**Day 6: Introduction of Aspect Oriented Programming**

1. What is AOP? What is the value of it?

AOP is Aspect Oriented Programming.

Aspect-oriented is used to identifies code scattering and tangling to crosscutting concerns.

It is the separation of concerns, Increased Modularity, reduces code, removes dependencies.

**AOP is a programming paradigm that aims to increase modularity by allowing the separation of cross-cutting concerns.** It does so by adding additional behavior to existing code without modification of the code itself.

Instead, we can declare this new code and these new behaviors separately.

Spring’s [AOP framework](https://docs.spring.io/spring/docs/current/spring-framework-reference/core.html#aop) helps us implement these cross-cutting concerns.

* Single Responsibility Principle
* Scattering & Tangling

2. Explain what a pointcut is. What a join point is.   
How are they related?

Pointcut is an expression that defines a set of Join points

Join point is where advice code in applied in application and it always class method in spring AOP

Join points is the methods in Spring application that specified by a Pointcut in order to criteria to match Join points to advice.

A Pointcut is a predicate that helps match an Advice to be applied by an Aspect at a particular JoinPoint.

Joinpoint define pointcut

Day7:

hasRole, hasAuhority, hasPermission

**Day 8: Batch Processing and Validation**

|  |  |
| --- | --- |
| Question 1 of 3  What is the basic function of Batch processing? Explain it based on the 3 components of a batch step.   * ETL, Extra, Transform and Load  |  | | --- | |  | |
| Question 2 of 3  What is JSR 303. Give an example  -> Validation framework, Annotation for validation   |  | | --- | |  | |
| Question 3 of 3  What are JSR 303 Groups?  -> |

**Day 9: Spring Integration: Remoting Technologies**

**Q.1 What are the two services offered by JMS?**

- Two services offered by JMS is Point-to-Point (PTP) and Publish-Subscribe systems.  
- PTP built around the concept of a message queue and each message has only one consumer. It is multiple producers  
- Publish-Subscribe systems use a “topic” to send and receive messages and each message has multiple consumers. It is single producers

**Q.2 Why is AMQP valuable? What weakness of JMS does it overcome?**

AMQP valuable is queues and exchanges  
- Queues are only consumed by a single receiver and AMQP doesn’t publish directly to queues. The message is published to an exchange routed to one queue or multiple queues  
The weakness of JMS doesn't have queues and topics and routing key.

**Q.3 What is an exchange?**

- Exchanges are the message routing agents; accept messages from producer routes to queues. It has a topic and direct exchange.  
- Direct exchange is Queue binding requires a direct match based on a  
simple Routing Key. It's beyond basic JMS Pub/Sub  
- The topic exchange is Queue binding requires a direct match based on a  
complex routing Key. It is beyond basic JMS Pub/Sub

**Day 10:**

Q.1 What is the purpose of EAI?

EA is Enterprise Application Integration. It is used for the integration of service and data.

Enterprise architecting the integration solutions is a complex task.

* The whole to manage date, transform, storage in business role
* To avoid the massive many department that need to work together

Q.2 Explain the 3 main components of Spring integration and how they interact.

|  |
| --- |
|  |

|  |
| --- |
|  |

The 3 main components of Spring integration is a message, message channel, message endpoint

- Message a generic wrapper for any Java object combined with metadata used by the framework while handling that object. It consists of a payload and header(s).

- A Message channel is a component through which messages are moved so it can be thought of as a pipe between message producer and consumer. [ PTP or Pub/Sub].

- A message endpoint isolates application code from the infrastructure. In other words, it is an abstraction layer between the application code and the messaging framework. They are message producers & consumers

Q.3 Spring Integration supports Transport Conversion.   
Explain how it works in terms of a external server that produces JMS and another external server that consumes AMQP.

In term of an external server that produces JMS and another external server that consumers like HQ to warehouse Order process use case.

we use a Listen for RabbitMQ Order message for RabbitAdapter

and we use a transform Order Message to RouteOrder Message and then we used Broadcast RouteOrder Message to Warehouse for JMS Adapter in order to solve the HQ to warehouse order process problem that HQ receives an order from the customer request via phone and Headquarters generates a Customer Order and then they broadcast the Order to the Warehouse through RabbitMQ. The Warehouse “listens” for orders on a JMS Queue. The Warehouse accept & process an Order as a RouteOrder

Delivery food item, many transports that can use like airplane, train, like JMS and AMQP is like transport mechanism. Central place

**Day 10: Spring MVC**

Q.1 Explain the MVC Pattern.

MVC pattern designs that use Model View Controller. A controller can talk to Model and View. Controller talks to Model in order to retrieve data and provide data back to the controller and then controller transfer those data to View in order to render information to the client.

Q.2 Explain data binding in Spring MVC.

Data binding in Spring MVC is automatically mapped request parameters domain objects

- It simplifies the code by removing repetitive tasks

- It is the built-in Data Binding handles simple String to data type

conversions

It is to handle the complex relationship and we use @RequestMapping for data binding.

**Day 11: REST web service**

Q.1 What are the REST architectural constraints?

REpresentational State Transfer.

- In client-server, we have to a separation of concerns. The separates clients from servers.

- REST is the Stateless because the client–server communication is further constrained by no client context being stored on the server between requests.

- Uniform interface: In REST, the individual resources are identified in requests. We can identify the resource for URL. Unique identify.

* Stateless: more scalable , the server and session don’t know and care about information from user.

Q.2 What does idempotent mean? Give REST examples.

- Idempotent means that multiple calls with the same operation don’t change the representation.

- clients can make that same call repeatedly while producing the same result. In other words, making multiple identical requests has the same effect as making a single request. Note that while idempotent operations produce the same result on the server (no side effects), the response itself may not be the same (e.g. a resource's state may change between requests).

The PUT and DELETE methods are defined to be idempotent. However, there is a caveat on DELETE. The problem with DELETE, which if successful would normally return a 200 (OK) or 204 (No Content), will often return a 404 (Not Found) on subsequent calls, unless the service is configured to "mark" resources for deletion without actually deleting them. However, when the service actually deletes the resource, the next call will not find the resource to delete it and return a 404. However, the state on the server is the same after each DELETE call, but the response is different.

GET, HEAD, OPTIONS and TRACE methods are defined as safe, meaning they are only intended for retrieving data. This makes them idempotent as well since multiple, identical requests will behave the same.

Q.3 Consider the following command sent to a REST server:

DELETE https://mum.edu/members/2/addresses/1 HTTP/1.1  
  
What would be the Rest server implementation?

DELETE /members/2/addresses/1 - Deletes address #5 for member #2

**Exam 2:**

1. T/F item AOP

6-7 AOP definition,

8. AOP Terminology Relationship

10. @Aspect J

11. Spring AOP – PROXY

12. Proxy Pattern

14. Advice Type

15. Point cut Designators

17. More on args () pointcut

Implicit and Explicit

19. Pointcut & Advice Method Signatures

2. Describe/explain REST related scenarios

1. Command like last quiz (first two) what response from server.

2.

3. Annotation a spring MVC with RESTful controller, Bidirectional, conduct model and server. Issue and problem to take domain model for RESTful

3. Conceptual

4. Implementation Validation, Annotate domain model and controller. How validation in MVC part.

5. Conceptual question related to Validation group.

Use case, how they use, why they use it, example,

Lesson 8:

2. Enterprises,

3 Modern batch Processing

5. Spring Scheduling Annotation

7, 8 ETL: how it works

11. Chunk Based Processing

12. Error Handling, Skip Retry Restart

Lesson 8: - Validation

2. Data Validation is fundamental

3. Approaches

4. JSR 303, 349

5. Spring core Technology

6. Validation Property Annotation

7. Domain Object annotations. want to see good syntax

8. Error message externalized in. properties file

10. Data Validation Application. Conceptual idea

12. Constraint Composition

13. JSR 303 groups

15.BPM

17. How grown

19.

Lesson 11: Spring MVC

Controller allow

12. Data Binding

14. Data Binding – Relationship

16. Spring MVC Annotations

18. Mapping Example

19. Mapping Example. Do Validation

Lesson 12: Web Service

GET/POST/ DELETE

5. RESTful API HTTP method

8. Spring REST

9.

10. 11 web service controllers

12. Spring MVC

19. what it means and describe

24. JSON lazy Loading. Whole issue bidirectional and understand of concept

29.

28. 26 to