Hi I’m Rama Krishna, I’ve 3 years of experience in development of .net WEB apis and having experience with devops toos like Jenkins, SonarQube, Ansible, Docker

And my recent Project engagements are

Global Solution Delivery

It is A comprehensive collection of Competency, Skills, Artifacts, Case Studies, Account Planning, which provides valuable insights to achieve specific objectives and goals of the Organization

I’ve wored as a backend developer and devops engineer in this Project

Project depth: it majorly focus on the employee skill traking and the Organizations documets like Proposals(Client Projects traking win, loss, in-progress), case studies,templates, trainings

.NET Interview Questions

Q1. What do you know about OOPS & what are the pillars of OOPs concept?

Ans. OOPS is a programming pattern centered around the concept of objects, which represent real-world entities.

Here are the **four pillars of OOP:** Encapsulation, Inheritance, Polymorphism, Abstraction

Encapsulation:

Encapsulation is about bundling data (fields) and methods (functions) together in a single unit (class) while restricting direct access to some of the object's details. It provides controlled access through **getters** and **setters.**

In Code: Encapsulation is achieved using **access modifiers** like private, public, protected, and internal.

Inheritance:

Inheritance allows a class (child) to inherit properties and methods from another class (parent). It promotes code reuse and establishes a hierarchical relationship between classes.

In Code: A base class is extended by a derived class using the : operator in .NET.

Polymorphism:

Polymorphism means "many forms." It allows methods in different classes to share the same name but behave differently. It can be achieved through **method overloading** (compile-time) or **method overriding** (runtime).

In Code: Use virtual and override for runtime polymorphism, or simply define multiple methods with the same name but different parameters for compile-time polymorphism.

Abstraction:

Abstraction focuses on showing only essential details to the user while hiding implementation details. It is implemented using **abstract classes** and **interfaces** in .NET

NOTE: 1) We can't create the objects for the Abstract class because in the Abstract class we may have the declared methods.

2) The class which inherits the abstract class that class must implement the Abstract methods available in the Abstract class.

3) We can have the constructor for the abstract class and the Constructor is utilized for Object's Properties Initialization purpose only

Q2. What is the difference between Interface and Abstraction?

Ans.

 Abstract Class: A class that provides a blueprint for other classes. It can have both **abstract methods** (no implementation) and **concrete methods** (with implementation).
Purpose: To define common behaviors for related classes and allow code reuse.

 Interface: A completely abstract type that defines a contract. All methods in an mplicitly **abstract** (no implementation) and **public**.
Purpose: To ensure unrelated classes follow the same set of rules or behaviors.

Note:

The main difference is that an abstract class is a partial blueprint, allowing both defined and undefined methods, while an interface is a complete contract that must be implemented fully. For example, an abstract class like Animal might have a Speak method that subclasses like Dog or Cat can override, but it can also include a predefined Eat method. An interface, on the other hand, defines a strict set of rules, like a IPayment interface that forces all payment providers (e.g., PayPal, Stripe) to implement a ProcessPayment method."

Q3. Explain about middleware and what is the use of USE and RUN in middleware?

Ans. These are piece of codes that we want to run while serving the http request and response.

Middlewares can be added to the pipeline by using the **USE** and **RUN** methods.

USE : The Use method allows chaining of middleware components

RUN : The Run method, on the other hand, is terminal and ends the pipeline by generating a response

Q4. About exception handling / Global Exception Handling?

Ans. To ensures that runtime errors are managed without crashing the application.

By using the try catch blocks and GlobalException Handling.

1) If we use try catch blocks we need to handle exception in each method whereas in GlobalException Handling we write a consolidated handling.

Can we have multiple catch blocks ?

Yes, we can have multiple catch blocks for one try block. In case if we have the multiple catch blocks we need to maintain the Order of Exception classes in the catch block.

we always need to write the child exception class in the first catch blocks followed by Parent Exception classes. Other wise all the exceptions will be catched by the parent exception class catch block

Q5. Explain me about the Dependency Injection?

Ans. Is a design pattern used to create objects using the frame work rather than creating the Objects directly by the class itself.

Q6. What type of Instances/objects we can create using dependency Injection?

Or

Explain me about Singleton, Scoped and Transient

Ans. **<u>Singleton:</u>** in singleton for the entire application one object is created and that single object is used for each request.

<u>Scoped:</u> In scoped for each request individual object is created and that object is utilized for the request only.

<u>Transient:</u> In Transient for each request Individual Object is created and if we inject it multiple times in the same request then individual objects will be created for each Injecion.

Q7. What is the use of Async and Await key words?

Ans. The async and await keywords in .NET are used to enable **asynchronous programming**, which allows applications to perform non-blocking operations.

1) Async key word we use at method level and the await use at instruction level
2) By using this it won't block the main thread to complete the long running instructions
3) it internally uses the tasks
4) if we keep the await at instruction level it returns the either void/object otherwise it will give the task.

Q8. What are the Access modifiers & explain me about the access modifiers?

Ans. Public, private, Protected, Internal

public: The type or member can be accessed by any other code in the same assembly or another assembly that references it.

private: The type or member can be accessed only by code in the same class.

protected: The type or member can be accessed only by code in the same class or struct, or in a class that is derived from that class.

internal: The type or member can be accessed by any code in the same assembly, but not from another assembly.

Q9. Do you know about SOLID Principles and explain?

Ans.

Q10. What are the Http methods/verbs?

Ans. Post, Get, Put, Delete

Post is used for the Create the entities

Get is used to get the data

Put is used to update the Entities

Delete is used to delete the entities

Q11. What is the difference between Post and Put methods/verbs?

Ans. In General we use post for Creation and Put for Update the entities.

We can also use the Put for creation as well and both put and post accepts the body as request objects

Q12. **Do you know about the out and ref key ward and what is the difference between them?**

Ans. We can pass the Out Key variables to the methods and the data which is assigned to Out variable we can use in the caller method, and it won't accepts the Input data.

We can pass the Ref Key variables to the methods and the data which is assigned to Ref variable we can use in the caller method, and it accepts the Input data.

Q13. **How you test the APIs in your system?**

Ans. We can test the APIs using Postman and swagger.

As swagger is integrated with .net will use mostly swagger.

Q14. **What is Authentication and Authorization?**

Ans.

Q15. **What is extension method and can we create extension method?**

Ans. Extension Methods are a way to add new methods to existing classes or interfaces without modifying their source code

Q16. **What is the use of Program and startup class files?**

Ans. the Program class and the Startup class work together to configure and execute an application:

- **Program class**

The entry point of the application, where the Main method is located. The Program class sets up the web host, defines configurations, and links to the Startup class.

- **Startup class**

The control center of the application, where services and the request pipeline are configured. The Startup class includes the ConfigureServices and Configure methods:

- **ConfigureServices**: Configures the app's services, which are reusable components that provide app functionality.

  - **Configure**: Creates the app's request processing pipeline. This method is executed immediately after the ConfigureServices method

Q17. What is delegate and what are different types of delegates available in net?

Ans. **Delegates** are a type in .NET that represents a reference to a method. They are similar to function

Delegates allow methods to be passed as parameters, assigned to variables,

Types of Delegates:

1) **Func ;** Represents a method that returns a value.
2) **Predicate :** Represents a method that returns a boolean value.
3) **Action ;** Represents a method that returns void.

Q18. Do you know about Entity Framework and what is use of it?

Ans. Entity Framework is used to access the data from the databases.

Q19. Do you know about Linq?

Ans. Yes, **LINQ** is a powerful feature in .NET that allows developers to query collections, databases, XML, and other data sources directly within the programming language

Q20. What is Generic and what is the use of generics?

Ans. **Generics** allow you to write classes, methods that work with any data type while maintaining type safety. Instead of specifying a concrete data type (like int, string, etc.), you define a placeholder type that can be replaced with any data type when the code is used.

Generics help reduce redundancy, improve code reusability, and ensure that code is type-safe, preventing runtime errors that could arise from type mismatches.

Q21. What are IEnumerable and IQueryable and what is the difference between them?

Ans. IEnumerable<T> and IQueryable<T> are both interfaces used to represent collections of data. The key difference is in how and where the data is queried

IEnuerable is used for in-memory operations

IQueryable is used to query external data sources.

Q22. What is sealed key word and use of it?

Ans. the sealed keyword is used to indicate that a class cannot be inherited from, or that a method cannot be overridden in derived classes. This keyword is applied to classes and methods to restrict further inheritance or modification.

Q23. What type of approaches you know in entity frame work?

Ans. Database first and Code first approaches

Q24. What is the difference between the Database first and Code first approaches?

Ans. In database first approach first we define the schema later we scaffold the DB and gets the entites.

In Code first approach we define the required entity models and then we migrate the to the DB

Q25. Which approach you have worked on?

Ans. Database first approach.

Q26. How you perform validation for the incoming request objects in .net?

Ans. By using Data annotation and Fluent Validation.

Q27. What is the difference between the Data annotation and Fluent Validation?

Ans. Data annotations are defined above the properties where as in Fluent validation we write a separate class for validations for each Property using Linq

Q28. What is the use of Using Key word ?

Ans. 1) We use Using key word to import the packages to the current class which are already available.

2)While establishing the DB connection also we use Using key word because it automatically closes the DB connection after the using block end. We no need to close the DB connection explicitly.

Q29. What is CORS policy and what is the use of it?

Ans. The CORS policy is used to control how web browsers handle requests that come from a different domain, protocol, or port than the one the web application is running on.

Q30. What are the webapi Return types?

Ans.

IActionResult: This is an interface in ASP.NET Core MVC that defines a contract representing the result of an action method. It allows returning various ActionResult types, including HTTP status codes.

ActionResult<T>: This is a return type for web API controller action.

HttpResponseMessage: Web API converts the return value into an HTTP response message. This gives the developer a lot of control over the response message.

IHttpActionResult: This interface defines a command pattern for creating HTTP responses. The controller returns an IHttpActionResult, and the pipeline invokes it to create the response.

Q31. What is Dispose and finalize methods?

Ans.

both Dispose and Finalize are methods that clean up unmanaged resources**:**

Dispose : A developer explicitly calls this method in the code when a class is no longer needed.

Finalize : The garbage collector automatically calls this method when an object goes out of scope

Q32. What Is the use Web Api?

Ans.

allows software applications to communicate and exchange data over the internet:

Web API allows access to service data from web browsers, mobile apps, and other devices

Q33. What is string and string Builder?

Ans.

String:

A String instance is immutable, which means, we cannot change it after it was created. If we perform any operation on a String it will return a new instance (creates a new instance in memory) instead of modifying the existing instance value.

StringBuilder: StringBuilder is mutable, that is, if we perform any operation on StringBuilder it will update the existing instance value and it will not create a new instance.

StringBuffer : StringBuffer is mutable, and it will holds an initial buffer size init when we create a stringbuffer object, if we modify the value it will increases the buffer accordingly.

Q34. Diff between Array and ArrayList?

Ans. Array is fixed size and need to have the same type values.

ArrayList is dynamic in size and can hold any type of values.

Primary key: primary key is uniqely identifying a record in a table in all data bases, there can be only one primary key in a table and also primarykey does not accept the duplicate/null values.

Foreign Key: In all DB's if we want to establish the relationship b/w tables then we are using referencial integrity constraints foreign key.

one table forgin key must belong to the another table primary key and also these two columns must belongs to same data type. always forgin key values, always based on primary key values only.

Difference between deleting and Truncate?

Ans

whenever we are using delete from tablename then automatically deleted data internally stored in buffer we can also get back this data using rollback.

whenever we are using truncate table tablename then all rows are perminently deleted, we can not get it back this data by using rollback, because **truncate** is a **DDL** command and also **DDL** command transactions are automatically commited.

Explain the difference between UNION and UNION ALL?

The **UNION** and **UNION ALL** operators combine the results of two or more SELECT queries, but they handle duplicate rows differently.

UNION: Removes duplicate rows, so the result will only have unique rows. This makes UNION a bit slower because it needs extra processing to find and remove duplicates.

UNION ALL: Includes all rows from the queries, even if they are duplicates. It is faster because it doesn’t check for duplicates.

Explain the difference between WHERE and HAVING clauses?

1. Use WHERE to filter individual rows and HAVING to filter grouped data.
2. WHERE works before GROUP BY, and HAVING works after.
3. WHERE cannot use aggregate functions, but HAVING can.

What is the difference between GROUP BY and PARTITION BY?

Group by clause is used to arrange similar data items into a set of logical groups. when ever we are using group by clause database server selects similar data items from a table column and then reduces no of data item in each group.

PARTITION BY, on the other hand, is used in window functions. It divides the result set into partitions (subsets of rows), but unlike GROUP BY, it doesn't reduce the number of rows in the output. It allows you to perform calculations over subsets of rows, like calculating running totals or averages with in each partition.