



Top

ExpressJs

Concepts



What's **Express.js**

Think of **Express.js** as your web server's “**traffic manager**” – it directs **requests** where they need to go.

It's a **minimal** yet powerful **Node.js** framework that simplifies **web** and **API** development.

Analogy: Node.js is the engine, Express.js is the steering wheel.



What's Middleware

Middleware = checkpoints between request and response.

They can log, verify, modify, or stop a request.

Real-world analogy: Security guard checking tickets before letting you in.

Tech analogy: Like user Authorization. User can't see some pages if not logged in



What's Routing



Routing decides “**who handles what.**”

Each route maps an **HTTP method and URL to a function.**

Example:

```
app.get("/home", (req,res)=>{  
  res.send("Welcome!") })
```

Pro tip: Use `express.Router()` to organize routes like a clean neighborhood.

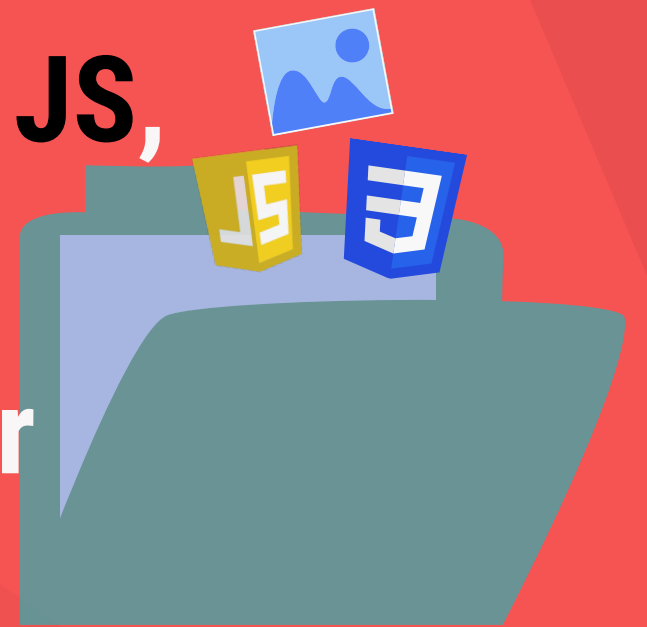
Serve Static Files

Want to serve your **HTML, CSS, JS,**
or **images?**

Just plug in after Initializing your
app:

```
app.use(express.static("Folder"))
```

In your route `res.sendFile()`



Request & Response

req is what the user sends — like an order form.

res is your reply — the finished product.

They're the heart of every Express interaction.

Body Parsing

When data comes in (like **JSON** or **form inputs**), Express helps **unpack** it.

Use:

```
app.use(express.json())
```

```
app.use(express.urlencoded({ extended: true}))
```

Analogy: It's like a waiter unwrapping your food order so the chef can understand it.

Don't Forget To Use CORS

next() Middleware

next() passes control to the next middleware.

Rough Idea:

The User send request and you have many middlewares like for Checking Verified email, loggedin

You need to use **next()** after each **middleware** complete its **designated task**.

Analogy: Think of it like a relay race — the baton (request) keeps moving until it reaches the finisher (response).

Forget **next()**? Your app might freeze mid-race.

Connecting Database

Express can talk to any DB using **ORMs** like:

- **Mongoose** → MongoDB
- **Prisma** → PostgreSQL

Analogy: Express is the waiter taking your query to the kitchen (database) and bringing back the result.



Express.js is simple,
but mastering these
concepts is crucial.