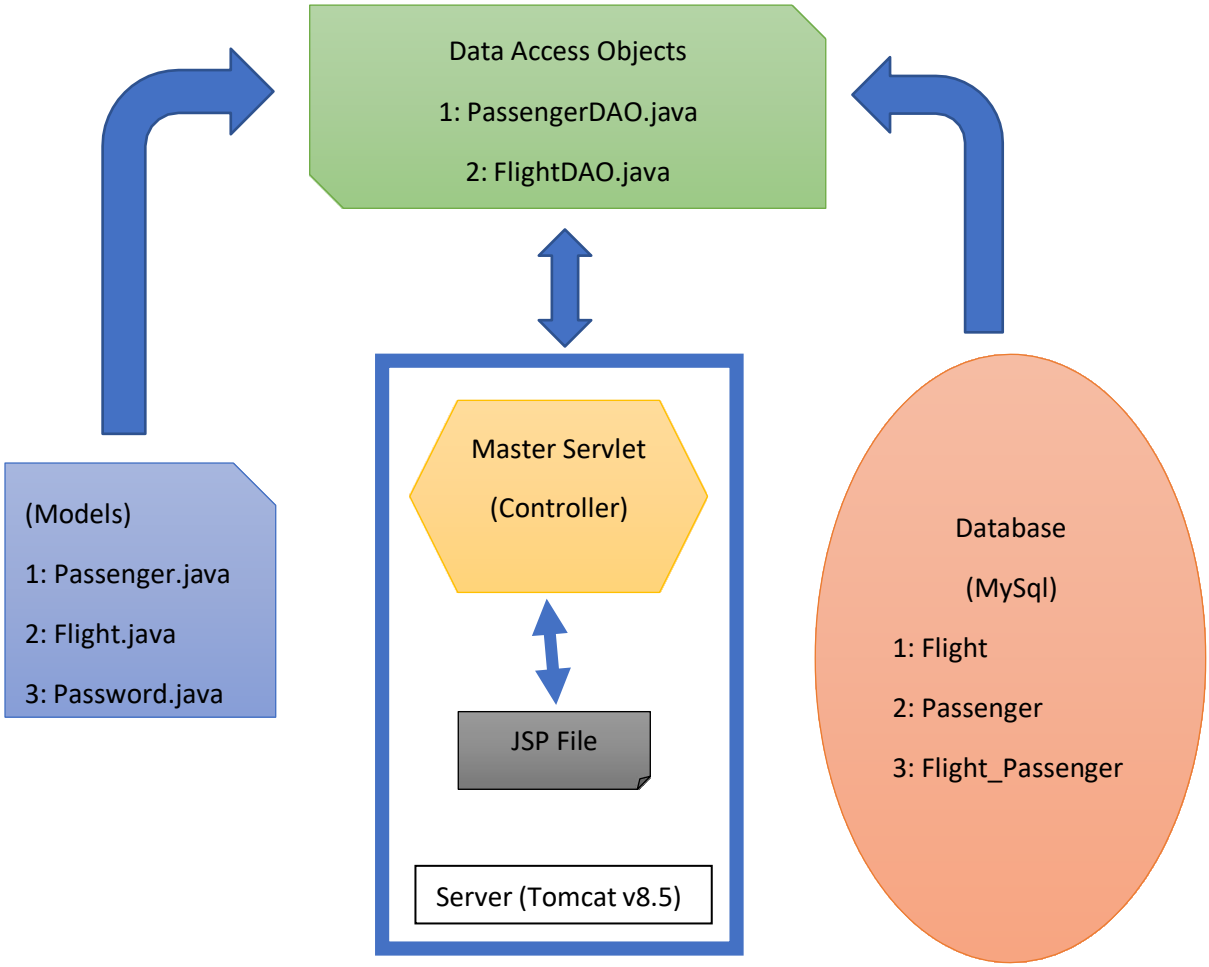




```
mysql> use demo;
Database changed
mysql> desc flight;
+--------------+-------------+------+-----+---------+----------------+
| Field        | Type        | Null | Key | Default | Extra          |
+--------------+-------------+------+-----+---------+----------------+
| flight_id    | int         | NO   | PRI | NULL    | auto_increment |
| flight_name  | varchar(255)| YES  |     | NULL    |                |
| Boarding_Date| varchar(255)| YES  |     | NULL    |                |
| flight_number| int         | YES  |     | NULL    |                |
| Source       | varchar(255)| YES  |     | NULL    |                |
| Ticket_price | float       | YES  |     | NULL    |                |
| Destination  | varchar(255)| YES  |     | NULL    |                |
+--------------+-------------+------+-----+---------+----------------+
7 rows in set (0.00 sec)

mysql> select * from  flight;
+-----------+-------------+---------------+---------------+----------+--------------+-------------+
| flight_id | flight_name | Boarding_Date | flight_number | Source   | Ticket_price | Destination |
+-----------+-------------+---------------+---------------+----------+--------------+-------------+
|         1 | Indigo      | 2023-01-30    |           426 | Bengalore|         5500 | Bhubaneswar |
|         2 | Vistara     | 2023-01-25    |           673 | Hyderabad|         3500 | Bangalore   |
|         3 | Air India   | 2023-01-27    |           367 | Bengalore|         3000 | Chennai     |
+-----------+-------------+---------------+---------------+----------+--------------+-------------+
3 rows in set (0.00 sec)
```

And a third table named (flight_passenger) containing the mapping of primary key of both the table is done.

```
mysql> select * from flight_passenger;
+-----------+-------------+
| flight_id | passenger_id |
+-----------+-------------+
|         4 |           1 |
|         4 |           2 |
+-----------+-------------+
2 rows in set (0.00 sec)

mysql> select * from flight;
+-----------+-------------+---------------+---------------+----------+--------------+-------------+
| flight_id | flight_name | Boarding_Date | flight_number | Source   | Ticket_price | Destination |
+-----------+-------------+---------------+---------------+----------+--------------+-------------+
|         1 | Indigo      | 2023-01-30    |           426 | Bengalore|         5500 | Bhubaneswar |
|         2 | Vistara     | 2023-01-25    |           673 | Hyderabad|         3500 | Bangalore   |
|         3 | Air India   | 2023-01-27    |           367 | Bengalore|         3000 | Chennai     |
|         4 | Vistara     | 2023-01-25    |           673 | Hyderabad|         3500 | Bangalore   |
+-----------+-------------+---------------+---------------+----------+--------------+-------------+
4 rows in set (0.00 sec)
```

# Flow Diagram




- Core Concepts used in this project are mostly maven, hibernate, MySQL, jdbc connector, associations, java, CRUD operations in a database, web development.

Step 11> Stop

## Conclusion

1: The prototype is robust and platform independent.

2: User can easily use the prototype and safely exit out of it.

3: As a developer, we can enhance it by introducing several new features such as guards along each web pages as currently its statically connected with each along with backend as will not allow to go back once admin has been logout, routing, custom validators and can have more user-friendly by adding styling (CSS, Bootstrap), custom loaders.

4: Though this prototype is tightly connected, the data will only persist in database until server is running and gets reset with each restarting of sever because of manual configuration of hibernate.

5: This prototype can also be implemented with multithreading to enable better performance.

6: And lastly, this prototype can be upgraded by implementing with securities patches to make it more versatile and secure in both local environment and global and later can be configured dynamically with connection of database through hibernate.