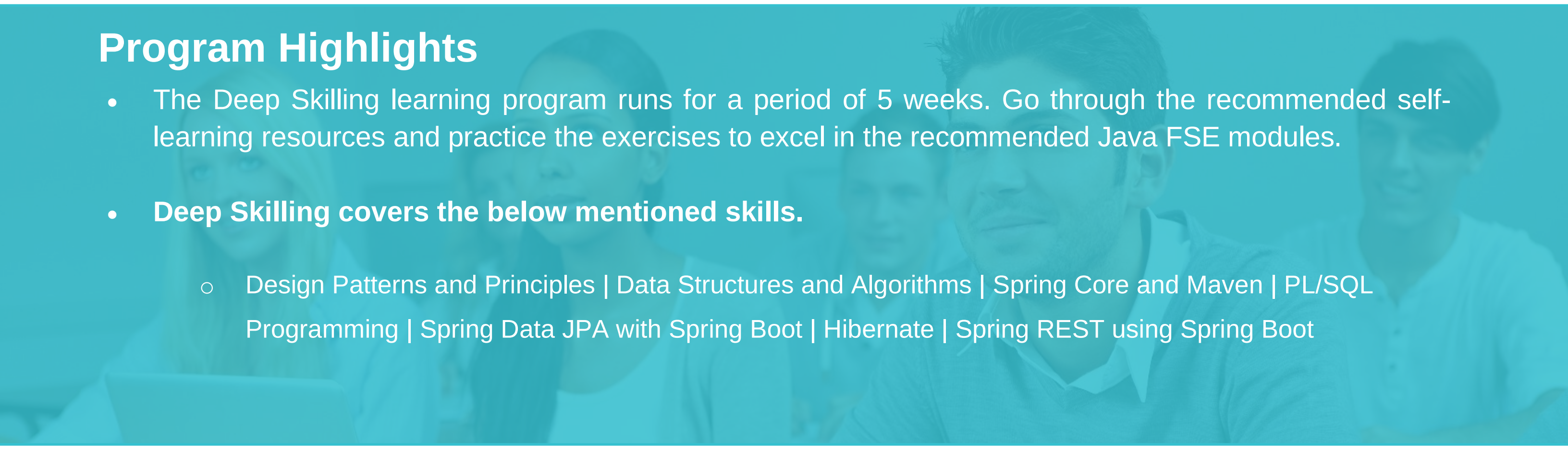
Digital Nurture 3.0

Deep Skilling Handbook - Java FSE

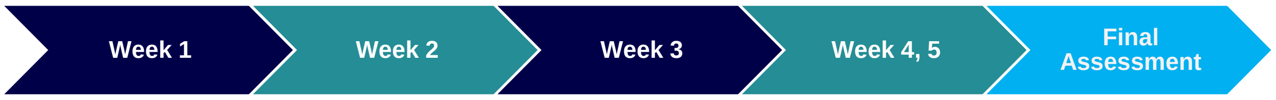


Program Highlights

- The Deep Skilling learning program runs for a period of 5 weeks. Go through the recommended self-learning resources and practice the exercises to excel in the recommended Java FSE modules.
- **Deep Skilling covers the below mentioned skills.**
    - Design Patterns and Principles | Data Structures and Algorithms | Spring Core and Maven | PL/SQL Programming | Spring Data JPA with Spring Boot | Hibernate | Spring REST using Spring Boot

Recommended Program Sequence

The learning journey contains below modules, followed by a Deep Skilling Final Assessment.



Week 1	Week 2	Week 3	Week 4, 5	Final Assessment
• Design Patterns and Principles • Data Structures and Algorithms • Hands-On Exercises	• Spring Core and Maven • PL/SQL Programming • Hands-On Exercises	• Spring Data JPA with Spring Boot, Hibernate • Hands-On Exercises	• Spring REST using Spring Boot 3 • Hands-On Exercises	• Deep Skilling Final Assessment

Program Completion Criteria

To successfully complete this learning program, candidates must meet the following criteria:

Weekly Hands-on Exercises:

- Candidates must complete the hands-on exercises assigned for each week. These exercises are designed to reinforce the learning objectives and ensure practical understanding of the material.

Final KBA Assessment:

- Candidates must pass the final Knowledge-Based Assessment (KBA). This assessment will evaluate their comprehensive understanding and application of the concepts covered throughout the program.

Learning Approach

DN 3.0 Deep Skilling Program adopts a comprehensive and blended learning approach to ensure an engaging and effective educational experience.

The program comprises two essential learning components:

- **Self-paced learning** through open-source learning reference links.
- **Weekly SME connect** sessions conducted by experts.

Self-Paced Learning using Open-source Reference Links	➢ Please refer to the learning reference links provided to learn and understand the recommended concepts. ➢ **We expect you to dedicate 8-10 hours of focused attention weekly** to your learning modules.
SME Connect Session	➢ We have scheduled sessions with Subject Matter Experts (SMEs) that are designed to deepen your understanding of complex topics. ➢ **100% attendance is mandatory** for SME Connect sessions. ➢ Engage actively in doubt clarification sessions, asking questions and seeking clarification on challenging topics. Benefit from the experts' experience and insights to gain a deeper understanding of the subject matter.

Disclaimer: *Cognizant does not claim ownership or responsibility for the content or any issues with the links provided, as they are merely references available on the internet. Candidates are free to leverage additional sources beyond what has been provided to enhance their skill capabilities for Deep Skilling*.

Effective Learning Strategies

Create a Study Schedule	Set Clear Goals	Stay Organized	Active Learning
▪ Dedicate specific times each week for learning. ▪ Aim for 8-10 hours of study per week.	▪ Define what you want to achieve each week. ▪ Break down tasks into manageable chunks.	▪ Keep track of your progress and deadlines. ▪ Use tools like calendars, to-do lists, and reminders.	▪ Engage with the material through hands-on exercises. ▪ Practice coding and solve problems regularly.

Duration Recommendation

Weekly Learning: 8-10 hours	Total Program Duration: 5 weeks
▪ Allocate 1-2 hours daily, or schedule longer sessions on weekends. ▪ Balance your learning with regular academic commitments.	▪ Follow the weekly recommendations consistently. ▪ Ensure to complete all exercises and review sessions.

Where to Practice?

Online Code Editors	Installed Software Tools
▪ Use platforms like Repl.it, OnlineGDB, and JSFiddle for practice. ▪ Ideal for quick testing and sharing code snippets.	▪ Set up Integrated Development Environments (IDEs) like VS Code, IntelliJ IDEA, Spring Tool Suite, or Eclipse. ▪ Install necessary tools and libraries as per course requirements.

Exercise Instructions

In this learning program, you will be required to complete weekly exercises designed to reinforce the concepts learned during the week. These exercises are hosted on a public GitHub repository and must be downloaded and solved on a weekly basis. Follow the instructions below to ensure a smooth and productive exercise workflow:

1. **Accessing the Exercises:**
   - Each week, exercises will be made available in our public GitHub repository.
   - The repository URL is *https://github.com/trinity2040/Digital-Nurture-3.0*
   - Navigate to the repository and locate the folder named **Java FSE** Inside it, you will find a set of exercises for each week.

2. **Downloading the Exercises:**
   - Download the files for the week's exercises by clicking the download button on the GitHub repository page.
3. **Solving the Exercises:**
   - Solve the problem statements provided in the downloaded files.

  - Ensure that you understand the problem requirements and apply the concepts learned during the week.
  - Take your time to think through the solutions and code them accurately.

4. **Self-Evaluation:**
   - After completing the exercises, evaluate your solutions based on the problem criteria.
   - Compare your approach with any provided hints or solutions if available.
   - Reflect on any mistakes or areas where you can improve.

5. **Submitting Solutions:**
   - Firstly, organize your solutions week-wise and keep them in a folder.
   - Create a **public repository** in your personal GitHub account, upload your solution folder, and share the URL with the POC on demand.

6. **Additional Support:**
   - If you encounter difficulties or have questions about the exercises, seek help from peers.
   - Utilize the resources and links provided in this handbook for further assistance.

Module 1 - Design Patterns and Principles Week #: 1

Overview:

This module introduces learners to essential design principles and patterns that are crucial for creating robust and maintainable software. Learners will delve into the SOLID principles, which include SRP, OCP, LSP, ISP, and DIP. They will also explore common design patterns, such as Creational, Structural, and Behavioral patterns, which provide reusable solutions to common design challenges. The module combines theoretical insights with practical exercises, helping learners understand and apply these concepts in their projects, ultimately enhancing their ability to write clean, efficient, and scalable code.

Learning Objectives:

After completing this module, users will be able to:

- Grasp the importance of the Single Responsibility Principle (SRP).
- Implement the Open/Closed Principle (OCP) to create extendable software.
- Ensure software components adhere to the Liskov Substitution Principle (LSP).
- Design interfaces following the Interface Segregation Principle (ISP).
- Apply the Dependency Inversion Principle (DIP) for flexible dependency management.
- Recognize and apply Creational Patterns.
- Utilize Structural Patterns to build robust systems.
- Employ Behavioral Patterns for effective object interaction.
- Improve code reusability and reduce duplication through design patterns.
- Solve common software design problems using appropriate design patterns.
- Evaluate and choose the right design pattern for a given problem context.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
SOLID Principles	What & Why, **The SOLID Principles of Object-Oriented Programming** - The Single Responsibility Principle, The Open-Closed Principle, The Liskov Substitution Principle, The Interface Segregation Principle, The Dependency Inversion Principle	https://www.baeldung.com/solid-principles
Commonly used Design Patterns	GoF, **Creational Patterns** - Singleton Pattern, Factory Method Pattern, Builder Pattern; **Structural Patterns** - Adapter Pattern, Decorator Pattern, Proxy Pattern; **Behavioral Patterns** - Observer Pattern, Strategy Pattern, Command Pattern; **Architectural Patterns** - Model-View-Controller (MVC), Dependency Injection	https://medium.com/@softwaretechsolution/design-pattern-81ef65829de2

Check Your Understanding:

- Design Patterns Quiz

Hands-On:

- Complete the respective Week 1 hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 2 - Data Structures and Algorithms   Week #: 1

Overview:

This module focuses on core data structures and algorithms, foundational for writing efficient and scalable code. Learners will explore key data structures such as arrays, linked lists, understanding their implementation and application scenarios. Additionally, they will delve into fundamental algorithms including searching (linear search, binary search) and sorting (bubble sort, quick sort, merge sort). The module emphasizes practical implementation through coding exercises, enabling learners to apply these concepts effectively in real-world programming challenges.

Learning Objectives:

After completing this module, learners will be able to:

- Choose appropriate data structures based on problem requirements.
- Apply searching algorithms to find elements in data structures.
- Utilize sorting algorithms to order data efficiently.
- Solve coding problems using appropriate data structures and algorithms.
- Analyze algorithmic complexities to optimize performance.
- Design efficient and scalable solutions for software development tasks using learned concepts.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Analysis of Algorithms	Introduction, Why DS& Algorithm, Types of DS, Notations, Time and Space Complexity, Start using frameworks for describing and analyzing algorithms, Begin using asymptotic notation to express running-time analysis, Asymptotic notations for run-time analysis of algorithms, Best Case, Average Case, Worst case analysis of an algorithm, Finding Time Complexity of few iterative and recursive algorithms	https://www.geeksforgeeks.org/design-and-analysis-of-algorithms/
Sorting	Bubble, Insertion, Heap Sort, Quick Sort, Merge Sort - Worst, Average and Best-Case analysis	https://www.geeksforgeeks.org/sorting-algorithms/
Arrays	Array Traversal - Array representation in Memory, Measuring Time complexity, Searching, Traversal in Arrays, When to use Arrays	https://www.geeksforgeeks.org/array-data-structure-guide/
Linked List	Single Linked List, Circular Single Linked List, Double Linked List, Circular Double Linked List - Search, Inert, Traverse, Delete operations, Time complexity	https://www.geeksforgeeks.org/linked-list-in-java/
Searching	Linear Search, Binary Search	https://www.geeksforgeeks.org/searching-algorithms/#basics-of-searching-algorithms

Check Your Understanding:

- Data Structures and Algorithms Quiz

Hands-On:

- Complete the respective Week 1 hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 3 - Spring Core and Maven Week #: 2

Overview:

This module covers essential topics in Spring Core and Maven for Java application development. It starts with an introduction to the Spring Framework, emphasizing its benefits in Java programming. Learners will set up projects using Maven for efficient dependency management and build automation. Key concepts include the Spring IoC container for managing application components, configuring Spring beans with XML and annotations, and implementing dependency injection for flexible and maintainable code. Additionally, the module covers Aspect-Oriented Programming (AOP) in Spring, Spring MVC for web development, Object-Relational Mapping (ORM) for database interactions, and introduces Spring Boot for rapid application deployment. Through practical exercises, learners will gain hands-on experience to apply these skills effectively in real-world Java projects.

Learning Objectives:

After completing this module, learners will be able to:

- Understand the core principles and advantages of using Spring in Java applications.
- Explain IoC and DI concepts and their benefits in loosely coupled systems.
- Identify and describe key modules of the Spring Framework and their purposes.
- List and discuss the advantages Spring offers over traditional Java development.
- Explain the role of Maven in Java projects and its benefits in dependency management.
- Create a new Maven project and understand its directory structure.
- Configure Maven to include necessary Spring dependencies for a project.
- Customize Maven settings.xml and pom.xml files for efficient project builds and dependency management.
- Differentiate between BeanFactory and ApplicationContext in managing Spring beans.
- Define beans and their dependencies using XML-based configuration in Spring.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Spring Framework	Overview of the Spring Framework, Inversion of Control (IoC) and Dependency Injection (DI), Spring modules: Core, AOP,	https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html

	Data Access, ORM, MVC, etc., Benefits of using Spring in Java applications	
Setting up a Spring Project with Maven	Introduction to Maven build tool, Creating a new Maven project, Adding Spring dependencies in the pom.xml file, Configuring Maven for building and managing dependencies	https://www.studytonight.com/spring-framework/spring-maven-project
Spring IoC Container	Understanding the IoC container, Configuring the Spring IoC container using XML, Defining beans and their dependencies, ApplicationContext and BeanFactory	https://www.baeldung.com/inversion-control-and-dependency-injection-in-spring
Spring Bean Configuration	Using annotations for bean configuration, Component scanning and Stereotype annotations, Java-based configuration with @Configuration, Mixing XML and Java-based configurations	https://www.baeldung.com/spring-bean-annotations
Dependency Injection in Spring	Constructor injection, Setter injection, Autowiring dependencies, Qualifiers for resolving autowiring conflicts, Using @Resource and @Inject annotations	https://docs.spring.io/spring-framework/reference/core/beans/dependencies/factory-collaborators.html
Spring AOP (Aspect-Oriented Programming)	Introduction to AOP concepts, Creating aspects and advice, Pointcuts and Joinpoints, AOP proxying mechanisms, Integrating AOP with Spring applications	https://www.geeksforgeeks.org/aspect-oriented-programming-and-aop-in-spring-framework/
Spring MVC and ORM	Overview of MVC and ORM, Configuration, Controller Layer, Model Layer, View Layer, Form Handling, Querying, Validation, Exception Handling	https://www.geeksforgeeks.org/spring-mvc-framework/
Spring Boot (Introduction)	Overview of Spring Boot, Simplifying Spring configuration with Boot, Creating a Spring Boot application, Auto-configuration and convention over configuration	https://www.geeksforgeeks.org/spring-boot/

Check Your Understanding:

- Maven Quiz
- Spring Quiz

Hands-On:

- Complete the respective Week 2 hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 4 - PL/SQL Programming — Week #: 2

Overview:

This module introduces learners to the fundamental concepts of PL/SQL, a procedural extension of SQL used for developing efficient database applications. Learners will explore PL/SQL's syntax, structure, and essential constructs such as variables, control structures, exception handling, cursors, stored procedures, functions, packages, and triggers. By understanding these concepts, learners will gain proficiency in leveraging PL/SQL to enhance database management, automate tasks, and enforce business rules effectively.

Learning Objectives:

After completing this module, learners will be able to:

- Develop efficient database applications using PL/SQL constructs.
- Implement error handling strategies and optimize data retrieval using cursors.
- Design and manage stored procedures, functions, packages, and triggers for enhanced database functionality.
- Apply PL/SQL effectively to automate tasks and enforce business rules within database environments.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to PL/SQL	What is PL/SQL?, Importance of PL/SQL in Database Management, Differences between SQL and PL/SQL	https://www.geeksforgeeks.org/plsql-introduction/
PL/SQL Environment	Overview of PL/SQL Environment, Understanding PL/SQL Block Structure, Anonymous Blocks vs. Named Blocks	https://www.educba.com/pl-sql-block-structure/
Basic PL/SQL Syntax	Declaring Variables, Data Types in PL/SQL, Assigning Values to Variables	https://www.tutorialspoint.com/plsql/plsql_basic_syntax.htm
Control Structures	Conditional Statements (IF-THEN-ELSE), CASE Statements, Loops (FOR, WHILE, and LOOP)	https://docs.oracle.com/cd/A97630_01/appdev.920/a96624/04_struc.htm

Error Handling	Understanding Exceptions, Predefined Exceptions, User-Defined Exceptions, Exception Handling in PL/SQL	https://docs.oracle.com/cd/B13789_01/appdev.101/b10807/07_errs.htm
Cursors	Introduction to Cursors, Implicit vs. Explicit Cursors, Cursor Operations (OPEN, FETCH, CLOSE)	https://www.javatpoint.com/pl-sql-cursor
Procedures and Functions	Creating Stored Procedures, Creating Functions, Parameters (IN, OUT, IN OUT), Differences between Procedures and Functions	https://docs.oracle.com/en/database/other-databases/timesten/22.1/plsql-developer/pl-sql-procedures-and-functions.html
Packages	Introduction to Packages, Creating Package Specifications, Creating Package Bodies, Benefits of Using Packages	https://docs.oracle.com/en/database/oracle/oracle-database/23/lnpls/plsql-packages.html
Triggers	Introduction to Triggers, Types of Triggers (BEFORE, AFTER, INSTEAD OF), Creating and Managing Triggers	https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/plsql-triggers.html

Check Your Understanding:

- PL/SQL Quiz

Hands-On:

- Complete the respective Week 2 hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 5 - Spring Data JPA with Spring Boot, Hibernate   Week #: 3

Overview:

This module focuses on integrating Spring Data JPA with Spring Boot, leveraging Hibernate for optimized database interactions. Learners will gain practical skills in project setup, entity mapping, repository management, query optimization, CRUD operations, pagination, sorting, entity auditing, data projections, and customization of data sources. The module equips learners to develop efficient and scalable applications using widely adopted frameworks and tools.

Learning Objectives:

After completing this module, learners will be able to:

- Understand the integration of Spring Data JPA with Spring Boot for effective database management.
- Develop skills in setting up projects and configuring database connectivity using Spring Data JPA.

- Gain proficiency in defining and mapping JPA entities within Spring applications.
- Implement repositories and utilize query methods for data access and manipulation.
- Perform CRUD operations and optimize queries for efficient data retrieval.
- Utilize pagination and sorting techniques to manage large datasets effectively.
- Implement entity auditing to track changes in database records.
- Create data projections to retrieve specific data subsets efficiently.
- Customize data source configurations and manage multiple databases in Spring Boot.
- Explore Hibernate-specific features and optimizations for database interactions.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Spring Data JPA	Overview of Spring Data JPA, Relationship between JPA and Spring Data JPA, Advantages of using Spring Data JPA in Spring Boot	https://www.baeldung.com/the-persistence-layer-with-spring-data-jpa
Setting Up a Spring Boot Project with Spring Data JPA	Creating a Spring Boot project, Adding Spring Data JPA dependencies, Configuring the application properties for database connection	https://www.javaguides.net/2021/12/how-to-use-spring-data-jpa-in-spring-boot-project.html
Entity Mapping	Introduction to JPA entities, Mapping entities to database tables, Defining primary keys and relationships, Using annotations like @Entity, @Table, @Id, @GeneratedValue, etc.	https://salithachathuranga94.medium.com/object-relational-mapping-with-spring-boot-jpa-and-hibernate-18cdfc51b4f0
Spring Data Repositories	Overview of Spring Data repositories, Creating repositories for entities, Derived queries from method names, Using Query DSL with @Query annotation, Custom query methods	https://docs.spring.io/spring-data/data-commons/docs/1.6.1.RELEASE/reference/html/repositories.html
CRUD Operations with Spring Data JPA	Implementing basic CRUD operations, Using JpaRepository methods for common operations, Executing custom queries with the repository	https://medium.com/javarevisited/spring-boot-jpa-crud-with-example-bbd219b5d4a6
Query Methods and Named Queries	Defining query methods in repositories, Using keywords in query methods, Named queries with @NamedQuery and @NamedQueries, Executing dynamic queries	https://docs.spring.io/spring-data/jpa/reference/jpa/query-methods.html

Pagination and Sorting	Implementing pagination with Page and Pageable, Sorting query results, Combining pagination and sorting	https://www.baeldung.com/spring-data-jpa-pagination-sorting
Auditing with Spring Data JPA	Enabling entity auditing, Using @CreatedBy, @LastModifiedBy, @CreatedDate, and @LastModifiedDate, Configuring auditing properties	https://docs.spring.io/spring-data/jpa/reference/auditing.html
Spring Data JPA Projections	Creating projections for specific data subsets, Interface-based and class-based projections, Using @Value and constructor expressions, Controlling the fetched data with projections	https://docs.spring.io/spring-data/jpa/reference/repositories/projections.html
Spring Data JPA and Spring Boot Integration	Leveraging Spring Boot auto-configuration, Customizing data source configuration, Externalizing configuration with application properties, Managing multiple data sources	https://www.geeksforgeeks.org/spring-boot-spring-data-jpa/
Spring Data JPA and Hibernate	Introduction to Spring Data JPA and Hibernate, **Hibernate-specific Features** - Leveraging Hibernate-specific annotations, Configuring Hibernate dialect and properties, Batch processing with Hibernate;	https://medium.com/@burakkocakeu/jpa-hibernate-and-spring-data-jpa-efa71feb82ac

Check Your Understanding:

- Spring Data JPA Quiz
- Hibernate Quiz

Hands-On:

- Complete the respective Week 3 hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Overview:

This module focuses on building RESTful services with Spring Boot, covering essential concepts from REST architecture to advanced features like security, testing, and API documentation. Learners will gain hands-on experience in setting up projects, creating REST controllers, handling HTTP methods, managing data transfer objects (DTOs), implementing CRUD operations, supporting content negotiation, integrating Spring Boot Actuator for monitoring, securing endpoints with Spring Security, and testing REST services effectively.

Learning Objectives:

After completing this module, learners will be able to:

- Understand RESTful architecture and its application in modern web development.
- Set up Spring Boot projects for developing RESTful services efficiently.
- Implement request mappings and handle HTTP methods in REST controllers.
- Customize responses and manage exceptions effectively in REST APIs.
- Utilize Data Transfer Objects (DTOs) for data mapping and serialization.
- Implement CRUD operations and validate input data in REST endpoints.
- Integrate HATEOAS principles to enhance API navigability and usability.
- Configure content negotiation to support various media types in responses.
- Monitor and manage RESTful services using Spring Boot Actuator.
- Secure REST endpoints with Spring Security, including authentication and authorization mechanisms.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Spring REST and Spring Boot 3	Overview of RESTful architecture, Introduction to Spring REST, Benefits of using Spring Boot for RESTful services, Setting up a Spring Boot project for REST, What's New in Spring Boot 3?	https://www.javatpoint.com/restful-web-services-spring-boot https://www.baeldung.com/spring-boot-3-spring-6-new
Building a Simple REST Controller	Creating a basic REST controller, Defining request mappings, Handling HTTP methods (GET, POST, PUT, DELETE), Returning JSON responses	https://www.geeksforgeeks.org/easiest-way-to-create-rest-api-using-spring-boot/
Request and Response Handling	Handling path variables and query parameters, Request body and form data processing, Customizing response status and headers,	https://medium.com/@tericcabrel/validate-request-body-and-parameter-in-spring-boot-53ca77f97fe9

	Exception handling in REST controllers	https://belowthemalt.com/2023/01/12/how-do-you-add-custom-response-headers-in-a-spring-boot-application/
RESTful Resource Representation with DTOs	Introduction to Data Transfer Objects (DTOs), Mapping entities to DTOs, Customizing JSON serialization and deserialization, Managing versioning and backward compatibility	https://www.moesif.com/blog/technical/api-design/REST-API-Design-Best-Practices-for-Sub-and-Nested-Resources/
RESTful CRUD Operations	Implementing Create, Read, Update, and Delete operations, Utilizing HTTP methods for CRUD operations, Validating input data with annotations, Optimistic locking for concurrent updates	https://www.basedash.com/blog/what-are-crud-operations-in-a-rest-api
RESTful HATEOAS	Understanding HATEOAS (Hypermedia as the Engine of Application State), Adding links to resources, Building and consuming hypermedia-driven APIs	https://www.geeksforgeeks.org/hateoas-ahy-its-needed-in-restful-api/
Content Negotiation and Media Types	Configuring content negotiation, Supporting different media types (JSON, XML), Using the Accept header for content negotiation, Producing and consuming custom media types	https://maheshbonagiri.medium.com/spring-boot-and-content-negotiation-183b20eaa425
Spring Boot Actuator for REST Monitoring	Integrating Spring Boot Actuator, Monitoring and managing RESTful services, Exposing custom metrics, Securing and customizing Actuator endpoints	https://www.javatpoint.com/restful-web-services-spring-boot-actuator
Security and Authentication in RESTful APIs	Securing RESTful endpoints with Spring Security, Implementing authentication and authorization, Token-based authentication (JWT), Handling Cross-Origin Resource Sharing (CORS)	https://www.toptal.com/spring/spring-boot-oauth2-jwt-rest-protection
Testing RESTful APIs	Unit testing REST controllers with JUnit and Mockito, Integration testing for REST services, Using Spring Test and MockMvc, Test coverage and best practices	https://www.baeldung.com/integration-testing-a-rest-api

Documenting RESTful APIs	Introduction to API documentation tools (Swagger, Springdoc), Documenting REST APIs with Swagger/OpenAPI, Generating API documentation, Best practices for API documentation	https://www.geeksforgeeks.org/spring-boot-rest-api-documentation-using-swagger/

Check Your Understanding:

- REST Quiz

Hands-On:

- Complete the respective Week 4, 5 hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

What's Next?

Congratulations on successfully completing the 5-week DN 3.0 Deep Skilling learning program!

As you have now finished this important phase of your learning, you will be taking a **Knowledge-Based Assessment (KBA)** to certify your skills. **This assessment will cover all the skills and topics you have learned during the past five weeks**, ensuring you have a comprehensive understanding of the material.

We wish you the best of luck on your assessment and look forward to seeing you apply your newly acquired knowledge and skills. Good luck!