Ticket service app Assumptions:

------------------------

1. Depending on the requirement, we can increase/decrease the number of seats to hold per

Customer. In this application, I used 10 seats.

1. Depending on the requirement, we can increase/decrease the Seat Hold expiration time.

In this Application, I used one minute.

1. Seats can be reserved in only one level at a time. In a single transaction, we can’t reserve

Multiple seats in differenet levels.

1. Seats can be reserved only if they are on Hold. We need Seat Hold Confirmation Id to

Reserve the seat.

1. Once the Seat is reserved, that particular seat will be removed from the available seats
2. Customer Email is validated by Regular Expression
3. Unit Test Cases have been written for the three methods of the Ticket Service Interface.

Assumption

----------------

1. As per the Ticket Service Homework, disk storage, REST API and front-end GUI are not

strictly required. So I used the core Java to develop the ticket service app.

1. The ticket service app can be done more efficiently with the help of spring, REST and

Hibernate.

3 . If we implement the APP by using above technologies, the following steps are my design assumptions.

1. Expose each of these services as a REST API

1.1 We can use Jersey JAX-RS for developing RESTful Web services clients

1.2 Create and Configuring a Client Instance

1.3 Create an instance of the Web resource.

1.4 Send Requests to the Resource.

1.5 Receive responses from the resource. The response can be xml/json.

2. Spring container can be used to instantiate the beans. Dependency Injection can be implemented by Constructor or Setter injections.

3. In the Data access layer we integrate Spring with Hibernate/iBatis

4. Spring JDBCTemplate or Hibernate Template can be used to perform CRUD operations on the Database

5. We can store and retrieve the VenueLevel information from the Database.

6. SeatHod confirmation code and Reservation confirmation code can be saved to the Database.