1. **Using examples, distinguish between integration tests and unit tests**

* Unit testing means testing the smaller units of your application, like classes and methods.

e.g

MyUnit Class

**public class** MyUnit {  
  
 **public** String concatenate(String unit, String test){  
 **return** unit + test;  
 }  
}

MyUnitTest Class

**import** org.junit.Test;  
**import static** org.junit.Assert.\*;  
  
**public class** MyUnitTest {  
  
 @Test  
 **public void** testConcatenate() {  
 MyUnit myUnit = **new** MyUnit();  
  
 String result = myUnit.concatenate(**"unit"**, **"test"**);  
  
 *assertEquals*(**"unitest"**, result);  
  
 }  
}

* Integration Testing a testing where software modules are integrated logically and tested as a group.

e.g

An Employee Management System has 3 modules say **'Login Page', ‘Employee’** and **'Delete Employee’** and each of them is integrated logically.

| **Test Case ID** | **Test Case Objective** | **Test Case Description** | **Expected Result** |
| --- | --- | --- | --- |
| **1** | Check the interface link between the Login and Employee module | Enter login credentials and click on the Login button | To be directed to the Employee |
| **2** | Check the interface link between the **Employee** and **Delete** **Employee** Module | From **Employee** select the specific *Employee* and click a delete button | Selected *Employee* should disappear from **Employee** module |

1. **What are checked and unchecked exceptions? When do you use them? Explain those using examples in code**

**Checked exceptions**are the exceptions for which the programmer **has** to **specify the condition/situation**for which that exception will be raised. The **exceptional situation**is defined by the programmer as per his/her need.

For example, I need to raise an exception if a user enters a negative number. In this case, **the exception condition i.e. a negative number as input is specified by me.**The code for the same will be:

*//checked exception***public void** check (**int** number) **throws** IOException {  
 **try** {  
 **if** (number < 0) {  
 **throw new** IOException(**"value cannot be negative"**);  
 }  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
}

**Unchecked exceptions**are those exceptions that are automatically generated by the runtime environment. The programmer doesn’t have to specify the condition for checking the situation that might cause an exception.

*//unchecked exception example code***public static void** main(String args[]) {  
 String exceptionArr[] = { **"Exception"**, **"checked"**, **"unchecked"**, **"throws"**, **"notfound"** };  
 **try** {  
 String myString = exceptionArr[7]; *// This throws ArrayIndexOutOfBoundException* System.***out***.println(myString);  
 } **catch** (ArrayIndexOutOfBoundsException e) {  
 System.***out***.println(**"ArrayIndexOutOfBoundException appears here. Better handle it in code above :)"** + e);  
 }  
}

1. **Highlight how database replication works and how that is different from using sync module. Does that provide flexibilities over direct server access at the central location? Would you suggest a better solution or tool than what is mentioned?**

We have a **master** and **slave**.

The master is where all the changes happens. The slave receives a copy of the changes applied at the master server. This happens so fast in order that the slave is always in sync with the master.

**Database Replication** implies that there are two or more copies of (all) the **data.**

Using **Sync module** implies that two or more copies of **data** are being kept *up-to-date*, but not necessarily that each copy contains all of the **data** (although this is typically the case for database syncing).

Yes this provide flexibilities as opposed to central server access.

1. **Demonstrate Model View Controller(MVC) architecture in JavaScript. Use AngularJs framework or reactJS to explain those**.

Am not experience in AngularJs or reactJS but I can easily learn them

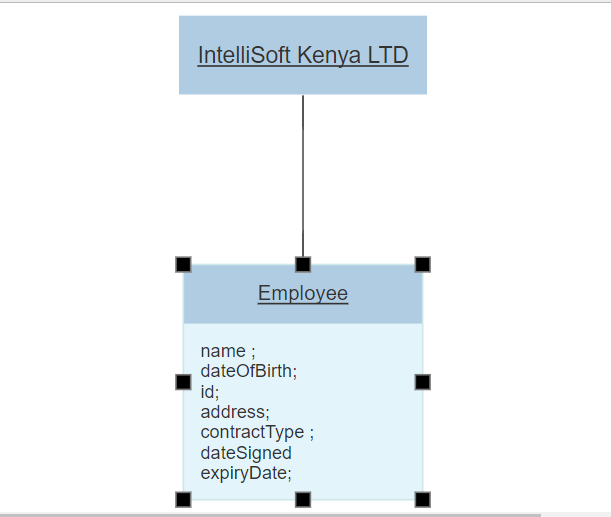
1. **Highlight the benefits of using spring and what subprojects do you know?(2 marks)**

Below are the benefits of Spring:

* **Lightweight:**Spring Framework is lightweight with respect to size and transparency.
* **Inversion Of Control (IOC):**In Spring Framework, loose coupling is achieved using Inversion of Control. The objects give their own dependencies instead of creating or looking for dependent objects.
* **Aspect Oriented Programming (AOP):**By separating application business logic from system services, Spring Framework supports Aspect Oriented Programming and enables cohesive development.
* **Container:**Spring Framework creates and manages the life cycle and configuration of application objects.
* **MVC Framework:**Spring Framework is a MVC web application framework. This framework is configurable via interfaces and accommodates multiple view technologies.
* **Transaction Management:**For transaction management,Spring framework provides a generic abstraction layer. It is not tied to J2EE environments and it can be used in container-less environments

Spring subprojects include :Spring Data JPA, Spring Data Rest, Spring Data JDBC Spring Cloud CLI, Spring Cloud CLI

1. **IntelliSoft Kenya Limited urgently needs a system to manage their employees. The system should collect names, date of birth, unique identifier, address, contract information(when signed, expiry date and type). You are required to model this system using UML, OOP principles, flow diagrams. Provide a practical solution through java code, demonstrate industrial coding accepted styles**



1. **What is Dependency Injection? How can we inject beans in Spring? Explain clearly using code examples**

**Dependency** **injection** **is** basically providing the objects that an object needs (its dependencies) instead of having it construct them by itself.

We can inject beans in spring either using **annotations** or **xml**. I’ll use annotations method in my code

1. **Project X has many methods(around 1000) that commit and write information to the database. All methods have been prefixed with the word save. The requirement is before any of these methods are called. A check must be performed to see if the parameters passed are valid. By use of an aspect, demonstrate using code how that can be realized with fewer lines of code and in the most efficient way(10 marks).**
2. **What are the important benefits of using the Hibernate Framework? Using examples, show how Hibernate and Java persistence API can be used together with spring framework. Show how ORM maps to class objects that are directly mapped to database tables. Show how maven is used for dependency injection**

**Productivity:** It helps developers get rid of writing complex and tedious SQL statement, no more need of JDBC APIs for result set or data handling. It makes developers more concentrate on the business logic and increase the project’s productivity.

**Maintainability:** It helps reduce the lines of code, makes system more understandable and emphasizes more on business logic rather than persistence work (SQLs). More important, a system with less code is more easier to refactor.

**Portability:** It abstracts our application away from the underlying SQL database and sql dialect. Switching to other SQL database requires few changes in Hibernate configuration file (Write once / run-anywhere).