Module 5

1. MVC:

- MVC is a design pattern used to decouple user-interface (view), data (model), and application logic (controller). This pattern helps to achieve separation of concerns.

- Using the MVC pattern for websites, requests are routed to a Controller that is responsible for working with the Model to perform actions and/or retrieve data.

- ASP.NET MVC 5 is a web framework based on Model-View-Controller (MVC) architecture.

- Developers can build dynamic web applications using ASP.NET MVC framework that enables a clean separation of concerns, fast development.

- The Controller chooses the View to display and provides it with the Model. The View renders the final page, based on the data in the Model.

    - Models & data:
    - Create clean model classes and easily bind them to your database. Declaratively define validation rules, using C# attributes, which are applied on the client and server.
    - ASP.NET supports many database engines including SQLite, SQL Server, MySQL, PostgreSQL, DB2 and more, as well as non-relational stores such as MongoDB, Redis, and Azure Cosmos DB.

    - Controllers:
    - Simply route requests to controller actions, implemented as normal C# methods. Data from the request path, query string, and request body are automatically bound to method parameters.

    - Views :
    - The Razor syntax provides a simple, clean, and lightweight way to render HTML content based on your view. Razor lets you render a page using C#, producing fully HTML5 compliant web pages.

2. REST:

- REST is the acronym that stands for: Representational State Transfer. REST is an architectural style of distributed system.

- It is based upon the set of principles that describes how network resources are defined and addressed.

- RESTful services uses HTTP (Hyper Text Transfer Protocol) to communicate. REST system interface with external systems as web resources identified by URIs.

- Http verbs plays a very important role in the Restful Web API. The most common Http verbs are GET, PUT, POST and DELETE and these correspond to CRUD (Create, Read, Update and Delete) operations respectively.

3. Parameter Binding:

- Action methods in Web API controllers can have one or more parameters of different types.
- It can be either primitive type or complex type. Web API binds action method parameters with the URL's query string or with the request body depending on the parameter type.
- By default, if the parameter type is of .NET primitive types such as int, bool, double, string, DateTime, decimal, or any other type that can be converted from string type, then it sets the value of a parameter from the query string.
- And if the parameter type is the complex type, then Web API tries to get the value from the request body by default.

- The following table lists the default rules for parameter binding

HTTP Method	Query String	Request Body
GET	Primitive Type, Complex Type	NA
POST	Primitive Type	Complex Type
PUT	Primitive Type	Complex Type
PATCH	Primitive Type	Complex Type
DELETE	Primitive Type, Complex Type	NA

4. Serialization:

- The process of storing an object to a physical storage is called serialization. The process of reading a serialized object back into memory is deserialization.

- In simple words serialization in C# is a process of storing the object instance to a persistent storage.

- Serialization stores state of objects i.e. member variable values to persostant storage such as a disk. Deserialization is reverse of serialization.

- It is a process of reading objects from a file where they have been stored.

- Serialization in C# is the process of bringing an object into a form that it can be written on stream. It's the process of converting the object into a form so that it can be stored on a file, database, or memory; or, it can be transferred across the network.

- Its main purpose is to save the state of the object so that it can be recreated when needed.

- Deserialization in C# is the reverse process of serialization. It is the process of getting back the serialized object so that it can be loaded into memory. It resurrects the state of the object by setting properties, fields etc.

- **Types**

1 Binary Serialization
2 XML Serialization
3 JSON Serialization

5. Routing:

- Route defines the URL pattern and handler information. All the configured routes of an application stored in RouteTable and will be used by the Routing engine to determine appropriate handler class or file for an incoming request.

- Configure a Route
- Every MVC application must configure (register) at least one route configured by the MVC framework by default. You can register a route in RouteConfig class, which is in RouteConfig.cs under App_Start folder.

- URL Pattern

- The URL pattern is considered only after the domain name part in the URL. For example, the URL pattern *"{controller}/{action}/{id}"* would look like localhost:1234/{controller}/{action}/{id}.
- Anything after "localhost:1234/" would be considered as a controller name.

- Multiple Routes

- You can also configure a custom route using the MapRoute extension method.
- You need to provide at least two parameters in MapRoute, route name, and URL pattern. The Defaults parameter is optional.