INTERNSHIP PROGRESS REPORT

**About .NET**

.NET is a software framework developed by Microsoft that provides a programming model for building and deploying web, desktop, mobile, gaming, and IoT applications. It provides a common set of libraries, languages, and tools that developers can use to build applications that run on multiple platforms, including Windows, Linux, and macOS.

The .NET framework consists of several components, including the Common Language Runtime (CLR), the Base Class Library (BCL), and various development frameworks and libraries. The CLR is responsible for managing the execution of code written in .NET languages, while the BCL provides a set of core classes and types that can be used by .NET applications.

The .NET framework supports several programming languages, including C#, F#, and Visual Basic, and provides a variety of development tools, including Visual Studio and Visual Studio Code. It also includes various runtime environments, including .NET Core and .NET Framework.

.NET offers several benefits to developers, including enhanced security, automatic memory management, and the ability to develop and deploy applications on multiple platforms. It also provides a rich set of APIs and libraries that can help developers build applications quickly and easily.



High-level diagram of the .NET architecture:

Here's a breakdown of the components in the .NET architecture:

* Common Type System (CTS): Defines how types are declared, used, and managed in .NET. This includes data types, classes, interfaces, and other constructs used by .NET applications.
* Common Language Runtime (CLR): Provides a runtime environment for executing .NET applications. It includes a Just-In-Time (JIT) compiler that compiles .NET code into machine code at runtime, automatic memory management, and security features.
* Development Frameworks: Provide a set of tools and libraries for developing .NET applications. Examples include ASP.NET for web development, WinForms for desktop development, and Xamarin for mobile development.
* Libraries: Provide a set of pre-built components and classes that can be used in .NET applications. Examples include the System namespace in the Base Class Library (BCL), which provides core functionality such as IO, threading, and networking.
* .NET Application: The application built using .NET that runs on the CLR. It can be a desktop, web, mobile, or IoT application.

Overall, the .NET architecture provides a robust and flexible platform for building applications that can run on multiple platforms and devices.

**Frameworks in .NET**

A framework in .NET refers to a collection of libraries, tools, and language compilers that provide a foundation for developing software applications. It provides a set of reusable components, classes, and APIs that simplify common programming tasks and make it easier to build complex applications.

There are several frameworks in .NET, including:

* .NET Framework: This is the original framework that was released by Microsoft in 2002. It provides a set of libraries, runtime environments, and development tools for building Windows desktop applications, web applications, and server applications.
* .NET Core: This is a cross-platform framework that was introduced in 2016. It provides a set of libraries, runtime environments, and development tools for building applications that can run on Windows, Linux, and macOS. It's used to build web applications, microservices, desktop applications, and gaming applications.
* Xamarin: This is a framework for building cross-platform mobile applications using C#. It allows developers to build native mobile applications for iOS, Android, and Windows using a shared codebase.
* ASP.NET: This is a framework for building web applications using .NET. It provides a set of libraries, runtime environments, and development tools for building web applications, web services, and APIs.
* Entity Framework: This is an object-relational mapping (ORM) framework that allows developers to work with databases using .NET objects. It provides a set of libraries and tools for mapping database tables to .NET objects, querying databases using LINQ, and performing CRUD (Create, Read, Update, Delete) operations.

Overall, frameworks in .NET provide developers with a foundation for building software applications and simplify common programming tasks. They make it easier to build complex applications, reduce development time, and improve the maintainability of code.

**C# Programing Language Used**

C# (pronounced "C sharp") is a modern, object-oriented programming language developed by Microsoft in the early 2000s as part of the .NET initiative. It's designed to be simple, powerful, and easy to learn, and it's widely used for building a wide range of software applications, including desktop applications, web applications, games, and mobile apps.

Here are some key features of C#:

* Object-oriented programming: C# is an object-oriented language, which means that it's designed to work with objects, classes, and inheritance. This allows developers to write code that's organized, modular, and reusable.
* Strongly typed: C# is a strongly typed language, which means that every variable and object must have a declared data type. This makes code more reliable and less error-prone.
* Garbage collection: C# includes automatic memory management, which means that the language automatically manages memory allocation and deallocation, reducing the risk of memory leaks and other memory-related issues.
* Integrated development environment (IDE): Microsoft provides a powerful IDE called Visual Studio for developing C# applications. It includes a code editor, debugger, compiler, and many other features that make development easier and more efficient.
* Cross-platform: With the introduction of .NET Core, C# can now be used to build cross-platform applications that can run on Windows, Linux, and macOS.
* LINQ: Language-Integrated Query (LINQ) is a powerful feature in C# that allows developers to write queries against data sources using a syntax similar to SQL. This makes it easier to work with data in C# applications.

Overall, C# is a powerful and versatile programming language that's well-suited for building a wide range of applications. It's widely used in the software industry, and it's a great language for both beginners and experienced developers.

**Entity Framework**

Entity Framework (EF) is an object-relational mapping (ORM) framework that allows developers to work with databases using C# objects. It's a powerful tool for building data-driven applications, and it simplifies many common database-related tasks, such as querying, inserting, updating, and deleting data.

Here are some key features of Entity Framework:

* Mapping: Entity Framework provides a mapping layer that maps database tables to C# classes and properties. This allows developers to work with the database using C# objects, which are often easier to work with than raw SQL.
* LINQ: Entity Framework supports Language-Integrated Query (LINQ), which is a powerful feature in C# that allows developers to write queries against data sources using a syntax similar to SQL. This makes it easier to work with data in Entity Framework.
* Automatic change tracking: Entity Framework automatically tracks changes made to objects in memory and applies those changes to the database when SaveChanges() is called. This makes it easy to perform CRUD (Create, Read, Update, Delete) operations.
* Database migrations: Entity Framework allows developers to create and manage database schema changes using code-first migrations. This allows developers to easily make changes to the database schema without manually writing SQL scripts.
* Performance optimizations: Entity Framework includes a number of performance optimizations, such as caching, lazy loading, and batched queries. These optimizations help to improve the performance of Entity Framework applications.

Overall, Entity Framework is a powerful and versatile tool for building data-driven applications in C#. It simplifies many common database-related tasks and allows developers to work with databases using C# objects, which can be easier to work with than raw SQL.



Entity framework working diagram

**SQL Database**

SQL (Structured Query Language) Database is a relational database management system (RDBMS) that uses SQL as its main language for managing and manipulating data. SQL databases store data in tables, which are composed of rows and columns. Each column has a data type that defines the type of data it can store, such as text, numbers, or dates.

SQL databases support a wide range of operations, including creating, modifying, and deleting tables, inserting and updating data, and querying data using SQL statements. SQL databases also provide advanced features such as indexing for faster data retrieval, transaction management for ensuring data consistency, and security features such as user authentication and data encryption.

Some popular SQL database management systems include Microsoft SQL Server, MySQL, Oracle Database, PostgreSQL, and SQLite. SQL databases are widely used in various industries such as finance, healthcare, e-commerce, and many others, due to their robustness, scalability, and flexibility.

**Problem Identification**

Our main Aim is to learn .NET framework and use the company data source to connect to main application and derive conclusion from it. We used OOP’s (Object Oriented Program) according to industry practice. We develop an application which can able to talk to Company’s data source and apply some complex and efficient manipulation techniques. Our main aim is to develop the application which can establish the connection with database and create the relationship among the existence entities. There are some major problems we face along with the development . Such as:

Performance issues: Entity Framework can sometimes generate inefficient SQL queries, leading to slow query execution times and poor overall application performance.

* Mapping problems: Entity Framework relies on mappings between database tables and C# classes, and errors in these mappings can cause issues with data retrieval and manipulation.
* Concurrency issues: When multiple users are updating the same data simultaneously, conflicts can arise if Entity Framework is not handling concurrency correctly.
* Memory issues: Entity Framework can generate large object graphs, which can cause memory issues in applications that manipulate large amounts of data.
* Database design issues: Entity Framework can have difficulties working with databases that have complex schemas or non-standard data types.
* Debugging issues: Debugging Entity Framework-related issues can sometimes be challenging, as it involves working with complex queries and mappings.

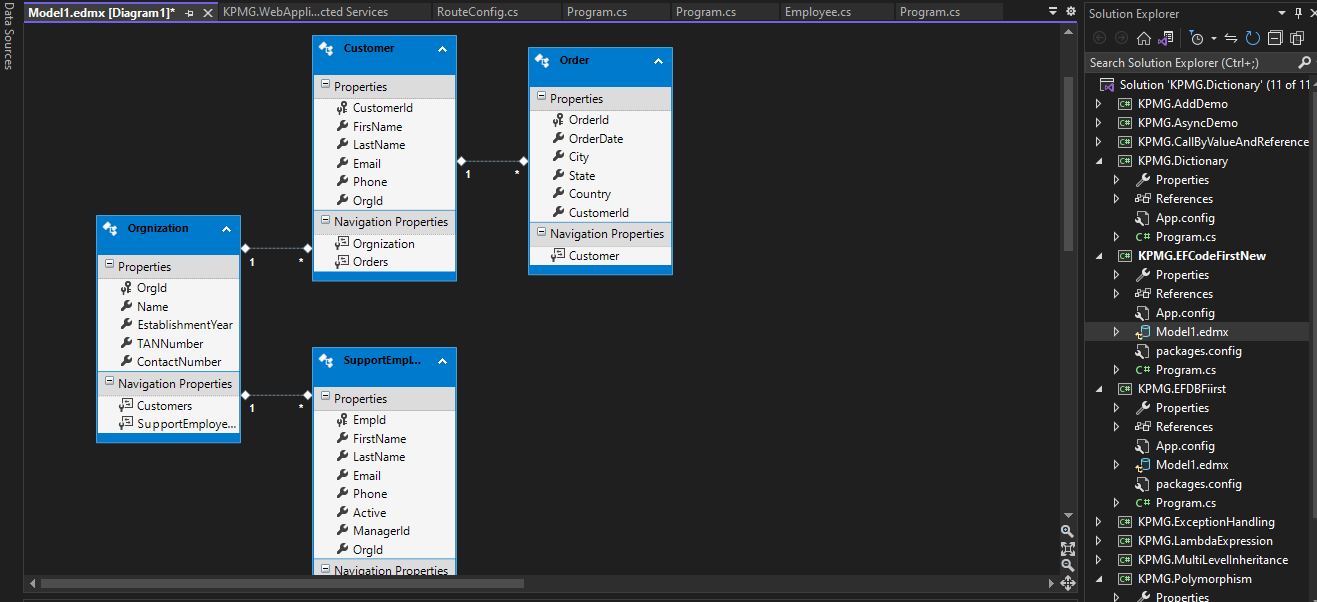
**Objective**

We make sure that we solve the underline concerns and develop a workflow to map end to end connectivity between application and data source. some common objectives include:

* Scalability: The application should be able to handle increasing amounts of data and users without sacrificing performance.
* Maintainability: The application should be easy to modify and maintain over time, with clear separation of concerns between different components.
* Security: The application should be designed to prevent unauthorized access to data and to protect against common security threats such as SQL injection attacks.
* Reliability: The application should be designed to minimize downtime and to handle errors gracefully, with built-in mechanisms for logging and error handling.
* Flexibility: The application should be designed to accommodate changing business requirements, with the ability to easily modify data models and application logic.
* Integration: The application should be designed to integrate with other systems and tools, such as third-party APIs and reporting tools.

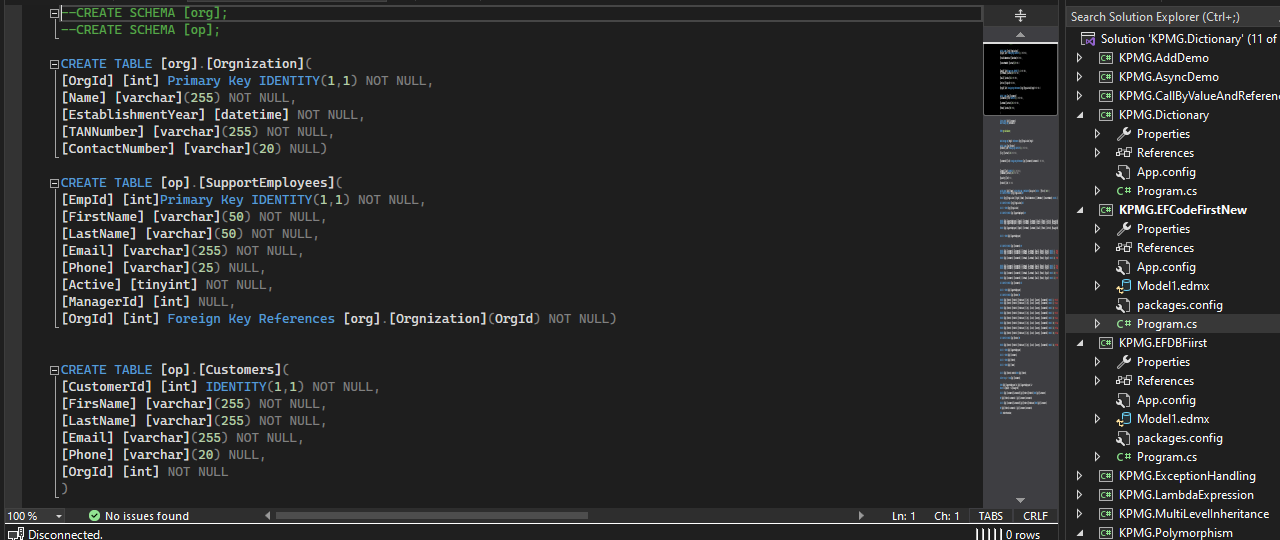
**Solution Provided.**

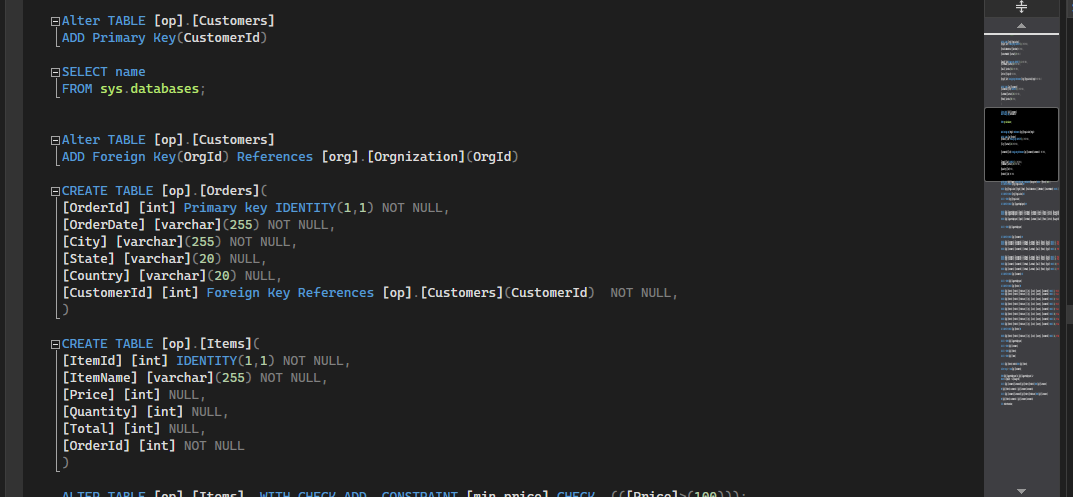
Utilizing database design best practices: Developers can design the database schema using best practices such as normalization and appropriate use of data types to improve data integrity and performance. following are the ER Diagram we established .

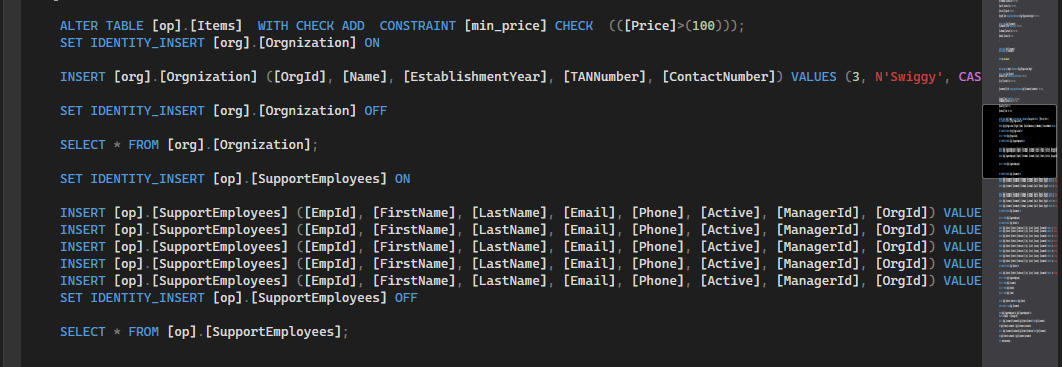


We Implemented the script to connect with the company’s database and perform operation on data-source using Entity framework.

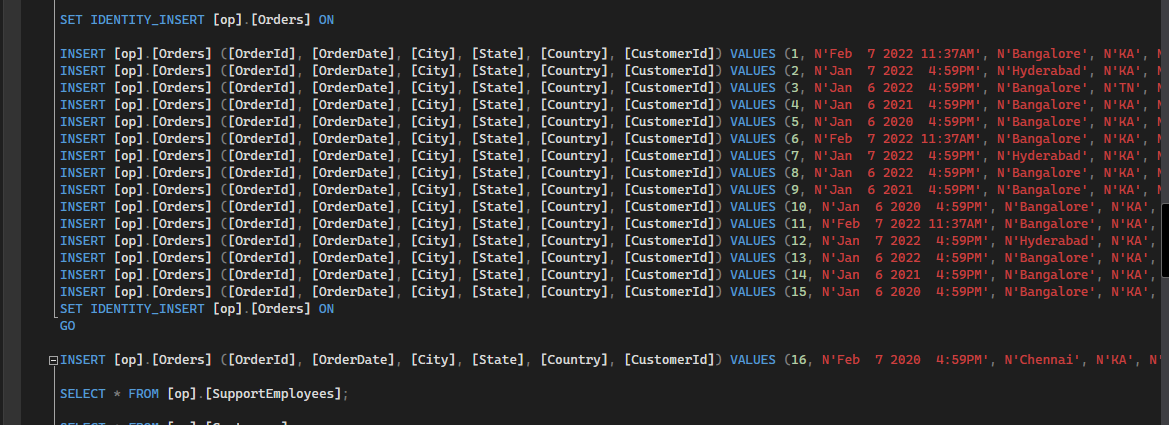
Before implementation we created the test entities to test the Entity framework logic

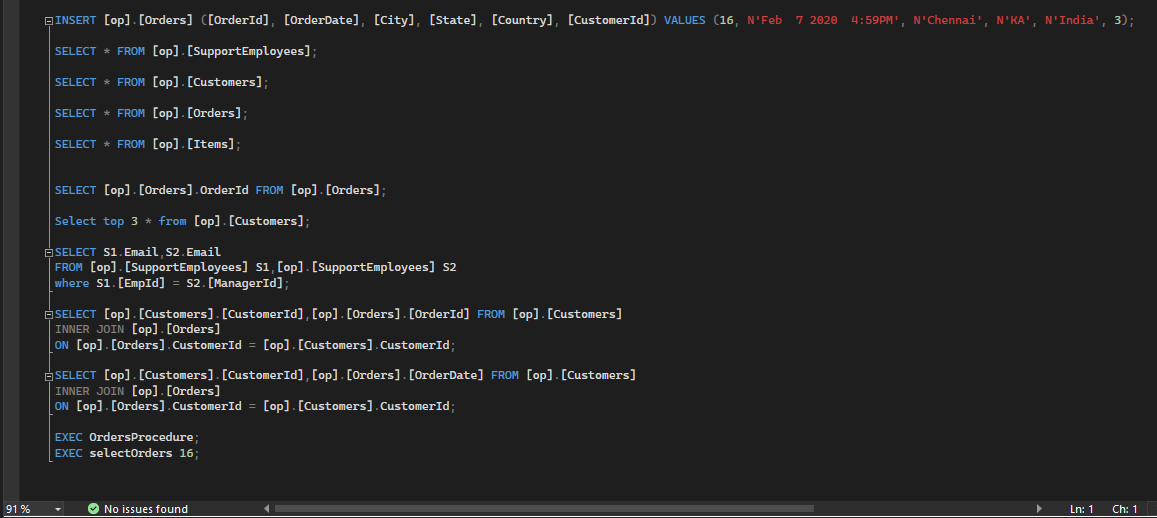


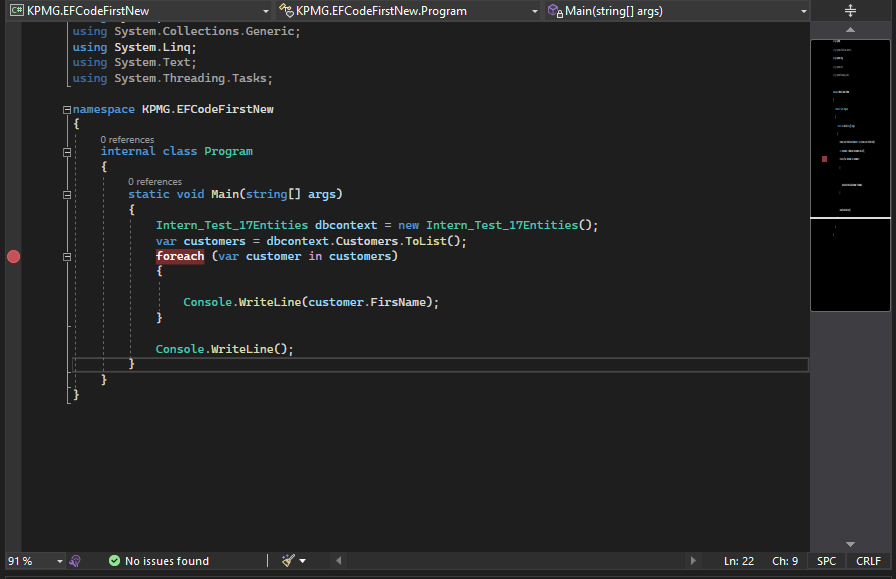












**Note: Disclosure of some code is not allowed under the KPMG confidential policies.**

**Result and Conclusion**

We learn and develop the critical and efficient .NET Application which can able to connect the backend SQL Database with Async operations. And increases the efficiency and processing of Database for large numbers of requests.