CRUD Viva Questions and Answers (Node.js, Express.js, REST APIs, MySQL

Node.js Questions:

1. What is a callback function in Node.js?

Answer: A callback function is a function that is passed as an argument to another function and is executed once the asynchronous operation is completed. It allows Node.js to handle multiple tasks without blocking the execution of the program.

2. What are streams in Node.js?

Answer: Streams are used to handle large volumes of data in chunks, instead of loading everything into memory at once. They are a powerful tool for managing I/O operations efficiently. There are four types of streams in Node.js: Readable, Writable, Duplex, and Transform.

3. What is the difference between npm and npx?

Answer:

- npm (Node Package Manager) is used to install and manage packages for Node.js.
- npx is a tool that comes with npm to execute binaries from packages without installing them globally.

4. What is nodemon?

Answer: Nodemon is a development tool that automatically restarts a Node.js application whenever file changes in the directory are detected. It helps in improving development speed by eliminating the need to manually restart the server.

5. What is the purpose of 'require' in Node.js?

Answer: The require() function is used to load modules, JSON files, or local files into your application. It allows you to use functionality from other files or libraries in your Node.js code.

6. What is the event-driven architecture in Node.js?

Answer: Node.js uses an event-driven architecture, meaning that actions in the program (such as requests or I/O operations) trigger events. These events are handled by event listeners or callback functions, allowing for non-blocking asynchronous operations.

7. What are some core modules in Node.js?

Answer: Some core modules in Node.js are:

- http (for creating servers)
- fs (file system operations)
- path (working with file paths)
- events (event-driven programming)
- url (URL parsing)
- os (operating system-related utilities)

Express.js Questions:

1. What is Express.js?

Answer: Express.js is a web application framework built on top of Node.js. It simplifies the process of building web servers and APIs by providing utilities to handle routing, middleware, and HTTP requests.

2. What is middleware in Express.js?

Answer: Middleware functions are functions that have access to the request, response, and the next middleware function in the request-response cycle. They can modify the request or response objects or end the request-response cycle. Common middleware includes express.json() and express.static().

3. What is routing in Express? How do you handle routing in Express?

Answer: Routing in Express refers to defining how the server responds to client requests to specific endpoints. You can handle routing using methods like app.get(), app.post(), app.put(), and app.delete().

4. What is the role of app.use and app.listen in Express.js?

Answer:

- app.use() is used to define middleware functions or to serve static files.
- app.listen() is used to start the server and listen for incoming requests on a specified port.

Postman/Thunder Client Questions:

1. What is Postman/Thunder Client used for?

Answer: Postman and Thunder Client are tools used for testing APIs. They allow you to send various types of HTTP requests (GET, POST, PUT, DELETE) to an API and examine the response.

2. How do you use Postman to test a POST request?

Answer: In Postman, select the POST method, enter the API endpoint URL, go to the "Body" tab, select raw, and choose JSON format to input your data. Then click "Send" to see the response.

3. How do you use Postman to test a GET request?

Answer: In Postman, select the GET method, enter the API URL, and click "Send." Postman will show the response from the server.

4. How do you use Postman to test a PUT request?

Answer: In Postman, select the PUT method, enter the API endpoint, and in the "Body" tab, input the data to update. Click "Send" to see the response.

5. How do you use Postman to test a DELETE request?

Answer: In Postman, select the DELETE method, enter the API URL, and click "Send" to test the deletion operation.

REST API Questions:

1. What is a REST API?

Answer: A REST (Representational State Transfer) API is an architectural style for building web services. It uses HTTP methods (GET, POST, PUT, DELETE) to perform CRUD operations on resources, which are typically represented in JSON format.

2. What are the main principles of REST?

Answer: The main principles of REST include:

- Stateless: Each request from a client must contain all information needed to process it.
- Client-Server: The client and server are separate, allowing them to evolve independently.
- Uniform Interface: A standardized way of interacting with resources (e.g., using HTTP methods).

3. What is the difference between PUT and PATCH in REST APIs?

Answer:

- PUT is used to update a resource completely.
- PATCH is used to partially update a resource.

4. What is the difference between GET and POST in REST APIs?

Answer:

- GET is used to retrieve data from the server.
- POST is used to send data to the server (such as creating or submitting a new resource).

5. What are various status codes?

Answer: Some common HTTP status codes are:

- 200 OK: The request was successful.
- 201 Created: A resource has been created successfully.
- 400 Bad Request: The request could not be understood or was missing required parameters.
- 404 Not Found: The requested resource could not be found.
- 500 Internal Server Error: A generic error occurred on the server.

MySQL and CRUD Operations Questions:

1. How do you connect to MySQL from Node.js?

Answer: You can use the mysql2 package in Node.js to connect to MySQL:

```
const mysql = require("mysql2");
const connection = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "password",
```

```
  database: "dbname"
});
connection.connect();
```

2. What are CRUD operations?

Answer: CRUD operations refer to the four basic operations that can be performed on data:

- Create: Insert new data (e.g., INSERT in SQL).
- Read: Retrieve existing data (e.g., SELECT in SQL).
- Update: Modify existing data (e.g., UPDATE in SQL).
- Delete: Remove data (e.g., DELETE in SQL).