cognizant

Digital Nurture 4.0

Deep Skilling Handbook - DotNet 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 DotNet FSE modules.

- **Deep Skilling covers the below mentioned skills.**

    - Design Patterns and Principles | Data Structures and Algorithms | Advanced SQL | Unit testing framework | ORM Framework | Microservices and WebApi | Containerization | Version control | DevOps concepts | Front-end JS Framework | 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 are 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:

   - C# - Covered in upskilling
   - SQL - Basic (covered in upskilling) and Advanced (covered in deep skilling)
   - NUnit and Moq

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:
   - Entity Framework Core 8.0
   - ASP.NET Core 8.0 Web API
   - Microservices Architecture using ASP.NET Core Web API – This is an architecture pattern.
   - 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

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’.

Program Completion Criteria

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

Hands-on Exercises:

- Candidates must complete the 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 recommendations consistently. • Ensure to complete the mandatory exercises and review sessions.

Where to Practice?

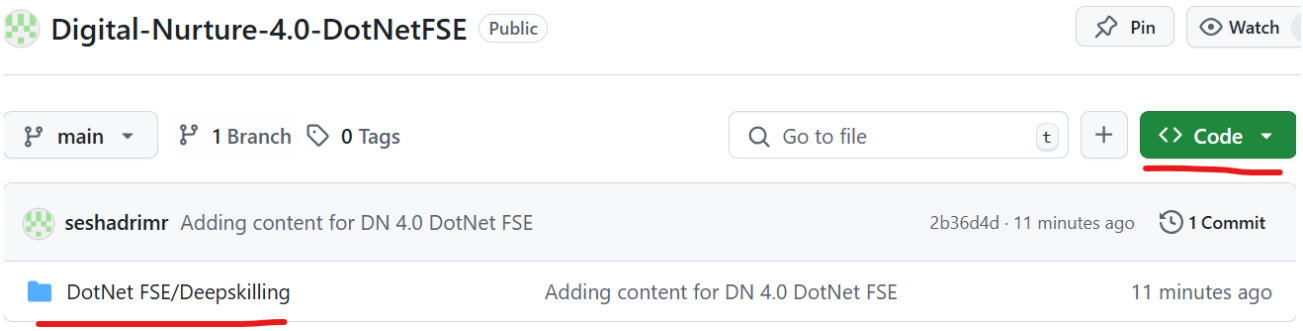
Installed Software Tools
• The skills covered require 'Visual studio community edition' to be used. Download it from Visual Studio Community | Download Latest Free Version • Install Sql server management studio from Install SQL Server Management Studio | Microsoft Learn • Install Visual studio code from 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 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 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 'DotNet 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 - DotNet 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 programming languages and frameworks are built.

Relevant skills:

- Design Patterns and Principles
- Data Structures and Algorithms

Please plan to complete the skills in 1 weeks time.

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 -	https://www.geeksforgeeks.org/sorting-algorithms/

	Worst, Average and Best-Case analysis	
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-DotNet/
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 skill's 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:

- C# - Covered in upskilling
- SQL – Basic (covered in upskilling) and Advanced (covered in deepskilling)
- NUnit and Moq

Please plan to complete this in 1 week.

Module 3 - Advanced SQL Using SQL Server

Overview:

This module covers a little advanced topics in SQL server for programming in it, apply logic to formulate Stored procedures and execute them, transactions to ensure data integrity.

Learning Objectives:

After completing this module, learners will be able to:

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Advanced Concepts	Introduction to SQL Server, SQL Server Editions, Key Components and Services of SQL Server, SQL Server Instances, SQL Server Management Studio	https://www.geeksforgeeks.org/introduction-of-ms-sql-server/ https://learn.microsoft.com/en-us/ssms/download-sql-server-management-studio-ssms
	OVER(), PARTITION BY, ROW_NUMBER, RANK, DENSE_RANK, GROUPING SETS, CUBE, ROLLUP, WITH Statement, Common Table Expression (CTE), Recursive CTE, MERGE, PIVOT, UNPIVOT	https://www.sqlservertutorial.net/sql-server-window-functions/ https://www.sqlservertutorial.net/sql-server-basics/ https://www.geeksforgeeks.org/unpivot-in-sql-server/
Views and Indexes	Introduction to Views, Creating Views, Modifying Views, Dropping Views, Types of Views, Updating Views, Indexed Views, Security and Views, Introduction to Indexes, Types of Indexes, Creating Indexes, Modifying Indexes, Dropping Indexes, Covering Indexes, Index Fragmentation, Indexing and Query Optimization	https://www.sqlservertutorial.net/sql-server-views/ https://www.sqlservertutorial.net/sql-server-indexes/ https://www.mssqltips.com/sqlservertip/4331/sql-server-index-fragmentation-overview/ https://www.geeksforgeeks.org/best-practices-for-sql-query-optimizations/
Stored Procedures and User-Defined Functions	What is a Stored Procedure?, Benefits of Using Stored Procedures, Types of Stored Procedures - User-defined, Temporary, System, Extended User-Defined, Create a Stored Procedure, Modify a Stored Procedure, Delete a Stored Procedure, Execute a Stored Procedure, Grant Permissions on a Stored Procedure, Return Data from a Stored Procedure, Rename a Stored Procedure, Parameters, What are Functions in SQL Server?, Benefits of user-defined functions, Types of functions-Scalar functions, Table-valued functions, System functions, Built-in system functions, Create User-defined Functions, Modify User-defined Functions, Delete User-	https://www.sqlservertutorial.net/sql-server-stored-procedures/ https://learn.microsoft.com/en-us/sql/relational-databases/stored-procedures/grant-permissions-on-a-stored-procedure?view=sql-server-ver16 https://www.sqlservertutorial.net/sql-server-user-defined-functions/

	defined Functions, Execute User-defined Functions	
Triggers and Cursors	What are Triggers?, When we use Triggers?, Types of SQL Server Triggers, DML Triggers - After Triggers,Instead Of Triggers, Logon Triggers, Create Trigger, Modify Triggers using SSMS, DELETE Triggers, Advantages of Triggers, Disadvantages of Triggers, What are Cursors?, Life Cycle of the Cursor, Uses of SQL Server Cursor, Types of Cursors in SQL Server, Limitations of SQL Server Cursor, Alternates to Cursor	https://www.sqlservertutorial.net/sql-server-triggers/ https://www.sqlservertutorial.net/sql-server-stored-procedures/sql-server-cursor/ https://www.scholarhat.com/tutorial/sqlserver/sql-server-cursor-alternatives
Exception Handling	What are Exceptions?, Types of SQL Server Exceptions - System Defined, User Defined, Retrieving detailed information on the error, TRY/CATCH, Using Nested TRY...CATCH Constructs, THROW and RAISERROR	https://www.mssqltips.com/sqlservertip/7997/sql-server-try-catch-raiserror-throw-error-handling/
Transactions	What are transactions?, ACID Properties, Transaction States, Transaction Management Statements, Savepoints, Implicit vs. Explicit Transactions, Nested Transactions, Transaction Isolation Levels, Locking and Blocking, Transaction Deadlocks	https://www.sqlservertutorial.net/sql-server-basics/sql-server-transaction/ https://www.geeksforgeeks.org/sql-server-transaction/ https://dotnettutorials.net/lesson/sql-server-savepoints-transaction/ https://www.sqlservercentral.com/articles/isolation-levels-in-sql-server https://www.sqlservertutorial.net/sql-server-administration/sql-server-deadlock/

Hands-On:

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

Module 4 – NUnit and Moq

Overview:

This module focuses on the Unit testing concepts through Microsoft .Net's NUnit and Mocking framework.

Learning Objectives:

After completing this module, learners will be able to:

- Explain what automated testing is and its benefits.
- Demonstrate unit testing concepts using NUnit framework.
- Demonstrate the usage of mocking framework to mock the dependencies in unit testing completion.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Getting Started	What is Automated Testing, Benefits of Automated Testing, Types of Tests, Test Pyramid, Popular Testing Frameworks, Using NUnit in Visual Studio, What is Test-Driven Development	https://www.geeksforgeeks.org/automation-testing-software-testing/ https://www.geeksforgeeks.org/benefits-of-automation-testing/ https://martinfowler.com/articles/practical-test-pyramid.html https://www.geeksforgeeks.org/test-driven-development-tdd/
Fundamentals of Unit Testing	Characteristics of Good Unit Tests, What to Test and What Not to Test, Naming and Organizing Tests, Black-box Testing, Set Up and Tear Down, Parameterized Tests, Ignoring Tests, Writing Trustworthy Tests	https://www.tutorialspoint.com/unit-testing-tutorial-for-beginners-concepts-types-tools https://www.geeksforgeeks.org/software-engineering-black-box-testing/
Core Unit Testing Techniques	Testing Strings, Testing Arrays and Collections, Testing Return Type of Methods, Testing Void Methods, Testing Methods that Throw Exceptions, Testing Private Methods, Code Coverage	https://www.c-sharpcorner.com/article/introduction-to-nunit-testing-framework/ https://dotnetpattern.com/nunit-introduction
Breaking External Dependencies	Loosely-coupled and Testable Code, Refactoring Towards a Loosely coupled Design, Dependency Injection via Method Parameters, Dependency Injection via Properties, Dependency Injection	https://dotnettutorials.net/lesson/dependency-injection-design-pattern-csharp/ https://dotnettutorials.net/lesson/dependency-injection-design-pattern/ https://dotnettutorials.net/lesson/unity-container-asp-net-mvc/

	via Constructor, Dependency Injection Frameworks, Mocking Frameworks, Creating Mock Objects Using Moq, State-based vs. Interaction Testing, Testing the Interaction Between Two Objects	https://www.codemag.com/Article/2305041/Using-Moq-A-Simple-Guide-to-Mocking-for-.NET

Hands-On:

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

FSE construct - Products and Frameworks

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' time.

Relevant skills:

- Entity Framework Core 8.0 - 1 weeks
- ASP.NET Core 8.0 Web API – 1.5 weeks
- Microservices Architecture using ASP.NET Core Web API – This is an architecture pattern. - 0.5 week
- React & Angular overview – 2 weeks

Module 5 - Entity Framework Core 8.0

Overview:

This module introduces the learner to the Object Relational Framework concept through Entity framework of Microsoft .NET.

Learning Objectives:

After completing this module, learners will be able to:

- Explain what ORM framework is, the features in Entity framework Core supporting the concepts of ORM.
- Demonstrate database model creation using EF core.
- Demonstrate the database operations via the model created; use LINQ queries to perform the operation.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Overview of EF Core 8 and .NET 8 Integration	What is ORM (Object-Relational Mapping)?, EF Core vs EF Framework: Key Differences, New Features in EF Core 8	https://dotnettutorials.net/lesson/entity-framework-core/
Setting up EF Core in a .NET 8 Project	Installing EF Core Packages via NuGet, Configuring DbContext, Connecting to SQL Server, Basic EF Core CLI Commands (Add Migration, Update Database)	https://dotnettutorials.net/lesson/install-entity-framework-core/ https://dotnettutorials.net/lesson/dbcontext-entity-framework-core/ https://dotnettutorials.net/lesson/database-connection-string-in-entity-framework-core/
Creating a Simple Database Model	Defining Entities and Relationships Primary Keys, Foreign Keys, and Navigation Properties, Code-First Approach Overview	https://dotnettutorials.net/lesson/relationships-in-entity-framework-core/ https://dotnettutorials.net/lesson/student-management-application-using-ef-core-code-first-approach/ https://www.c-sharpcorner.com/blogs/entity-framework-code-first-vs-database-first-approach
Performing Basic CRUD Operations	Inserting Records into the Database (AddAsync), Retrieving Data with Find, FirstOrDefault, and ToListAsync, Updating Records and Tracking Changes, Deleting Records (Remove vs DeleteRange)	https://dotnettutorials.net/lesson/crud-operations-in-entity-framework-core/
LINQ Queries in EF Core 8	Writing Queries Using Where, Select, and OrderBy, Projection into DTOs (Data Transfer Objects), Filtering and Aggregating Data, Asynchronous Queries with ToListAsync()	https://dotnettutorials.net/lesson/linq-to-entities-in-entity-framework-core/
EF Core Migrations and Database Updates	Adding, Removing, and Updating Migrations, Seeding Data during Migrations, Managing Database Schema Changes	https://www.entityframeworktutorial.net/efcore/entity-framework-core-migration.aspx
Handling Relationships and Data Loading	Eager, Lazy, and Explicit Loading Explained, Configuring One-to-One, One-to-Many, and	https://blog.jetbrains.com/dotnet/2023/09/21/eager-lazy-and-explicit-loading-with-entity-framework-core/

	Many-to-Many Relationships, Navigating Circular References	https://dotnettutorials.net/lesson/relationships-in-entity-framework-core/
Performance Optimizations and Best Practices	Query Caching and Tracking Behavior (AsNoTracking), Batch Processing and Bulk Operations, Handling Concurrency with RowVersion Columns, Using Compiled Queries for Performance Boost	https://dotnetevangelist.net/post/performance-optimization-tricks/

Hands-On:

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

Module 6 - ASP.NET Core 8.0 Web API

Overview:

This module introduces the learner to the concepts of ASP. Net Core, RESTful services, its implementation in Microsoft .NET through Web Api.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the concept of RESTful web service, Web API & Microservice
- List the types of Action Verbs
- Demonstrate creation of a simple Web API - With Read, Write actions
- Demonstrate Swagger installation to Web API and Web API listing on browser
- Demonstrate JWT-based authentication and authorization

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links
Introduction to Web APIs and ASP.NET Core	Overview of Web APIs, REST vs. SOAP, Setting up .NET 8 Development Environment, ASP.NET Core Web API project structure	https://dotnettutorials.net/lesson/introduction-to-asp-net-core-web-api/ https://dotnettutorials.net/lesson/environment-setup-asp-net-core-web-api/ https://dotnettutorials.net/lesson/asp-net-core-web-api-project-in-visual-studio-2019/ https://dotnettutorials.net/lesson/asp-net-core-web-api-files-and-folders/

Building RESTful APIs with ASP.NET Core	Creating controllers, actions, and routes, CRUD operations with Entity Framework Core, JSON formatting and serialization	https://dotnettutorials.net/lesson/controllers-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/models-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/routing-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/get-method-in-web-api/ https://dotnettutorials.net/lesson/post-method-in-web-api/ https://dotnettutorials.net/lesson/put-method-in-web-api/ https://dotnettutorials.net/lesson/delete-method-in-web-api/
Advanced API Features	Attribute routing and query parameters, Middleware and custom filters, Implementing JWT-based authentication and authorization	https://dotnettutorials.net/lesson/variables-and-query-strings-in-routing/ https://dotnettutorials.net/lesson/filters-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/action-filters-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/authorization-filters-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/jwt-authentication-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/role-based-jwt-authentication-in-asp-net-core-web-api/
Consuming and Creating SOAP Services	Introduction to SOAP APIs and WCF, Consuming SOAP services in ASP.NET Core, Creating SOAP services using WCF	https://www.tutorialspoint.com/soap/what_is_soap.htm https://www.tutorialspoint.com/wcf/wcf_overview.htm https://www.geeksforgeeks.org/difference-between-wcf-and-web-service/
API Security and Exception Handling	Global exception handling and error responses, Logging with Serilog, Securing APIs with API keys and Oauth	https://dotnettutorials.net/lesson/exception-filters-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/cors-in-asp-net-core-web-api/

		https://dotnettutorials.net/lesson/logging-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/custom-logging-provider-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/logging-using-serilog-in-asp-net-core-web-api/ https://www.c-sharpcorner.com/article/using-api-key-authentication-to-secure-asp-net-core-web-api/
API Documentation and Testing	Integrating Swagger/OpenAPI for API documentation, Using Postman for manual API testing, Automating API testing with REST Client	https://dotnettutorials.net/lesson/swagger-api-in-asp-net-core-web-api/ https://www.softwaretestingmaterial.com/postman-tutorial/

Hands-On:

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

Module 7 - Microservices Architecture using ASP.NET Core Web API

Overview:

This module focuses on the importance of loose coupling and logically independent services following design principles. The architectural pattern of Microservices using ASP. Net core Web Api.

Learning Objectives:

After completing this module, learners will be able to:

- Explain the architecture pattern of Microservices.
- Explain the advantages and challenges of Microservices.
- Compare the Microservices and Monolith architecture patterns.
- Demonstrate the service discovery pattern in ASP .NET Core Web Api.
- Explain the importance of monitoring and logging in microservices.
- Explain the deployment strategies for microservices.
- Explain an overview of automating deployment with Docker and Kubernetes.

Self-Learning (Open-source links):

Key Topics	Sub-topics	Learning Reference Links

Introduction to Microservices Architecture	Overview of microservices architecture, Advantages and challenges of microservices, Comparison with monolithic architecture, Setting up an ASP.NET Core Web API project	https://dotnettutorials.net/lesson/microservices-using-asp-net-core/
Microservice Communication and Data Management	Inter-service communication methods (HTTP, messaging, gRPC), Service discovery and registration Implementing communication patterns in ASP.NET Core Web API, Database per service vs. shared database patterns, Implementing data access with Entity Framework Core in microservices, Data consistency and transactions in microservices	https://www.geeksforgeeks.org/inter-service-communication-in-microservices/
Security, Monitoring, and Deployment	Authentication and authorization in microservices, Implementing JWT (JSON Web Tokens) authentication, Securing ASP.NET Core Web API endpoints, Importance of monitoring and logging in microservices, Logging strategies in ASP.NET Core Web API, implementing health checks and metrics endpoints, Deployment strategies for microservices (rolling updates, blue-green deployments), Setting up CI/CD pipelines with Azure DevOps	https://dotnettutorials.net/lesson/jwt-authentication-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/role-based-jwt-authentication-in-asp-net-core-web-api/ https://dotnettutorials.net/lesson/logging-in-asp-net-core-web-api/ https://medium.com/@jeslurrahman/implementing-health-checks-in-net-8-c3ba10af83c3 https://medium.com/@hitendra.patel2986/deployment-strategies-for-microservices-a-complete-guide-8f403e6d8b3e https://medium.com/@sarshar.samoo/getting-started-with-ci-cd-pipelines-for-asp-net-core-api-using-azure-devops-1d637afd4e1f https://medium.com/@ucheblessed/ci-cd-with-docker-and-kubernetes-deploying-containerized-applications-7556f0727517

	or GitHub Actions, Automating deployment with Docker and Kubernetes	

Hands-On:

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

Module 8 – 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 a 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,	https://www.geeksforgeeks.org/reactjs-components/ https://www.geeksforgeeks.org/reactjs-importing-exporting/

	Pass data, Default props, State and props	
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 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	https://www.geeksforgeeks.org/how-to-fetch-data-from-an-api-in-reactjs/

Hands-On:

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

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

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

Please plan to complete this in 1 week.

Module 9 – 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, email); **Creating a Git Repository**-Initializing a New Repository, Cloning an Existing Repository;	https://www.geeksforgeeks.org/git-introduction/
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**-	https://www.geeksforgeeks.org/branching-strategies-in-git/ https://www.geeksforgeeks.org/git-merge/

	Merging Branches, Handling Merge Conflicts; **Branching Strategies**-Feature Branching, Release Branching, Git Flow Workflow;	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**-Centralized Workflow, Feature Branch Workflow, Forking Workflow, Gitflow Workflow	https://www.geeksforgeeks.org/git-fork/ https://www.geeksforgeeks.org/git-workflows-for-agile-development-teams/

Hands-On:

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

Module 10 – 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 11 – 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,	https://www.geeksforgeeks.org/docker-tutorial/

	pull, Append a command, Exec	
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 Method, The Docker Build Context	https://www.geeksforgeeks.org/docker-tutorial/
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 features 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 learnt in this tenure**, 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!