Travel Management System

Team: Arunsundar Kannan, Ashish Tak, Prajnya Satish (Team #39)

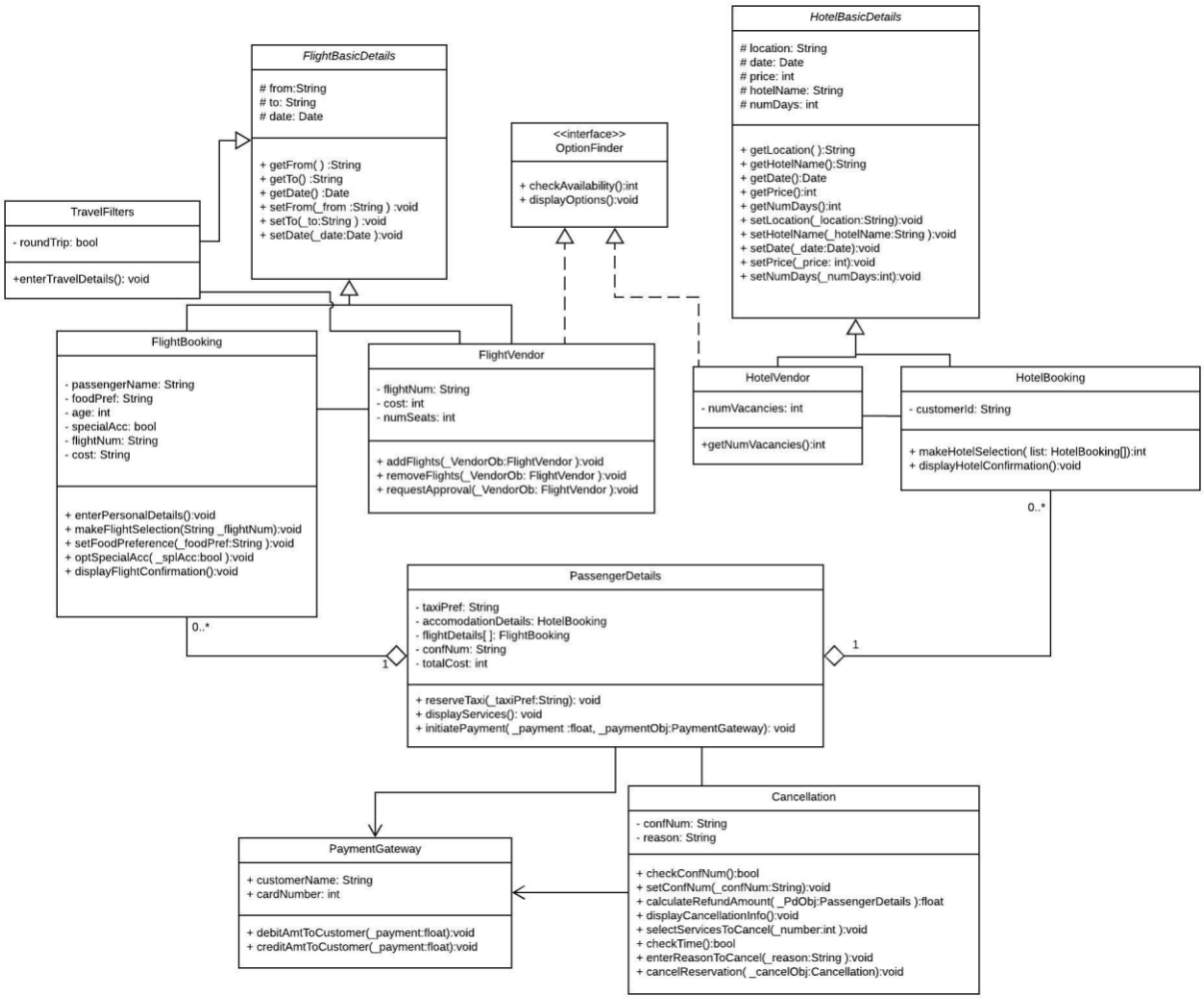
Title: Travel Management System

Project Description: A travel management system to enable a traveller to book flight tickets and manage other aspects of his journey such as food preference, accommodation and commute on the same platform. The customer books for all the services required together. If he so wishes, he could also cancel the booking on the same application.

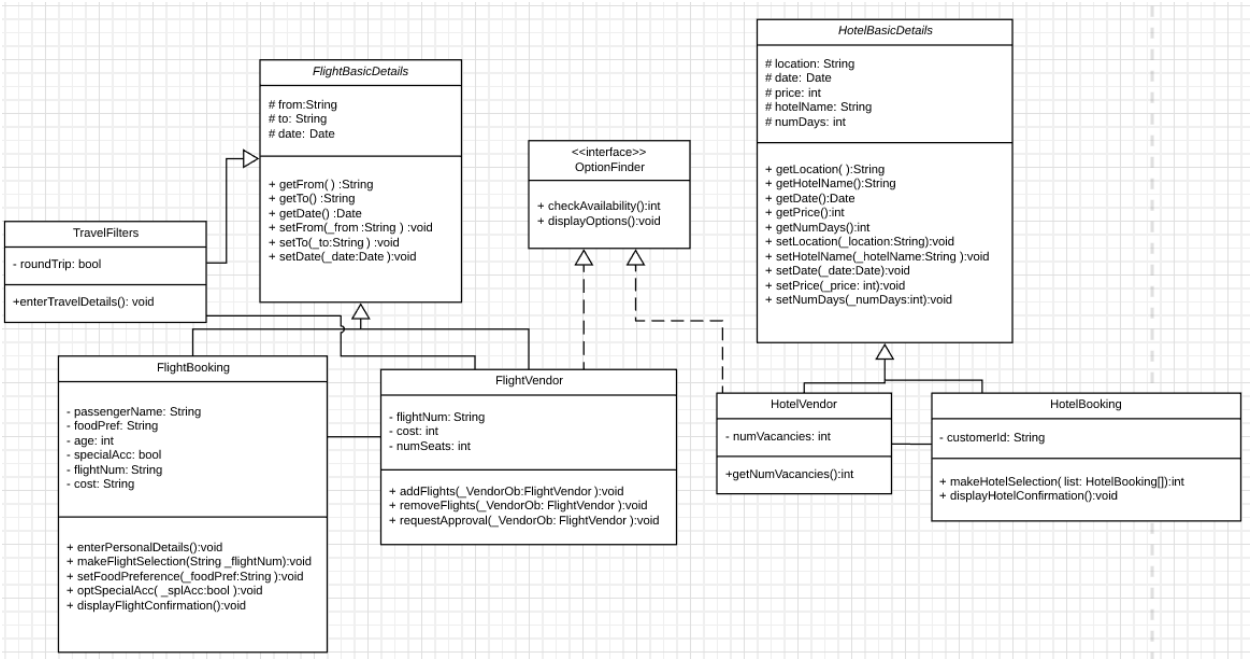
Framework: Spring

Data Storage: MySQL Database

Previous Class Diagram (with corrections):



Completed Class Diagram:



Summary: Corrections for the previous class diagram were made according to the comments given for Project's Part 2 report. Implementation of the abstract classes and the interface was completed by the first week. The project can be broadly divided into 2 modules: Flight module and Hotel module. The implementation of the concrete flight and hotel classes was done in the second week, along with the methods of the classes which were tested with model driver classes.

Breakdown:

Work done by Arunsundar Kannan:

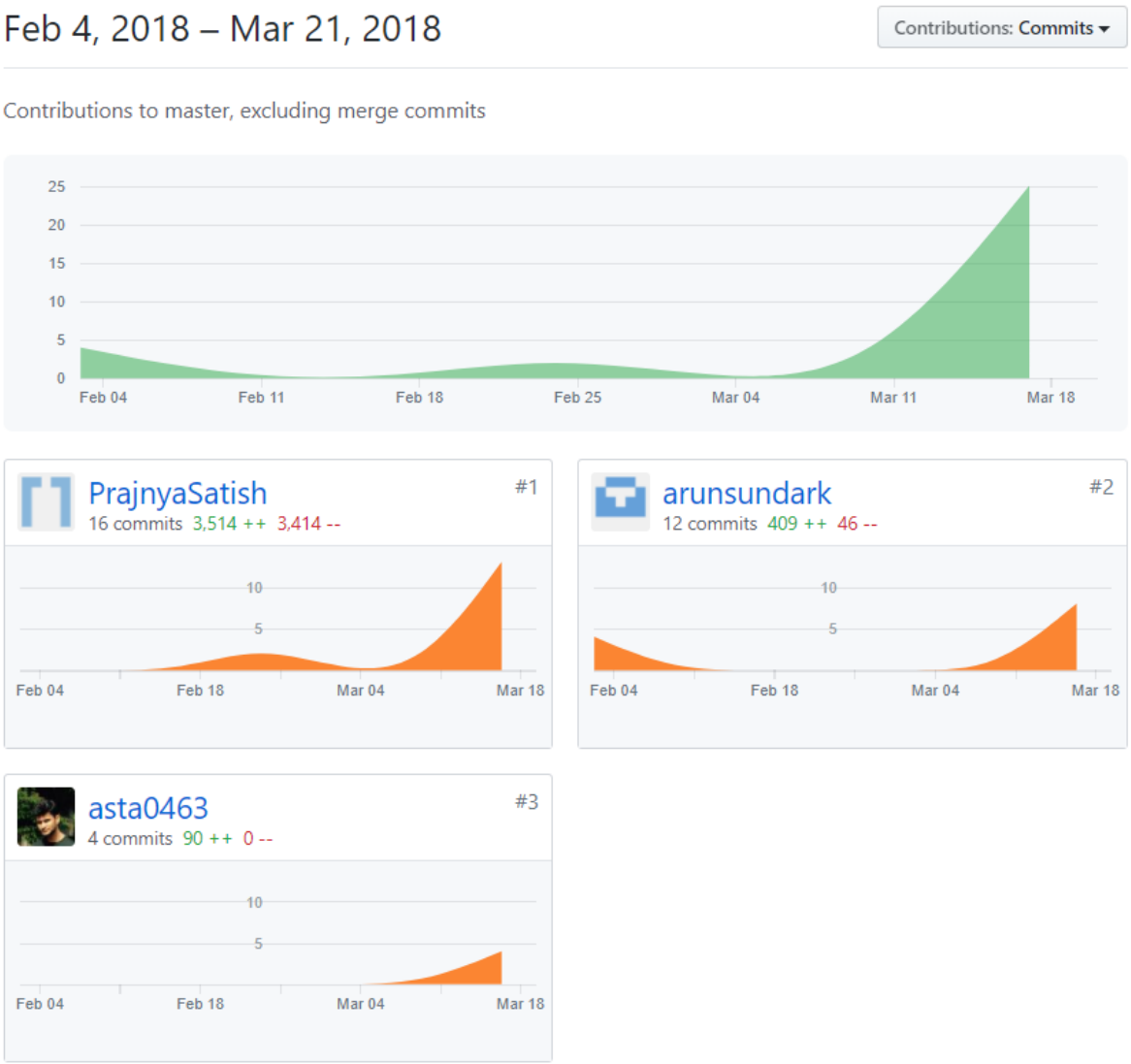
1.Implementation of the abstract FlightBasicDetails Class
2.Implementation of TravelFilters Class
3.Implementation of FlightBooking Class
4.Testing of individual modules of the flight classes

Work done by Ashish Tak:

1.Implementation of HotelBasicDetails Class
2.Implementation of HotelBooking Class
3.Implementation of Testing of individual modules of the Hotel classes
4.Corrections for the previous class diagram

Work done by Prajnya Sathish:
1.Implementation of FlightVendor Class
2.Implementation of HotelVendor Class
3.Testing of individual modules of the flight classes

Github Commit Graph:



Estimation of remaining effort: As per the class diagram submitted for Part2, the major remaining task is the combination of the Ticket objects for the Flight and the Hotel. The total cost obtained from these 2 types of objects would be passed to the payment gateway. Another important use of the consolidated ticket object (an object of the PassengerDetails class) is that it would be the one that's referred to whenever the customer goes for cancellation of any of the services.

As progress is being made in the implementation of the system, it has been observed that it would be better to make the Flight and Hotel Vendor classes as abstract and implement a Strategy design pattern to have a concrete class for each vendor, so that the algorithm for calculation of the cost and the number of vacancies can be customized as per the vendor.

Next Iteration: The tasks planned for the next iteration can be broadly divided into the below 3:

- Completing the functionalities for the remainder of the classes as per the submitted class diagram
- Incorporating the Spring Framework into the system
- Unit Testing of the functionalities and parallel inspection of the architecture to determine which design patterns would be the best fit