Express.js Interview Questions and Answers

What is Express.js?

Express.js is a lightweight and fast web application framework for Node.js, used to build web applications and APIs.

Why use Express.js instead of Node.js alone?

Express.js simplifies handling requests, responses, routing, and middleware, reducing boilerplate code compared to pure Node.js.

What are the main features of Express.js?

- Middleware support
- Routing
- Template engines
- Error handling
- HTTP utilities

How does Express.js handle HTTP requests?

Express uses a routing system to map HTTP methods (GET, POST, etc.) to specific endpoints.

What is middleware in Express.js?

Middleware functions process requests before they reach the final route handler.

What are the types of middleware in Express.js?

- Built-in middleware
- Third-party middleware
- Application-level middleware
- Router-level middleware
- Error-handling middleware

What is routing in Express.js?

Routing determines how an application responds to client requests based on URLs and HTTP methods.

What is the difference between app.use() and app.get()?

app.use() applies middleware to all requests, while app.get() handles only GET requests for a specific route.

What is the purpose of the next() function in middleware?

It passes control to the next middleware function in the request-response cycle.

What is a route parameter in Express.js?

A route parameter is a dynamic segment in a URL used to capture values (e.g., /user/:id).

What is the difference between query parameters and route parameters?

Query parameters are in the URL after ? (e.g., /search?q=express), while route parameters are part of the URL pattern (/user/:id).

How can you handle errors in Express.js?

By using error-handling middleware with four parameters (err, req, res, next).

What are template engines in Express.js?

Template engines like EJS, Pug, and Handlebars generate dynamic HTML pages.

What is the difference between synchronous and asynchronous middleware in Express.js?

Synchronous middleware executes sequentially, while asynchronous middleware (using async/await or Promises) may execute independently.

What is Express Router?

Express Router allows organizing routes in separate files or modules to keep code modular.

What is the purpose of the req and res objects in Express?

- req contains details about the HTTP request.
- res provides methods to send a response.

What is the difference between res.send() and res.json()?

- res.send() sends a response of any type (string, object, buffer, etc.).
- res.json() sends a JSON response with appropriate headers.

How can you serve static files in Express.js?

Using express.static() middleware to serve assets like images, CSS, and JavaScript.

How does Express handle file uploads?

Express does not handle file uploads directly; libraries like multer are used.

How do you redirect in Express.js?

Using res.redirect() to forward users to another URL.

What is CORS in Express.js?

Cross-Origin Resource Sharing (CORS) controls which external domains can access API resources.

How do you enable CORS in an Express app?

Using the cors middleware (npm install cors).

What is the difference between local and global middleware?

- Local middleware applies to specific routes.
- Global middleware applies to all routes.

How do you handle authentication in Express.js?

Using strategies like JWT, OAuth, or session-based authentication.

What are sessions in Express.js?

Sessions store user-related data on the server using express-session.

What is the difference between sessions and cookies?

- Sessions store data on the server.
- Cookies store data in the browser.

What is helmet in Express.js?

Helmet is a middleware that enhances security by setting various HTTP headers.

How does rate limiting work in Express.js?

Using middleware like express-rate-limit to control request frequency per user.

What is the purpose of Morgan in Express.js?

Morgan is a logging middleware for monitoring HTTP requests.

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

- PUT updates the entire resource.
- PATCH updates only specific fields.

How can you optimize Express.js applications?

- Use caching
- Optimize database queries
- Implement gzip compression
- Minimize middleware usage

What is clustering in Express.js?

Clustering allows running multiple instances of an Express app to utilize multi-core CPUs.

How do you handle heavy loads in Express.js?

- Load balancing
- Using Redis for caching
- Implementing rate limiting

What is process management in Express.js?

Tools like PM2 help manage, monitor, and restart Express processes.

How can you handle timeouts in Express.js?

Using req.setTimeout() or third-party middleware like connect-timeout.

How do you prevent SQL Injection in Express.js?

- Use parameterized queries
- Use ORM/ODM libraries like Sequelize or Mongoose

What is CSRF, and how do you prevent it?

Cross-Site Request Forgery (CSRF) tricks users into making unwanted requests; prevent it using CSRF tokens (csurf package).

How do you prevent XSS in Express.js?

- Escape user input
- Use security middleware like helmet

What is a Content Security Policy (CSP)?

CSP restricts resources like scripts and styles to prevent code injection attacks.

How do you secure Express sessions?

- Use secure cookies
- Store sessions in a database
- Rotate session keys regularly

How does Express differ from Koa?

- Koa has no built-in middleware, while Express includes routing and middleware support.
- Koa uses async/await natively.

What is a singleton pattern in Express.js?

A design pattern ensuring a single instance of a module is used across the app.

What is dependency injection in Express.js?

A technique where dependencies (services, configs) are injected into a function/module rather than being created inside it.

How do you log errors in Express.js?

Using logging libraries like Winston or Bunyan.

What are WebSockets, and how do they work with Express.js?

WebSockets enable real-time communication, often integrated using socket.io with Express.

What is graceful shutdown in Express.js?

Ensuring all requests finish before stopping the server using process.on('SIGTERM', callback).

What is the difference between development and production mode in Express.js?

- Development mode has detailed logs and debugging enabled.
- Production mode focuses on performance and security.

How do you configure environment variables in Express.js?

Using the dotenv package to manage sensitive information.

What is the use of app.locals in Express.js?

It stores variables accessible in templates and throughout the app.

What is the future of Express.js?

While Express remains popular, newer frameworks like Fastify and Koa offer alternatives with better performance and modern features.