Advanced Java Assignment 2021

Name	ADITEYA BARAL	**SRN**	PES1201800366	**Application assigned**	IRCTC

Sl. No	**Functionality**	**Java features used**
1	Live and real-time update of transit timings for trains	1. **JDBC**: A relation in the database will need to be populated with the transit times of each train, including other details such as source and destination, train information and so on. This relation will be updated periodically to store updated information for each train. **Classes**: Connection, DriverManager, PreparedStatement, Statement, CallableStatement, ResultSet **Methods**: executeQuery() 2. **Multithreading**: To fetch updated results periodically, a separate thread is spawned which obtains updated transit timings for each train. This activity is performed in the background and in a separate thread toMultiple users can attempt not freeze the user's activity in the main thread. **Classes**: Thread, Runnable

		Methods: start(), join(), isAlive(), sleep() 3. **JSP**: JSP is used to dynamically update the contents of the webpage to list all the updated transit timings periodically after they have been retrieved from a database. This allows the webpage to automatically refresh on its own to show new/updated results. <% ResultSet result = statement.executeQuery('SELECT * FROM TRANSIT'); while(result.next()) { %> <%= result.next() %> <% } %> **Methods**: jspInit(), jspDestroy()
2	Alerts or Announcement Board	1. **JDBC**: A relation in the database will be used to store all incoming as well as past alerts and announcements so that they can be retrieved from any component of the application. The relation will store details like the DateTime of the announcement, the alert message, the validity of the alert, category and so on. **Classes**: Connection, DriverManager, PreparedStatement, Statement, CallableStatement, ResultSet **Methods**: executeQuery()

		2. **Multithreading**: Alerts and announcements will be updated periodically in a separate thread in the background for the same to avoid blocking the user's activity thread.
		
		Classes: Thread, Runnable
		
		Methods: start(), join(), isAlive(), sleep()
		
		3. **JSP**: JSP is used to dynamically load the updated webpage contents after new alerts and announcements have been retrieved and updated and to display them on the webpage for users to view and read.
		
		<%
		ResultSet alerts=
		statement.executeQuery('SELECT * FROM ALERTS ORDER BY DATE DESCENDING');
		while(result.next())
		{ %>
		  <%= result.next() %>
		<% } %>
		
		Methods: jspInit(), jspDestroy()

3	Validation and verification of input credentials while signing in or registering	1. **JDBC**: When a user creates an account, their details are encrypted and stored in the Users relation in the database. Information such as name, contact number, email, password, address and so on are stored to create a profile for the user. The application uses these stored details the next time the same user tries to log in.

		Classes: Connection, DriverManager, PreparedStatement, Statement, CallableStatement, ResultSet **Methods**: executeQuery() 2. **Multithreading**: When a user clicks on sign-in/register and enters their details, a new thread may be spawned to either enter these details into the database in the Users relation (registration) or verify these details with ones that are already stored in the database (signing in). The user's activity thread can only proceed after this thread returns the status of the transaction (registration/sign-in was successful or unsuccessful) **Classes**: Thread, Runnable **Methods**: start(), join(), isAlive(), sleep() 3. **JSP**: JSP is used to dynamically update the webpage after entry validation is performed and display appropriate alerts for invalid entries made by the user. These occur when an entry field is left NULL or an email address field does not contain a valid email address, or the entered credentials are invalid. In such cases, the webpage will automatically display a small alert next to the field prompting the user to correct the entry before invoking the thread to verify credentials.

		<%
		if (!checkCredentialValid(username))
		{
		%>
		 Invalid credentials
		<% } %>
		%>
		
		Methods: jspInit(), jspDestroy()

4	Booking seats and transactions	1. **JDBC**: Relations in the database will store information about seats available in each train, the price of each seat, location, whether the seat is available or booked, category of the seat etc. When a user finally picks their chosen seats and confirms the booking, these changes need to be propagated to this relation, along with updates in other necessary relations such as the user's current order, booking history and global booking details. If a transaction fails, the entire process needs to be undone and all changes made need to be rolled back. **Classes**: Connection, DriverManager, PreparedStatement, Statement, CallableStatement, ResultSet, Savepoint **Methods**: executeQuery(), commit, setAutoCommit(), rollback() 2. **Multithreading**: Since multiple users can book seats at the same time, there will be multiple user activity threads. It is possible for multiple users to attempt to book the same seat. In such a case, appropriate locking mechanisms are

		implemented to ensure that when a user first clicks on an available seat, the seat gets blocked by that thread and appears booked to all other threads. If the user does not proceed with the transaction or the transaction fails, the thread releases the lock on the seat and it becomes available to the other threads. Threads are also used to update the status of available seats in real-time in the background so that when a user tries to make a booking they get to view only the available seats. **Classes**: Thread, Runnable, Lock, ReentrantLock **Methods**: start(), join(), isAlive(), sleep(), lock() 3. **JSP**: To display only the most recently updated available seats to a user, the webpage must be dynamically updated every time the user visits the booking section and attempts to choose a seat. JSP updates the content in real-time and hence allows the application to render an updated seat matrix to every user. It is also used to display helpful alerts if a user's entry is invalid (for example, if the user wishes to book 3 seats but has chosen only 2) or if the user's search request is invalid or if the user is yet to log in/register. <% ResultSet available_seats= statement.executeQuery('SELECT * FROM SEATS WHERE

		TRAIN_ID=1234 AND DATE=2021-11-25'); while(result.next()) { %> <%= result.next() %> <% } %> **Methods**: jspInit(), jspDestroy()