Dapper in .NET – Course Overview

Duration: 5 Days

Audience:

- .NET Developers
- Backend Developers
- SQL Developers
- Application Architects

Tools Covered:

- Dapper (micro-ORM)
- .NET (Core/Framework)
- SQL Server / PostgreSQL / MySQL
- ADO.NET concepts
- ASP.NET Core Web API integration

Prerequisites:

- SQL fundamentals
- Relational database concepts
- Basic .NET and C# understanding
- Familiarity with Entity Framework or ADO.NET (helpful but not mandatory)

Day 1 – Fundamentals (Duration: 6–7 hours)

- Module 1: Introduction to Dapper
    - Micro-ORM concept, comparison with EF Core
- Module 2: Setup & Configuration
    - Installing via NuGet, configuring database connections
- Module 3: CRUD Operations
    - Insert, Update, Delete, Select with Execute and Query
- Module 4: Object Mapping
    - Mapping results to POCOs, handling nulls and conversions

Day 2 – Advanced Querying (Duration: 6–7 hours)

- Module 5: Parameterized Queries
    - Preventing SQL injection, using parameters effectively
- Module 6: Multi-Mapping
    - Handling joins, mapping related entities
- Module 7: QueryMultiple
    - Working with multiple result sets in one query
- Module 8: Transactions
    - Using IDbTransaction for atomic operations
- Module 9: Stored Procedures
    - Executing stored procedures with input/output parameters

Day 3 – Performance & Best Practices (Duration: 6–7 hours)

- Module 10: Performance
    - Dapper vs EF Core benchmarks, when to use each
- Module 11: Connection Management
    - Best practices for pooling and resource handling
- Module 12: Error Handling
    - Logging and structured exception handling
- Module 13: Repository Pattern
    - Abstracting queries for maintainability

Day 4 – Integration & Hybrid Approach (Duration: 6–7 hours)

- Module 14: Hybrid Approach
    - Combining EF Core and Dapper in one project
- Module 15: Testing
    - Unit testing repositories with xUnit/NUnit
- Module 16: Integration
    - Building a small ASP.NET Core Web API with Dapper

Day 5 – Capstone Project & Real-World Scenarios (Duration: 6–7 hours)

- Module 17: End-to-End Project
    - Designing a layered architecture with Dapper
    - Implementing repositories, transactions, and stored procedures
- Module 18: Deployment & Optimization
    - Deploying ASP.NET Core Web API with Dapper to Azure
    - Performance tuning and monitoring
- Module 19: Case Studies & Wrap-Up
    - Real-world scenarios of hybrid EF Core + Dapper usage
    - Best practices checklist and Q&A