🕐 SESSION 1 (45 min) — Introduction to Templating Engines + EJS

1. ⭐ What is a Templating Engine?

A templating engine lets you:

- Write **HTML + dynamic JS variables**
- Reuse layouts, headers, footers
- Avoid writing long `res.send("<h1>...</h1>")` code
- Build clean full-stack applications

Templating engines generate **HTML on the server** using templates.

Popular engines:

- **EJS** (looks like HTML + JS)
- **Pug** (indentation-based syntax)
- Handlebars
- Mustache

Today we focus on **EJS** and **Pug**.

---

2. ⭐ Setting Up EJS

Install EJS:

```
npm install ejs
```

Configure Express to use EJS:

```
const express = require("express");
const app = express();

app.set("view engine", "ejs");
```

Create folder structure:

```
project/
 ├── views/
 │     ├── index.ejs
 │     ├── about.ejs
 ├── app.js
 └── package.json
```

---

3. ⭐ EJS Basic Example

Create `views/index.ejs`:

```
<!DOCTYPE html>
<html>
  <head>
    <title>EJS Example</title>
  </head>
  <body>
    <h1>Welcome <%= username %>!</h1>
    <p>Your role is: <%= role %></p>
  </body>
</html>
```

Pass data from Express:

```
app.get("/", (req, res) => {
  res.render("index", { username: "Dielli", role: "Instructor" });
});
```

Your browser shows:

```
Welcome Dielli!
Your role is: Instructor
```

---

4. ⭐ Passing Arrays & Loops

Example:

```
app.get("/students", (req, res) => {
  const students = ["Ardit", "Nora", "Erion"];
  res.render("students", { list: students });
});
```

`students.ejs`:

```
<h2>Student List</h2>

<ul>
  <% list.forEach(student => { %>
    <li><%= student %></li>
  <% }) %>
</ul>
```

🧩 Mini Exercise (EJS)

Students should:

- Create a route `/blog`
- Pass title, author, paragraph
- Render it inside a `.ejs` view

Example:

```
res.render("blog", {
  title: "Why I Love JavaScript",
  author: "Student Name",
  body: "JavaScript is amazing because..."
});
```

🕑 SESSION 2 (45 min) — Pug + Comparing Engines + Passing Data

1. ⭐ Setting Up Pug

Install:

```
npm install pug
```

Configure Express:

```
app.set("view engine", "pug");
```

---

2. ⭐ Pug File Example

Folder:

```
views/
 ├── home.pug
```

`home.pug`:

```
doctype html
html
  head
    title Pug Example
  body
    h1 Welcome #{username}
```

```
    p Your role is #{role}
```

Pass data:

```
app.get("/pug-home", (req, res) => {
  res.render("home", { username: "Dielli", role: "Instructor" });
});
```

---

3. ⭐ Pug Loop Example

JS:

```
app.get("/pug-students", (req, res) => {
  res.render("students", { list: ["Ardit", "Nora", "Erion"] });
});
```

students.pug:

```
h2 Student List
ul
  each student in list
    li= student
```

🟧 EJS vs Pug (Quick Comparison)

Feature	EJS	Pug
Syntax	HTML + JS	Minimal, indentation-based
Easy to learn	⭐⭐⭐⭐	⭐⭐⭐
Readability	Very readable	Compact
Best for	Beginners, frontend devs	Minimalist code lovers
Flexibility	Very high	High

Students usually prefer **EJS** because it looks like normal HTML.

🧠 Recap Discussion (Ask Students)

1. What problem do templating engines solve?
2. How do you pass variables from Express to a template?
3. What's the main difference between Pug and EJS syntax?
4. Which engine feels more natural to you? Why?
5. When would you use loops and conditionals in templates?

🏠 Homework

Students must build a **“Mini Blog Preview”** using EJS:

Requirements:

- Route: `/posts`
- Pass an array of 3 blog posts:

```
[
  { title: "Post 1", author: "Student", body: "..." },
  { title: "Post 2", author: "Student", body: "..." },
  { title: "Post 3", author: "Student", body: "..." }
]
```

- Render all posts in a clean HTML layout
- Show title + author + preview text
- Use a loop inside EJS
- Add a link back to home