**First Web Api using .Net core**

## **1. Concept of RESTful Web Service, Web API & Microservice**

### **🔹 RESTful Web Service**

* REST stands for **Representational State Transfer**.
* It is an **architectural style** for designing networked applications.
* Communicates via **HTTP methods** (GET, POST, PUT, DELETE).
* **Resources** (data) are identified using **URIs**.
* Uses standard HTTP status codes and formats (JSON, XML).

### **🔹 Web API**

* A **Web API** is an interface for building **HTTP services**.
* Allows apps to interact with each other over the web.
* In .NET, **ASP.NET Core Web API** is used to build RESTful services.
* Supports **multiple data formats** like JSON (default), XML, etc.

### **🔹 Microservice**

* A **Microservice** is a small, independent, loosely coupled service that performs a specific business function.
* Microservices architecture:  
  + Each service can be **developed, deployed, and scaled independently**.
  + Communicates via **lightweight protocols** (like REST, gRPC).

## **2. Features of REST Architecture**

| **Feature** | **Description** |
| --- | --- |
| **Stateless** | Each request is independent; no session state on server. |
| **Client-Server** | Separation of concerns – client UI and server logic. |
| **Cacheable** | Responses can be cached to improve performance. |
| **Uniform Interface** | Consistent URI structure and HTTP verbs. |
| **Layered System** | Can use intermediaries like gateways or load balancers. |
| **Supports multiple formats** | Not restricted to XML; supports JSON, text, etc. |

## **3. HttpRequest & HttpResponse**

### **🔹 HttpRequest**

* Represents the **request** sent by the client to the server.
* Contains:  
  + Method (GET, POST, etc.)
  + Headers
  + Query parameters
  + Request body

### **🔹 HttpResponse**

* Represents the **response** returned by the server to the client.
* Contains:  
  + Status code (200 OK, 404 Not Found, etc.)
  + Response body (JSON/XML)
  + Headers

## **4. Types of Action Verbs in Web API**

| **Verb** | **Attribute in Code** | **Purpose** |
| --- | --- | --- |
| GET | [HttpGet] | Read data |
| POST | [HttpPost] | Create new data |
| PUT | [HttpPut] | Update existing data |
| DELETE | [HttpDelete] | Remove data |

## **5. Common HTTP Status Codes in Web API**

| **Status Code** | **Description** | **Method Example** |
| --- | --- | --- |
| 200 OK | Success | return Ok(data) |
| 400 BadRequest | Client error | return BadRequest() |
| 401 Unauthorized | Authentication failed | return Unauthorized() |
| 500 InternalServerError | Server error | return StatusCode(500) |

## **6. Simple Web API with Read/Write Actions**

### **Structure:**

* Controller inherits from ApiController.
* Uses **Action Verbs** like [HttpGet], [HttpPost].
* Returns appropriate **HttpResponses**.

## **7. Configuration Files in Web API**

### **🔹 In .NET Core (5.0, 6.0, 7.0)**

| **File** | **Purpose** |
| --- | --- |
| Program.cs / Startup.cs | Registers services, middleware, routing |
| appsettings.json | Stores configuration (e.g., connection strings, keys) |
| launchSettings.json | Sets up profiles for debugging (port, browser, etc.) |

### **🔹 In .NET Framework (4.5)**

| **File** | **Purpose** |
| --- | --- |
| Web.config | Main config file (app settings, DB strings, etc.) |
| Route.config | Used to define custom route mappings (optional) |
| WebApi.config | Web API-specific route and configuration file |