.NET Platform Report

1. Introduction

.NET is a free, open-source developer platform developed by Microsoft for building different types of applications. It supports C#, F#, and VB.NET and allows cross-platform development on Windows, macOS, and Linux.

2. .NET Versions Overview

2.1 Classic .NET Framework (2002 - 2019)

• Windows-only.
• Latest major version: .NET Framework 4.8.
• Best for legacy enterprise applications and Windows Forms/WPF apps.
• Installed as part of the Windows OS.

2.2 .NET Core (2016 - 2020)

• Cross-platform: Windows, Linux, and macOS.
• Versions: .NET Core 1.0 → 3.1
• Modular and lightweight.
• CLI-based development and dependency management via NuGet.
• Supports modern APIs and performance improvements.

2.3 .NET 5 and Beyond

• Unified platform from Microsoft (merging .NET Core and .NET Framework).
• Versions:
- .NET 5 (2020)
- .NET 6 (LTS, 2021)
- .NET 7 (2022)
- .NET 8 (LTS, 2023)
• Single SDK, runtime, and base class library.
• Enhanced performance, cloud-native development, Blazor, MAUI support.

3. Namespaces in .NET

3.1 What is a Namespace?

A namespace is a container that holds classes, structs, interfaces, enums, and other namespaces to organize large code projects.

3.2 Common .NET Namespaces

Namespace	Description
System	Core functionalities like base data types, exceptions, console.
System.Collections.Generic	Generic collections like List, Dictionary.
System.Linq	LINQ support for querying collections.
Microsoft.AspNetCore.Mvc	MVC pattern in ASP.NET Core web apps.
System.IO	File and stream handling.
System.Threading.Tasks	Asynchronous and parallel programming support.

4. .NET Core

4.1 Features

- Open-source and modular.
- Command-line interface (CLI).
- Cross-platform.
- Supports containers (Docker).
- Supports Microservices architecture.
- Uses NuGet for package management.

4.2 Project Structure

Typical .NET Core app folders:

- Program.cs – entry point.
- Startup.cs – configuration of services and middleware.
- appsettings.json – configuration settings.

4.3 Advantages Over .NET Framework

- Better performance.
- Cross-platform development.
- Easier deployment.
- Frequent updates.

5. Solutions and Projects in .NET

5.1 What is a Solution (.sln)?

A Solution is a container that holds one or more related Projects. Managed via Visual Studio or CLI. File extension: .sln

5.2 What is a Project?

Represents a single application or library. Contains files like .csproj, source code, resources, and dependencies.

5.3 Common Solution Types

- Web Apps (ASP.NET Core)
- Console Apps
- Class Libraries
- Unit Test Projects
- Blazor Apps
- MAUI (Mobile/Desktop) Projects

5.4 CLI Examples

```
dotnet new sln -n MySolution
dotnet new webapi -n MyWebAPI
dotnet sln add MyWebAPI/MyWebAPI.csproj
dotnet build
dotnet run --project MyWebAPI
```

6. Conclusion

The .NET platform has evolved significantly from the Windows-only .NET Framework to the modern cross-platform .NET Core and unified .NET versions. Namespaces provide organization and modularity, while solutions and projects structure the development process for scalable applications.