cognizant

Digital Nurture 4.0

Deep Skilling Handbook - Java FSE

Program Highlights

- The Deep Skilling learning program runs for a period of 8 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 | Microservices and Frameworks | Test driven design | Logging framework | Version control | DevOps concepts | Front-end JS Framework | Cloud fundamentals | GenAI fundamentals

Recommended Program Sequence

The learning journey contains the modules mentioned above, followed by a Deep Skilling Final Assessment. It is recommended to follow a sequence of skills learning, the details of which is given below.

Reason for the sequence recommendation:

Any engineering competency landscape can be classified into four major constructs

1. Engineering Concepts

The core concepts for any engineering landscape. This is the foundation on which all programming languages and frameworks are built.

Relevant skills:

- Design Patterns and Principles
- Data Structures and Algorithms

2. Programming Languages

The languages are used to implement the necessary logic for the requirements.

Relevant skills:

- Java - Covered in upskilling
- SQL – Basic (covered in upskilling) and Advanced (covered in deep skilling)
- TDD using JUnit5 and Mockito
- SLF4J logging framework

3. Products and Frameworks

There are numerous frameworks in the programming family to build applications. These enable the programmers to develop different types of applications by providing a gamut of features provided over a single integrated environment.

Relevant skills:

- Spring Core and Maven

- Spring Data JPA with Spring Boot, Hibernate

- Spring REST using Spring Boot 3
- Microservices with Spring Boot 3 and Spring Cloud
- React & Angular overview

4. Platforms

These are the enablers as a base to host the application developed using the Engineering concepts, programming languages, product and frameworks to make it accessible by all users, reliable and scalable.

Relevant skills:

- GIT
- CI/CD
- Containerization using Docker
- Cloud Fundamentals

Apart from all the mentioned skills, the very widely emerging concentration that any industry is looking into is the 'Generative Artificial Intelligence' or the 'GenAI'.

Let us get an introduction to it. You can go through it in the module 'Gen AI Fundamentals and Tools'.

Program Completion Criteria

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

Hands-on Exercises:

- Candidates must complete hands-on exercises mapped against every skill. These exercises are designed to reinforce learning objectives and ensure a 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 4.0 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 solving problems regularly.

Duration Recommendation

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

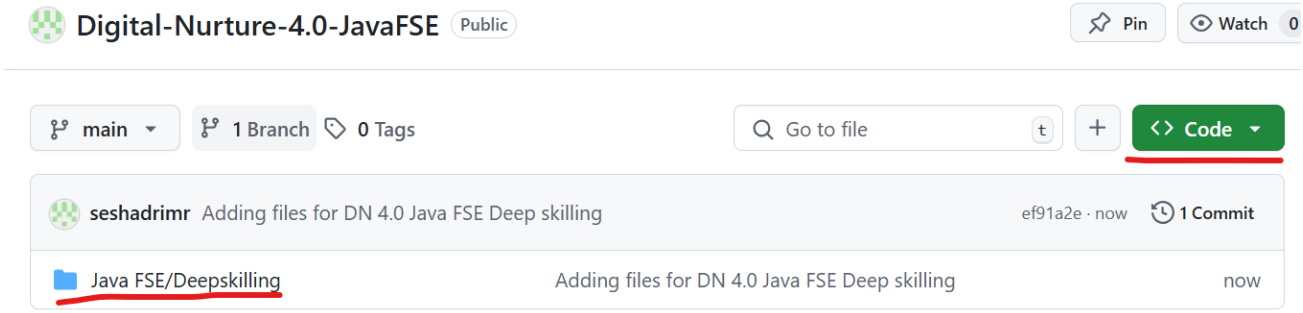
Where to Practice?

Online Code Editors	Installed Software Tools
• Use platforms like Repl.it, OnlineGDB, and Programiz 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 Visual studio code from Download Visual Studio Code - Mac, Linux, Windows • Install necessary tools and libraries as per course requirements.

Exercise Instructions

In this learning program, you will be required to complete 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 along with the learning. Follow the instructions below to ensure a smooth and productive exercise workflow:

1. **Accessing Exercises:**
   - Each week, exercises will be made available in our public GitHub repository.
   - The repository URL is this
2. **Downloading Exercises:**
   - You'll see the options 'Java FSE' as shown in the screenshot below. Click the 'Code' button as shown in the screenshot and choose the option 'Download zip'. This will download all the required Hands-on content for your practice.



3. **Solving the Exercises:**
   - You'll need to complete **Mandatory hands-on**, identified for every skill. The details of such hands-on is provided in the file 'DN 4.0 - Java FSE Mandatory hands-on detail.xlsx' in the same GitHub repository. The remaining ones are additional ones, which is up to you to attempt them.
   - 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 hints or solutions provided 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 your peers.
   - Utilize the resources and links provided in this handbook for further assistance.

Upskilling skills – A quick recap

Hi Champ – Welcome to the Deep skilling of Digital Nurture 4.0 program. Before you delve deep into the skills into this, with utmost sincerity, please do go through the upskilling handbook and refresh the basic programming skills to keep yourself abridged.

Always remember – The Object-Oriented Programming concepts are the core for any Programming language or frameworks.

Correlate the concepts you learn with real-life to understand and relate better for better understanding and life-long knowledge retention.

All the best!

FSE construct - Engineering concepts

Engineering Concepts

The core concepts for any engineering landscape. This is the foundation on which all the programming languages and frameworks are all built.

Relevant skills:

- Design Patterns and Principles
- Data Structures and Algorithms

Please plan to complete the skills in 1 week.

Module 1 - Design Patterns and Principles

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 skills' 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

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, and understand 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 skills' hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

FSE construct - Programming Languages

The languages are used to implement the necessary logic for the requirements.

Relevant skills:

- Java - Covered in upskilling
- SQL – Basic (covered in upskilling) and Advanced (covered in deep skilling)
- TDD using JUnit5 and Mockito
- SLF4J logging framework

Please plan to complete this in 1 week

Module 3 - PL/SQL Programming

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.tutorialspoint.com/plsql/plsql_cursors.htm
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 skills' hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 4 – Test driven development and Logging framework

Overview:

This module focuses on the concepts of Test-driven development and tools used for it. The features of tools like Junit and Mockito for unit testing, mock the dependencies to complete the unit testing. Basics of automation testing tools and logging framework.

Learning Objectives:

After completing this module, learners will be able to:

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to TDD	What is TDD?, Benefits of TDD, TDD vs. Traditional Testing	https://www.geeksforgeeks.org/test-driven-development-tdd/ https://developer.ibm.com/articles/5-steps-of-test-driven-development/
TDD Cycle	Red-Green-Refactor Cycle, Writing Failing Tests, Writing Minimal Code to Pass Tests, Refactoring Code	https://www.geeksforgeeks.org/test-driven-development-tdd/ https://medium.com/@rubenghosh968/test-driven-development-55af68347fdd
Introduction to Unit Testing	What is Unit Testing? Importance of Unit Testing, Unit Testing vs. Integration Testing	https://www.geeksforgeeks.org/unit-testing-software-testing/ https://www.geeksforgeeks.org/difference-between-unit-testing-and-integration-testing/
JUnit Framework	Setting Up Junit, Writing Basic JUnit Tests, Assertions in JUnit	https://www.geeksforgeeks.org/introduction-to-junit-5/
Test Structure	Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods	https://www.geeksforgeeks.org/unit-testing-best-practices/ https://semaphore.io/blog/aaa-pattern-test-automation https://www.geeksforgeeks.org/introduction-to-junit-5/
Advanced JUnit Features	Parameterized tests, Test suites and categories, Test execution order, Exception testing, Timeout and performance testing	https://www.geeksforgeeks.org/junit-5-how-to-write-parameterized-tests/ https://www.geeksforgeeks.org/junit-5-parameterizedtest/

		https://www.geeksforgeeks.org/junit-5-test-suites-with-example/ https://www.geeksforgeeks.org/junit-5-test-execution-order/ https://www.geeksforgeeks.org/test-execution-for-software-testing/ https://www.javacodegeeks.com/2017/06/testing-exceptions-junit-5.html https://www.geeksforgeeks.org/junit-5-timeout/ https://www.geeksforgeeks.org/performance-testing-software-testing/
Mockito Basics	Introduction to Mockito, Mocking and stubbing, Verifying interactions, Argument matching, Handling void methods	https://www.javacodegeeks.com/mockito-tutorials
Testing Spring Applications with JUnit and Mockito	Overview of Spring testing, testing controllers, services, and repositories, Integration testing with Spring Boot, Mocking dependencies in Spring tests	https://www.geeksforgeeks.org/testing-in-spring-boot/ https://www.geeksforgeeks.org/autowired-injectmocks-spring-boot-tests/
Mocking External Dependencies	Mocking databases and repositories, mocking external services (e.g., RESTful APIs), Mocking file I/O and network interactions, Strategies for testing code with external dependencies	https://www.javacodegeeks.com/2024/12/mocking-repositories-and-daos-in-java-with-mockito.html
Introduction to Automation Testing	What is Automation Testing? Benefits of Automation Testing, Manual Testing vs. Automation Testing	https://www.geeksforgeeks.org/automation-testing-software-testing/
Types of Automated Tests	Unit Tests, Integration Tests, Functional Tests, End-to-End Tests, Performance Tests	https://www.geeksforgeeks.org/automation-testing-software-testing/
Automation Testing Process	Test Tool Selection, Defining Scope of Automation, Planning, Design, and Development, Test Execution, Maintenance	https://www.geeksforgeeks.org/automation-testing-software-testing/
Test Automation Frameworks	Types of Frameworks (e.g., Data-Driven, Keyword-Driven, Hybrid), Designing a Test Automation Framework, Best Practices for Framework Development	https://www.geeksforgeeks.org/automation-testing-software-testing/
Popular Automation Tools	Selenium, Appium, JUnit and TestNG, Cucumber, Cypress	https://www.geeksforgeeks.org/automation-testing-software-testing/

SLF4J	SLF4J vs. Log4J vs. Lombok, SLF4J - Env Setup, Sample Logging, SLF4j - error messages, warning levels, parameterized logging, different appenders	https://www.tutorialspoint.com/slf4j/index.htm

Hands-On:

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

FSE construct - Products and Frameworks

There are numerous frameworks in the programming family to build applications. These enable the programmers to develop different types of applications by providing a gamut of features provided over a single integrated environment.

Please plan to complete this in 5 weeks.
Relevant skills:

- Spring Core and Maven – 0.5 week
- Spring Data JPA with Spring Boot, Hibernate – 0.5 week
- Spring REST using Spring Boot 3 – 1.5 weeks
- Microservices with Spring Boot 3 and Spring Cloud – 0.5 week
- React & Angular overview – 2 weeks

Module 5 - Spring Core and Maven

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, Data Access, ORM, MVC, etc., Benefits of using Spring in Java applications	https://docs.spring.io/spring-framework/docs/3.2.x/spring-framework-reference/html/overview.html
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 skills' hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 6 - Spring Data JPA with Spring Boot, Hibernate

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	https://docs.spring.io/spring-data/jpa/reference/auditing.html

	@LastModifiedDate, Configuring auditing properties	
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 skills' hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 7 - Spring REST using Spring Boot 3

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.geeksforgeeks.org/spring-boot-introduction-to-restful-web-services/ 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, Exception handling in REST controllers	https://medium.com/@tericcabrel/validate-request-body-and-parameter-in-spring-boot-53ca77f97fe9 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	https://blog.treblle.com/understanding-restful-api-crud-operations/

	methods for CRUD operations, Validating input data with annotations, Optimistic locking for concurrent updates	
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-and-why-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.geeksforgeeks.org/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 skills' hands-on exercises to reinforce your learning. Refer to the **Exercise Instructions** section above to access the practice content.

Module 8 - Microservices with Spring Boot 3 and Spring Cloud

Overview:

This module focuses on Microservices with Spring Boot 3 and Spring Cloud.

Learning Objectives:

After completing this module, learners will be able to:

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Microservices	What are Microservices? Benefits of Microservices, Microservices vs. Monolithic Architecture	https://www.geeksforgeeks.org/microservices/
Principles of Microservices	Single Responsibility Principle, Decentralized Data Management, Continuous Delivery and Deployment, Scalability and Flexibility	https://www.geeksforgeeks.org/microservices/
Challenges of Microservices	Complexity Management, Inter-Service Communication, Data Consistency, Monitoring and Logging	https://www.geeksforgeeks.org/microservices/
Microservices Architecture Overview	Components of Microservices Architecture, Service Discovery, API Gateway, Load Balancing	https://www.geeksforgeeks.org/microservices/
Design Patterns	Service Registry and Discovery, Circuit Breaker, API Composition, Saga Pattern	https://www.geeksforgeeks.org/microservices/
Inter-Service Communication	Synchronous Communication (REST, gRPC), Asynchronous Communication (Message Queues,	https://www.geeksforgeeks.org/inter-service-communication-in-microservices/

	Event Streaming), Handling Failures and Retries	
Data Management	Database per Service, Event Sourcing, CQRS (Command Query Responsibility Segregation)	https://www.geeksforgeeks.org/cqrs-design-pattern-in-microservices/
Security	Authentication and Authorization, Secure Communication (HTTPS, OAuth2), Security Best Practices	https://www.geeksforgeeks.org/authentication-and-authorization-in-microservices/
Spring Cloud for Microservices	Introduction to Spring Cloud, Features and components of Spring Cloud, Configuring microservices with Spring Cloud, Service discovery and registration with Spring Cloud Netflix Eureka	https://www.geeksforgeeks.org/what-is-spring-cloud/ https://howtodoinjava.com/spring-cloud/spring-cloud-components/ https://www.geeksforgeeks.org/managing-configuration-for-microservices-with-spring-cloud-config/ https://www.geeksforgeeks.org/spring-cloud-how-to-register-microservices-using-netflix-eureka/
Spring Security for Microservices	Overview of Spring Security, Securing microservices using Spring Security, Authentication and authorization in a microservices environment, Configuring security for RESTful APIs	https://www.geeksforgeeks.org/spring-security-tutorial/ https://www.geeksforgeeks.org/best-practices-to-secure-microservices-with-spring-security/ https://www.geeksforgeeks.org/spring-security-tutorial/#spring-security-core https://www.geeksforgeeks.org/spring-security-tutorial/#spring-security-authentication-and-authorization https://www.geeksforgeeks.org/securing-rest-apis-with-spring-security/
Centralized Authentication and Authorization	Implementing centralized authentication with OAuth 2.1/OIDC, Configuring Authorization Servers and Resource Servers, Using JSON Web Tokens (JWT) for secure communication, Single Sign-On (SSO) in a microservices architecture	https://studentofjava.blog/centralized-authentication-with-spring-security-oauth/ https://www.geeksforgeeks.org/workflow-of-oauth-2-0/ https://www.geeksforgeeks.org/json-web-token-jwt/ https://examples.javacodegeeks.com/java-development/enterprise-java/jwt-tutorial-for-beginners/ https://www.geeksforgeeks.org/single-sign-on-in-microservice-architecture/

Microservices Communication with Spring Cloud	Inter-service communication patterns, Using Spring Cloud Feign for declarative REST clients, Service orchestration and choreography, Circuit Breaker pattern with Spring Cloud Circuit Breaker	https://www.geeksforgeeks.org/inter-service-communication-in-microservices/ https://www.geeksforgeeks.org/what-is-feign-client-in-microservices/ https://www.geeksforgeeks.org/orchestration-vs-choreography/
API Gateway and Edge Services	Introduction to API Gateways, Configuring API Gateway with Spring Cloud Gateway, Implementing edge services for routing and filtering, Load balancing and resilience patterns in an API Gateway	https://www.geeksforgeeks.org/api-gateway-patterns-in-microservices/ https://www.geeksforgeeks.org/spring-cloud-gateway/ https://rathoreaparna678.medium.com/api-gateway-and-edge-services-zuul-spring-cloud-gateway-dc7567e9bc94 https://www.geeksforgeeks.org/edge-pattern-in-microservices/ https://umatechnology.org/edge-routing-techniques-for-stateless-microservices-supported-by-modern-rum-tools/ https://www.geeksforgeeks.org/microservices-resilience-patterns/ https://www.geeksforgeeks.org/load-balancing-in-spring-boot-microservices/
Fault Tolerance and Resilience	Implementing fault tolerance with Spring Cloud Hystrix, Circuit Breaker and fallback mechanisms, Retrying and fallback strategies for resilience, Handling transient faults in microservices	https://www.tpointtech.com/fault-tolerance-with-hystrix https://www.geeksforgeeks.org/implementing-a-basic-circuit-breaker-with-hystrix-in-spring-boot-microservices/ https://medium.com/@AlexanderObregon/spring-microservices-resilience-with-retry-and-fallback-mechanisms-8500208fc463 https://www.geeksforgeeks.org/resilient-microservices-design/
Spring Cloud Config	Externalized configuration in microservices, Configuring microservices using	https://www.netjstech.com/2023/10/microservice-externalized-configuration-config-server.html#:~:text=In%20this%20tutorial%2C%20you%27ll%20see%20how%20to%20externalize,man

	Spring Cloud Config, Dynamic configuration updates and refresh, Managing configuration properties for different environments	age%20external%20properties%20for%20applications%20across%20all%20environments. https://www.geeksforgeeks.org/managing-configuration-for-microservices-with-spring-cloud-config/ https://www.geeksforgeeks.org/dynamic-configuration-updates-with-spring-cloud-config/ https://www.geeksforgeeks.org/spring-boot-managing-application-properties-with-profiles/
Monitoring and Metrics in Microservices	Introduction to microservices monitoring, Using Spring Boot Actuator for monitoring endpoints, Integrating monitoring tools (Prometheus, Grafana), Application and system-level metrics in microservices	https://www.techtarget.com/searchapparchitecture/tip/The-basics-of-monitoring-and-observability-in-microservices https://www.baeldung.com/spring-boot-actuators https://www.geeksforgeeks.org/prometheus-monitoring/ https://www.geeksforgeeks.org/what-is-grafana/ https://www.geeksforgeeks.org/efficient-load-balancing-and-metrics-monitoring-in-spring-cloud-microservices/
Security Best Practices in Microservices	Role-based access control (RBAC) in microservices, Securing communication between microservices, Securing sensitive data in microservices, Implementing security policies and practices	https://www.geeksforgeeks.org/authentication-and-authorization-in-microservices https://www.geeksforgeeks.org/authentication-and-authorization-in-microservices/#importance-of-security-in-microservices-architecture https://www.geeksforgeeks.org/authentication-and-authorization-in-microservices/#securing-communication-between-microservices

Hands-On:

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

Module 9 – Microservice frameworks

Overview:

This module focuses on Microservice frameworks like Micronaut and Quarkus.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the features and benefits of Microservice frameworks.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Micronaut	What is Micronaut?, Key features and benefits, Use cases and applications	https://studentofjava.blog/what-is-micronaut-a-beginners-guide-to-the-lightweight-java-framework/
Introduction to Quarkus	What is Quarkus?, Key features and benefits, Use cases and applications	https://www.baeldung.com/quarkus-io https://dzone.com/articles/spring-boot-quarkus-or-micronaut

Module 10 – Single Page Application framework - React

Overview:

This module focuses on a very widely used, simple Front-end tool to build a Single Page Application (SPA), ReactJS. It concentrates on the need for this framework, the functionalities in brings us with, the ease of the feature usage to build forms for user consumption.

Learning Objectives:

After completing this module, learners will be able to:

- Define SPA and its benefits
- Define React and identify its working
- Demonstrate on create-react-app
- Explain React components
- Demonstrate the various components of a React app

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to SPA	What is Single Page Application (SPA).	https://www.geeksforgeeks.org/what-is-single-page-application/
Introduction to React	What is React, How does it work, React features, create-react-app, React virtual DOM	https://www.geeksforgeeks.org/react/ https://www.geeksforgeeks.org/reactjs-introduction/

		https://www.geeksforgeeks.org/reactjs-virtual-dom/
React Components and Props	Functional components, Class components, Component constructor, Components in files, Pass data, Default props, State and props	https://www.geeksforgeeks.org/reactjs-components/ https://www.geeksforgeeks.org/reactjs-importing-exporting/
React ES6 and JSX	What is ES6, Classes, Class inheritance, Arrow functions, this, var, let, const, What is JSX, Nested elements in JSX, JSX attributes, JSX styling	https://www.geeksforgeeks.org/reactjs-es6/ https://www.w3schools.com/react/react_es6.asp https://www.geeksforgeeks.org/reactjs-jsx-introduction/
React Events	React event object, Event handlers, Passing arguments to event handlers	https://www.geeksforgeeks.org/react-js-events/
Conditional Rendering	Element variables, Inline if with logical && operator, Inline if-else with conditional operator, How to prevent component from rendering	https://www.geeksforgeeks.org/reactjs-conditional-rendering/
React List and Keys	Displaying a list on UI, map(), Keys, Extracting components with keys	https://www.geeksforgeeks.org/reactjs-lists/ https://www.geeksforgeeks.org/reactjs-keys/
React Forms	Controlled inputs, Uncontrolled inputs, Validation, Displaying error messages, textarea tag, select tag	https://www.geeksforgeeks.org/reactjs-forms/ https://codezup.com/react-form-validation-error-handling/
Calling API with React	How React Clients interact with a database, general discussion on the different ways of	https://www.geeksforgeeks.org/how-to-fetch-data-from-an-api-in-reactjs/

	interacting with an API from React App(Fetch Api, Axios, Jquery and XmlHttpRequest), Implementation of API interaction from React App using Fetch Api and Axios	

Hands-On:

- Complete the respective skill's hands-on exercises to reinforce your learning. Please use Visual Studio Code for the Hands-on. Refer to the **Exercise Instructions** section above to access the practice content.

Module 10 – Single Page Application framework – Angular Overview

Overview:

This module focuses on an awareness of another widely used complete framework on Single Page Application (SPA), Angular.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the basic components of an Angular application.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Understanding Angular Architecture	Components, Modules, Templates, Services and Dependency Injection, Directives	https://www.geeksforgeeks.org/angular-tutorial/ https://www.geeksforgeeks.org/build-an-app-with-angular-and-angular-cli/ https://www.geeksforgeeks.org/angular-components-overview/ https://www.geeksforgeeks.org/angular-2/

Check your understanding:

Angular quiz ▶

FSE construct – Platforms & GenAI

These are the enablers as a base to host the application developed using the Engineering concepts, programming languages, product and frameworks to make it accessible by all users, reliable and scalable.

Relevant skills:

- GIT
- CI/CD
- Containerization using Docker
- Cloud Fundamentals

Please plan to complete this in 1 week.

Module 11 – Version control - GIT

Overview:

This module focuses on the most widely used code repository tool, GIT.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the version control concepts.
- Demonstrate the branch clone, code push, forking operations
- Explain the concept of branch workflows

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Version Control	**Version Control Concepts**-Definition and Purpose, Benefits of Version Control, Types of Version Control Systems	https://www.geeksforgeeks.org/git-tutorial/ https://www.geeksforgeeks.org/version-control-systems/
Understanding Git	**What is Git**-Introduction to Git, Git as a Distributed Version Control System (DVCS); **Git Components**-Working Directory, Staging Area, Repository	https://www.geeksforgeeks.org/version-control-systems/ https://www.geeksforgeeks.org/git-introduction/ https://www.geeksforgeeks.org/what-is-a-git-repository/
Setting Up Git	**Installing Git**-Download and Installation Steps, Configuring Basic Settings (username,	https://www.geeksforgeeks.org/git-introduction/

	email); **Creating a Git Repository**-Initializing a New Repository, Cloning an Existing Repository;	
Basic Git Commands	**git init and git clone** - Initializing a New Repository, Cloning an Existing Repository; **git add**-Staging Changes, Using Wildcards; **git commit**-Committing Changes, Adding Commit Messages; **git status**-Checking the Status of Your Repository; **git log**-Viewing Commit History, Options for Customizing Log Output;	https://www.geeksforgeeks.org/basic-git-commands-with-examples/ https://www.geeksforgeeks.org/essential-git-commands/
Branching and Merging	**Introduction to Branching**-Creating and Managing Branches, Switching Between Branches; **Merging Changes**-Merging Branches, Handling Merge Conflicts; **Branching Strategies**-Feature Branching, Release Branching, Git Flow Workflow;	https://www.geeksforgeeks.org/branching-strategies-in-git/ https://www.geeksforgeeks.org/git-merge/ https://www.geeksforgeeks.org/how-to-merge-two-branches-in-git/
Remote Repositories	**Adding Remote Repositories**-Linking Local and Remote Repositories, Multiple Remotes; **git pull and git push**-Pulling Changes from a Remote Repository, Pushing Changes to a Remote Repository; **Handling Remote Branches**-Tracking and Creating Remote Branches;	https://www.geeksforgeeks.org/what-is-a-git-repository/ https://www.geeksforgeeks.org/what-is-a-git-repository/#git-push-and-pull-commands
Collaborating with Git	**Forking and Pull Requests**-Forking a Repository, Creating and Managing Pull Requests; **Git Collaboration Workflows**-	https://www.geeksforgeeks.org/git-fork/ https://www.geeksforgeeks.org/git-workflows-for-agile-development-teams/

	Centralized Workflow, Feature Branch Workflow, Forking Workflow, Gitflow Workflow	

Hands-On:

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

Module 12 – DevOps and CICD

Overview:

This module focuses on an awareness of the concept of Development and Operations done together in the current business development model.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the concepts of DevOps
- Explain the details of the components of DevOps like Continuous Integration and Continuous Delivery
- List the popular tools used in DevOps

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to DevOps	What is DevOps?, Goals and Benefits of DevOps, Key DevOps Practices	https://www.geeksforgeeks.org/devops-tutorial/ https://www.geeksforgeeks.org/introduction-to-devops/
Understanding CI/CD	What is Continuous Integration (CI)?, What is Continuous Deployment/Delivery (CD)?, Differences between CI and CD, Benefits of CI/CD	https://www.geeksforgeeks.org/introduction-to-devops/ https://www.geeksforgeeks.org/what-is-ci-cd/
CI/CD Tools and Platforms	Overview of Popular CI/CD Tools (e.g., Jenkins, GitHub Actions, GitLab CI/CD, CircleCI)	https://www.geeksforgeeks.org/introduction-to-devops/

Check your understanding:

DevOps quiz ▶

Module 13 - Cloud fundamentals

Overview:

This module focuses on an awareness of the concepts in Cloud computing.

Learning Objectives:

After completing this module, learners will be able to:

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Cloud Computing	Traditional IT Deployment, Virtualization, Service-Oriented Architecture (SOA), Cloud vs. On-Premises Data Centers, Pros and Cons of Cloud Computing	https://www.flackbox.com/traditional-it-deployment-models-on-prem-colo https://www.geeksforgeeks.org/fundamentals-of-software-architecture/ https://www.geeksforgeeks.org/cloud-computing/
Cloud Service Models	Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS)	https://www.geeksforgeeks.org/cloud-computing/ https://www.geeksforgeeks.org/cloud-based-services/
Cloud Deployment Models	Public Cloud Model, Private Cloud Model, Hybrid Cloud Model, Community Cloud Model	https://www.geeksforgeeks.org/cloud-computing/ https://www.geeksforgeeks.org/cloud-deployment-models/
Cloud Service Providers	AWS, Azure, GCP	https://www.geeksforgeeks.org/aws-tutorial/ https://www.geeksforgeeks.org/what-is-microsoft-azure/ https://www.geeksforgeeks.org/google-cloud-platform-gcp/

Check your understanding:

Cloud Quiz ▶

Module 14 – Containerization using Docker

Overview:

This module focuses on an awareness of the most widely used application containerization tool, Docker.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the concept and features of Docker.
- List the basic Docker commands.
- Explain the concepts of Docker storage, networking, and orchestration.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Docker Commands	Basic Docker Commands - Run, ps, stop, rm, images, rmi, pull, Append a command, Exec	https://www.geeksforgeeks.org/docker-tutorial/
Docker Run	How to Use the docker run Command, Run a Container Under a Specific Name, Run a Container in the Background (Detached Mode), Run a Container Interactively, Run a Container and Publish Container Ports, Run a Docker Container and Remove it Once the Process is Complete	https://www.geeksforgeeks.org/docker-tutorial/
Docker Images	What is a Docker Image?, Image Layers, Container Layer, Parent Image, Base Image, Docker Manifest, Container Registries, Container Repositories, How to Create a Docker Image - Interactive Method, Dockerfile	https://www.geeksforgeeks.org/docker-tutorial/

	Method, The Docker Build Context	
Docker Compose	What is Docker Compose?, Benefits of Docker Compose, Basic Commands in Docker Compose, Install Docker Compose, Create the Compose file,The YAML Configuration file	https://www.geeksforgeeks.org/docker-tutorial/
Docker Engine	What is Docker Engine-Docker CLI, REST API, Docker Deamon	https://www.geeksforgeeks.org/docker-tutorial/
Docker Storage	Storage Drivers, Data Volumes, Changing the Storage Driver for a Container, Creating a Volume, Listing all the Volumes	https://www.geeksforgeeks.org/docker-tutorial/
Docker Networking	Default Networks, Listing All Docker Networks, Inspecting a Docker network, Creating Your Own New Network	https://www.geeksforgeeks.org/docker-tutorial/
Container Orchestration	What is container orchestration?, Why do we need container orchestration?, What are the benefits of container orchestration?, What is Kubernetes container orchestration?, Container orchestration versus Docker	https://www.geeksforgeeks.org/docker-tutorial/

Check your understanding:

Docker quiz ►

Gen AI Fundamentals

Overview:

This module focuses on the very famous concepts of Generative Artificial Intelligence, different tools and their features, and the application of GenAI tools for real-time problems.

Learning Objectives:

After completing this module, learners will be able to:

- Explain what Generative AI is and its history
- Explain the types of GenAI models
- Explain few widely used GenAI tools and its practical applications

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Generative AI	What is Generative AI? Overview of Generative AI applications, History and evolution of Generative AI	https://www.gartner.com/en/topics/generative-ai https://www.geeksforgeeks.org/what-is-generative-ai/
Types of Generative AI Models	GANs (Generative Adversarial Networks), VAEs (Variational Autoencoders)	https://www.geeksforgeeks.org/generative-ai-models/ https://www.geeksforgeeks.org/what-is-generative-ai/ https://www.geeksforgeeks.org/variational-autoencoders/
Popular Generative AI Tools	Overview of OpenAI's GPT, Introduction to Google's BERT	https://www.geeksforgeeks.org/generative-ai-models/ https://www.geeksforgeeks.org/github-copilot/ https://www.geeksforgeeks.org/explanation-of-bert-model-nlp/
Practical Applications of Generative AI	Text generation and completion, Image generation and editing, Music and audio synthesis	https://www.geeksforgeeks.org/applications-of-ai/

What's Next?

Congratulations on successfully completing the 8-week DN 4.0 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 with your assessment and look forward to seeing you apply your newly acquired knowledge and skills. Good luck!