[redacted]

SpringBoot MCQs

Question - 1
Mocking Beans

Given the following implementation of *DataService* which uses *ApiClient* component to call external APIs:

```
@Component
public class ApiClient {
  public void call() {
   //not relevant code
  }
}

@Service
public class DataService {
  @Autowired
  private ApiClient apiClient;
  public void collectData() {
    apiClient.call();
  }
}
```

The requirement is to implement the test and don't call external API during the test execution. The following code has been implemented:

```
@SpringBootTest
class DataServiceTest {
  <CODE HERE>
  @Autowired
  DataService dataService;
  @Test
  void collectData() {
    dataService.collectData();
    //test code
  }
}
```

What should be inserted instead of *<CODE HERE>* to achieve the goal?

- @Mock ApiClient apiClient; @InjectMocks
- @InjectMocks
- @MockBean ApiClient apiClient;
- @Captor ApiClient apiClient;
- @SpyBean ApiClient apiClient;

Question - 2
Secure Method

Given the following controller method, assume that *usersService* is correctly implemented and autowired.

```
<CODE_HERE>
@GetMapping("/users/{id}")
public ResponseEntity<UserResponse> get(@PathVariable @NotNull UUID id) {
  return ResponseEntity.ok(usersService.get(id));
}
```

The requirement is to make sure that only users with any of *ROLE_ADMIN* or *ROLE_USER_MANAGER* roles assigned will be able to execute this method. What can be put in place of '<CODE_HERE>' to implement this requirement using Spring Security?

- ◯ @PostAuthorize("hasRole('ROLE_ADMIN') or hasRole('ROLE_USER_MANAGER')")
- ◯ @PreAuthorize("hasAnyRole('ROLE_ADMIN','ROLE_USER_MANAGER')")
- ◯ @Secured("hasRole('ROLE_ADMIN') or hasRole('ROLE_USER_MANAGER')")
- ◯ @PreAuthorize("hasRole('ROLE_ADMIN') and hasRole('ROLE_USER_MANAGER')")

Question - 3
Transactional Tests

Given the following test code, which of the statements is true?

```
@ExtendWith(SpringExtension.class)
@Transactional
@ContextConfiguration
class UserRepositoryTest {
  @Test
  @Commit
  void test1() {
    /* non relevant code */
  }
  @Test
  @Rollback(false)
  void test2() {
    /* non relevant code */
  }
}
```

- ◯ Only the transaction for method test1() will be committed.
- ◯ Only the transaction for method test2() will be committed.
- ◯ Transactions for both methods will be committed.
- ◯ @Transactional will lead to a runtime error when running the tests.

Question - 4
Postgres Specific Service

The requirement is to implement a Spring Boot *@Service* that should be loaded to the spring context only if the *org.postgresql.Driver* class is present on the classpath, and the *application.properties* file contains the property *database.vendor=postgres*.

```
<CODE HERE>
@Service
public class PostgresSpecificService {
    /* not relevant code*/
  }
```

Which of the following annotation options can replace *<CODE HERE>* to achieve this?

- @ConditionalOnProperty(name = "database.vendor", value = "postgres") @ConditionalOnClass(name = "org.postgresql.Driver")
- @ConditionalOnProperty(prefix = "database", name = "vendor", havingValue = "postgres") @ConditionalOnClass(name = "org.postgresql.Driver")
- @ConditionalOnProperty(name = "database.vendor", havingValue = "postgres") @ConditionalOnMissingBean(org.postgresql.Driver.class)
- @ConditionalOnProperty(prefix = "database", name = "vendor", havingValue = "postgres") @ConditionalOnBean(name = "org.postgresql.Driver")
- None of the above

Question - 5
Bean Scopes

Consider the following code.

```
<CODE HERE>
public class Processor {
  public void process() {
    /* not relevant code */
  }
}

@RestController
public class ProcessController {
  @Autowired
  private Processor processor;
  @GetMapping("/process")
  public void process() {
    processor.process();
  }
}
```

What should be inserted in place of *<CODE HERE>* to have a new instance of processor created every time the */process* endpoint is called?

- @Scope("prototype") @Component
- @Prototype
- @Component @Scope(scopeName = "prototype", proxyMode= ScopedProxyMode.TARGET_CLASS)
- @Component(`autowireCandidate=true`)
- @Service(alwaysNew=true)

Question - 6
Spring Circular Bean Trap

Consider the following code.

Circular.java

```
package spring.circular;

public interface Circular {
    void doCircularThings();
}
```

CircularBeanA.java

```
package spring.circular;
import org.springframework.stereotype.Component;
import javax.annotation.PostConstruct;

//X
@Component
public class CircularBeanA implements Circular {
    private Circular circularBeanB;

    public CircularBeanA(
        //Y
        Circular circularBeanB) {
        this.circularBeanB = circularBeanB;
    }

    @Override
    public void doCircularThings() {
        System.out.println("CircularBeanA: did bad things");
    }

    @PostConstruct
    private void init() {
        System.out.println("CircularBeanA:  initialized");
    }
}
```

CircularBeanB.java

```
package spring.circular;

import org.springframework.context.annotation.Lazy;
import org.springframework.stereotype.Component;
import javax.annotation.PostConstruct;

@Component
public class CircularBeanB implements Circular {
    private Circular circularBeanA;

    public CircularBeanB(
        @Lazy
        //Z
        Circular circularBeanA) {
        this.circularBeanA = circularBeanA;
    }

    @Override
    public void doCircularThings() {

    }

    @PostConstruct
    public void init() {
        System.out.println("CircularBeanB: initialized");
        circularBeanA.doCircularThings();
    }
}
```

Which of the following options are true regarding this code?

- ◯ The application runs successfully and it'll output
  CircularBeanA: initialized
  CircularBeanB: initialized
  CircularBeanA: did bad things
- ◯ The application won't run.
  The code doesn't compile.
- ◯ The application runs but it will throw `NoUniqueBeanDefinitionException` and it will exit.
- ◯ If @Qualifier("circularBeanA") annotation is put on Z and @Qualifier("circularBeanB") put on Y, the application will not throw BeanCurrentlyInCreationException and runs successfully.
- ◯ The application runs and it will write
  CircularBeanB: initialized into console then
  it will throw org.springframework.beans.factory.BeanCurrentlyInCreationException.
- ◯ If @PostConstruct annotation is removed in CircularBeanB, it will run successfully and will print
  CircularBeanA: initialized into console
- ◯ if @Primary annotation is put in X place, it will run successfully.

Question - 7
Spring AOP Usage

Consider the following code.

NotifierMetricLogger.java

```
package spring.listener;

import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;
import org.aspectj.lang.annotation.Before;
import org.aspectj.lang.annotation.AdviceName;
import org.springframework.stereotype.Component;
import org.aspectj.lang.JoinPoint;
import org.aspectj.lang.ProceedingJoinPoint;
import java.util.logging.Logger;

//X
@Component
public class NotifierMetricLogger {
    private static final Logger log = Logger.getLogger(NotifierAspect.class.getName());
    //Y
    public Object beforeNotifyLogging(ProceedingJoinPoint joinPoint) throws Throwable {
        long startDate = System.currentTimeMillis();
        Object proceed = joinPoint.proceed();
        long executionTime = System.currentTimeMillis() - startDate;
        log.info("Notify process time :" + executionTime);
        return proceed;
    }
}
```

TwitterNotifier.java

```
package spring.service.impl;
```

```
import spring.service.Notifier;
import org.springframework.stereotype.Component;
import java.util.logging.Logger;

@Component
public class TwitterNotifier implements Notifier {
    private static final Logger log = Logger.getLogger(TwitterNotifier.class.getName());
    @Override
    public void notify(String message) {
        log.info("TwitterNotifier: " + message);
        //send notification to home page
    }
}
```

Notifier.java

```
package spring.service;

public interface Notifier {
    void notify(String message);
}
```

Assuming the Spring Boot application is configured to use AOP with @EnableAspectJAutoProxy(proxyTargetClass = true) annotation, to capture *TwitterNotifier.notify(String message)* method's process time, which of the following options should be placed in the *X* and *Y* positions in NotifierMetricLogger.java?

- X = @Aspect
  Y = @Before("execution(* spring.service.impl.*.notify(..))")
- X = @AdviceName("NotifierMetricLogger")
  Y = @Around("execution(* spring.service.impl.*.notify(..))")
- X = @AdviceName("NotifierMetricLogger")
  Y = @Before("execution(* spring.service.impl.*.notify(..))")
- X = @Aspect
  Y = @Around("execution(* spring.service.Notifier.notify(..))")

Question - 8
Bean definition enhancement

During the startup of a Spring Boot application, it needs to read bean configuration metadata and change it before the container instantiates any beans.

How can this be achieved in an efficient and scalable way?

- Implement BeanPostProcessor.
- Implement BeanFactoryPostProcessor.
- It is not possible to change beans metadata on runtime. All beans metadata is defined at compile time.
- Implement Aspect.

Question - 9
Behavior Inheritance

A Spring Boot application has the following hierarchy of classes.

```
public class Animal {
  @PostConstruct
  private void init() {
    System.out.println("Animal  init");
  }
}

@Component
public class Cat extends Animal{
  @PostConstruct
  public void init() {
    System.out.println("Cat init");
  }
}

@Lazy
@Component
public class Dog extends Animal{
  @PostConstruct
  public void init() {
    System.out.println("Dog init");
  }
}
```

What is the output?

- ◯ IllegalBeanDefinitionException: @PostConstruct should be applied to a public method
- ◯ Cat init Dog init
- ◯ Animal init Cat init
- ◯ Animal init Cat init Animal init Dog init
- ◯ Animal init Cat init Dog init or Animal init Dog init Cat init

Question - 10
Multiple Beans Definition

A Spring application has an interface called *Server*, and two implementations: *ServerA* and *ServerB*. There is a class, *ServerManager*, that uses the *Server* bean as a dependency.

```
public interface Server {
}

@Service
public class ServerA implements Server {
}

@Service
public class ServerB extends ServerA {
}


@Service
public class ServerManager {
  @Autowired
```

```
    Server server;
  }
```

Which of the following statements is true about this code?

- The code throws `InterfaceNotInstantiatableException`.
- The code runs fine. A random Server implementation is injected into the `server` field.
- The code throws `NoUniqueBeanDefinitionException`.
- The code does not compile.

Question - 11
Testing a Spring Application

There is a Spring boot web application that uses a relational database for data storage. The *org.springframework.boot:spring-boot-starter-data-jdbc* starter is used in the implementation of the data access layer. Now tests are needed that cover the functionality of the data layer in isolation.

What Spring test annotation is recommended when constructing the test context?

- @DataJpaTest
- @DataJdbcTest
- @SpringBootTest
- @WebMvcTest

Question - 12
Path Variable

In the Spring MVC controller, which of these are valid uses of the @PathVariable annotation?

- @RequestMapping(value="/users/{userId}/addresses/{addressId}") public String viewUserAddress(@PathVariable String userId, @PathVariable String addressId, Model m)
- @RequestMapping(value="/users/{userId}") public String viewUser(@PathVariable("users") String user, Model m)
- @RequestMapping(value="/users/{userId}") public String viewUser(@PathVariable String userId, Model m)
- @RequestMapping(value="/users/{userId}") public String viewUser(@PathVariable("userId") String personnelId, Model m)

Question - 13
Component Dependency

The following @Configuration contains definitions for 2 beans: ServiceA and ServiceB. Imports are omitted.

```
@Configuration
public class ServiceConfiguration {
  @Bean
  public ServiceA serviceA(ServiceB serviceB) {
    return new ServiceA(serviceB);
  }
  @Bean
  public ServiceB serviceB(ServiceC serviceC) {
    return new ServiceB(serviceC);
  }
  @Bean
  public ServiceC serviceC(ServiceA serviceA) {
    return new ServiceC(serviceA);
  }
}
```

Which of the following statements is true?

- This code does not compile.
- This code runs fine and creates 3 Beans: `ServiceA`, `ServiceB` and `ServiceC`.
- The code throws `BeanCurrentlyInCreationException` when run.
- The code throws `StackoverflowError` when run.

Question - 14
Application Properties Override

In the classpath of a Spring Boot application, there are 2 files with properties *application.properties* and *application-prod.properties*. It is required to always load properties from *application.properties* and override with values in the *application-prod.properties* file only when the application is deployed on the production server.

What should the value of the environment be to achieve this on the production server?

- spring.properties=application-prod.properties
- spring.profiles.active=application-prod
- spring.profiles.active=prod
- environment=prod

Question - 15
Spring Dependency Injection

```
@Service
public ProductService {
    private ProductRepository productRepository;
    private ProductMapper productMapper;

    @Autowired
    public ProductService(ProductRepository productRepository,
                          ProductMapper productMapper){
        this.productRepository = productRepository;
        this.productMapper = productMapper;
    }

}
```

Which type of injection is implemented in the ProductService?

- Setter
- Getter
- Property
- Construction

the processing pipeline easier than with a centralized system.