1 What is the difference between Node.js and Express?

- **Node.js** is a JavaScript runtime built on Chrome's V8 engine. It allows you to run JavaScript outside the browser.
- **Express** is a minimal and flexible web framework built on top of Node.js that simplifies server creation and routing.

---

2 Explain how routing works in Express internally.

Express routing works based on:

- HTTP Method (GET, POST, etc.)
- URL Path

Internally:

- Express stores routes in a stack.
- When a request comes, it matches method + path.
- If matched → executes callback.
- If not → moves to next middleware.

---

3 What is the purpose of express.json() middleware?

express.json() parses incoming JSON request bodies.

Without it:

req.body → undefined

With it:

app.use(express.json());

---

4 Different HTTP Methods in REST APIs

Method	Purpose
GET	Fetch data
POST	Create data
PUT	Update entire resource
PATCH	Update partial resource
DELETE	Delete resource

---

5 Difference between req.params, req.query, and req.body

Property	Used For	Example
req.params	URL parameters	/user/:id
req.query	Query string	?page=2
req.body	Data sent in request body	POST data

---

6 How to create a dynamic route?

Example:

```
app.get("/user/:id", (req, res) => {
  res.send(`User ID: ${req.params.id}`);
});
```

---

7 What happens if two routes have same path & method?

Example:

```
app.get("/", handler1);
app.get("/", handler2);
```

The first matching route runs.
If next() is used → second route may execute.

---

8 How does route mounting work?

Route mounting organizes routes using Router.

Example:

```
const userRouter = express.Router();

app.use("/users", userRouter);
```

---

9 What is middleware?

Middleware is a function that:

- Has access to req, res, next
- Runs before sending response
- Can modify request or response

Example:

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

  console.log("Request received");

  next();

});
```

---

10 **Properties inside req object**

Common properties:

- req.params
- req.query
- req.body
- req.headers

- req.method
- req.url
- req.path

---

11 How to handle 404 errors?

```
app.use((req, res) => {
  res.status(404).send("Page Not Found");
});
```

---

12 Difference between res.send() and res.json()

res.send()	**res.json()**
Sends any type	Sends JSON only
Auto detects type	Converts to JSON

Example:

```
res.send("Hello");
res.json({ name: "Karthi" });
```

---

13 MVC Pattern in Express

Flow:

Route → Controller → Model → Response

Example:

routes/user.js

```
router.get("/", userController.getUsers);
```

controllers/userController.js

```
exports.getUsers = (req, res) => {
  res.json({ message: "Users fetched" });
};
```

---

14 Difference between PUT and PATCH

PUT	**PATCH**
Updates full resource	Updates partial resource
Replaces data completely	Modifies specific fields

---

15 Full Request-Response Lifecycle in Express

1. Client sends request
2. Express receives request
3. Middleware runs
4. Route matches
5. Controller executes logic
6. Database interaction (optional)
7. Response sent using res
8. Connection closed

1.Basic Server Setup:

```
import express from "express";

const app = express();

app.listen(3000, () => {

  console.log("Server is running on http://localhost:3000");

});
```

2.Simple GET Route:

```
import express from "express";

const app = express();
```

```
app.get("/", (req, res) => {

  res.send("Welcome to my server");

});

app.listen(3000, () => {

  console.log("Server is running on http://localhost:3000");

});
```