**Why Json files?**

Developers often prefer JSON because it simplifies the exchange of data between different technologies. JSON has smaller file sizes and faster data transmission. XML tag structure is more complex to write and read and results in bulky files. JSON is safer than XML.

**Why RabbtMQ**

RabbitMQ is well-suited for use cases that require **instant messaging, task queues, and communication between microservices**. It provides reliable message delivery, supports various messaging patterns, and offers rich message routing and processing features. RabbitMQ has low latency. It sends thousands of messages per second. **Kafka** has real-time transmission of up to millions of messages per second. RabbitMQ supports a broad range of languages and legacy protocols.

**What is called REST API?**

Full form of REST API is Representational State Transfer Application Programming Interface more commonly known as REST API web service. It means when a RESTful API is called, the server will transfer a representation of the requested resource's state to the client system.

**What is the difference between a REST API and an API?**

APIs can refer to any type of interface that enables communication between different systems. REST APIs are a specific type of API that adheres to the constraints of the REST architecture. RESTful APIs use HTTP requests to interact with data, while traditional APIs can use a variety of protocols.

**Why NodeJS (alternate – C#.net, python, AngularJS)**

Node. JS is a JavaScript runtime environment that allows you to easily build scalable network applications. Node JS allows developers to write both client-side and server-side code in JavaScript, which means that you can reuse code across both sides of your app. This can help to reduce development time and make your codebase more maintainable\

* High Performance.
* Scalability.
* Easy to Learn.
* Reduces Loading time using caching.
* Improves Response time and boosts performance.
* Large Community Support.
* Cost-Effective.
* Extensibility.

**Why Python?**

Python is commonly used for developing websites and software, task automation, data analysis, and data visualisation. Since it's relatively easy to learn, Python has been adopted by many non-programmers, such as accountants and scientists, for a variety of everyday tasks. **Advantages/Benefits of Python**

* Presence of Third Party Modules: ...
* Extensive Support Libraries: ...
* Open Source and Community Development: ...
* Learning Ease and Support Available: ...
* User-friendly Data Structures: ...
* Productivity and Speed:

**What to use .NET Framework or .NET Core?**

.NET Core is to be used for the server application when –

1. There is cross-platform need.
2. Targeting microservices or using Docker containers.
3. Need of high performance and scalable systems.
4. Side by side need of .NET versions per application.

.NET Framework is to be used for the server application when –

1. Application is to be built to run only on Windows.
2. If application uses .NET framework technologies not available for .NET Core.
3. Applications that are already running on .NET Framework.

Conclusion-

If the application has to be built from scratch, choose .Net Core but, if it is already existing good in the .Net Framework and you are not planning to migrate it, then it is a better one.

**Below is a table of differences between .Net Core and .Net Framework:**

| **BASED ON** | **.NET Core** | **.NET Framework** |
| --- | --- | --- |
| Open Source | .Net Core is an open source. | Certain components of the .Net Framework are open source. |
| Cross-Platform | Works on the principle of “build once, run anywhere”. It is compatible with various operating systems — Windows, Linux, and Mac OS as it is cross-platform. | .NET Framework is compatible with the windows operating system. Although, it was developed to support software and applications on all operating systems. |
| Application Models | .Net Core does not support desktop application development and it rather focuses on the web, windows mobile, and windows store. | .Net Framework is used for the development of both desktop and web applications as well as it supports windows forms and WPF applications. |
| Installation | .NET Core is packaged and installed independently of the underlying operating system as it is cross-platform. | .NET Framework is installed as a single package for Windows operating system. |
| Support for Micro-Services and REST Services | .Net Core supports the development and implementation of micro-services and the user has to create a REST API for its implementation. | .Net Framework does not support the development and implementation of microservices but it supports the REST API services. |
| Performance and Scalability | .NET Core offers high performance and scalability. | .Net Framework is less effective in comparison to .Net Core in terms of performance and scalability of applications. |
| Compatibility | .NET Core is compatible with various operating systems — Windows, Linux, and Mac OS. | .NET Framework is compatible only with the Windows operating system. |
| Android Development | .NET Core is compatible with open-source mobile application platforms, i.e. Xamarin, through the .NET Standard Library. Developers use Xamarin’s tools to configure the mobile app for specific mobile devices such as iOS, Android, and Windows phones. | .NET Framework does not support any framework for mobile application development. |
| Packaging and Shipping | .Net Core is shipped as a collection of Nugget packages. | All the libraries of .Net Framework are packaged and shipped together. |
| Deployment Model | Whenever the updated version of .NET Core gets initiated; it is updated instantly on one machine at a time, thereby getting updated in new directories/folders in the existing application without affecting it. Thus, .NET Core has a good and flexible deployment model. | In the case of .Net Framework, when the updated version is released it is first deployed on the Internet Information Server only. |
| Support | It has support for microservices. | It does not support creation and microservices. |
| WCF Services | It has no support for WCF services. | It has excellent support for WCF services. |
| Rest APIs | Supports Rest APIs | It also supports REST Services. |
| CLI Tools | .NET Core provides light-weight editors and command-line tools for all supported platforms. | .Net Framework is heavy for Command Line Interface and developers prefer to work on the lightweight Command Line Interface. |
| Security | .NET Core does not have features like Code Access Security. | Code access security feature is present in .NET Framework. |