**Rest API Best Practices**

**What is Rest API ?**

**Ans:** A REST API is an application programming interface that adheres to the constraints of REST architectural style and enables interaction with RESTful web services. Interconnected networks make up the web. A web service is a set of open protocols and standards used for exchanging data between client-server applications

**File Folder Structure :**

* For Monolithic Application MVC (Model, View, Controller)
* For Rest API MC (Model, Controller )

**JSON Best practices**

**What is JSON?**

**Ans:** JSON stands for JavaScript Object Notation.JSON is a lightweight format for storing and transporting data.JSON is often used when data is sent from a server to a web page.JSON is "self-describing" and easy to understand

**Uses of JSON:**

**Ans:**

* It is used while writing JavaScript based applications that includes browser extensions and websites.
* JSON format is used for serializing and transmitting structured data over network connection.
* It is primarily used to transmit data between a server and web applications.
* Web services and APIs use JSON format to provide public data.

**Characteristics of JSON:**

**Ans:**

* JSON is easy to read and write.
* It is a lightweight text-based interchange format.
* JSON is language independent.

**Understanding JSON Structure :**

**Ans:** Data objects are separated by commas · Curly braces {} hold objects · Square brackets [] hold arrays

**JSON - Data Types:**

**Ans:**

In JSON, values must be one of the following data types:

* a string
* a number
* an object (JSON object)
* an array
* a boolean
* null

**Request Best Practices**

**When to use the GET method?**

**Ans:**

* GET is used to request something from a server with less amount of data to pass.
* When nothing should change on the server because of your action.
* When request only retrieves data from a web server by specifying parameters
* Get method only carries request url & header not request body.

**When to use the POST method?**

**Ans:**

* POST should be used when the server state changes due to that action.
* When a request needs its body, to pass a large amount of data.
* When want to upload documents , images , video from client to server

**Request Body:**

**Ans:**

* Request body should be structured in JSON Array/ Object pattern
* Request body hold multipart/ form-data like images, audio, video etc
* Request body should not hold any auth related information.
* Request body should associated with specific request data model, setter getter can used for this

**Request Header:**

**Ans:**

* Request header should carry all security related information, like token, auth etc.
* Only string **Key:Pair** value is allowed for header .
* Request header should provide user agent information of client application.
* If necessary CSRF/ XSRF should provide via header.

**Response Best Practices**

**Response Header:**

**Ans:**

* Provide proper http response status code.
* Provide proper content type, file type if any.
* Provide cache status if any.
* Authentication token should provide via response header.
* Only string data is allowed for response header.
* Provide content length if any.
* Provide response date and time.

**Response Body:**

**Ans:**

* Avoid providing response status, code, message via response body
* Use JSON best practices for JSON response body.
* For single result, can use String, Boolean directly.
* Provide proper JSON encode-decode before writing JSON Body.

**Response Cookies:**

**Ans:**

* A Restful API may send cookies just like a regular Web Application that serves HTML
* Avoid using response cookies as it is violate stateless principle.
* If required use cookie encryption, decryption and other policies

**Web Security Practices**

**Security Practices May Varies :**

**Ans:**

* May varies from application to application
* May varies from developer to developer
* May varies from environment to environment
* May varies from use case to use case

**But we have to know :**

**Ans:**

* The best practices
* Know about the security layers
* Security placement

**Output Header:**

**Ans:**

* Provide proper http response status code.
* Provide proper content type, file type if any.
* Provide cache status if any.
* Authentication token should provide via response header.
* Only string data is allowed for response header.
* Provide content length if any.
* Provide response date and time.
* Follow request-response model described before.

**Output Body:**

**Ans:**

* Avoid providing response status, code, message via response body
* Use JSON best practices for JSON response body.
* For single result, can use String, Boolean directly.
* Provide proper JSON encode-decode before writing JSON Body.
* Follow discussion on JSON described before.

**Request Rate limit- Throttling :**

**Ans:** We need to make sure our APIs are running as efficiently as possible. Otherwise, everyone using your database

will suffer from slow performance. Performance isn’t the only reason to limit API requests, either. API limiting, which also known

as rate is limiting, is an essential component of Internet security, as DoS attacks can tank a server with unlimited API requests.

Rate limiting also helps make your API scalable. If your API blows up in popularity, there can be unexpected spikes in traffic,

causing severe lag time.

**CSRF/XSRF Protection :**

**Ans:**

Cross-site request forgery attacks (CSRF or XSRF for short) are used to send malicious requests from an authenticated user to a web application.

* Use request-response header to pass CSRF token
* CSRF token should be unique for every session
* For self API CSRF token works well.

**JWT (JSON WEB TOKEN):**

**Ans:**

* Compact and self-contained way for securely transmitting information between parties as a JSON object.
* Information can be verified and trusted because it is digitally signed.