

Praktikum 4 - Matakuliah Pilihan 1 (Web)

Program Studi: Teknik Informatika

Nama: Salsa Putri Ajriyanti

Nim: 3312401043

Kelas: Informatika A Web Pagi

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.

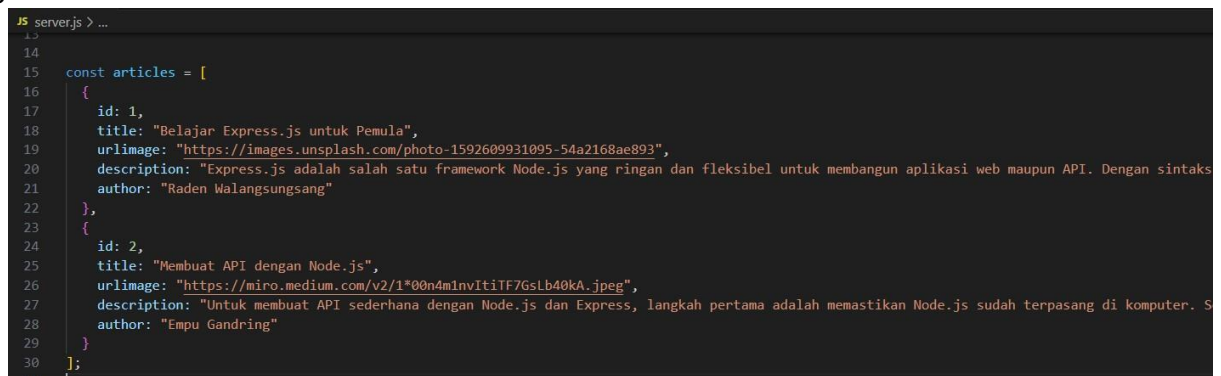
A. Membuat Endpoint test untuk artikel dengan Express.js

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 > ...
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 Walangsungsang"
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 }

```

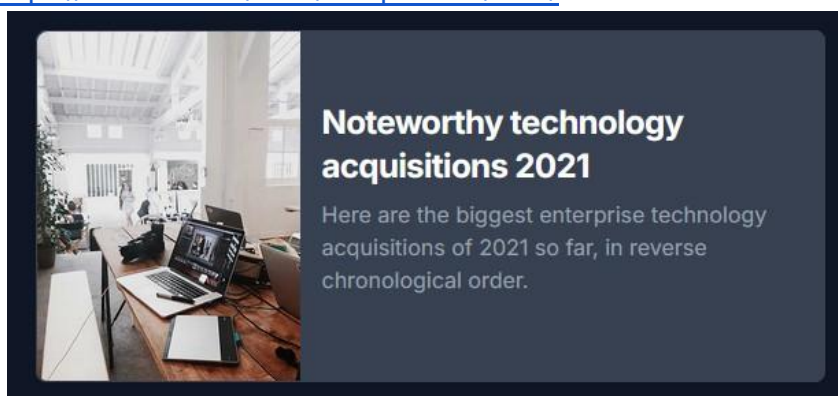
9. Pastikan API bisa diakses lewat browser.
10. Kemudian pada `src/app/articles/page.js` ketik kode program seperti dibawah ini

```
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 flowbite 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

.

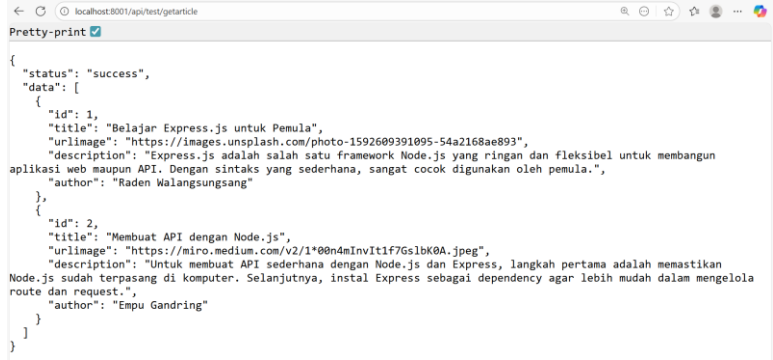
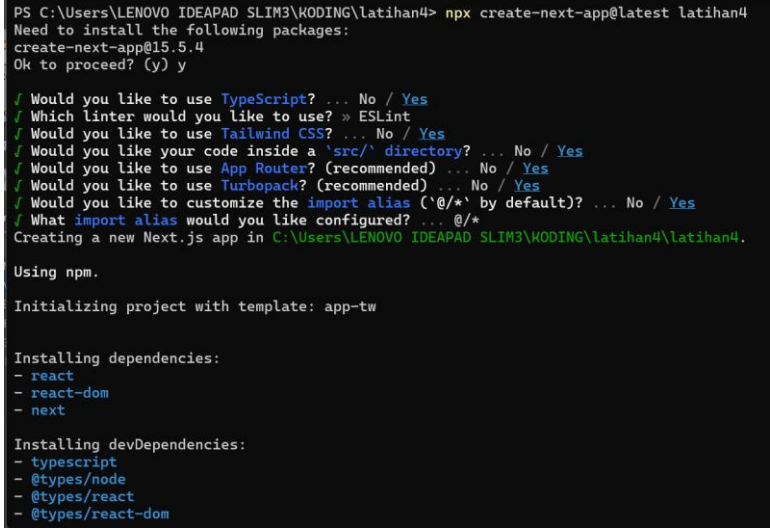
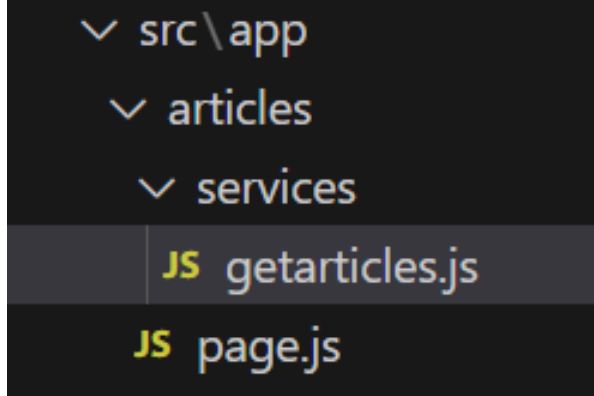
git commit -m "first commit" git branch -M main git remote add

origin https://github.com/agunghakase/Latihan4.git git push -u

origin main

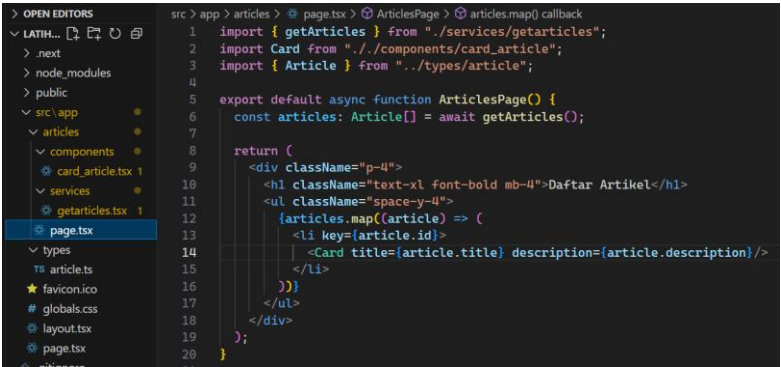
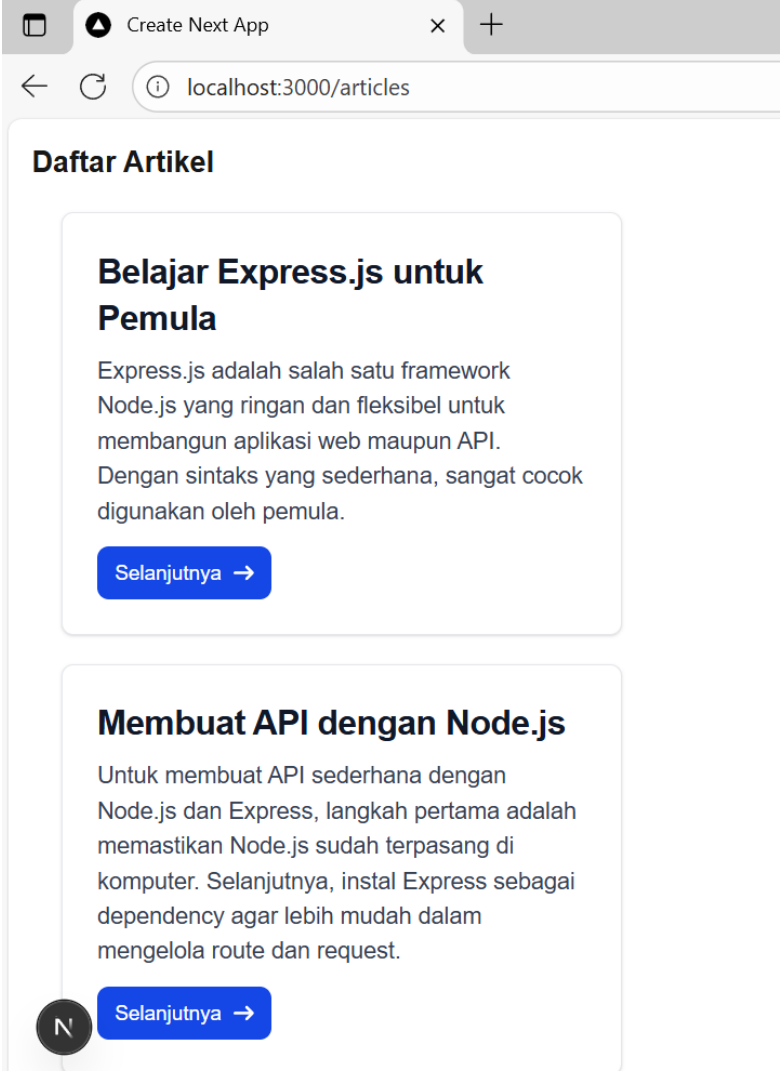
Hasil Pengerjaan

No.	Instruksi	Screenshot	Kendala/Saran
A.	Membuat Endpoint test untuk artikel dengan Express.js		
1.	Menyiapkan express js	<pre> C:\Users\LENOVO IDEAPAD SLIM3\KODING>cd api4 C:\Users\LENOVO IDEAPAD SLIM3\KODING\api4>npm init -y Wrote to C:\Users\LENOVO IDEAPAD SLIM3\KODING\api4\package.json: { "name": "api4", "version": "1.0.0", "description": "", "main": "index.js", "scripts": { "test": "echo \"Error: no test specified\" && exit 1", "start": "node server.js" }, "keywords": [], "author": "", "license": "ISC", "type": "commonjs", "dependencies": { "express": "^5.1.0" }, "devDependencies": {} } C:\Users\LENOVO IDEAPAD SLIM3\KODING\api4> </pre>	
2.	Buat file server.js dengan port di 8001, tidak perlu ada router	<pre> JS server.js > [?] articles > [?] description 1 const express = require('express'); 2 const app = express(); 3 const PORT = 8001; 4 5 app.listen(PORT, () => { 6 console.log(`Server is running on http://localhost:\${PORT}`); 7 }); 8 </pre>	
3.	Pada file server.js tambahkan array-objects dengan nama articles dan jalankan server.js	<pre> 9 const articles = [10 { 11 id: 1, 12 title: "Belajar Express.js untuk Pemula", 13 urlImage: "https://images.unsplash.com/photo-159269391095-54a2168ae893", 14 description: "Express.js adalah salah satu framework Node.js yang ringan dan fleksibel untuk membangun aplikasi web maupun API. Dengan", 15 author: "Raden Walangsungang" 16 }, 17 { 18 id: 2, 19 title: "Membuat API dengan Node.js", 20 urlImage: "https://miro.medium.com/v2/1+0b7mInvIt157Gclb88A.jpeg", 21 description: "Untuk membuat API sederhana dengan Node.js dan Express, langkah pertama adalah memastikan Node.js sudah terpasang di kom", 22 author: "Empu Gandring" 23 } 24]; 25 26 app.get('/api/test/getarticle', (req, res) => { 27 res.json({ 28 status: "success", 29 data: articles 30 }); 31 }); </pre>	

4.	Output dari endpoints /api/test/getarticle	 <pre> { "status": "success", "data": [{ "id": 1, "title": "Belajar Express.js untuk Pemula", "urlimage": "https://images.unsplash.com/photo-1592609391095-54a2168ae893", "description": "Express.js adalah salah satu framework Node.js yang ringan dan fleksibel untuk membangun aplikasi web maupun API. Dengan sintaks yang sederhana, sangat cocok digunakan oleh pemula.", "author": "Raden Walangsungsang" }, { "id": 2, "title": "Membuat API dengan Node.js", "urlimage": "https://miro.medium.com/v2/1*00n4mInvItif7Gs1bk0A.jpeg", "description": "Untuk membuat API sederhana dengan Node.js dan Express, langkah pertama adalah memastikan Node.js sudah terpasang di komputer. Selanjutnya, instal Express sebagai dependency agar lebih mudah dalam mengelola route dan request.", "author": "Empu Gandring" }] } </pre>	
B.	Membuat Service mengambil data dari API Endpoints di Next.js		
1.	Buat folder baru Latihan4 didalamnya install next.js Dan jalankan npm run dev	 <pre> PS C:\Users\LENOVO IDEAPAD SLIM3\KODING\latihan4> npx create-next-app@latest latihan4 Need to install the following packages: create-next-app@15.5.4 Ok to proceed? (y) y / 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 / What import alias would you like configured? ... @/* Creating a new Next.js app in C:\Users\LENOVO IDEAPAD SLIM3\KODING\latihan4\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 </pre>	
2.	Membuat folder articles dan services, serta membuat file page.js di dalam folder articles dan membuat getarticles.js di dalam folder services		

3.	file getarticles.js buat kode program	 <pre> 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 } 14 </pre>	
4.	pada src/app/articles/ page.js ketik kode program	 <pre> 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 14 <p>{article.description}</p> 15 <small className="text-gray-500"> 16 By: {article.author} 17 </small> 18 19))} 20 21 </div> 22); 23 } 24 </pre>	
5.	Cek output dengan localhost:3001/articles		
C.	Experiment 1		
1.	Ganti nama file untuk page.js dan getarticles.js menjadi page.tsx dan getarticles.tsx	 <pre> return (<div className="p-4"> <h1 className="text-xl font-bold mb-4">Daftar Artikel</h1> <ul className="space-y-2"> {articles.map((article) => (<li key={article.id} className="border p-3 rounded"> </pre>	yg terjadi yaitu pada file page.js terdapat error di bagian (article) untuk sekarang tujuannya Cuma jalan jadi tidak

			apa apa tapi tujuannya buat belajar TypeScript dengan benar, sebaiknya tulis type.ts nya
2.	Ambil component flowbite Horizontal Card dan Modifikasi src/app/articles/page .tsx dengan mengganti baris kode dengan flobite horizontal components.	<pre> 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-4"> 10 {articles.map((article) => (11 <li key={article.id}> 12 <a 13 href="#" 14 className="flex flex-col items-center bg-white border border-gray-200 rounded-lg shadow-sm md:flex-row md:space-between md:p-4" 15 > 16 21 <div className="flex flex-col justify-between p-4 loading-normal"> 22 <h5 className="mb-2 text-2xl font-bold tracking-tight text-gray-900 dark:text-white"> 23 {article.title} 24 </h5> 25 <p className="mb-3 font-normal text-gray-700 dark:text-gray-400"> 26 {article.description} 27 </p> 28 <small className="text-gray-500">By: {article.author}</small> 29 </div> 28 29 30)} 31 </pre>	
3.	Output		
D.	Experiment 2		

1.	Gunakan Dynamic components seperti pada Praktikum 2. Dengan menempatkan flowbite ke folder components di dalam /src/app/articles/component.tsx Kemudian pada page.tsx cukup dipanggil saja dengan di dalam panggil nama komponennya dan mengirim parameter		Setelah banyak yg di oprek dan hasil akhirnya seperti ini, untuk file .js ke .tsx harus ada file typenya dimana berisi tipe parameternya contohnya title:string
2.	Output		
E.	Github		

1.	Melakukan commit ke github	<pre>create mode 100644 latihan4/postcss.config.mjs create mode 100644 latihan4/public/file.svg create mode 100644 latihan4/public/globe.svg create mode 100644 latihan4/public/next.svg create mode 100644 latihan4/public/vercel.svg create mode 100644 latihan4/public/window.svg create mode 100644 latihan4/src/app/articles/components/card_article.tsx create mode 100644 latihan4/src/app/articles/services/getarticles.tsx create mode 100644 latihan4/src/app/favicon.ico create mode 100644 latihan4/src/app/globals.css create mode 100644 latihan4/src/app/layout.tsx create mode 100644 latihan4/src/app/page.tsx create mode 100644 latihan4/src/app/types/article.ts create mode 100644 latihan4/tsconfig.json C:\Users\LENOVO IDEAPAD SLIM3\WODING\PRAKTIKUM4 FE-BE>git branch -M main C:\Users\LENOVO IDEAPAD SLIM3\WODING\PRAKTIKUM4 FE-BE>git remote add origin https://github.com/salsaajriy/praktikum4-fe-be.git C:\Users\LENOVO IDEAPAD SLIM3\WODING\PRAKTIKUM4 FE-BE>git push -u origin main Enumerating objects: 751, done. Counting objects: 100% (751/751), done. Delta compression using up to 8 threads Compressing objects: 100% (692/692), done. Writing objects: 100% (751/751), 792.08 KiB 3.28 MiB/s, done. Total 751 (delta 145), reused 0 (delta 0), pack-reused 0 remote: Resolving deltas: 100% (145/145), done. To https://github.com/salsaajriy/praktikum4-fe-be.git * [new branch] main -> main branch 'main' set up to track 'origin/main'.</pre>	
2.	Output	