WHY RESTFUL API

A REST API(also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. REST stands for representational state transfer.

One of the key advantages of REST APIs is that they provide a great deal of flexibility. Data is not tied to resources or methods, so REST can handle multiple types of calls and return different data formats.
This flexibility allows developers to build an API that meets your needs while also meeting the needs of very diverse customers

There are 6 key constraints to think about when considering whether a RESTful API is the right type of API for your needs:

- Client-Server: This constraint operates on the concept that the client and the server should be separate from each other and allowed to evolve individually.
- Stateless: REST APIs are stateless, meaning that calls can be made independently of one another, and each call contains all of the data necessary to complete itself successfully.
- Cache: Because a stateless API can increase request overhead by handling large loads of incoming and outbound calls, a REST API should be designed to encourage the storage of cacheable data.
- Uniform Interface: The key to the decoupling client from server is having a uniform interface that allows independent evolution of the application without having the application’s services, or models and actions, tightly coupled to the API layer itself.

- Layered System: REST APIs have different layers of their architecture working together to build a hierarchy that helps create a more scalable and modular application.
- Code on Demand: Code on Demand allows for code or applets to be transmitted via the API for use within the application.

Why you need REST API

- Scalability. This protocol stands out due to its scalability. Thanks to the separation between client and server, a product may be scaled by a development team without much difficulty.
- Flexibility and portability. With the indispensable requirement for data from one of the requests to be properly sent, it is possible to perform a migration from one server to another or carry out changes on the database at any time. Front and back can therefore be hosted on different servers, which is a significant management advantage.
- Independence. With the separation between client and server, the protocol makes it easy for developments across a project to take place independently. In addition, the REST API adapts at all times to the working syntax and platform. This offers the opportunity to use multiple environments while developing.
- REST is compatible with a variety of messaging formats, including XML, YAML, and JSON. However SOAP only uses XML. This makes REST more flexible and lightweight for certain use cases.
- Caching is easier with REST