Unit I: Introduction to ASP.NET

1. Short note on .NET Framework

The .NET Framework is a software development platform developed by Microsoft for building and running applications. It provides a runtime environment called Common Language Runtime (CLR) and a large class library. The framework supports multiple programming languages such as C#, VB.NET, and F#. It ensures language interoperability, security, memory management, and exception handling. .NET Framework simplifies application development and deployment.

2. Features of .NET Framework

The .NET Framework provides features such as language interoperability, automatic memory management, security, exception handling, rich class libraries, scalability, and versioning support. It supports both desktop and web-based applications. The framework also enables faster development using reusable components.

3. Explain MSIL in detail

Microsoft Intermediate Language (MSIL) is a CPU-independent instruction set generated by .NET language compilers. Source code written in languages like C# is compiled into MSIL. MSIL is later converted into native machine code by the JIT compiler at runtime. This approach provides platform independence and optimized execution. MSIL also helps in enforcing security and type safety.

4. CLR architecture and its features

The Common Language Runtime (CLR) is the execution engine of the .NET Framework. It manages code execution, memory, thread handling, and security. CLR features include garbage collection, exception handling, type safety, and JIT compilation. It ensures reliable and secure application execution.

5. JIT compiler and its types

Just-In-Time (JIT) compiler converts MSIL code into native machine code at runtime. There are three types of JIT compilers: Pre-JIT compiles code at installation time, Normal JIT compiles code method by method at runtime, and Econo JIT compiles only required code to save memory. JIT improves performance and platform compatibility.

6. Short note on CLS

Common Language Specification (CLS) defines a set of rules that languages must follow to ensure interoperability. It allows objects created in one language to be used in another. CLS ensures consistency and compatibility across .NET languages.

7. What is C#?

C# is an object-oriented programming language developed by Microsoft. It is simple, secure, and modern. C# is widely used for developing web, desktop, and enterprise applications. It fully supports .NET Framework features.

8. Short note on CTS

Common Type System (CTS) defines how data types are declared and used in .NET. It ensures that data types are consistent across languages. CTS supports value types and reference types.

9. Managed and Unmanaged code

Managed code runs under the control of CLR and benefits from services like garbage collection and security. Unmanaged code runs outside the CLR and does not get automatic memory management. Managed code is safer and easier to maintain compared to unmanaged code.

10. Assemblies and their structure

An assembly is a compiled code library used for deployment and versioning. It contains MSIL code, metadata, manifest, and resources. Assemblies can be private or shared. They are the building blocks of .NET applications.

11. Namespace

A namespace is used to organize classes and avoid name conflicts. It provides a logical grouping of related classes. Namespaces improve code readability and maintainability.

12. FCL / BCL

Framework Class Library (FCL) or Base Class Library (BCL) is a collection of reusable classes. It provides support for file handling, database access, collections, and networking. It reduces development time and improves productivity.

13. Metadata and Garbage Collection

Metadata stores information about types, methods, and assemblies. Garbage Collection automatically manages memory by freeing unused objects. Together, they improve performance and reliability.

14. Value type vs Reference type

Value types store data directly and are stored in stack memory. Reference types store references and are stored in heap memory. Value types are faster, while reference types provide flexibility.

15. Boxing vs Unboxing

Boxing converts a value type into a reference type. Unboxing converts a reference type back into a value type. Boxing and unboxing impact performance.

16. Difference between ASP and ASP.NET

ASP is a scripting-based technology with limited features. ASP.NET is a fully object-oriented framework. ASP.NET provides better performance, security, and scalability.