

Praktikum 4 - Matakuliah Pilihan 1 (Web)

Program Studi: Teknik Informatika

Lakukan praktikum dibawah ini, dan buat screenshot untuk pembuktian mengerjakan setiap poin dengan mengisi tabel dibawah, kemudian tunjukkan hasil akhir dari men-share repository github yang telah dibuat.

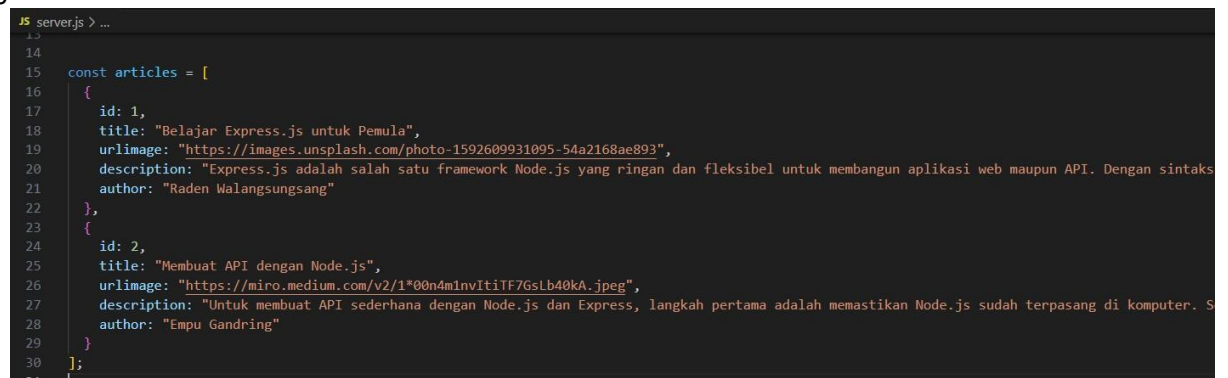
A. Membuat Endpoint test untuk artikel dengan Express.js

1. Lakukan langkah yang sama seperti praktikum 3, menyiapkan express js (npm init, npm install express, buat file [server.js](#))
2. Buat file [server.js](#) dengan port di 8001, tidak perlu ada router. Seperti pada gambar dibawah ini



```
JS server.js X
JS server.js > ...
1 // server.js
2 const express = require("express");
3 const app = express();
4 const port = 8001;
5
6 app.listen(port, () => {
7   console.log(`Server berjalan di http://localhost:${port}`);
8 });
9
```

3. Pada file [server.js](#) tambahkan array-objects dengan nama articles seperti pada gambar dibawah ini



```
JS server.js > ...
13
14
15 const articles = [
16   {
17     id: 1,
18     title: "Belajar Express.js untuk Pemula",
19     urlimage: "https://images.unsplash.com/photo-1592609931095-54a2168ae893",
20     description: "Express.js adalah salah satu framework Node.js yang ringan dan fleksibel untuk membangun aplikasi web maupun API. Dengan sintaks",
21     author: "Raden Walangsungang"
22   },
23   {
24     id: 2,
25     title: "Membuat API dengan Node.js",
26     urlimage: "https://miro.medium.com/v2/1*00n4m1nvItiTF7GsLb40kA.jpeg",
27     description: "Untuk membuat API sederhana dengan Node.js dan Express, langkah pertama adalah memastikan Node.js sudah terpasang di komputer. S",
28     author: "Empu Gandring"
29   }
30 ];
31
```

4. Untuk urlimage, dan description bisa cari berita dari web media lainnya, tidak harus sama.
5. Kemudian tambahkan endpoints baru (/api/test/getarticle) untuk mengakses artikel

```

31
32 // Endpoint GET /api/test/getarticle
33 app.get("/api/test/getarticle", (req, res) => {
34   res.json({
35     status: "success",
36     data: articles
37   });
38 });

```

6. Jalankan [server.js](#) dan pastikan bisa mengakses /api/test/getarticle

B. Membuat Service mengambil data dari API Endpoints di Next.js

1. Buat folder baru Latihan4 didalamnya install [next.js](#) seperti pada praktikum 2 Ketik :
`npx create-next-app@latest praktikum2`
2. Buka terminal baru, Jalankan dengan `npm run dev`.
3. (Pastikan [server.js](#) pada point A tidak dihentikan/distop)
Port 8001 untuk API / Express
Port 8000 untuk Frontend / Nextjs
4. Buat folder articles di dalam src/app sehingga menjadi /src/app/articles
5. Buat folder [page.js](#) kosong di dalamnya sehingga menjadi /src/app/articles/[page.js](#)
6. Buat folder services di dalam folder articles sehingga menjadi /src/app/articles/services, kemudian di dalamnya buat file getarticles.js
7. Pastikan folder hierarchy seperti pada gambar dibawah ini



8. Pada file [getarticles.js](#) buat kode program seperti ini, file ini berfungsi untuk mengambil data ke api endpoints dengan menggunakan nextjs

```

src > app > articles > services > JS getarticles.js > ...
1  export async function getArticles() {
2    try {
3      const res = await fetch("http://localhost:8001/api/test/getarticle");
4      if (!res.ok) {
5        throw new Error("Gagal mengambil data artikel");
6      }
7      const data = await res.json();
8      return data.data; // sesuai struktur di server.js { status, data }
9    } catch (error) {
10     console.error("Error fetching articles:", error);
11     return [];
12   }
13 }

```

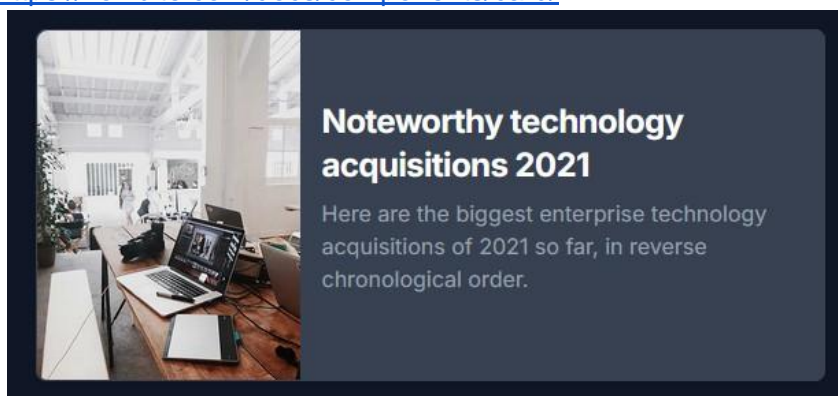
9. Pastikan API bisa diakses lewat browser.
10. Kemudian pada `src/app/articles/page.js` ketik kode program seperti dibawah ini

```
src > app > articles > JS page.js > ArticlesPage > articles.map() callback
1  import { getArticles } from "../services/getarticles";
2
3  export default async function ArticlesPage() {
4    const articles = await getArticles()
5
6    return (
7      <div className="p-4">
8        <h1 className="text-xl font-bold mb-4">Daftar Artikel</h1>
9        <ul className="space-y-2">
10         {articles.map((article) => (
11           <li key={article.id} className="border p-3 rounded">
12             <h2 className="font-semibold">{article.title}</h2>
13             <img src={article.urlimage} alt={article.title} className="w-40 my-2" />
14             <p>{article.description}</p>
15             <small className="text-gray-500">By: {article.author}</small>
16           </li>
17         )]}
18       </ul>
19     </div>
20   );
21 }
```

11. Kode program diatas untuk menampilkan artikel yang diambil dari API services dengan nama `getArticles()`
12. Setelah kedua program selesai dibuat, cek di browser dengan mengunjungi link <http://localhost:8000/articles>

C. Experiment 1

1. Ganti namafile untuk `page.js` dan `getarticles.js` dengan ekstensi `*.tsx` sehingga menjadi `page.tsx` dan `getarticles.tsx`, kemudian apakah yang terjadi?
2. Ambil component flowbite Horizontal Card pada link berikut <https://flowbite.com/docs/components/card/>



3. Modifikasi `src/app/articles/page.tsx` dengan mengganti baris kode dengan flobite horizontal components.

```
<li key={article.id} className="border p-3 rounded">
  ...
</li>
```

D. Experiment 2

1. Gunakan Dynamic components seperti pada Praktikum 2. Dengan menempatkan flowbite ke folder components di dalam /src/app/articles/components
2. Kemudian pada page.tsx cukup dipanggil saja dengan di dalam panggil nama komponennya dan mengirim parameter seperti pada gambar dibawah ini

```
{articles.map((article) => (  
  <li key={article.id} className="border p-3 rounded">  
    <Card img={article.urlimage} title={article.urlimage} desc={article.description } author={article.author } />  
  </li>  
))
```


3. Pastikan dynamic component tersimpan di folder components dibawah articles sehingga strukturnya seperti ini:




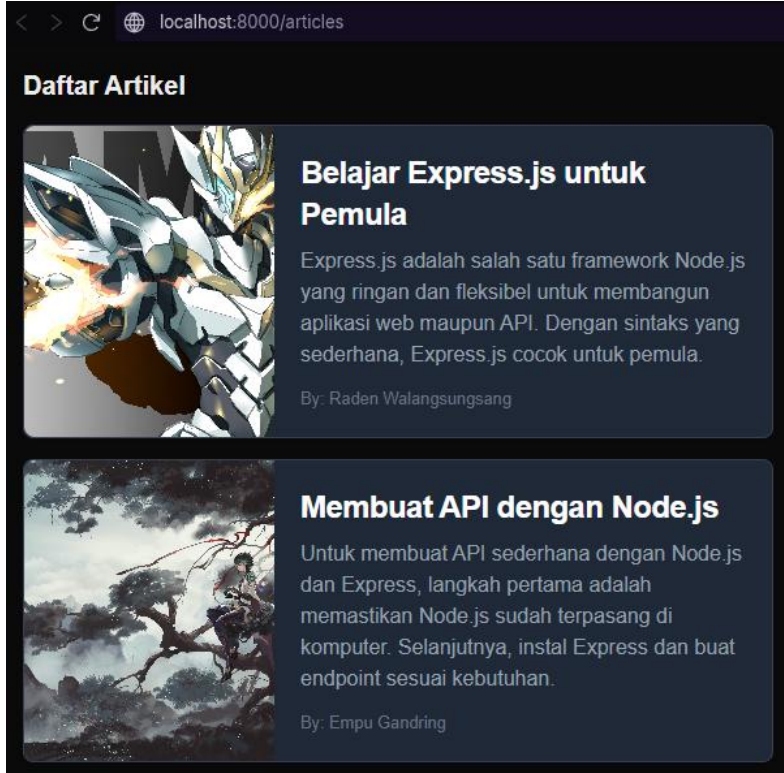
E. Github + Visual Code

1. Buat proyek di Github dengan nama **Latihan4**
git init
git add.
git commit -m "first commit"
git branch -M main
git remote add origin
<https://github.com/agunghakase/Latihan4.git>
git push -u origin main

Hasil Pengerjaan

No.	Instruksi	Screenshot	Kendala/ Saran
A.	Instalasi dan Konfigurasi		
1.	Step 1-2	<pre> C:\Users\User\ExpressJs\APIproject2>npm init -y Wrote to C:\Users\User\ExpressJs\APIproject2\package.json: { "name": "apiproject2", "version": "1.0.0", "description": "", "main": "index.js", "scripts": { "test": "echo \"Error: no test specified\" && exit 1" }, "keywords": [], "author": "", "license": "ISC", "type": "commonjs" } C:\Users\User\ExpressJs\APIproject2>npm install express added 68 packages, and audited 69 packages in 2s 16 packages are looking for funding run `npm fund` for details found 0 vulnerabilities C:\Users\User\ExpressJs\APIproject2> </pre>	
2.	Step 3-6	 <pre> { "status": "success", "data": { "id": 1, "title": "Relajar Express.js untuk Pemula", "urlImage": "https://img.freemove.com/photos-15238899361869-642262688893", "description": "Express.js adalah salah satu framework Node.js yang ringan dan fleksibel untuk membangun aplikasi web dengan API. Dengan titik yang sederhana, Express.js cocok untuk pemula.", "author": "Raden Setiawan-gungah" }, "id": 2, "title": "Membuat API dengan Node.js", "urlImage": "https://www.mediaman.com/4217460/Node.jsFullstack.jpg", "description": "Untuk membuat API sederhana dengan Node.js dan Express, langkah pertama adalah memastikan Node.js sudah terpasang di komputer. Selanjutnya, instal Express dan buat endpoint sesuai kebutuhan.", "author": "Tepu Gaudeng" } </pre>	
B.	Membuat Service mengambil data dari API Endpoints di Next.js		

1.	Step 1-3	<pre> C:\Users\User\ExpressJs>npx create-next-app@latest latihan4 ✓ Would you like to use TypeScript? ... No / Yes ✓ Which linter would you like to use? » ESLint ✓ Would you like to use Tailwind CSS? ... No / Yes ✓ Would you like your code inside a 'src/' directory? ... No / Yes ✓ Would you like to use App Router? (recommended) ... No / Yes ✓ Would you like to use Turbopack? (recommended) ... No / Yes ✓ Would you like to customize the import alias ('@/*' by default)? ... No / Yes Creating a new Next.js app in C:\Users\User\ExpressJs\latihan4. Using npm. Initializing project with template: app-tw Installing dependencies: - react - react-dom - next Installing devDependencies: - typescript - @types/node - @types/react - @types/react-dom - @tailwindcss/postcss - tailwindcss - eslint - eslint-config-next - @eslint/eslintrc added 397 packages, and audited 398 packages in 37s 165 packages are looking for funding run `npm fund` for details found 0 vulnerabilities Initialized a git repository. Success! Created latihan4 at C:\Users\User\ExpressJs\latihan4 </pre>	
2.	Step 4-7		

3.	Step 8-12		
C.	Experiment 1		
1.	Step 1-3		
D.	Experiment 2		
1.	Step 1-3	<pre>src > app > articles > components > @ card_articles.ts > @ CardArticle 1 type CardArticleProps = { 2 img: string; 3 title: string; 4 desc: string; 5 author: string; 6 }; 7 8 export default function CardArticle({ img, title, desc, author }: CardArticleProps) { 9 return (10 <div className="max-w-xl lg:white border border-gray-300 rounded-lg shadow-md dark:bg-gray-800 dark:border-gray-700 flex"> 11 12 <div className="p-5 flex flex-col justify-center"> 13 <h3 className="mb-2 text-2xl font-bold tracking-tight text-gray-900 dark:text-white">{title}</h3> 14 <div className="mb-3 font-normal text-gray-700 dark:text-gray-400">{desc}</div> 15 <small className="text-gray-500">by: {author}</small> 16 </div> 17 </div> 18); 19 }</pre>	

		<pre> src > app > articles > page.tsx > ArticlesPage 1 import { getArticles } from "../services/getarticles"; 2 import CardArticle from "../components/card_article"; 3 4 type Article = { 5 id: number; 6 title: string; 7 urlimage: string; 8 description: string; 9 author: string; 10 }; 11 12 export default async function ArticlesPage() { 13 const articles: Article[] = await getArticles(); 14 15 return (16 <div className="p-4"> 17 <h1 className="text-xl font-bold mb-4">Daftar Artikel</h1> 18 <ul className="space-y-4"> 19 {articles.map((article) => (20 <li key={article.id} className="border p-3 rounded"> 21 <CardArticle 22 img={article.urlimage} 23 title={article.title} 24 desc={article.description} 25 author={article.author} 26 /> 27 28))} 29 30 </div> 31); 32 </pre>	
2.	Output		
E.	Github + Visual Code		

1.

Github

