POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

DIKTISAINTEK BERDAMPAK KAMPUS INOVASI



Membangun Backend Modern dengan Express JS

Workshop Pemrograman WEB

Yunia Ikawati

Teknik Informatika-PENS

Review Node.js

- Dulu, JavaScript hanya bisa berjalan di dalam browser (seperti Chrome atau Firefox) untuk membuat animasi web.
- Node.js adalah lingkungan waktu jalan (runtime environment) yang memungkinkan kita menjalankan JavaScript di komputer/server kita, di luar browser.
- Berkat Node.js, JavaScript sekarang bisa digunakan untuk membuat sistem Backend (mengelola database, keamanan, file, dll).

Apa Itu Express JS?

- **Express JS** adalah kerangka kerja (*framework*) untuk Node.js.
- **Tujuan Utama:** Mempermudah dan mempercepat pembuatan server dan *REST API*.
- **Kenapa menggunakan Express?**
  - **Minimalis:** Tidak memaksakan banyak aturan ketat.
  - **Cepat:** Berjalan sangat ringan.
  - **Satu Bahasa:** Menggunakan JavaScript (sama dengan *Frontend*).
  - **Populer:** Standar industri, komunitasnya sangat besar (mudah cari jawaban di StackOverflow).

Persiapan Awal (Setup Project)

1. Pengecekan Kesiapan (Environment Check)

- Sebelum mulai *coding*, kita harus memastikan "mesin"-nya sudah siap. Buka Command Prompt/PowerShell (di Windows), lalu ketik:

```
PS C:\Users\yunia> node -v
v22.14.0
PS C:\Users\yunia> npm -v
11.3.0
```

2. Membuat "Rumah" untuk Proyek

```
PS C:\Users\yunia> mkdir belajar-express


    Directory: C:\Users\yunia


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         4/19/2026   2:09 AM                belajar-express


PS C:\Users\yunia> cd belajar-express
```

Persiapan Awal (Setup Project)

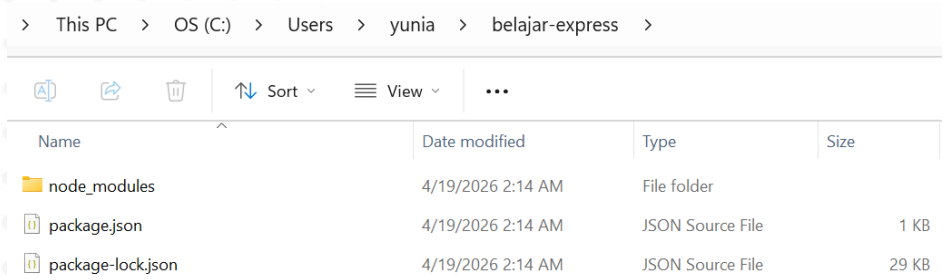
3. Inisialisasi Proyek (npm init)

- Setelah berada di dalam folder yang benar, saatnya membuat identitas untuk aplikasi kita.

```
PS C:\Users\yunia\belajar-express> npm init -y
```

4. Menginstal Express JS (npm install)

```
PS C:\Users\yunia\belajar-express> npm install express
```



Membuat file server.js atau index.js

1. Buka Folder Proyek di VS Code dengan mengetikkan < **code .** > pada terminal

```
PS C:\Users\yunia\belajar-express> code .
```

2. Buat File `server.js` dan simpan

```
// 1. Panggil library Express
const express = require('express');

// 2. Buat aplikasi Express
const app = express();
const port = 3000;

// 3. Buat Rute (Route) Dasar
app.get('/', (req, res) => {
  res.send('Halo, ini server Express pertama saya!');
});

// 4. Jalankan Server
app.listen(port, () => {
  console.log(`Server berjalan di http://localhost:${port}`);
});
```

Membuat file server.js atau index.js

3. Jalankan Server melalui terminal VS Code: **node server.js**

Klik menu **Terminal** di bagian atas (atau **View** > **Terminal**).

Memahami Routing

Bagaimana Server Merespons URL?

- *Routing* adalah cara kita menentukan apa yang terjadi saat *user* atau *Frontend* mengakses URL tertentu dengan metode tertentu.

4 Metode HTTP Utama (CRUD):

- GET: Membaca/Mengambil data (Read).
- POST: Membuat data baru (Create).
- PUT: Memperbarui data (Update).
- DELETE: Menghapus data (Delete).

- **Contoh Struktur Routing:**

```
app.METHOD('/alamat-url', (request, response) => { ... })
```

Request & Response (req, res)

Tanya Jawab antara Client dan Server

- Setiap kali rute diakses, Express memberikan dua objek penting: req dan res.
- **req (Request):** Data/permintaan yang masuk dari Client.
  - req.query: Untuk URL parameters (misal: ?nama=Budi).
  - req.params: Untuk URL dinamis (misal: /user/:id).
  - req.body: Untuk data yang dikirim melalui form/POST.
- **res (Response):** Jawaban yang dikirim balik ke Client.
  - res.send(): Mengirim teks biasa atau HTML.
  - res.json(): Mengirim data dalam format JSON (Sangat penting untuk API).

Praktik Membuat REST API Sederhana

Edit file server.js dan tambahkan data berikut:

server.js

```
const express = require('express');
const app = express();
const port = 3000;

// Rute dasar (yang sudah kita buat di Slide 5)
app.get('/', (req, res) => {
  res.send('Halo, ini server Express pertama saya!');
});

// ========================================
// 1. Kita buat "Database Sementara" menggunakan Array berisi Object
const mahasiswa = [
  { id: 1, nama: "Budi", jurusan: "Informatika" },
  { id: 2, nama: "Siti", jurusan: "Sistem Informasi" },
  { id: 3, nama: "Andi", jurusan: "Teknik Komputer" }
];

// 2. Kita buat Endpoint baru: GET /api/mahasiswa
app.get('/api/mahasiswa', (req, res) => {
  // res.json() digunakan khusus untuk mengirim data dalam format JSON
  res.json({
    status: "sukses",
    jumlah_data: mahasiswa.length,
    data: mahasiswa
  });
});
// ========================================

// Menyalakan server
app.listen(port, () => {
  console.log(`Server berjalan di http://localhost:${port}`);
});
```

Praktik Membuat REST API Sederhana

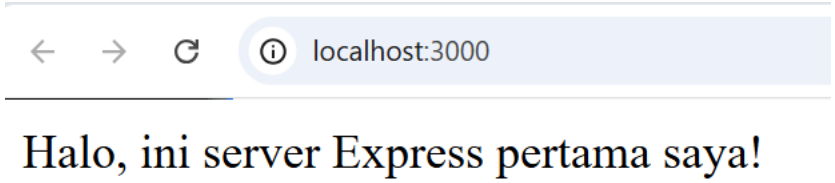
Langkah 2: Matikan dan Nyalakan Ulang Server (Restart):

1. Arahkan kursor ke terminal di bagian bawah VS Code
2. Klik di dalam area terminal tersebut, lalu tekan tombol **Ctrl + C**
3. Nyalakan lagi servernya dengan mengetik perintah: **node server.js**

```
PS C:\Users\yunia\belajar-express> ^C
PS C:\Users\yunia\belajar-express> node server.js
Server berjalan di http://localhost:3000
```

Langkah 3: Tes API di Browser

1. Buka *browser* (Chrome, Edge, dll).
2. Ketikkan alamat endpoint yang baru saja kita buat: **http://localhost:3000/api/mahasiswa**



Latihan: Membuat API "Daftar Mata Kuliah"

1. Buat folder proyek baru dan inisialisasi npm init -y.
2. Install express.
3. Buat file app.js dan jalankan server di port 8080.
4. Buat array data statis minimal berisi 3 mata kuliah (id, nama_mk, sks).
5. Buat sebuah rute GET /api/matakuliah yang akan merespons dengan data array tersebut dalam format JSON.
6. Tes hasilnya menggunakan browser.

Tugas Eksplorasi Mandiri

Untuk memperdalam pemahaman kalian tentang materi hari ini, silakan eksplorasi kode dari *developer* profesional. Pilih dan akses **minimal 2 dari 3** tautan referensi *project* Express JS berikut ini:

1. **MDN Web Docs - Local Library Project** *(Tutorial resmi dari Mozilla untuk membuat aplikasi manajemen perpustakaan)*
   👉 Tautan: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs
2. **Express.js Official GitHub Examples** *(Kumpulan puluhan mini-project dengan berbagai fitur dari pembuat Express langsung)* 👉 Tautan: https://github.com/expressjs/express/tree/master/examples
3. **FreeCodeCamp - Node & Express Handbook** *(Panduan dasar dan contoh proyek microservices yang ramah pemula)*
   👉 Tautan: https://www.freecodecamp.org/news/the-express-handbook/

Instruksi Tugas:

1. Buka dan baca dokumentasi atau kode repositori dari 2 tautan yang kalian pilih.
2. Pelajari struktur foldernya dan perhatikan bagaimana mereka menulis kode *routing* dan mengatur respons server.
3. Praktikkan (tulis ulang / *re-code*) salah satu contoh *project* atau fitur dari referensi tersebut di Visual Studio Code kalian masing-masing sampai berhasil dijalankan tanpa *error*. (Catatan: Jangan sekadar *copy-paste*, biasakan mengetik agar otot jari dan otak kalian terbiasa dengan sintaks JavaScript!)
4. Tunjukkan hasilnya! Simpan tangkapan layar (*screenshot*) yang membuktikan bahwa server/API kalian berhasil berjalan
5. Tujuan tugas ini adalah melatih kalian membaca dokumentasi resmi dan melihat standar penulisan kode di dunia industri.

Kesimpulan

- Node.js membuat JavaScript bisa berjalan di *server*.
- Express JS mempermudah kita membuat server dan API dengan Node.js.
- *Routing* mengatur jalan masuk data (GET, POST, dll).
- Aplikasi *Frontend* nantinya akan mengonsumsi JSON yang disediakan oleh Express JS.