CTUC506 – Web Development Using .NET

Complete 5-Mark Descriptive Answers

UNIT I – Introduction to ASP.NET

1. .NET Framework

The .NET Framework is a Microsoft development platform used to build and run applications. It provides a runtime environment known as the Common Language Runtime (CLR) and a large class library called the Framework Class Library (FCL). CLR manages memory, security, and execution of programs, while FCL provides reusable classes for file handling, database access, and networking.

2. Features of .NET Framework

Key features include language independence, automatic garbage collection, code security, interoperability, just-in-time compilation, and a rich class library. These features help developers create secure and efficient applications.

3. MSIL

Microsoft Intermediate Language is an intermediate code generated after compilation. It is platform independent and converted into machine code at runtime by the JIT compiler, ensuring portability and performance.

4. CLR Architecture

The Common Language Runtime manages program execution. Its features include garbage collection, exception handling, thread management, security, and JIT compilation.

5. JIT Compiler

The Just-In-Time compiler converts MSIL into native machine code during execution. Types include Pre-JIT, Normal JIT, and Econo JIT.

6. CLS

The Common Language Specification defines rules that ensure code written in one .NET language can be used by another.

7. C#

C# is an object-oriented programming language used to develop .NET applications. It supports features such as inheritance, polymorphism, and automatic memory management.

8. CTS

The Common Type System defines data types and ensures type safety across all .NET languages.

9. Managed vs Unmanaged Code

Managed code runs under CLR control, while unmanaged code runs directly on the operating system.

10. Assemblies

An assembly is a deployment unit containing MSIL code, metadata, and resources.

11. Namespace

A namespace organizes classes and avoids naming conflicts in large applications.

12. FCL / BCL

The Framework Class Library provides reusable classes for common programming tasks.

13. Metadata and Garbage Collection

Metadata stores information about types, while garbage collection automatically frees unused memory.

14. Value vs Reference Types
Value types store actual data, while reference types store references to data.

15. Boxing and Unboxing
Boxing converts a value type to an object, and unboxing converts an object back to a value type.

16. ASP vs ASP.NET
ASP.NET is object-oriented, faster, and more secure compared to classic ASP.

UNIT II – ASP.NET Application Development

1. Page Life Cycle
The ASP.NET page life cycle includes page request, initialization, loading, postback handling, rendering, and unloading.

2. Page_Load vs Page_Prerender
Page_Load occurs earlier, while Page_Prerender occurs just before rendering.

3. Page Directive
The Page directive defines page settings such as language and code-behind.

4. Web.config
Web.config stores application configuration such as security and connection strings.

5. Server Controls
Server controls include TextBox, Label, Button, CheckBox, RadioButton, Image, and Literal.

6. Literal vs Label
Literal is lightweight and faster, while Label supports formatting.

UNIT III – ADO.NET

1. Connected and Disconnected Architecture
Connected architecture maintains a live database connection, while disconnected architecture works offline using DataSet.

2. ADO.NET Components
Components include Connection, Command, DataReader, DataAdapter, and DataSet.

3. SqlConnection
SqlConnection establishes a connection to SQL Server.

4. SqlCommand
SqlCommand executes SQL queries and stored procedures.

5. DataReader vs DataSet
DataReader is fast and forward-only, while DataSet allows random access.

6. DataAdapter
DataAdapter synchronizes data between DataSet and database.

7. CRUD Operations
CRUD operations allow inserting, updating, deleting, and selecting records.

UNIT IV – MVC Architecture

1. MVC Architecture
MVC separates application logic into Model, View, and Controller.

2. MVC Interaction
Controller handles requests, Model manages data, and View displays output.

3. Routing
Routing maps URLs to controller actions.

4. Web Forms vs MVC
MVC provides more control and better performance.

5. Attribute Routing
Attribute routing defines routes using attributes.

6. View Engine
Razor is a simple and powerful view engine.

7. HTML Helpers
HTML Helpers generate HTML elements.

8. Layouts
Layouts provide a common structure for views.

9. Data Transfer
ViewBag, ViewData, TempData, and Models are used for data transfer.

UNIT V – SOA

1. ABC of WCF
Address, Binding, and Contract define WCF communication.

2. SOAP
SOAP is an XML-based messaging protocol.

3. REST
REST uses HTTP methods for communication.

4. JSON
JSON is a lightweight data exchange format.

5. WSDL
WSDL describes web services.

6. SOAP vs JSON
SOAP is heavy; JSON is lightweight.

7. REST Methods
GET and POST are common REST methods.

8. SOAP Service Design
SOAP services use data contracts.

9. WCF Attributes
ServiceContract, OperationContract, and DataContract define WCF services.

UNIT VI – AJAX and .NET Core

1. Advantages of AJAX

AJAX improves speed and user experience.

2. ScriptManager and UpdatePanel

ScriptManager manages scripts; UpdatePanel enables partial updates.

3. UpdatePanel Properties

ChildrenAsTriggers, Triggers, and UpdateMode control updates.

4. Features of .NET Core

.NET Core is cross-platform, fast, and open source.

5. Characteristics of .NET Core

It is modular, lightweight, and scalable.

6. .NET Core vs .NET Framework

.NET Core is cross-platform, while .NET Framework is Windows-only.