**1. What is Node.js and how does it work?**

* Node.js is a runtime environment that allows JavaScript to run outside the browser.
* It uses the V8 JavaScript engine.
* It handles multiple requests asynchronously using event-driven, non-blocking I/O.

**2. Advantages of using Node.js**

* Fast performance due to V8 engine.
* Asynchronous and non-blocking.
* Single programming language (JavaScript) for frontend and backend.
* Scalable and efficient.
* Huge ecosystem (NPM packages).

**3. What is Express.js?**

* Express.js is a minimal and flexible Node.js web application framework.
* It simplifies building APIs and web servers.

**4. Middleware functions in Express**

* Functions that have access to request, response, and next() function.
* Used for tasks like logging, authentication, parsing JSON, etc.

Example:

app.use((req, res, next) => {

console.log('Request URL:', req.url);

next();

});

**5. How do you handle errors in Express?**

* Use error-handling middleware with four arguments (err, req, res, next).

Example:

app.use((err, req, res, next) => {

res.status(500).send('Something broke!');

});

**6. Difference between app.use() and app.get()/app.post()**

|  |  |
| --- | --- |
| **Method** | **Purpose** |
| app.use() | Middleware for all HTTP methods |
| app.get() | Handle GET requests |
| app.post() | Handle POST requests |

**7. Purpose of app.listen() in Express**

* Starts a server and listens for incoming connections on the specified port.

Example:

app.listen(3000, () => {

console.log('Server running on port 3000');

});

**8. HTTP methods corresponding to CRUD operations**

|  |  |
| --- | --- |
| CRUD Operation | HTTP Method |
| Create | POST |
| Read | GET |
| Update | PUT/PATCH |
| Delete | DELETE |

**9. What is Multer and why do we use it?**

* Multer is a middleware for handling multipart/form-data.
* Used for uploading files (e.g., images) in Node.js applications.

**10. What is bcrypt and why is it used?**

* bcrypt is a library to hash passwords.
* Used to securely store passwords instead of saving plain text.

**11. How to hash a password using bcrypt**

Example:

const bcrypt = require('bcrypt');

const saltRounds = 10;

const hashedPassword = await bcrypt.hash('myPassword', saltRounds);

**12. What is JWT and how does it work?**

* JWT (JSON Web Token) is a secure way to transmit information between parties as a JSON object.
* It is used for authentication.
* The server generates a token and sends it to the client. The client sends it back for protected routes.

**13. Parts of a JWT**

* Header: Contains token type and hashing algorithm.
* Payload: Contains claims (user data).
* Signature: Used to verify the token hasn't been altered.

Structure:

Header.Payload.Signature