Data Access Object Design Pattern

(Interview Questions 2)

What is Data Access Object in Java? And where you have used it in your automation Framework?

- In Java, a DAO (Data Access Object) is a design pattern that provides ***an abstract interface for accessing data from a database.***
- It separates the logic for ***retrieving and storing data from the business logic of the application.***
- This separation of concerns **helps to improve the maintainability and flexibility of the application.**

Usage:

- A DAO typically consists of a ***set of methods that correspond to common database operations, such as reading, inserting, updating, and deleting records***
- These methods typically take arguments that correspond to the data being manipulated, and they return results that indicate the success or failure of the operation.
- In your Test Automation Framework you are interested only in the Reading Operation.

```java
public interface UserDao {
  User getUserById(int id);
  List<User> getAllUsers();
  void updateUser(User user);
  void deleteUser(int id);
}
```

```java
public class JdbcUserDao implements UserDao {
  private Connection connection;

  public JdbcUserDao(Connection connection) {
    this.connection = connection;
  }

  public User getUserById(int id) {
    // code to retrieve user from database using JDBC
  }

  public List<User> getAllUsers() {
    // code to retrieve all users from database using JDBC
  }

  public void updateUser(User user) {
    // code to update user in database using JDBC
  }

  public void deleteUser(int id) {
    // code to delete user from database using JDBC
  }
}
```

Drawback Of DAO:

- There are a few potential drawbacks to using a DAO design pattern in Java.
  - ***Increased complexity***:
    - Adding a DAO layer to an application can increase the overall complexity of the codebase, as it adds another layer of abstraction.
    - This can make it more difficult to understand and debug the application.

  - ***Performance overhead:*** Using a DAO can add a small amount of performance overhead to an application, as it requires additional method calls and object creation

  - ***Limited flexibility:***
    - A DAO may provide a fixed set of methods that correspond to common database operations, but it may not be able to handle more complex or specialized queries.

  - ***Tight coupling with the database:***
    - A DAO is tightly coupled with the database implementation, which can make it difficult to switch between different database technologies or configurations.
    - This can limit the flexibility of the application and increase the cost of maintenance.

Where to Use DAO in your Test Automation Framework?

DAO is component will only be used for DB validation.

Example:

- Where the user records have been updated properly by UI or API automation
- Verify if the Dropdowns are populated correctly on the UI