**Lab 1: Understanding ORM with a Retail Inventory System**

1. **What is ORM?**

• Explain how ORM maps C# classes to database tables.

• Benefits: Productivity, maintainability, and abstraction from SQL.

**ORM (Object Relational Mapping)** is a way of automatically mapping the objects (classes) you define in C# to relational tables in a database like SQL Server.

**For example:**

| **C# class** | **maps to** | **SQL table** |
| --- | --- | --- |
| Product | → | Products |
| Category | → | Categories |

* Each **property** of a C# class becomes a **column** of the table.
* Each **instance** of the class becomes a **row** of data.

So we do not have to write raw INSERT, SELECT, UPDATE SQL commands every time.  
Instead, we work with **C# objects** — EF Core takes care of generating and executing the SQL under the hood.

**Benefits of using an ORM like EF Core:**

* Faster development (no repetitive SQL code)
* Easier to maintain (changes in class automatically map to DB)
* Consistent with .NET patterns
* Supports modern features like LINQ, async, etc.

2. **EF Core vs EF Framework:**

• EF Core is cross-platform, lightweight, and supports modern features like

LINQ, async queries, and compiled queries.

• EF Framework (EF6) is Windows-only and more mature but less flexible.

· **EF Core**:

* Cross-platform (.NET 6/7/8)
* Lighter, faster
* Modern
* More flexible for migrations

· **EF Framework (EF6)**:

* Windows-only
* Older (but stable)
* Limited to synchronous operations

3. **EF Core 8.0 Features:**

* JSON column mapping (store JSON in a column natively)
* Compiled models for faster startup
* Bulk operations improved
* Interceptors (like middleware for EF Core)
* .NET 8 support

1. **Create a .NET Console App:**
2. **Install EF Core Packages:**

**OUTPUT:**

