**---------------------------------------------------------------**

**$ Rest API Backend Development Notes $**

**---------------------------------------------------------------**

**-> What is REST api?**

**Ans:** Representational State Transfer (REST) is an architectural style that defines a set of constraints to be used for creating web services.

**-> 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

**-> What is REST api?**

**Ans:**

* JSON is lightweight and self-describing.
* JSON is an open standard data-interchange format.
* JSON originated from JavaScript.
* JSON is easy to read and write.
* JSON is language independent.
* JSON supports data structures such as arrays and objects.

**-> What is the JSON structure?**

**Ans:**

* Curly braces {} hold objects
* The data are in key, value pairs
* Square brackets [] hold arrays
* Each data element is enclosed with quotes if it‘s a character, or without quotes if it is a numeric value
* Commas are used to separate pieces of data

**-> how many data types support json?**

**Ans:**

* string – Literal text that’s enclosed in quotes.
* number – Positive or negative integers or floating point numbers.
* object – A key, value pair enclosed in curly braces
* array – A collection of one or more JSON objects.
* boolean – A value of either true or false with no quotes.
* null – Indicates the absence of data for a key value pair, represented as “null” with no quotes.

**-> JSON String escape**

**Ans:**

The following characters are reserved characters and can not be used in JSON and must be properly escaped to be used in strings.

* Backspace to be replaced with \b
* Form feed to be replaced with \f
* Newline to be replaced with \n
* Carriage return to be replaced with \r
* Tab to be replaced with \t
* Double quote to be replaced with \"
* Backslash to be replaced with \\

**-> What is the Request Response Model?**

**Ans:** Every request is a combination of request header, body and request URL.

Body, Header, URL Parameter

**-> What is API Throttling?**

**Ans:** API throttling allows you to control the way an API is used. Throttling allows you to set permissions as to whether certain API calls are valid or not. Throttles indicate a temporary state, and are used to control the data that clients can access through an API. When a throttle is triggered, you can disconnect a user or just reduce the response rate. You can define a throttle at the application, API or user level.

**-> What is REST api?**

**Ans:** One of the most recognizable characteristics of REST is the predominant use of nouns in URIs. Restful URIs should not indicate any kind of CRUD (Create, Read, Update, and Delete) functionality. Instead, REST APIs should allow you to manipulate a resource.

Example : /users or /users/:id getUserInfo

**-> API Response Best Practices?**

**Ans:**

*Response Header:*

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

Response Body:

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

Response Cookies:

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

**-> 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 POSTmethod?**

**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

**-> REST api controller best practice?**

**Ans:**

* The controllers should always be as clean as possible. We shouldn’t place any business logic inside it.
* Controllers should be responsible for accepting http request
* Consider API versioning
* Use async/await if at all possible.
* Follow solid principles to manage controller classes.
* Mention which method is responsible for GET() and which for POST().
* Controller should be only responsible for calling the model, return response , redirect to action etc.

**-> REST api middleware best practice?**

**Ans:**

* Use to implement API key, user-agent restriction, CSRF, XSRF security, token based API authentication.
* Use to implement API request rate limit.
* Logging of incoming HTTP requests.
* Redirecting the users based on requests.
* Middleware can inspect a request and decorate it, or reject it, based on what it finds.
* Middleware is most often considered separate from your application logic.
* Middleware gives you enough freedom to create your own security mechanism.

**-> What is Request Rate Limiting?**

**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.

**-> What is CSRF 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.

**-> What is a Browser User Agent?**

**Ans:** User agent is a request header property, describing client identity like operating system, browser details, device details etc.Moreover every web crawler like Google crawler, Facebook crawler has specific user-agent name.

**-> what is the bearer authentication system?**

**Ans:** Bearer authentication (also called token authentication) is an HTTP authentication scheme that involves security tokens called bearer tokens, passed through request-response header. In General JSON Web Tokens JWT used for this

purposes.

**-> What is 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.

Json Web Token mostly uses authorization system, Data Exchange etc.