**If the project is based on Spring 4.3.\* and Hibernate 5.2, is it possible to mock DB in JUnit test? if not, how to do the JUnit test? Give cases to show how to do it.**

* We can mock the DB in junit using mockito framework or DBUnit.
* Consider example, we have a service tier which has the business rules and make as a interface for the front end and entity tier which has logic responsible for making calls to database.
* In this case, service is dependent on entity. If we have a junit that makes a connection with the database and if the database is down, our tests will fail.
* For this, we need to remove the dependency and allow junit to run independently even if the database is down.

**Solution 1: Use DBUnit**

* With DBUnit we can use embedded database such a HSQLDB to make our database exclusive to the running of out tests

**Solution 2: Use Mockito(Using mock annotation or call mock() method from mockito class)**

-We mock the calls to database and assume that it will return a value. We do this for both negative and positive scenarios. This way we eliminate the actual database call and remove dependency

Eg: using mock() from mockito class

**//Dao class to mock**

**BookDao bookDao = mock(BookDao.class)**

**@Test**

**Public void testGetAvalailableBooks() {**

**When(bookDao.getAvailableBooks().thenReturn(40));**

**}**

**Above is the simple example to mock the database call to get all available books from the database**