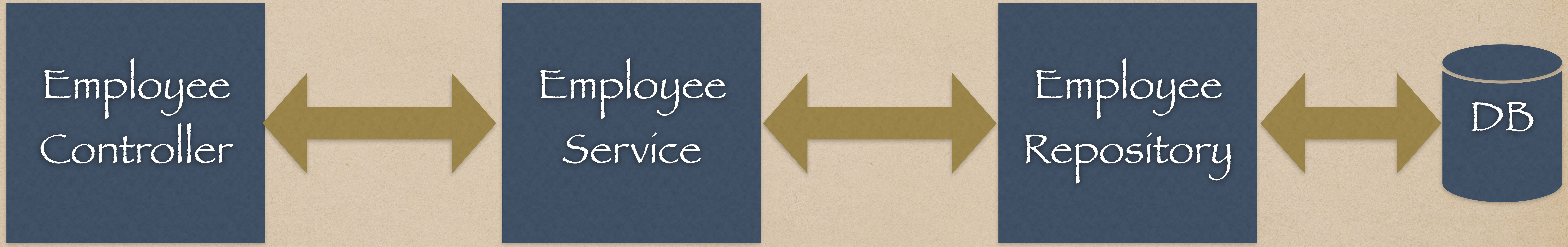
Spring Boot Application Integration Testing

By Ramesh Fadatare (Java Guides)

Integration Testing

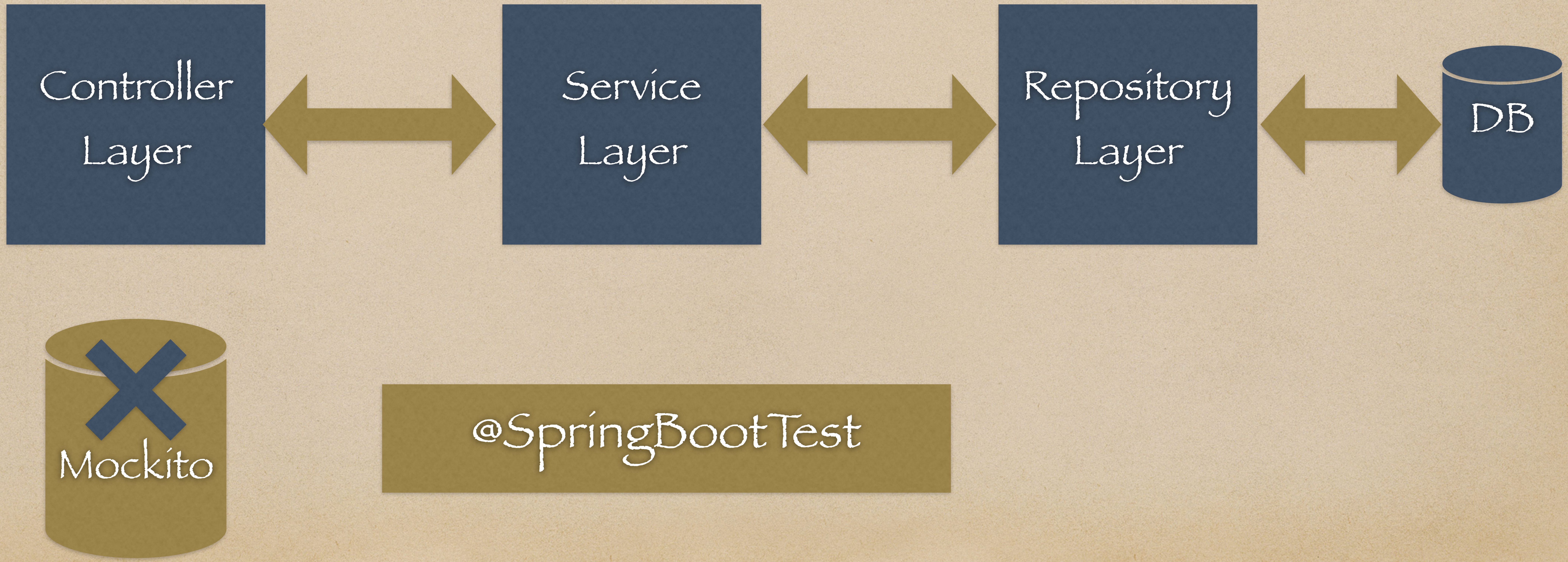
As the name suggests, integration tests focus on integrating different layers of the application. That also means no mocking is involved.

Basically, we write integration tests for testing a feature which may involve interaction with multiple components.

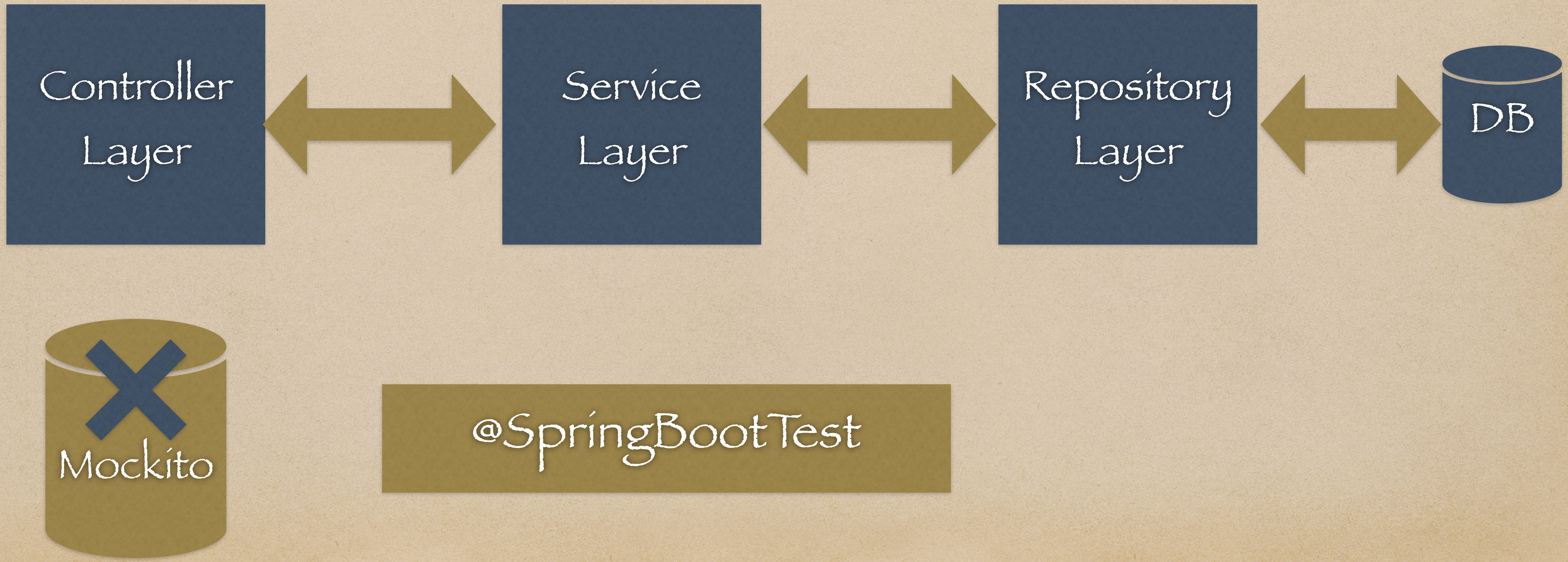


Examples:
Employee Management Feature (EmployeeRepository, EmployeeService, EmployeeController).
User Management Feature (UserController, UserService, and UserRepository).
Login Feature (LoginRespository, LoginController, Login Service) etc

Spring Boot Application Integration Testing



Spring Boot Application Integration Testing



Spring Boot Integration Testing

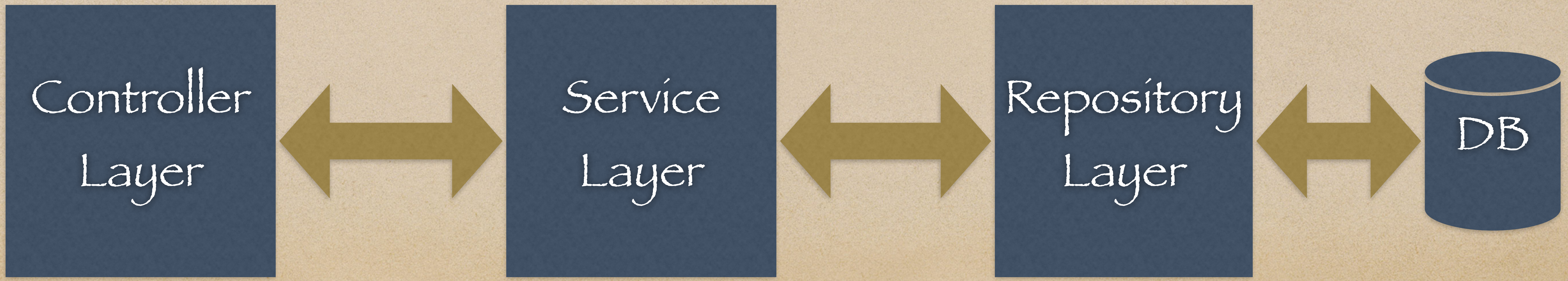
1. **@SpringBootTest annotation overview**
2. **Integration test save employee feature**
3. **Integration test get all employees feature**
4. **Integration test get employee by id feature**
5. **Integration test update employee feature**
6. **Integration test delete employee feature**

@SpringBootTest

Spring Boot provides **@SpringBootTest** annotation for Integration testing. This annotation creates an application context and loads full application context.

@SpringBootTest will bootstrap the full application context, which means we can **@Autowire** any bean that's picked up by component scanning into our test.

Integration testing - @SpringBootTest



@SpringBootTest

It starts the embedded server, creates a web environment and then enables **@Test** methods to do integration testing.

By default, **@SpringBootTest** does not start a server. We need to add attribute webEnvironment to further refine how your tests run. It has several options:

★ **MOCK(Default):** Loads a web ApplicationContext and provides a mock web environment

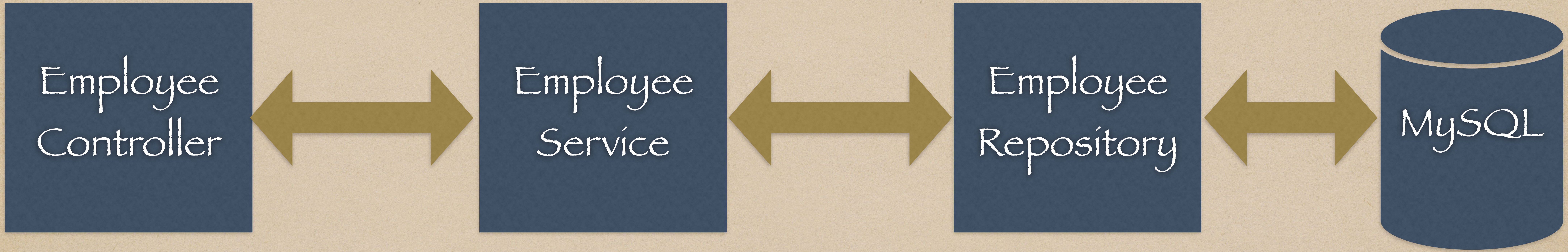
★ **RANDOM_PORT:** Loads a WebServerApplicationContext and provides a real web environment. The embedded server is started and listen on a random port. This is the one should be used for the integration test

★ **DEFINED_PORT:** Loads a WebServerApplicationContext and provides a real web environment.

★ **NONE:** Loads an ApplicationContext by using SpringApplication but does not provide any web environment

Steps for Integration Testing

1. We will add MySQL driver dependency to our Spring boot application. Of course you can use h2 in-memory database for integration testing



2. Add MySQL configuration in application.properties file

3.Write Integration tests for EmployeeController