# .NET Interview Questions – Version Systems (Rajkot)

## 1. .NET Basics

* What is .NET Framework / .NET Core / .NET 8?
* Difference between .NET Framework, .NET Core, and .NET 5/6/7/8.
* Explain CLR, CTS, CLS with examples.
* What are Assemblies and DLLs?

## 2. OOP Concepts in .NET

* Explain Inheritance, Polymorphism, Abstraction, Encapsulation with examples.
* Difference between Abstract Class and Interface.
* What are Constructors? Types of constructors in C#.
* Difference between static and instance members.
* Can we achieve multiple inheritance in C#? If not, how?

## 3. ASP.NET / ASP.NET Core MVC

* What is MVC? Explain the flow in ASP.NET Core MVC.
* Difference between ViewBag, ViewData, and TempData.
* What is Middleware in ASP.NET Core?
* Difference between IEnumerable and IQueryable.
* What is Dependency Injection (DI) in .NET Core?

## 4. .NET Web API

* What is REST API? Difference between REST and SOAP.
* What are HTTP verbs (GET, POST, PUT, DELETE) and how do they map to CRUD operations?
* Difference between IActionResult and ActionResult.
* How to secure Web API? (JWT Token, API Key, OAuth).
* Difference between Synchronous and Asynchronous methods.

## 5. Entity Framework / Database

* What is Entity Framework Core?
* Difference between Code First and Database First.
* What are Migrations in EF Core?
* How to call a Stored Procedure in .NET Core without EF (using ADO.NET)?

## 6. ADO.NET (if no EF is used)

* What are DataReader and DataAdapter?
* Difference between ExecuteScalar, ExecuteReader, ExecuteNonQuery.
* How to handle SQL Injection in ADO.NET?

## 7. Advanced / Practical Questions

* Difference between Task, Thread, and Async/Await in C#.
* What is Garbage Collection in .NET? Explain generations in GC.
* How does .NET manage memory?
* Explain the Repository Pattern.
* How to implement multiple delete in SQL without using SPLIT function?

# CRUD Example in .NET Web API (ADO.NET)

[ApiController]  
[Route("api/[controller]")]  
public class ProductsController : ControllerBase  
{  
 private readonly IConfiguration \_configuration;  
  
 public ProductsController(IConfiguration configuration)  
 {  
 \_configuration = configuration;  
 }  
  
 private SqlConnection GetConnection()  
 {  
 return new SqlConnection(\_configuration.GetConnectionString("DefaultConnection"));  
 }  
  
 // CREATE  
 [HttpPost]  
 public IActionResult Create(Product product)  
 {  
 using var conn = GetConnection();  
 conn.Open();  
 var cmd = new SqlCommand("INSERT INTO Products (Name, Price) VALUES (@n, @p)", conn);  
 cmd.Parameters.AddWithValue("@n", product.Name);  
 cmd.Parameters.AddWithValue("@p", product.Price);  
 cmd.ExecuteNonQuery();  
 return Ok("Product added");  
 }  
  
 // READ  
 [HttpGet]  
 public IActionResult GetAll()  
 {  
 var products = new List<Product>();  
 using var conn = GetConnection();  
 conn.Open();  
 var cmd = new SqlCommand("SELECT \* FROM Products", conn);  
 var reader = cmd.ExecuteReader();  
 while (reader.Read())  
 {  
 products.Add(new Product  
 {  
 Id = (int)reader["Id"],  
 Name = reader["Name"].ToString(),  
 Price = (decimal)reader["Price"]  
 });  
 }  
 return Ok(products);  
 }  
  
 // UPDATE  
 [HttpPut("{id}")]  
 public IActionResult Update(int id, Product product)  
 {  
 using var conn = GetConnection();  
 conn.Open();  
 var cmd = new SqlCommand("UPDATE Products SET Name=@n, Price=@p WHERE Id=@id", conn);  
 cmd.Parameters.AddWithValue("@n", product.Name);  
 cmd.Parameters.AddWithValue("@p", product.Price);  
 cmd.Parameters.AddWithValue("@id", id);  
 cmd.ExecuteNonQuery();  
 return Ok("Product updated");  
 }  
  
 // DELETE  
 [HttpDelete("{id}")]  
 public IActionResult Delete(int id)  
 {  
 using var conn = GetConnection();  
 conn.Open();  
 var cmd = new SqlCommand("DELETE FROM Products WHERE Id=@id", conn);  
 cmd.Parameters.AddWithValue("@id", id);  
 cmd.ExecuteNonQuery();  
 return Ok("Product deleted");  
 }  
}  
  
public class Product  
{  
 public int Id { get; set; }  
 public string Name { get; set; }  
 public decimal Price { get; set; }  
}