

LAPORAN PEMROGRAMAN FRAMEWORK

“SESSION 4 API”



Oleh:

Nama : Bagus Satria Putra
Kelas : 3F
Absen/NIM : 8 / 1841720146

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

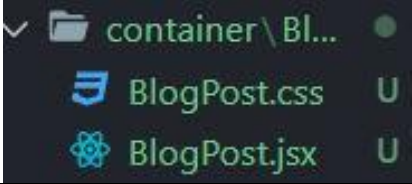
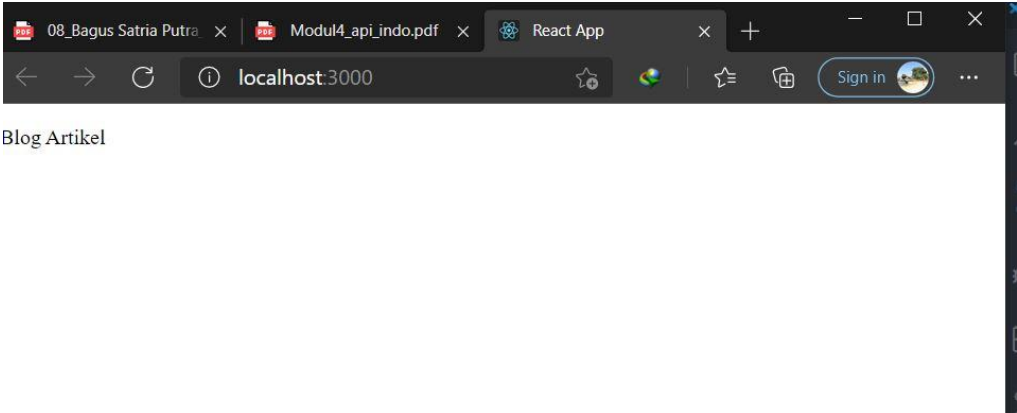
4 Maret 2021

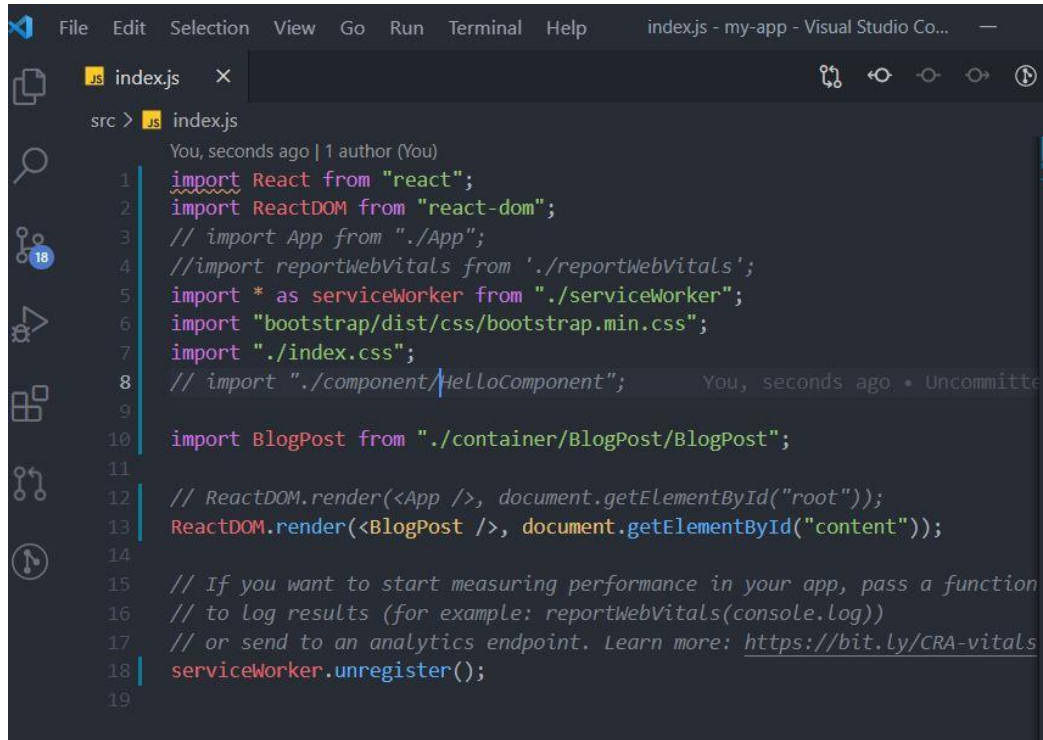
Untuk link youtube dan github ada di halaman terakhir.

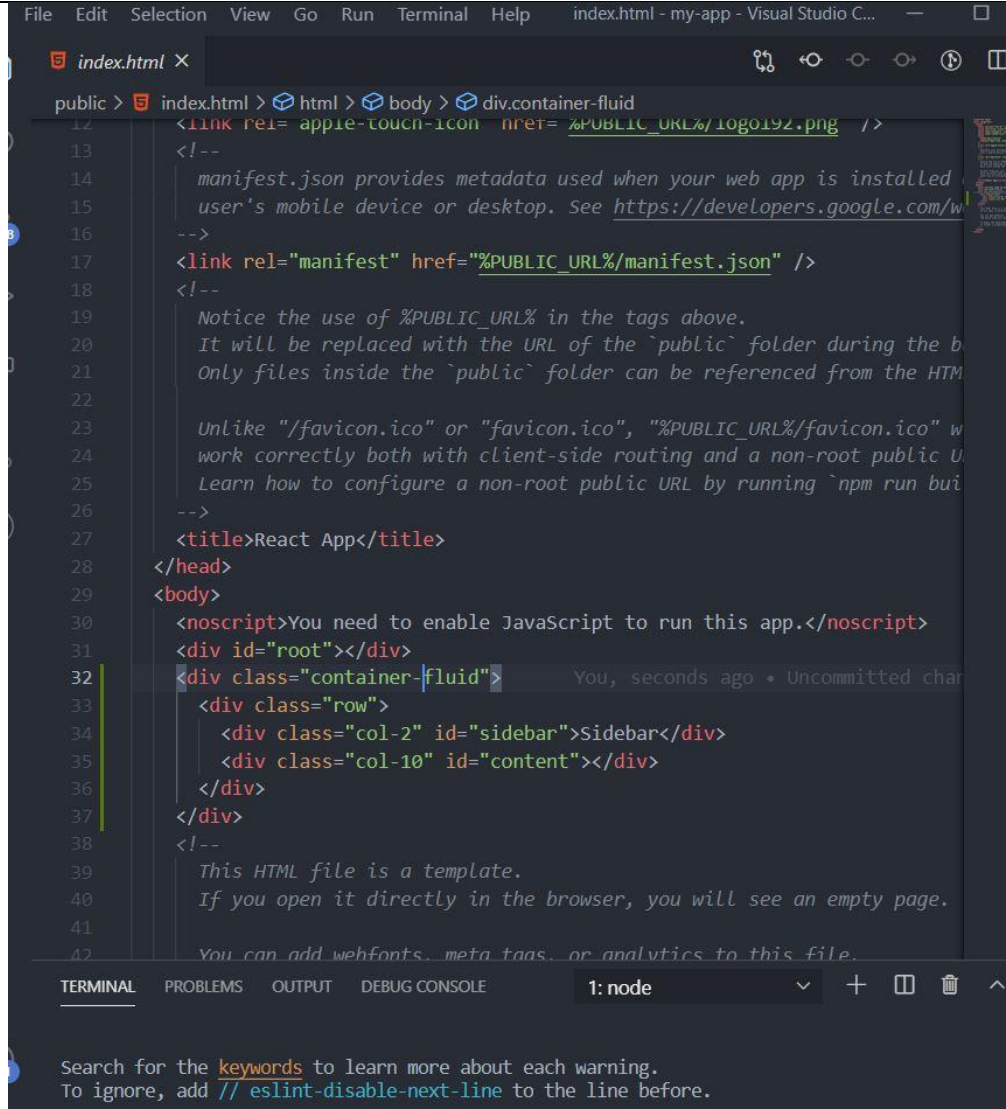
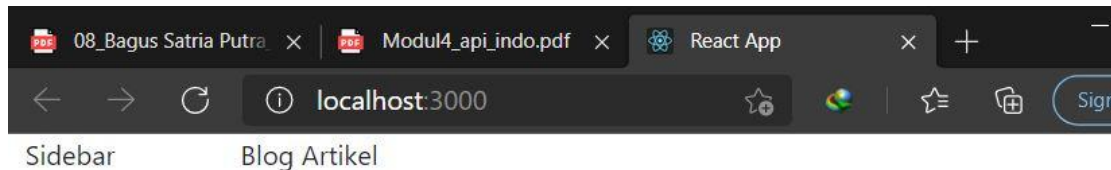
API

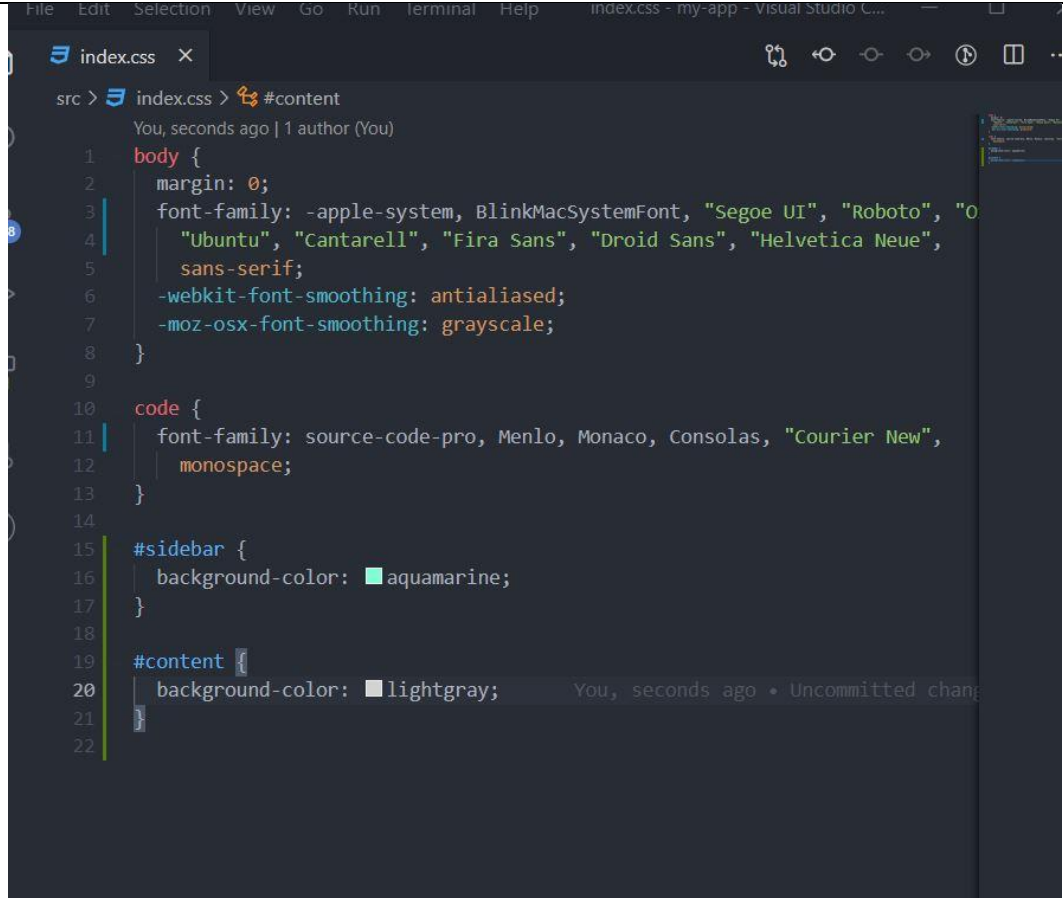
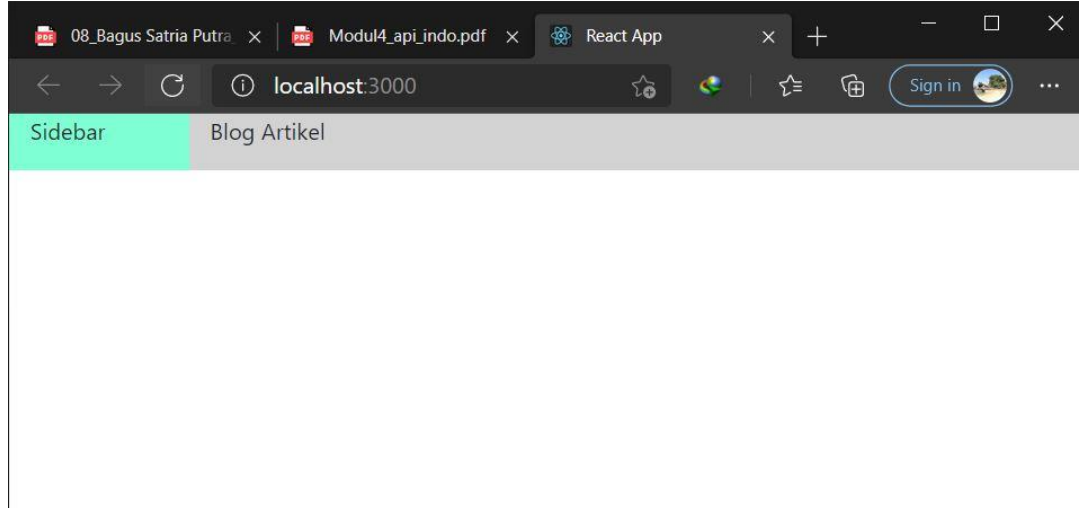
API adalah singkatan dari Application Programming Interface, dan memungkinkan developer untuk mengintegrasikan dua bagian dari aplikasi atau dengan aplikasi yang berbeda secara bersamaan. API terdiri dari berbagai elemen seperti function, protocols, dan tools lainnya yang memungkinkan developers untuk membuat aplikasi. Tujuan penggunaan API adalah untuk mempercepat proses development dengan menyediakan function secara terpisah sehingga developer tidak perlu membuat fitur yang serupa. Penerapan API akan sangat terasa jika fitur yang diinginkan sudah sangat kompleks. Gambar 1. API API dapat anda temui dalam kehidupan sehari-hari seperti saat anda memesan hotel, mengirimkan pesan, memesan makanan secara online maupun ketika mengunduh sebuah software. Kenapa menggunakan API? API membuat pemrograman menjadi lebih mudah. Kebutuhan kita sebagai pelanggan dan khususnya bagi developer sangat dimudahkan dengan adanya API. Dengan melihat hal tersebut, peran dari API sendiri sangat berat terlebih untuk membuat tampilan sebuah aplikasi menjadi interaktif, mudah untuk digunakan, dan bersahabat untuk pengguna. Tidak hanya itu, API juga digunakan untuk berkomunikasi antara layanan-layanan. API memiliki peran yang sangat penting dalam teknologi.

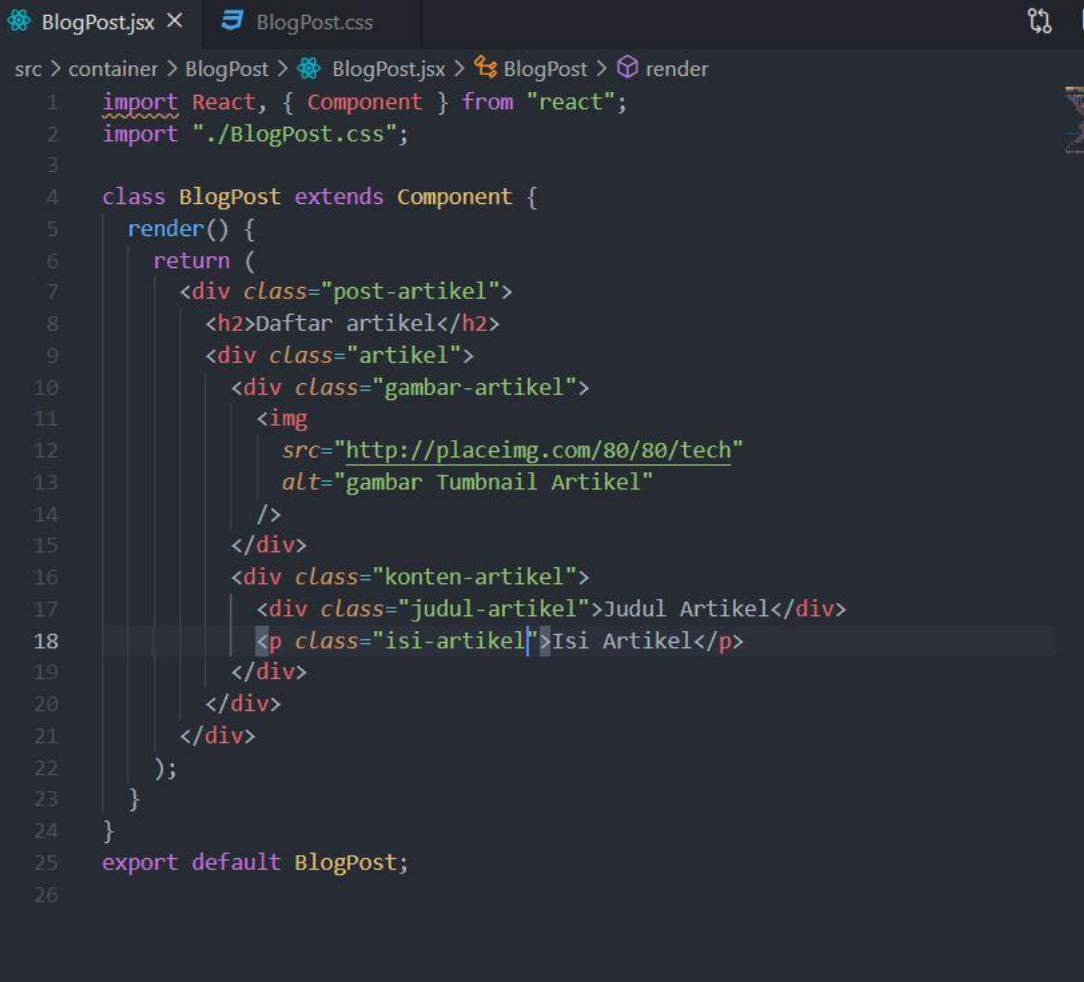
Keuntungan menggunakan API Bagi Para Developer Antara Lain: 1. Aplikasi. API membantu kinerja dari aplikasi lebih cepat dan fleksibel seperti layanan dan informasi yang diberikan karena API dapat memasuki komponen-komponen aplikasi. 2. Kustomisasi Dengan API, kustomisasi untuk konten dan layanan dapat dilakukan sesuai kebutuhan dan keinginan. 3. Fleksibel API membuat layanan menjadi lebih fleksibel. Hal tersebut karena API mendukung data migrasi lebih baik dan informasi yang didapat ditinjau lebih dekat. 4. Integrasi / integration API dapat menjamin pengiriman informasi lebih lancar dikarenakan API memungkinkan konten tertanam dari aplikasi maupun situs dengan mudah. Hal tersebut memberikan pengalaman yang terintegrasi bagi pengguna. 5. Lebih banyak data API memberikan banyak pilihan karena semua informasi yang dihasilkan di tingkat pemerintah tersedia untuk setiap warga negara.

No.	Keterangan
1.	<p>1. Buka Project React pada pertemuan sebelumnya dan jalankan “npm start” menggunakan cmd dalam direktori tersebut.</p> <p>2. Buat folder baru bernama “BlogPost” pada folder container (statefull component).</p> <p>3. Buat file BlogPost.jsx dan BlogPost.css di dalam folder “BlogPost”, seperti pada Gambar 1.2.</p> <p>Jawaban.</p> 
2.	<p>4. Buka file BlogPost.jsx dan ketikkan kode seperti Gambar 1.3.</p> <p>Jawaban :</p> 
3.	<p>5. Pada file index.js, lakukan import component BlogPost seperti Gambar 1.4</p> <p>6. Pada web browser akan tampil seperti pada Gambar 1.5</p> <p>Jawaban :</p> 

4.	
5.	<p>7. Import css bootstrap.min.css (css bootstrap yang sudah dikompresi) ke dalam index.js (seperti Gambar 1.6). Jika css tidak ditemukan, install lewat cmd dengan perintah “npm install bootstrap”</p> <p>Jawaban :</p>  <pre> 1 import React from "react"; 2 import ReactDOM from "react-dom"; 3 // import App from "./App"; 4 //import reportWebVitals from './reportWebVitals'; 5 import * as serviceWorker from "./serviceWorker"; 6 import "bootstrap/dist/css/bootstrap.min.css"; 7 import "./index.css"; 8 // import "./component/HelloComponent"; 9 10 import BlogPost from "../container/BlogPost/BlogPost"; 11 12 // ReactDOM.render(<App />, document.getElementById("root")); 13 ReactDOM.render(<BlogPost />, document.getElementById("content")); 14 15 // If you want to start measuring performance in your app, pass a function 16 // to log results (for example: reportWebVitals(console.log)) 17 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals 18 serviceWorker.unregister(); 19 </pre>
6.	<p>8. Modifikasi file index.html pada folder "public" seperti Gambar 1.7. Cermati code program yang ada dalam gambar!.</p> <p>Jawaban :</p>

	
7.	<p>9. Amati tampilan yang ada pada browser (seperti Gambar 1.8)</p> <p>Jawaban :</p> 
8.	<p>10. Buka file index.css dan tambahkan code css seperti Gambar 1.9, untuk menambah sedikit style pada halaman web</p> <p>Jawaban :</p>

	 <pre> src > index.css > #content You, seconds ago 1 author (You) 1 body { 2 margin: 0; 3 font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", "Roboto", "O 4 "Ubuntu", "Cantarell", "Fira Sans", "Droid Sans", "Helvetica Neue", 5 sans-serif; 6 -webkit-font-smoothing: antialiased; 7 -moz-osx-font-smoothing: grayscale; 8 } 9 10 code { 11 font-family: source-code-pro, Menlo, Monaco, Consolas, "Courier New", 12 monospace; 13 } 14 15 #sidebar { 16 background-color: aquamarine; 17 } 18 19 #content { 20 background-color: lightgray; 21 } 22 </pre>
9.	<p>11. Perhatikan kembali browser, dan lihat hasil tampilan seperti Gambar 1.10.</p> <p>Jawaban :</p> 
10.	<p>12. Ubah kode program untuk statefull component BlogPost.jsx menjadi seperti Gambar 1.11</p> <p>Jawaban :</p>

	 <pre> 1 import React, { Component } from "react"; 2 import "./BlogPost.css"; 3 4 class BlogPost extends Component { 5 render() { 6 return (7 <div class="post-artikel"> 8 <h2>Daftar artikel</h2> 9 <div class="artikel"> 10 <div class="gambar-artikel"> 11 15 </div> 16 <div class="konten-artikel"> 17 <div class="judul-artikel">Judul Artikel</div> 18 <p class="isi-artikel">Isi Artikel</p> 19 </div> 20 </div> 21 </div> 22); 23 } 24 } 25 export default BlogPost; 26 </pre>
11.	13. Tambahkan custom css ke BlogPost.css seperti Gambar 1.12 Jawaban :

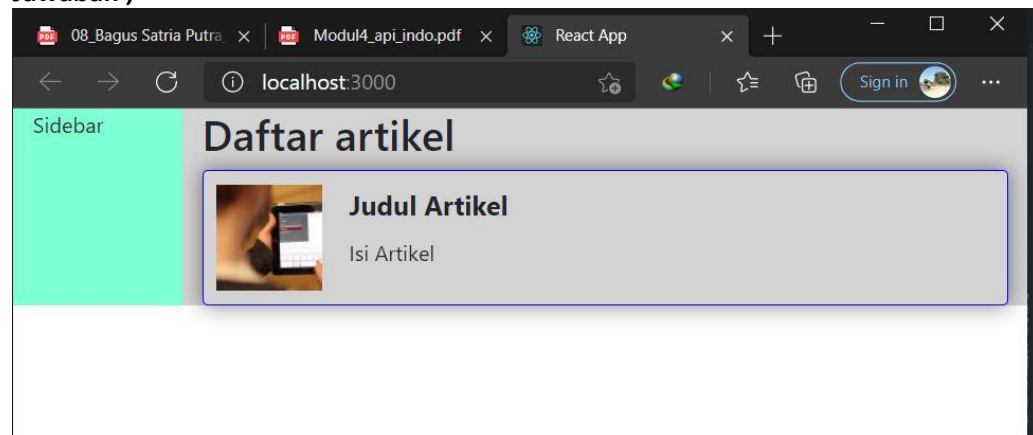

```

BlogPost.jsx  BlogPost.css X
src > container > BlogPost > BlogPost.css > .konten-artikel p.isi-artikel
1  .artikel {
2    width: 100%;
3    padding: 10px;
4    border: 1px solid blue;
5    border-radius: 4px;
6    box-shadow: 0 0 16px rgba(0, 0, 0, 0.5);
7    display: flex;
8  }
9
10 .gambar-artikel {
11   height: 80px;
12   width: 80px;
13   margin-right: 20px;
14   vertical-align: top;
15 }
16
17 .gambar-artikel img {
18   width: 100%;
19   height: 100%;
20   object-fit: cover;
21 }
22
23 .konten-artikel {
24   flex: 1;
25 }
26
27 .konten-artikel div.judul-artikel {
28   font-size: 20px;
29   font-weight: bold;
30   margin-bottom: 10px;

```

12. 14. Perhatikan tampilan browser.

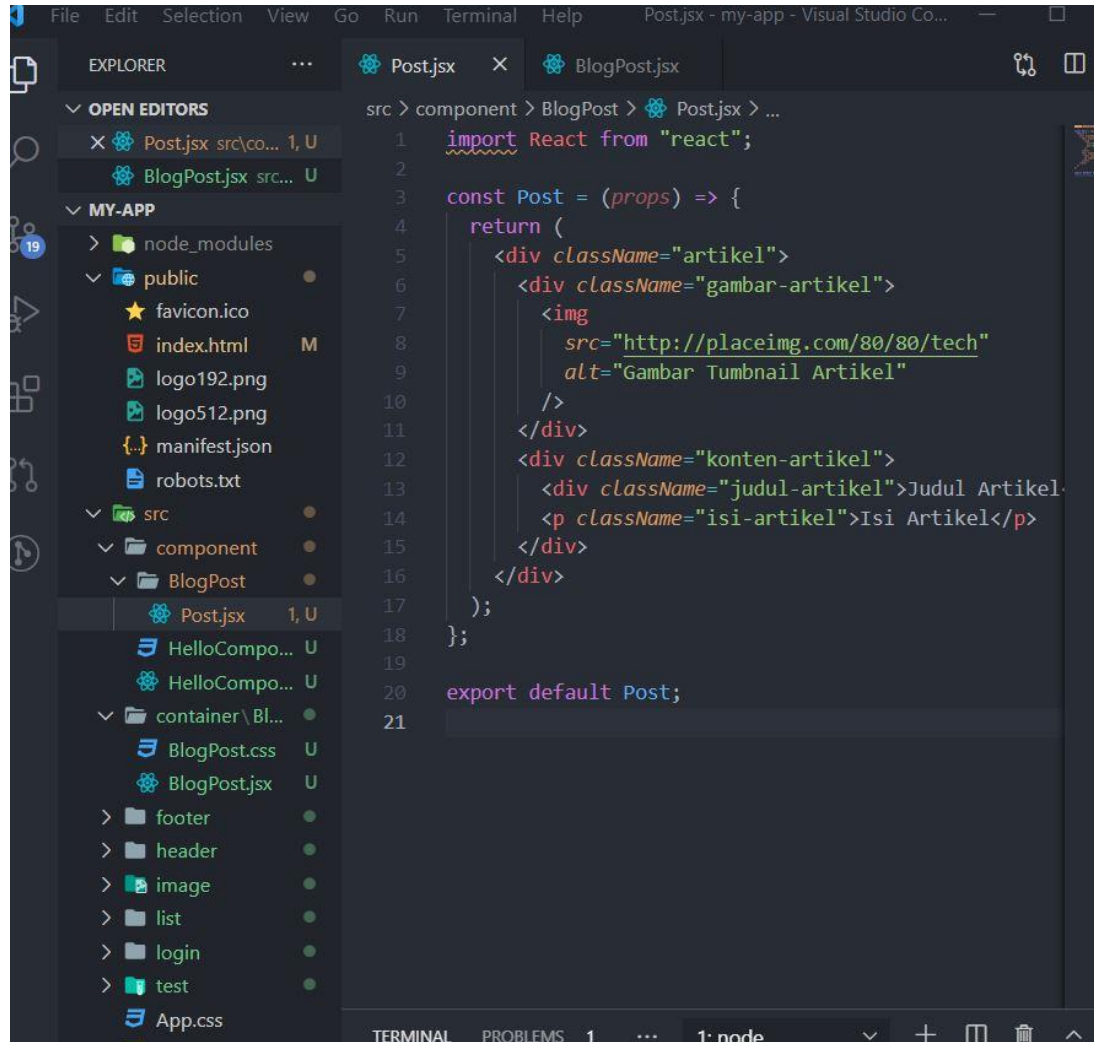
Jawaban ;



13. 15. Buat folder BlogPost pada folder component (stateless component), lalu buat file Post.jsx

16. Potong (cut) baris 9-17 pada statefull component BlogPost.jsx ke stateless component Post.jsx, dan modifikasi Post.jsx seperti Gambar 1.13.

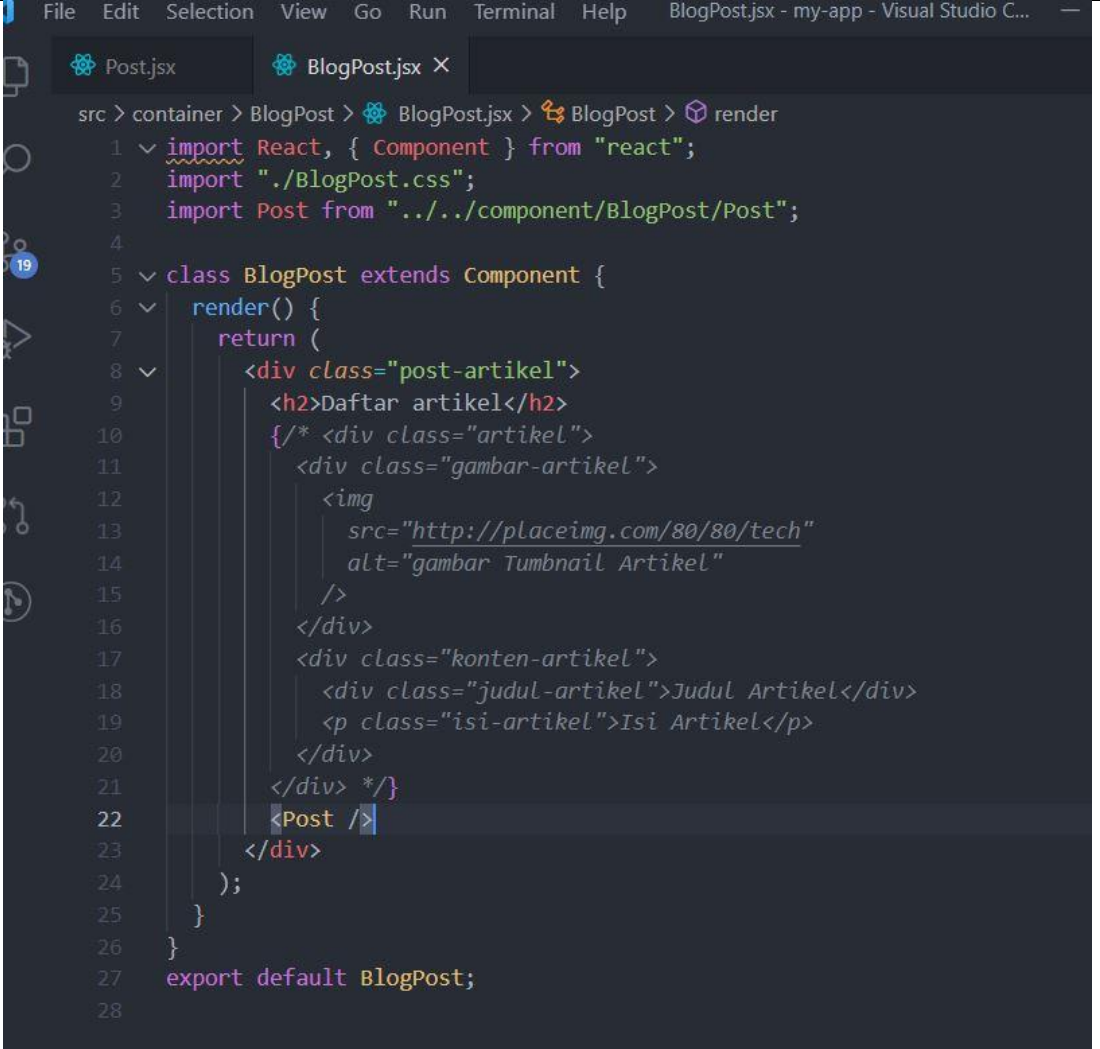

Jawaban :



```
1 import React from "react";
2
3 const Post = (props) => {
4   return (
5     <div className="artikel">
6       <div className="gambar-artikel">
7         
11       </div>
12       <div className="konten-artikel">
13         <div className="judul-artikel">Judul Artikel
14         <p className="isi-artikel">Isi Artikel</p>
15       </div>
16     </div>
17   );
18 };
19
20 export default Post;
21
```

14. 17. Untuk statefull component BlogPost.jsx pada baris 10, panggil stateless component Post.jsx seperti Gambar 1.14.

Jawaban :

	 <pre> src > container > BlogPost > BlogPost.jsx > BlogPost > render 1 import React, { Component } from "react"; 2 import "./BlogPost.css"; 3 import Post from "../../component/BlogPost/Post"; 4 5 class BlogPost extends Component { 6 render() { 7 return (8 <div class="post-artikel"> 9 <h2>Daftar artikel</h2> 10 /* <div class="artikel"> 11 <div class="gambar-artikel"> 12 16 </div> 17 <div class="konten-artikel"> 18 <div class="judul-artikel">Judul Artikel</div> 19 <p class="isi-artikel">Isi Artikel</p> 20 </div> 21 </div> */ 22 <Post /> 23 </div> 24); 25 } 26 } 27 export default BlogPost; 28 </pre>
15.	<p>18. Perhatikan hasil tampilan browser, apa yang terjadi?</p> <p>Jawaban :</p> 
16.	<p>19. Pada statefull component BlogPost.jsx, tambahkan parameter yang ingin dilempar ke stateless component untuk ditampilkan. Kode program bisa dilihat pada Gambar 1.15.</p> <p>Jawaban :</p>

	 <pre> 4 5 class BlogPost extends Component { 6 render() { 7 return (8 <div class="post-artikel"> 9 <h2>Daftar artikel</h2> 10 /* <div class="artikel"> 11 <div class="gambar-artikel"> 12 16 </div> 17 <div class="konten-artikel"> 18 <div class="judul-artikel">Judul Artikel</div> 19 <p class="isi-artikel">Isi Artikel</p> 20 </div> 21 </div> */ 22 <Post 23 judul="JTI Polinema" 24 isi="Jurusan Teknologi Informasi - Politeknik Negeri Malang" 25 /> 26 </div> 27); 28 } 29 } 30 export default BlogPost; 31 </pre> <p>Line 22:11: The href attribute is required for an anchor to be keyboard accessible. Provide a valid, navigable address as the href value. If you cannot provide an href, but still need the element to resemble a link, use a button and change it with appropriate styles. Learn more: https://github.com/evcohen/eslint-plugin-jsx-a11y/blob/master/docs/rules/anchor-is-valid.md</p> <p>Search for the keywords to learn more about each warning.</p>
17.	<p>20. Setelah itu pada stateless component Post.jsx tangkap parameter yang dilempar oleh statefull component seperti pada Gambar 1.16 dan lihat pada browser apa yang terjadi!.</p> <p>Jawaban :</p>

Visual Studio Code interface showing a React component file named `Post.jsx` in the `src > component > BlogPost` directory. The code defines a functional component `Post` that renders a div with a class `artikel`. Inside, there's a div with class `gambar-artikel` containing an image with a placeholder URL and alt text. Below that is a div with class `konten-artikel` containing a div with class `judul-artikel` and a paragraph with class `isi-artikel`. The component is exported as `Post`.

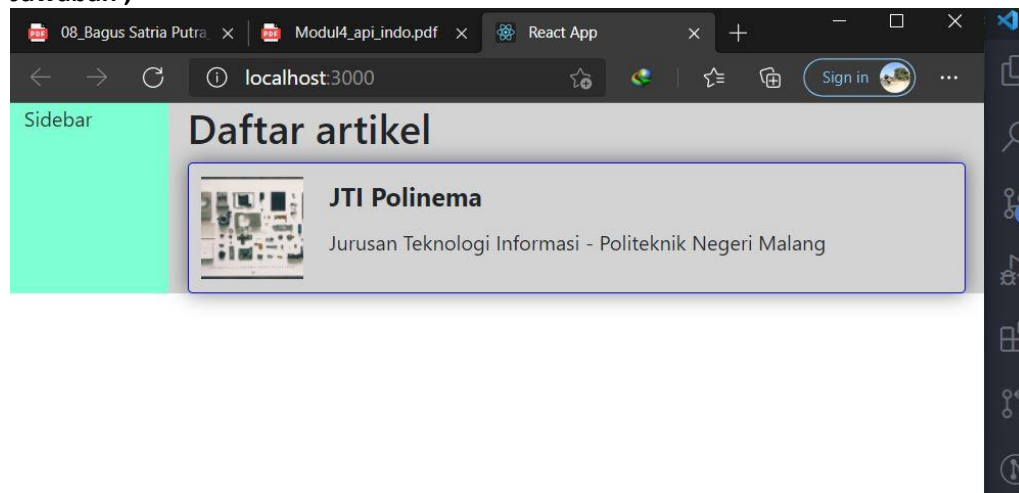
```
src > component > BlogPost > Post.jsx > Post
1  import React from "react";
2
3  const Post = (props) => {
4    return (
5      <div className="artikel">
6        <div className="gambar-artikel">
7          
11        </div>
12        <div className="konten-artikel">
13          <div className="judul-artikel">{props.judul}</div>
14          <p className="isi-artikel">{props.isi}</p>
15        </div>
16      </div>
17    );
18  };
19
20  export default Post;
```

TERMINAL PROBLEMS 1 OUTPUT DEBUG CONSOLE 1: node

Line 22:11: The href attribute is required for an anchor to be keyboard accessible. Provide a valid, navigable address as the href value. If you cannot provide an href, but still need the element to resemble a link, use a button and change it with appropriate styles. Learn more: <https://github.com/evcohen/eslint-plugin-jsx-a11y/blob/master/docs/rules/anchor-is-valid.md>

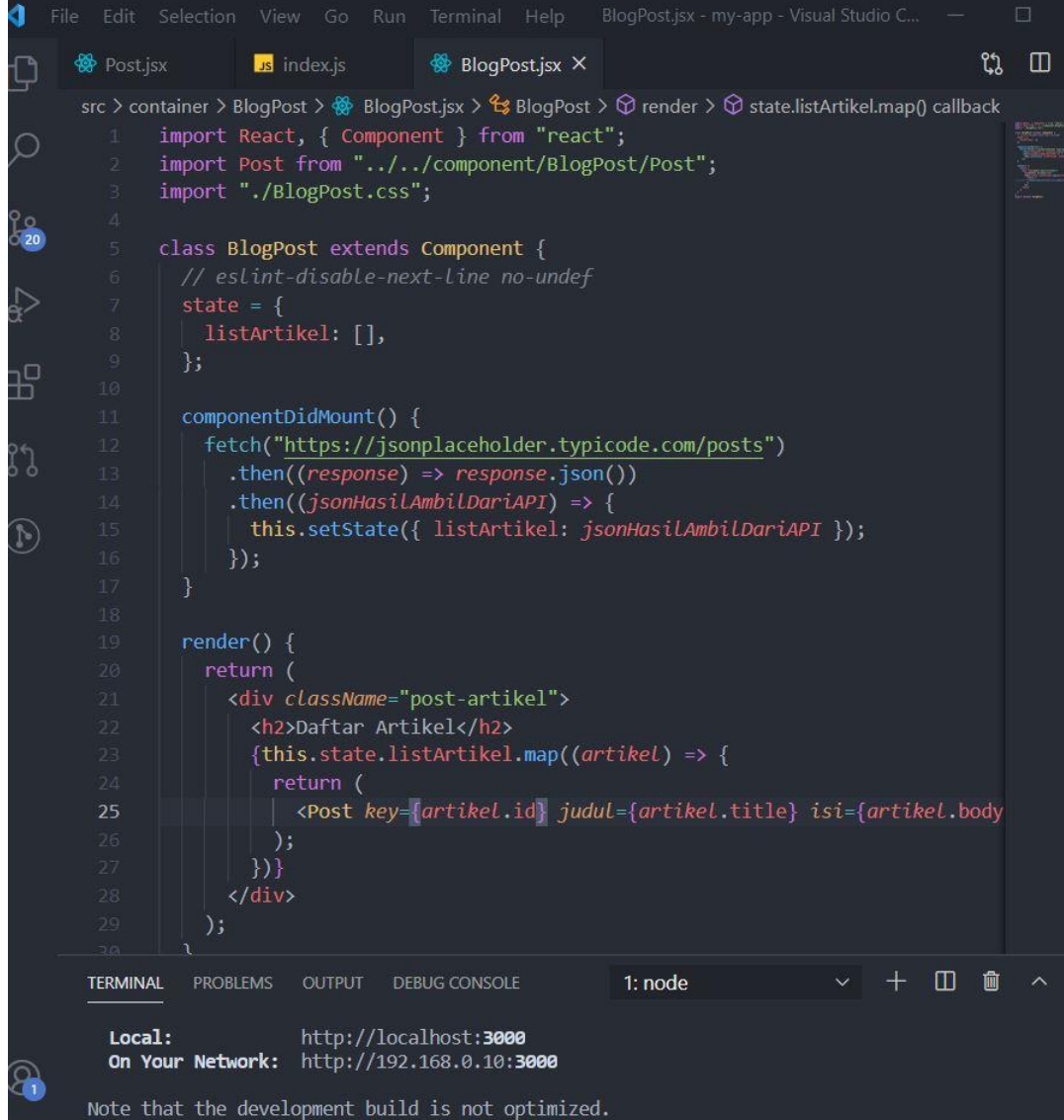
18.	21. Simpan, dan amati apa yang terjadi pada browser kalian!.
-----	--------------------------------------------------------------

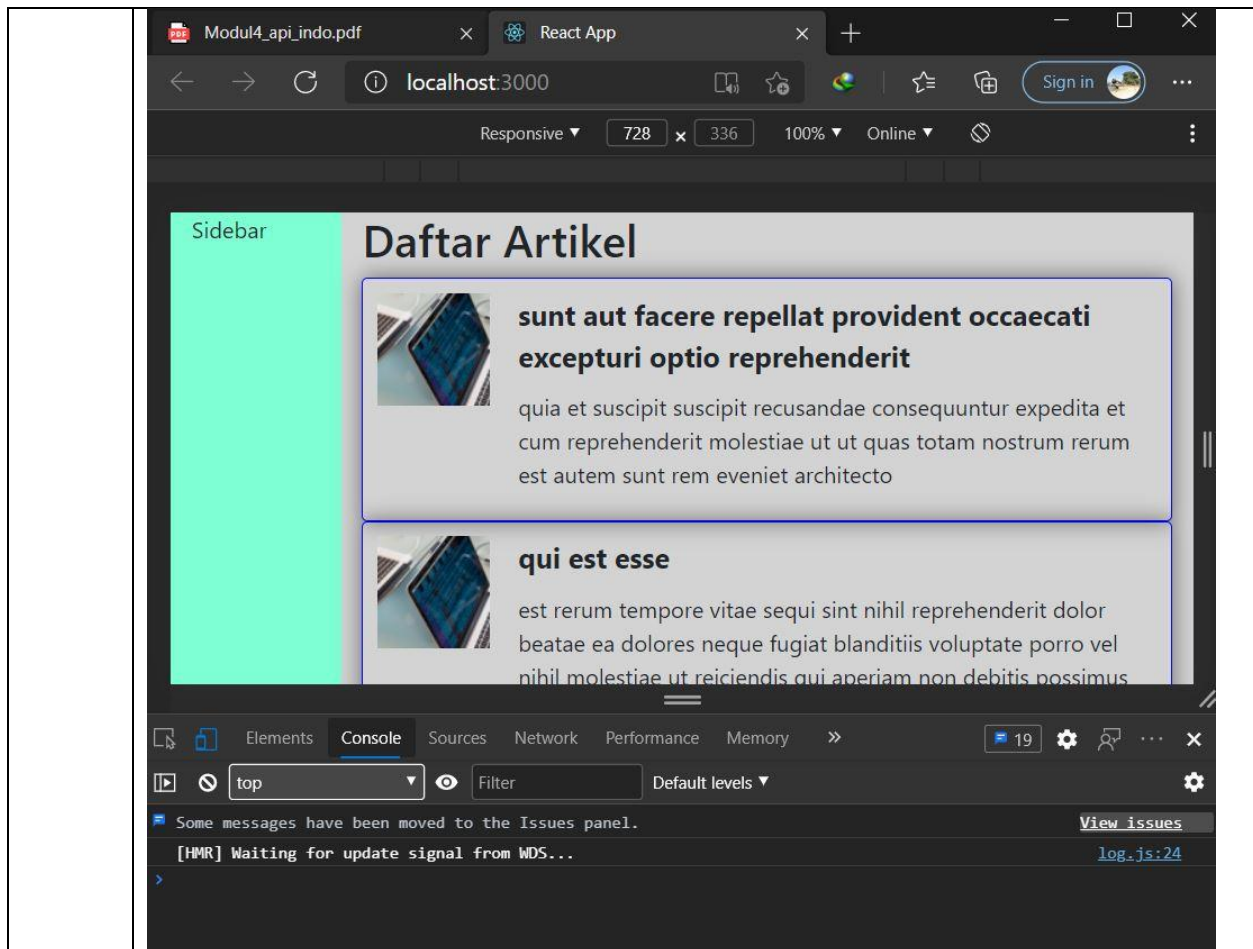
Jawaban ;



19.	22. Gunakan state untuk menyimpan data hasil request dari API
-----	---------------------------------------------------------------

	<p>23. data API yang akan kita gunakan adalah data dummy dari https://jsonplaceholder.typicode.com/posts, dimana memiliki 4 element data yaitu userid, id, title, body (seperti pada Gambar 1.17)</p> <p>24. Edit pada statefull component BlogPost.jsx seperti pada Gambar 1.18 dan perhatikan dengan seksama akan penjelasan di beberapa baris kode program tersebut.</p> <p>25. Lihat hasilnya pada browser. Kemudian klik kanan pada browser pilih "inspect element" kemudian pilih tab "console". Refresh browser dan amati apa yang terjadi.</p> <p>26. Jika terlihat seperti pada Gambar 1.19, maka terjadi kesalahan pada program yang kita buat.</p> <p>27. Jika terjadi hal demikian, hal ini terjadi karena dalam react "class" dalam tag html harus ditulis menjadi "className". selain itu, pada statefull component yang dinamis, harus ada "UNIQUE KEY" pada tiap komponen yang diproses sehingga komponen perlu diberi UNIQUE KEY.</p> <p>28. UNIQUE KEY dapat diambil dari element yang ada pada data API yang sudah kita ambil (contoh saat ini adalah element id pada data API (userid, id, title, body) yang akan kita gunakan untuk UNIQUE KEY. Lihat Gambar 1.20.</p> <p>Jawaban :</p>
--	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

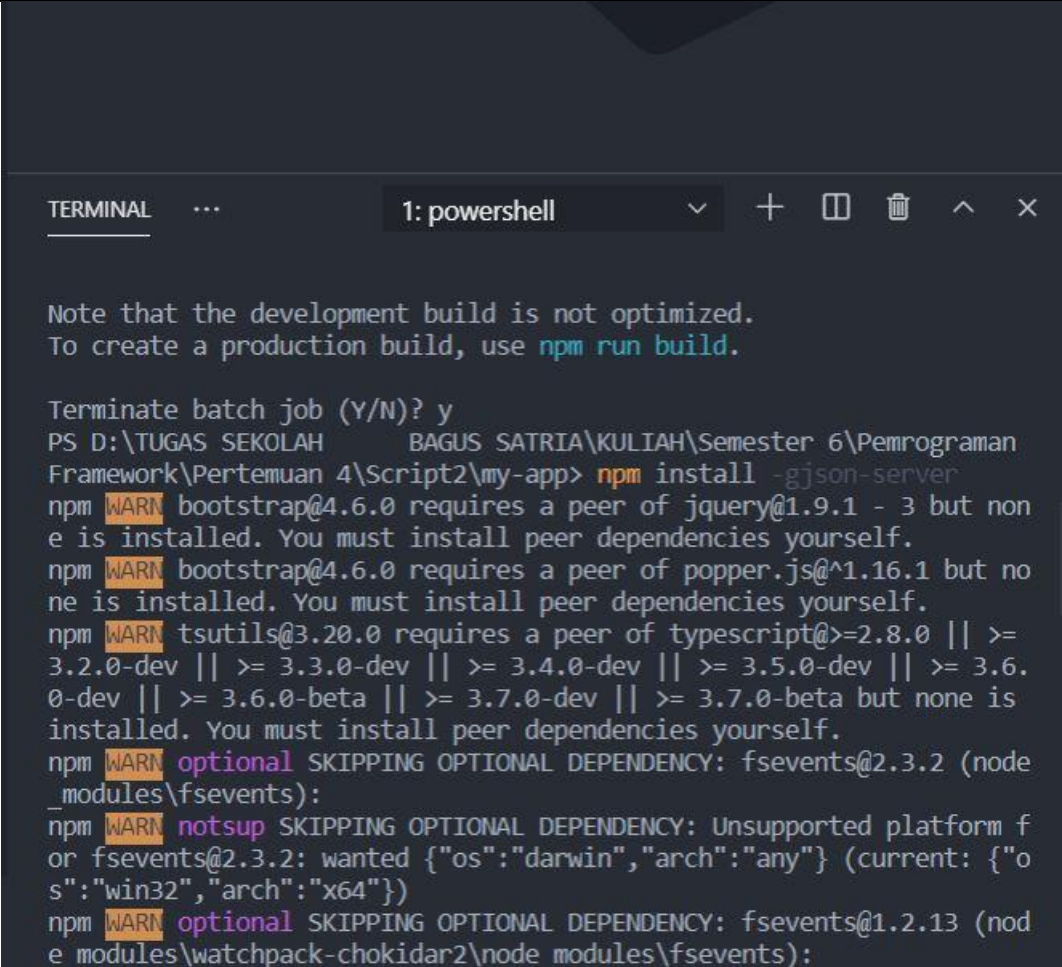
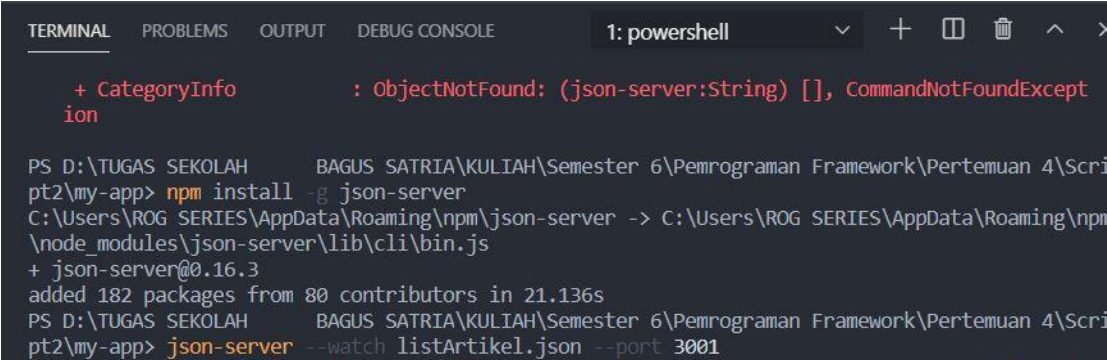
	 <pre> src > container > BlogPost > BlogPost.jsx > BlogPost > render > state.listArtikel.map() callback 1 import React, { Component } from "react"; 2 import Post from "../../component/BlogPost/Post"; 3 import "../BlogPost.css"; 4 5 class BlogPost extends Component { 6 // eslint-disable-next-line no-undef 7 state = { 8 listArtikel: [], 9 }; 10 11 componentDidMount() { 12 fetch("https://jsonplaceholder.typicode.com/posts") 13 .then((response) => response.json()) 14 .then((jsonHasilAmbilDariAPI) => { 15 this.setState({ listArtikel: jsonHasilAmbilDariAPI }); 16 }); 17 } 18 19 render() { 20 return (21 <div className="post-artikel"> 22 <h2>Daftar Artikel</h2> 23 {this.state.listArtikel.map((artikel) => { 24 return (25 <Post key={artikel.id} judul={artikel.title} isi={artikel.body 26); 27 })} 28 </div> 29); 30 } </pre> <p> Local: http://localhost:3000 On Your Network: http://192.168.0.10:3000 </p> <p>Note that the development build is not optimized.</p>
20.	29. Simpan dan lihat apa yang terjadi pada console browser (Gambar 1.21). Jawaban :



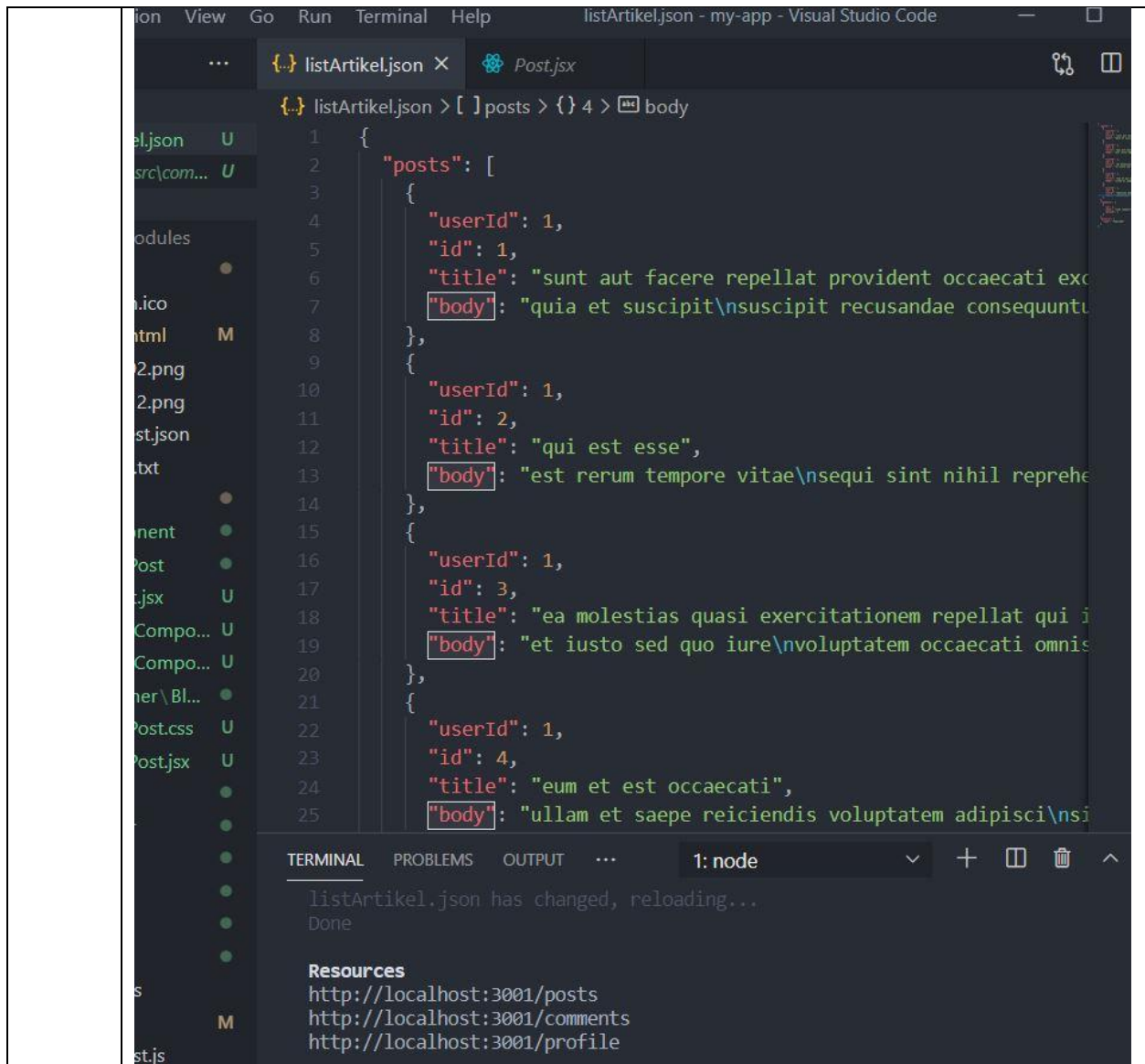
Praktikum 2

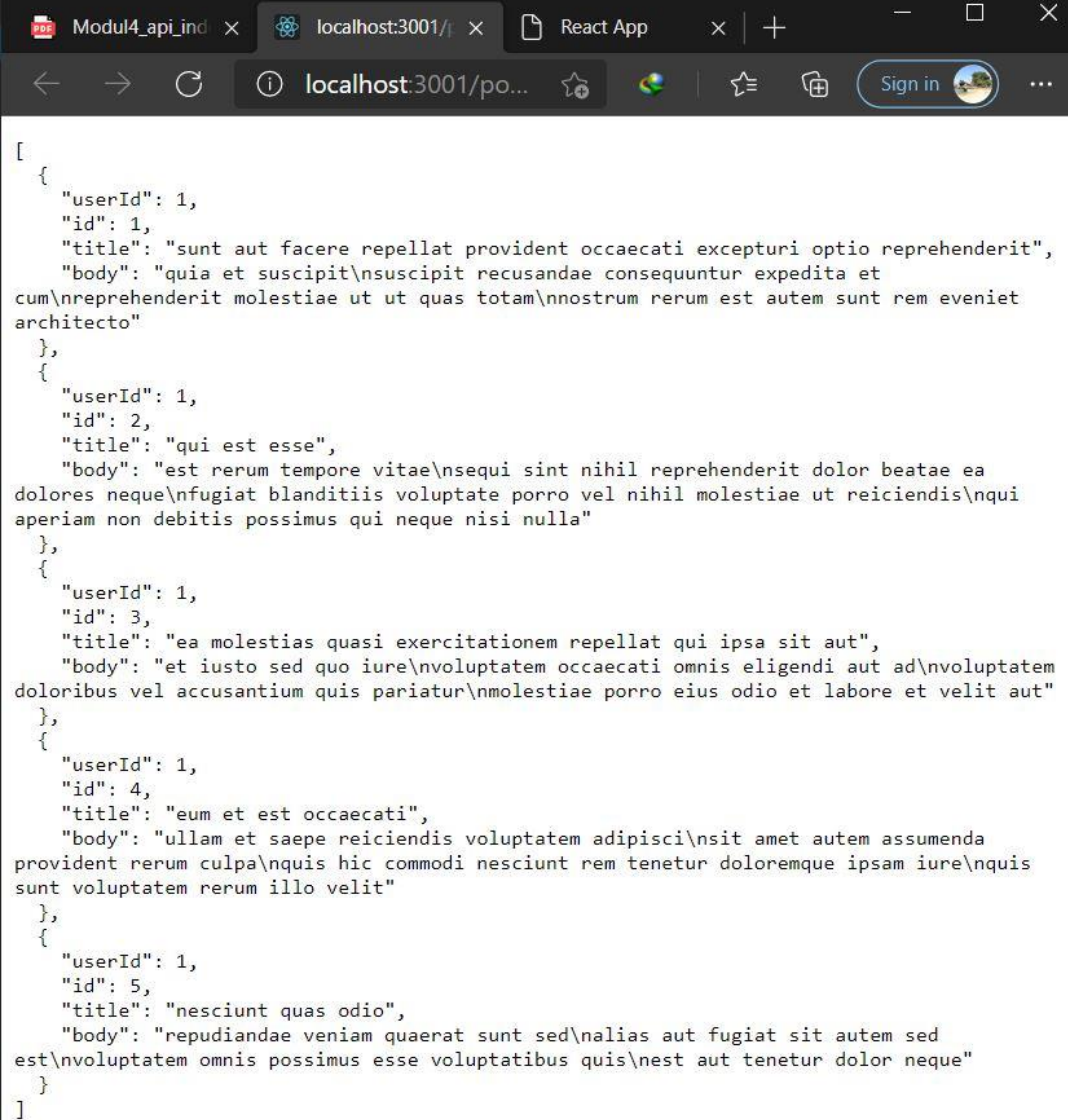
Interaksi dengan API menggunakan Fake API

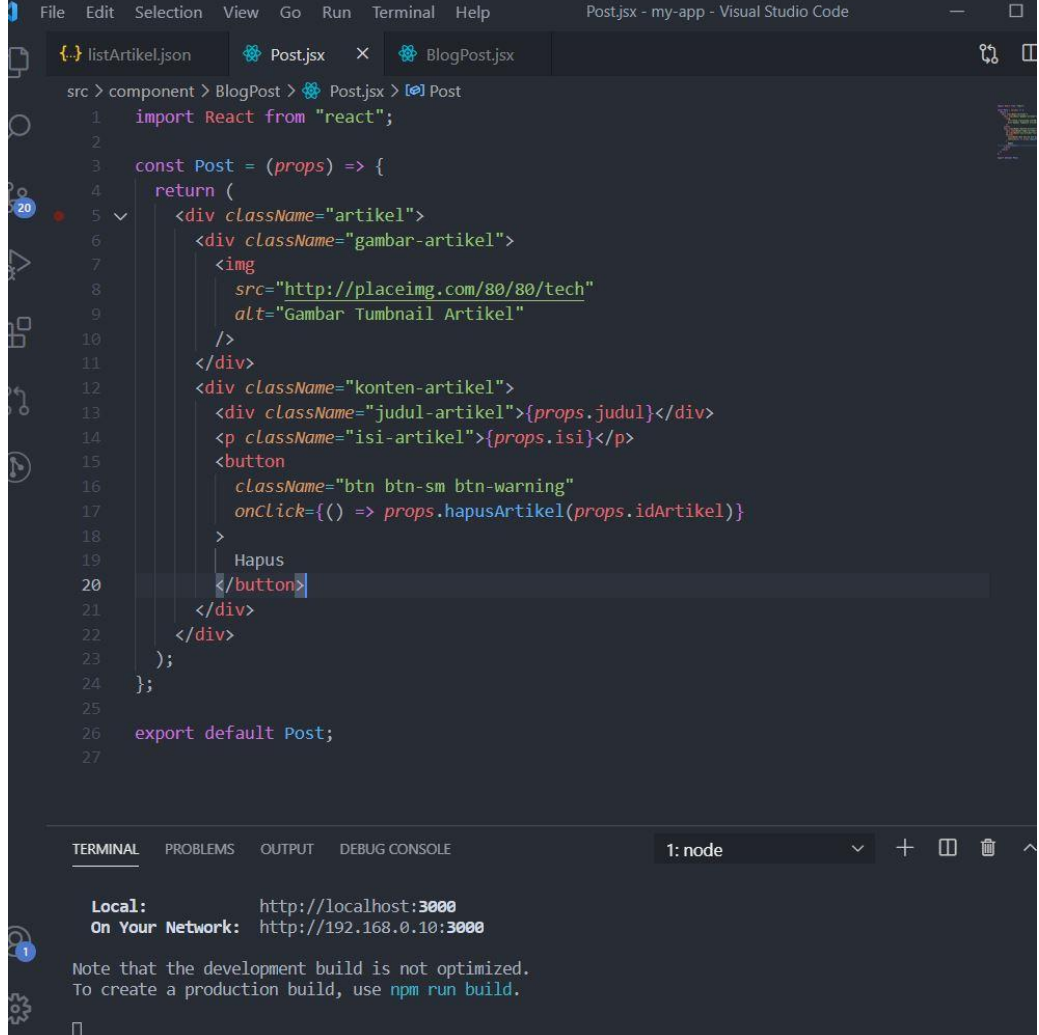
- 1 2.1 Install Fake API (JSON Server) Fake API/JSON Server bisa kita dapatkan di halaman <https://github.com/typicode/jsonserver>. Tahapan install dan membuat data json sendiri
1. Install pada direktori project reactjs kita dengan perintah `npm install -gjson-server`
Jawaban :

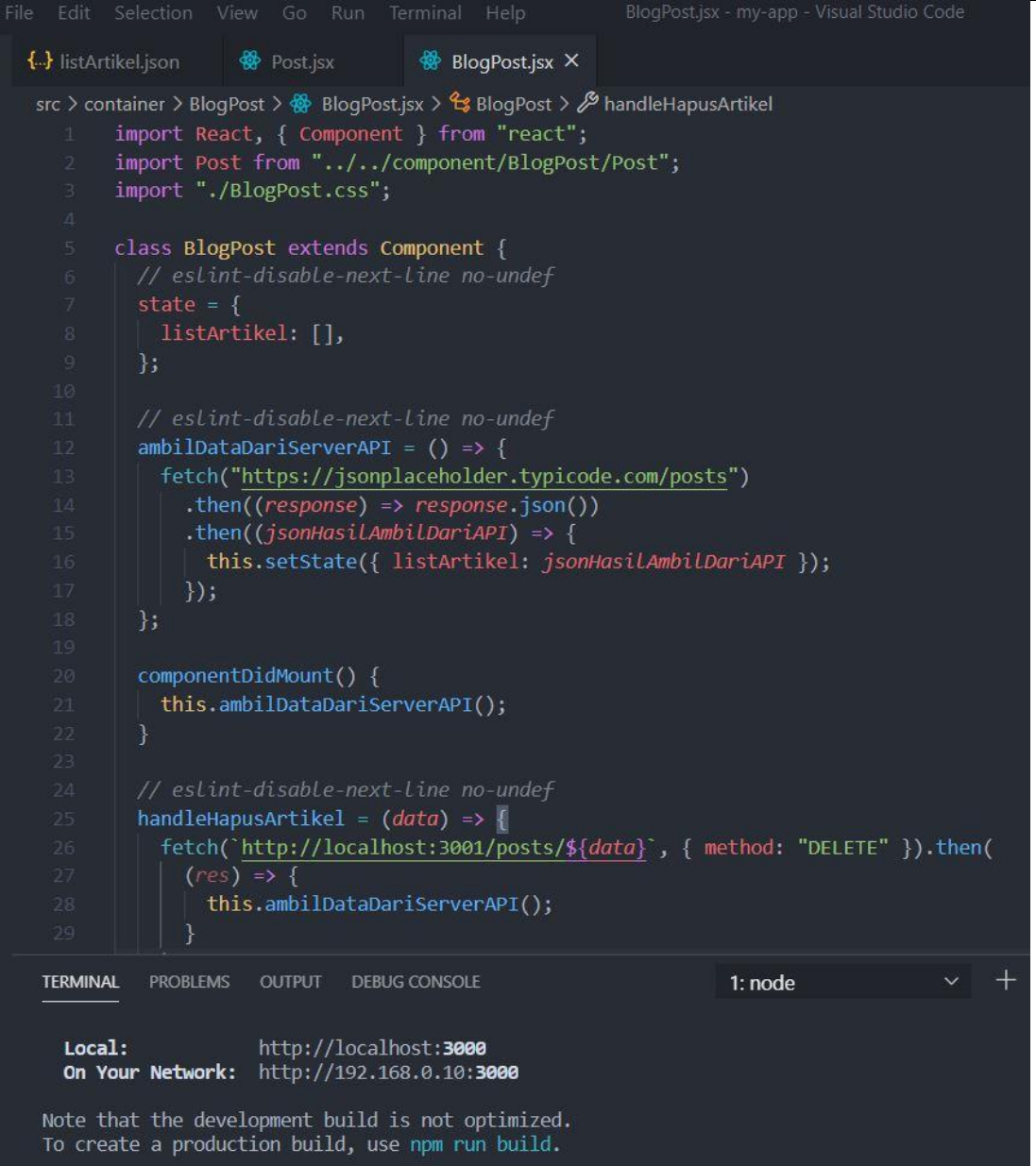
	 <pre> TERMINAL ... 1: powershell Note that the development build is not optimized. To create a production build, use <code>npm run build</code>. Terminate batch job (Y/N)? y PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app> npm install -g json-server npm WARN bootstrap@4.6.0 requires a peer of jquery@1.9.1 - 3 but none is installed. You must install peer dependencies yourself. npm WARN bootstrap@4.6.0 requires a peer of popper.js@^1.16.1 but none is installed. You must install peer dependencies yourself. npm WARN tsutils@3.20.0 requires a peer of typescript@>=2.8.0 >=3.2.0-dev >= 3.3.0-dev >= 3.4.0-dev >= 3.5.0-dev >= 3.6.0-dev >= 3.6.0-beta >= 3.7.0-dev >= 3.7.0-beta but none is installed. You must install peer dependencies yourself. npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.3.2 (node_modules\fsevents): npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"}) npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\watchpack-chokidar2\node_modules\fsevents): </pre>
2	<p>2. Copy-kan file json listArtikel.json yang sudah ada pada direktori project reactjs kita.</p> <p>3. Buka cmd baru pada direktori project, lalu ketik perintah <code>json-server --watch listArtikel.json --port 3001</code>.</p> <p>Jawaban :</p>  <pre> TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: powershell + CategoryInfo : ObjectNotFound: (json-server:String) [], CommandNotFoundException + ~~~~~ PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app> npm install -g json-server C:\Users\ROG SERIES\AppData\Roaming\npm\json-server -> C:\Users\ROG SERIES\AppData\Roaming\npm\node_modules\json-server\lib\cli\bin.js + json-server@0.16.3 added 182 packages from 80 contributors in 21.136s PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app> json-server --watch listArtikel.json --port 3001 </pre>
3	<p>4. Apabila pada cmd tampil seperti Gambar 2.1, maka server Fake API local kita telah siap</p> <p>Jawaban :</p>

	<pre> PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app> Get-ExecutionPolicy Restricted PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app> Set-ExecutionPolicy RemoteSigned -Scope CurrentUser PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app> json-server --watch listArtikel.json --port 3001 \{^_^}/ hi! Loading listArtikel.json Done Resources Home http://localhost:3001 Type s + enter at any time to create a snapshot of the database Watching... </pre>
4	<p>5. Kita cek url resource yang adapada Fake APIserver ke browser apakah bisa diakses. Ketik url <code>http://localhost:3001/posts</code> padabrowser</p> <p>Jawaban :</p>



	 <pre>[{ "userId": 1, "id": 1, "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit", "body": "cum\ndoloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut" }, { "userId": 1, "id": 2, "title": "qui est esse", "body": "est rerum tempore vitae\ndoloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut" }, { "userId": 1, "id": 3, "title": "ea molestias quasi exercitationem repellat qui ipsa sit aut", "body": "et iusto sed quo iure\ndoloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut" }, { "userId": 1, "id": 4, "title": "eum et est occaecati", "body": "ullam et saepe reiciendis voluptatem adipisci\ndoloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut" }, { "userId": 1, "id": 5, "title": "nesciunt quas odio", "body": "repudiandae veniam quaerat sunt sed\ndoloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut" }]</pre>
	<p>Praktikum 3</p> <p>Interaksi dengan API menggunakan method DELETE</p>
1	<p>3.1 Langkah Praktikum 3</p> <p>1. Buka stateless component Post. Tambahkan 1 baris kode program pada baris 10 seperti pada Gambar 3.1</p> <p>Jawaban :</p>

	 <pre>File Edit Selection View Go Run Terminal Help Postjsx - my-app - Visual Studio Code listArtikel.json Post.jsx BlogPost.jsx src > component > BlogPost > Post.jsx > [e] Post 1 import React from "react"; 2 3 const Post = (props) => { 4 return (5 <div className="artikel"> 6 <div className="gambar-artikel"> 7 11 </div> 12 <div className="konten-artikel"> 13 <div className="judul-artikel">{props.judul}</div> 14 <p className="isi-artikel">{props.isi}</p> 15 <button 16 className="btn btn-sm btn-warning" 17 onClick={() => props.hapusArtikel(props.idArtikel)} 18 > 19 Hapus 20 </button> 21 </div> 22 </div> 23); 24 }; 25 26 export default Post; 27</pre> <p>TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: node</p> <p>Local: http://localhost:3000 On Your Network: http://192.168.0.10:3000</p> <p>Note that the development build is not optimized. To create a production build, use <code>npm run build</code>.</p>
2	<p>2. Kemudian pada statefull component BlogPost, modifikasi kode program sebelumnya sesuai dengan Gambar 3.2</p> <p>Jawaban :</p>

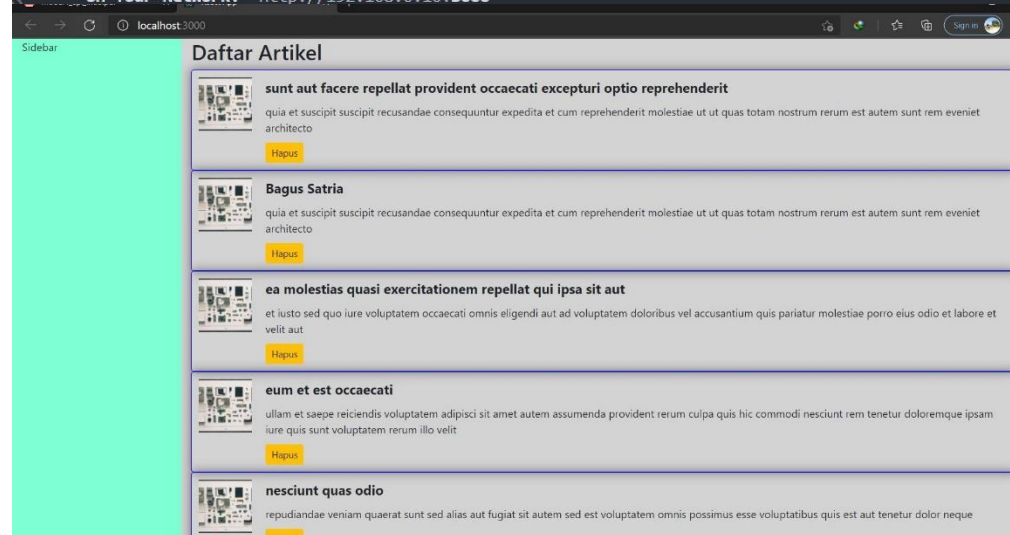
	 <pre> src > container > BlogPost > BlogPost.jsx > BlogPost > handleHapusArtikel 1 import React, { Component } from "react"; 2 import Post from "../../component/BlogPost/Post"; 3 import "./BlogPost.css"; 4 5 class BlogPost extends Component { 6 // eslint-disable-next-line no-undef 7 state = { 8 listArtikel: [], 9 }; 10 11 // eslint-disable-next-line no-undef 12 ambilDataDariServerAPI = () => { 13 fetch("https://jsonplaceholder.typicode.com/posts") 14 .then((response) => response.json()) 15 .then((jsonHasilAmbilDariAPI) => { 16 this.setState({ listArtikel: jsonHasilAmbilDariAPI }); 17 }); 18 }; 19 20 componentDidMount() { 21 this.ambilDataDariServerAPI(); 22 } 23 24 // eslint-disable-next-line no-undef 25 handleHapusArtikel = (data) => { 26 fetch(`http://localhost:3001/posts/\${data}`, { method: "DELETE" }).then(27 (res) => { 28 this.ambilDataDariServerAPI(); 29 } 30); 31 }; </pre> <p> TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: node + </p> <p> Local: http://localhost:3000 On Your Network: http://192.168.0.10:3000 </p> <p> Note that the development build is not optimized. To create a production build, use <code>npm run build</code>. </p>
3	3. Klik tombol hapus pada list artikel di browser. Amati apa yang terjadi Jawaban :

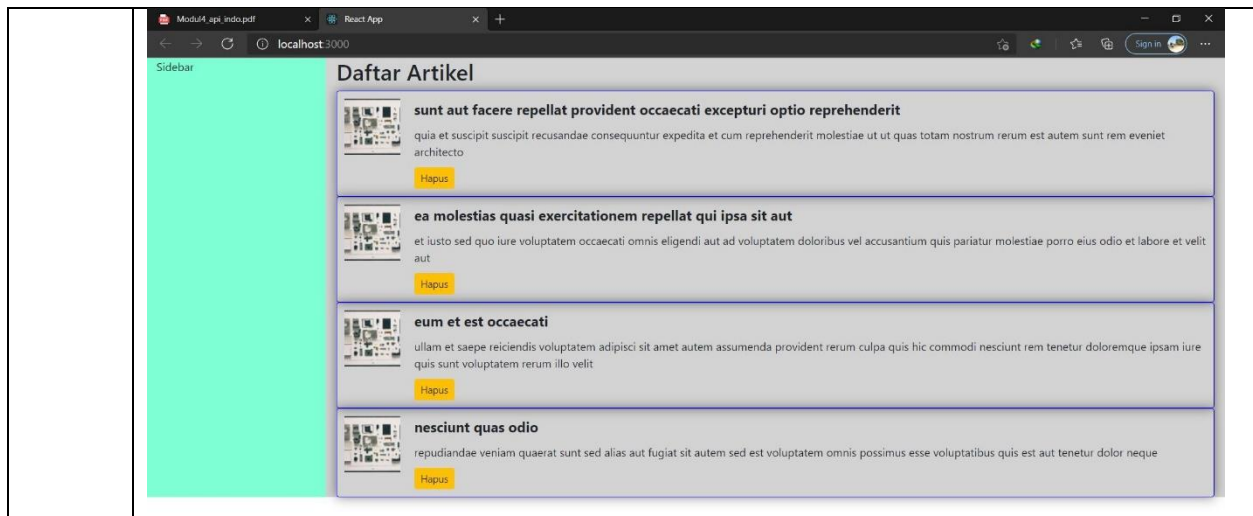
```
src > container > BlogPost > BlogPost.jsx > BlogPost > handleHapusArtikel

30  };
31  };
32
33  render() {
34    return (
35      <div className="post-artikel">
36        <h2>Daftar Artikel</h2>
37        {this.state.listArtikel.map((artikel) => {
38          return (
39            <Post
40              key={artikel.id}
41              judul={artikel.title}
42              isi={artikel.body}
43              idArtikel={artikel.id}
44              hapusArtikel={this.handleHapusArtikel}
45            />
46          );
47        })}
48      </div>
49    );
50  }
51 }
52 export default BlogPost;
53
```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: node

Local: http://localhost:3000
On Your Network: http://192.168.0.10:3000

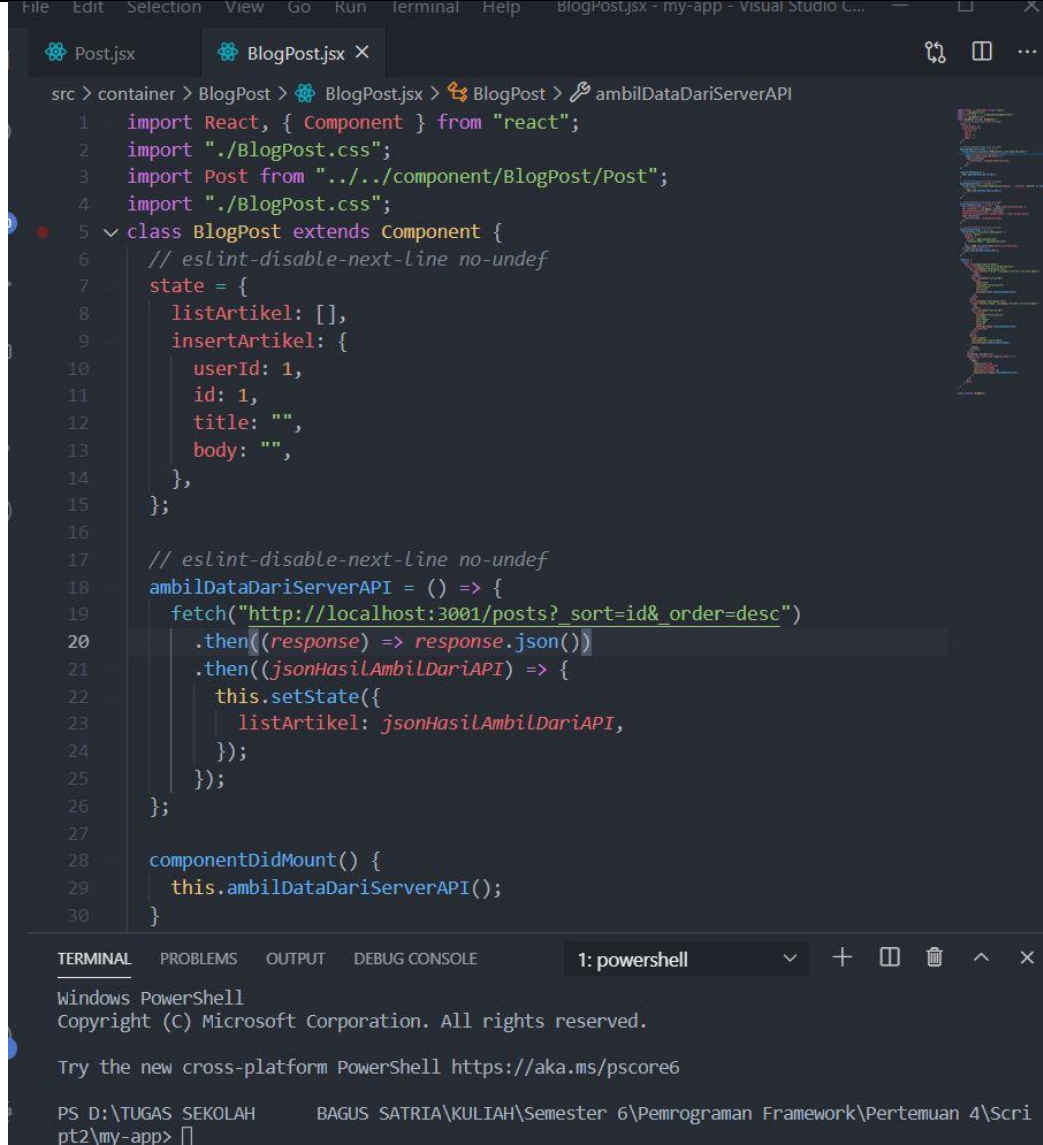


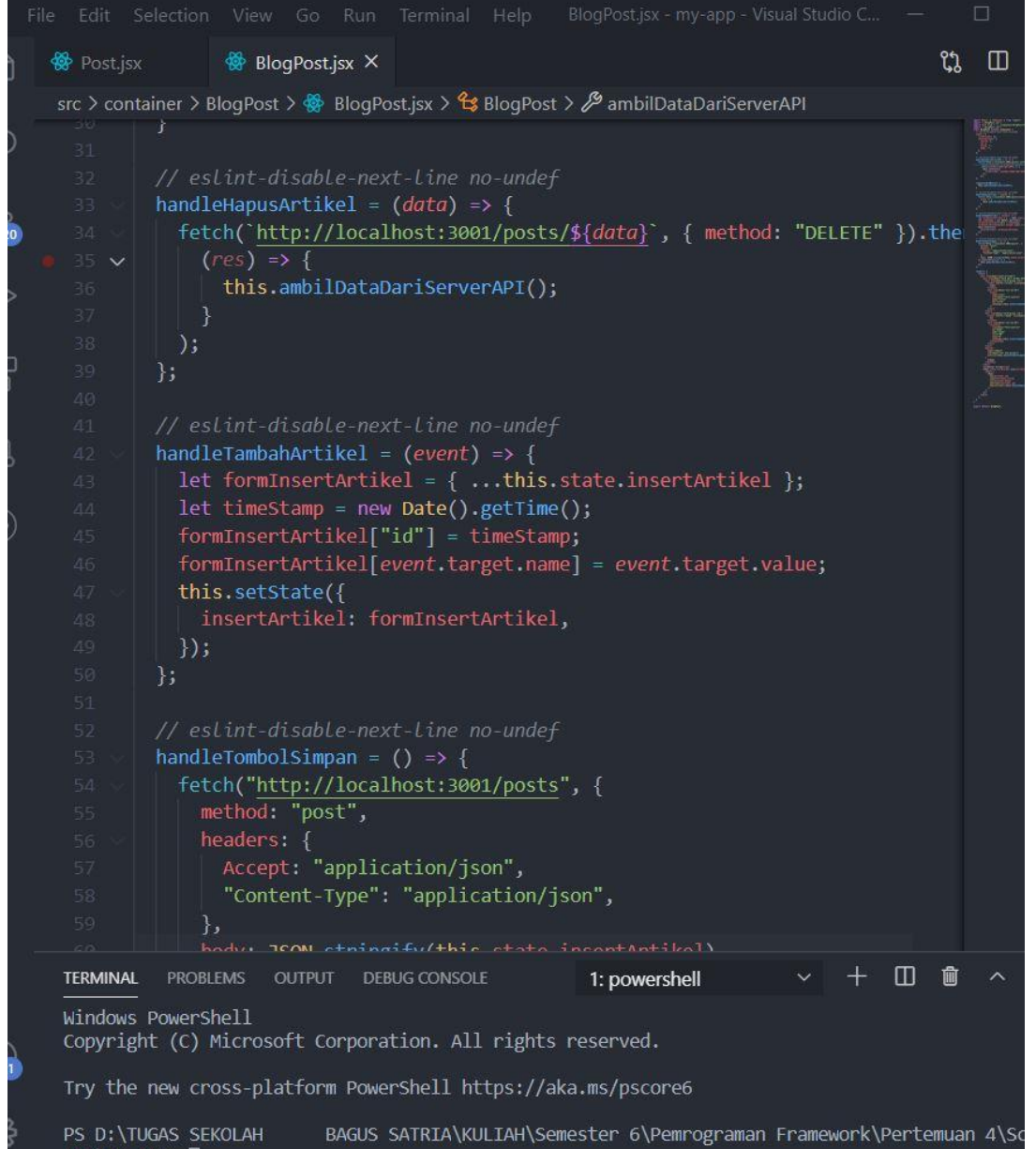


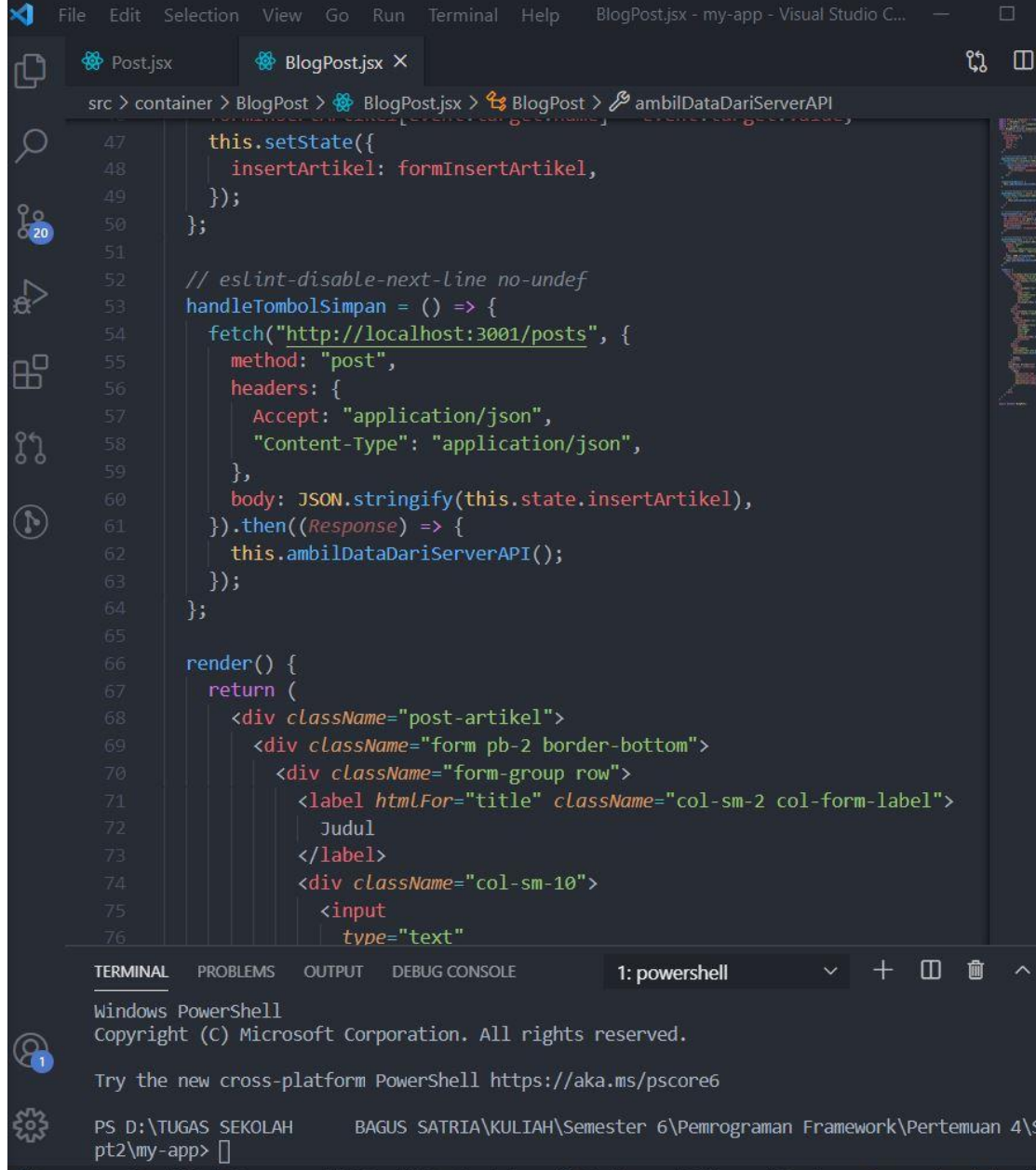
Praktikum 4

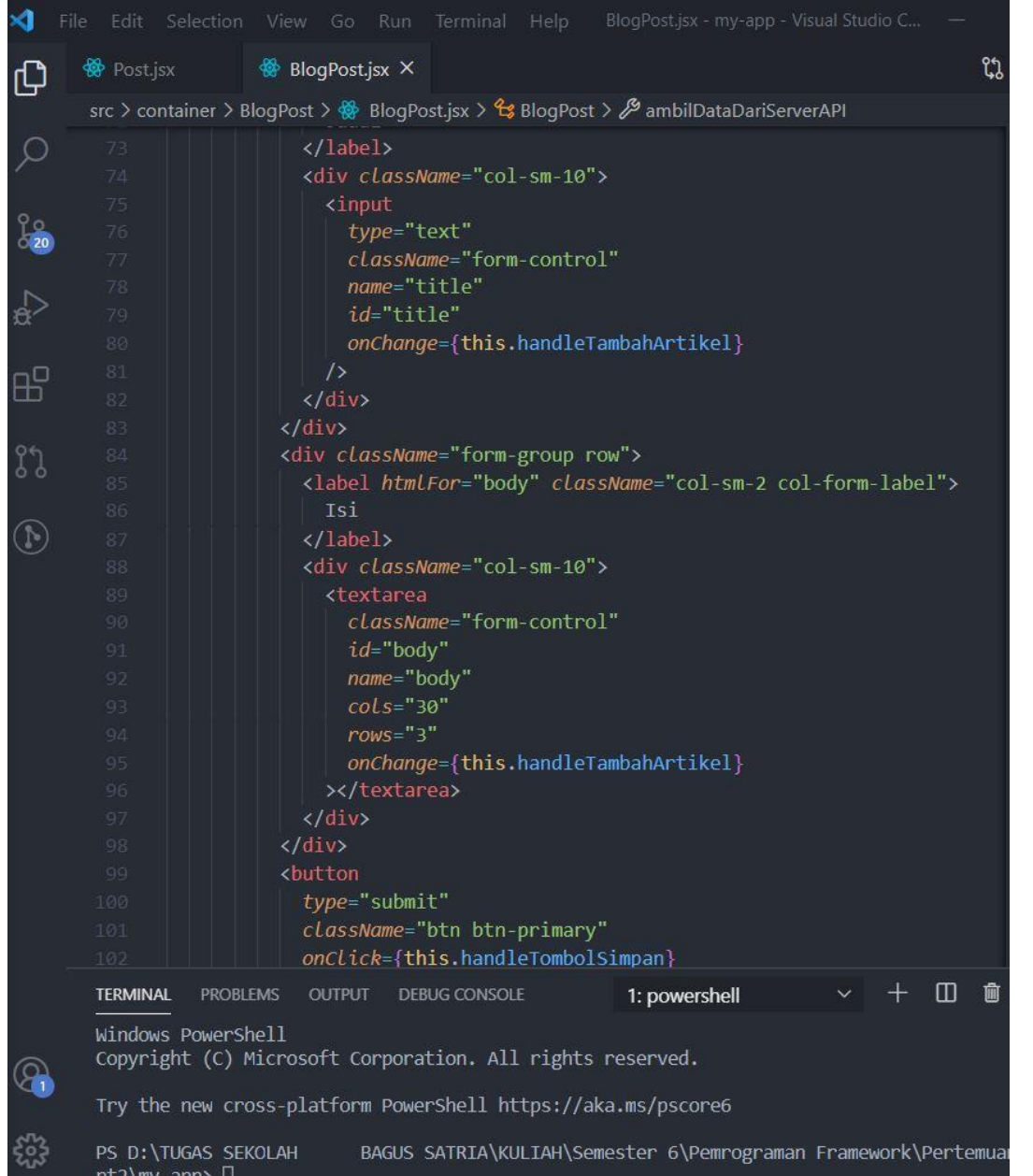
Interaksi dengan API menggunakan method POST

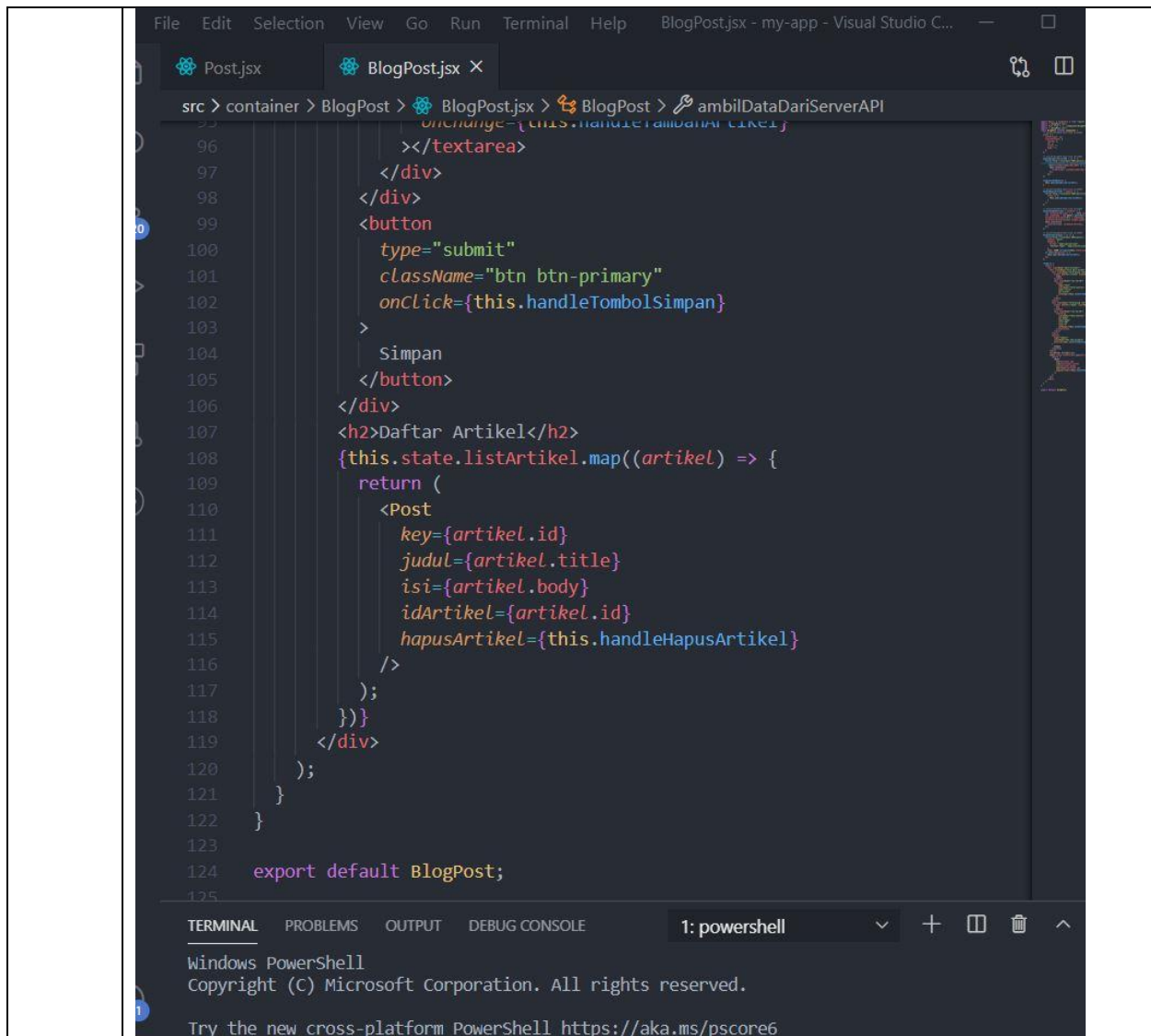
1	<p>4.1 Langkah Praktikum 4</p> <p>1. Buka statefull component BlogPost, dan modifikasi pada fungsi render() untuk menampilkan form input artikel yang berisi judul dan isi berita. seperti pada Gambar 4.1</p> <p>Jawaban :</p>
---	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

	 <pre> src > container > BlogPost > BlogPost.jsx > BlogPost > ambilDataDariServerAPI 1 import React, { Component } from "react"; 2 import "./BlogPost.css"; 3 import Post from "../../component/BlogPost/Post"; 4 import "./BlogPost.css"; 5 class BlogPost extends Component { 6 // eslint-disable-next-line no-undef 7 state = { 8 listArtikel: [], 9 insertArtikel: { 10 userId: 1, 11 id: 1, 12 title: "", 13 body: "", 14 }, 15 }; 16 17 // eslint-disable-next-line no-undef 18 ambilDataDariServerAPI = () => { 19 fetch("http://localhost:3001/posts? sort=id&_order=desc") 20 .then((response) => response.json()) 21 .then((jsonHasilAmbilDariAPI) => { 22 this.setState({ 23 listArtikel: jsonHasilAmbilDariAPI, 24 }); 25 }); 26 }; 27 28 componentDidMount() { 29 this.ambilDataDariServerAPI(); 30 } </pre> <p>TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: powershell</p> <p>Windows PowerShell Copyright (c) Microsoft Corporation. All rights reserved.</p> <p>Try the new cross-platform PowerShell https://aka.ms/pscore6</p> <p>PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Script2\my-app></p>
2	<p>2. Kemudian modifikasi BlogPost untuk bagian state dan request API dari server, seperti Gambar 4.2</p> <p>Jawaban :</p>

	 <pre> 30 } 31 32 // eslint-disable-next-line no-undef 33 handleHapusArtikel = (data) => { 34 fetch(`http://localhost:3001/posts/\${data}`, { method: "DELETE" }).then 35 (res) => { 36 this.ambilDataDariServerAPI(); 37 } 38); 39 }; 40 41 // eslint-disable-next-line no-undef 42 handleTambahArtikel = (event) => { 43 let formInsertArtikel = { ...this.state.insertArtikel }; 44 let timeStamp = new Date().getTime(); 45 formInsertArtikel["id"] = timeStamp; 46 formInsertArtikel[event.target.name] = event.target.value; 47 this.setState({ 48 insertArtikel: formInsertArtikel, 49 }); 50 }; 51 52 // eslint-disable-next-line no-undef 53 handleTombolSimpan = () => { 54 fetch("http://localhost:3001/posts", { 55 method: "post", 56 headers: { 57 Accept: "application/json", 58 "Content-Type": "application/json", 59 }, 60 body: JSON.stringify(this.state.insertArtikel) </pre> <p>TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: powershell</p> <p>Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved.</p> <p>Try the new cross-platform PowerShell https://aka.ms/pscore6</p> <p>PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\Sc...</p>
3	<p>3. Tambahkan untuk handle form tambah data artikel seperti Gambar 4.3</p> <p>Jawaban :</p>

	 <pre> 47 this.setState({ 48 insertArtikel: formInsertArtikel, 49 }); 50 }); 51 52 // eslint-disable-next-line no-undef 53 handleTombolSimpan = () => { 54 fetch("http://localhost:3001/posts", { 55 method: "post", 56 headers: { 57 Accept: "application/json", 58 "Content-Type": "application/json", 59 }, 60 body: JSON.stringify(this.state.insertArtikel), 61 }).then((Response) => { 62 this.ambilDataDariServerAPI(); 63 }); 64 }; 65 66 render() { 67 return (68 <div className="post-artikel"> 69 <div className="form pb-2 border-bottom"> 70 <div className="form-group row"> 71 <label htmlFor="title" className="col-sm-2 col-form-label"> 72 Judul 73 </label> 74 <div className="col-sm-10"> 75 <input 76 type="text" </pre> <p>TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE 1: powershell</p> <p>Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved.</p> <p>Try the new cross-platform PowerShell https://aka.ms/pscore6</p> <p>PS D:\TUGAS SEKOLAH BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan 4\pt2\my-app></p>
4	<p>4. Langkah terakhir tambahkan fungsi untuk handle tombol simpan artikel, seperti pada Gambar 4.4</p> <p>Jawaban :</p>

	 <p>The screenshot shows the Visual Studio Code interface. The main editor window displays the <code>BlogPost.jsx</code> file with the following code:</p> <pre> 73 </label> 74 <div className="col-sm-10"> 75 <input 76 type="text" 77 className="form-control" 78 name="title" 79 id="title" 80 onChange={this.handleTambahArtikel} 81 /> 82 </div> 83 </div> 84 <div className="form-group row"> 85 <label htmlFor="body" className="col-sm-2 col-form-label"> 86 Isi 87 </label> 88 <div className="col-sm-10"> 89 <textarea 90 className="form-control" 91 id="body" 92 name="body" 93 cols="30" 94 rows="3" 95 onChange={this.handleTambahArtikel} 96 >>/textarea> 97 </div> 98 </div> 99 <button 100 type="submit" 101 className="btn btn-primary" 102 onClick={this.handleTombolSimpan} </pre> <p>At the bottom, the TERMINAL panel shows a PowerShell session:</p> <pre> 1: powershell Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved. Try the new cross-platform PowerShell https://aka.ms/pscore6 PS D:\TUGAS SEKOLAH\BAGUS SATRIA\KULIAH\Semester 6\Pemrograman Framework\Pertemuan </pre>
5	<p>5. Simpan, lakukan percobaan penambahan data, dan amati perubahannya</p> <p>Jawaban :</p>



1.4 Pertanyaan Praktikum 1

a. Pada langkah 8, sekarang coba kalian ganti class container dengan container-fluid atau sebaliknya pada file "public/index.html" dan lihat apa perbedaannya.

1. Tampilan seperti apa yang kalian temukan setelah mencoba mengganti nama class tersebut?

2. Apa perbedaan dari container dan container-fluid ?

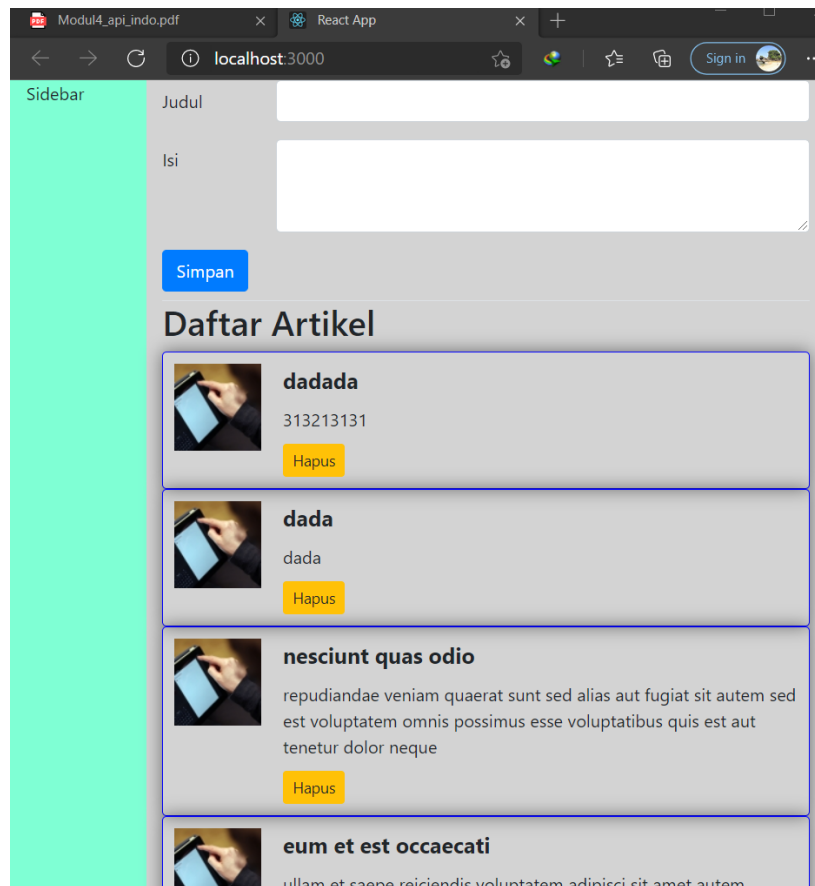
b. Jika kita ingin meng-import suatu component contoh component bootstrap, akan tetapi component dalam tersebut belum terdapat pada module ReactJS. Apa yang akan dilakukan untuk dapat menggunakan component tersebut? Bagaimana caranya?

Jawaban

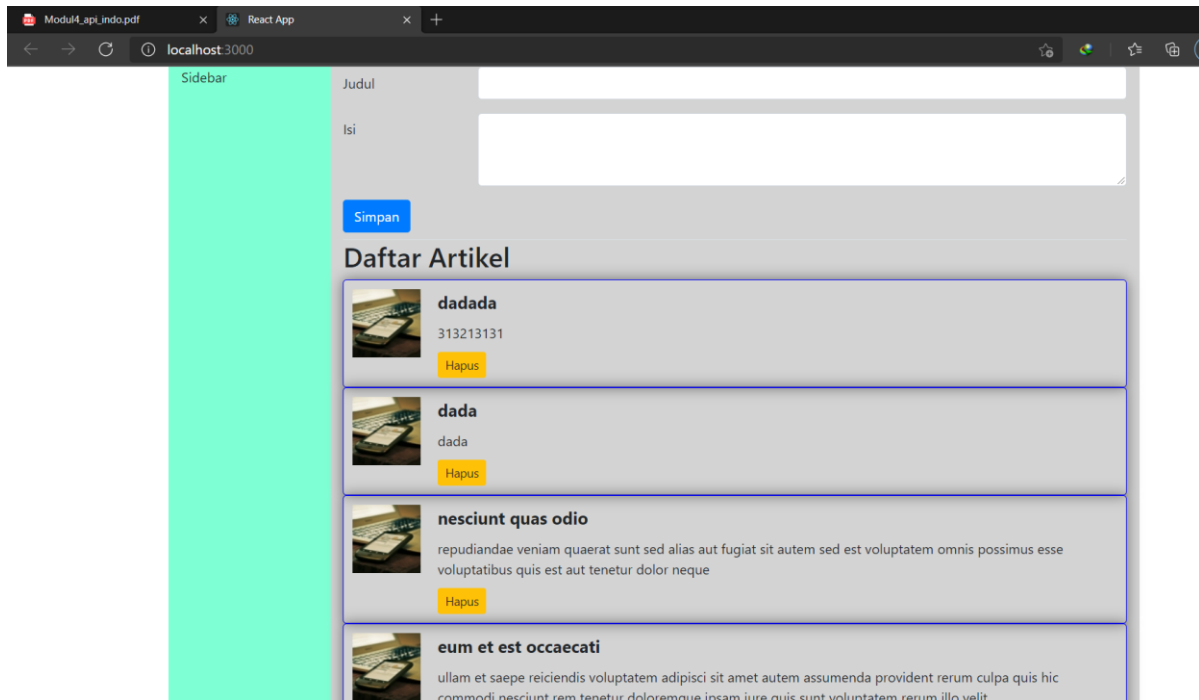
a.

1.

Container-fluid



Container



2. **container** berukuran tidak penuh, dan letaknya di tengah, maka **container-fluid** adalah kebalikannya, **container-fluid** memiliki lebar penuh.

b.

npm install bootstrap, atau download .zip bootstrap lalu copas di project kita.

2.2 Pertanyaan Praktikum 2

a. Kenapa json-server dijalankan pada port 3001? Kenapa tidak sama-sama dijalankan pada port 3000 seperti project react yang sudah kita buat?

b. Bagaimana jadinya kalau kita ganti port json-server menjadi 3000?

Jawaban

- Node.js dengan default maka port 3000 apabila json-server ingin menggunakan port 300 tidak bisa karena port 3000 sudah dipakai oleh node.js
- fungsi untuk mendirikan sebuah server REST API tiruan dengan usaha minimal. maka kita harus matikan node.js dulu

3.2 Pertanyaan Praktikum 3

- a. Apa yang terjadi setelah kalian klik tombol hapus?
- b. Perhatikan file listArtikel.json, apa yang terjadi pada file tersebut? Kenapa demikian?
- c. Fungsi handleHapusArtikel itu untuk apa?
- d. Jelaskan perbedaan fungsi componentDidMount() pada Gambar 1.18 dengan fungsi componentDidMount() pada Gambar 3.2 ?

Jawaban

- a. maka list terhapus
- b. apabila klik hapus di server secara otomatis data json di listArtikel terhapus
- c. melakukan panggilan data server lalu menggunakan method DELETE apabila salah satu data terhapus sesuai kita pilih maka langsung menjalankan fungsi ambilDataDariServer untuk melakukan update data json.
- d. componentDidMount pada Gambar 1.18 mengecek Ketika component telah dimounting maka panggil API, dengan mengubah response data dari URL API menjadi sebuah data json, lalu pada akhirnya data json hasilnya di masukkan ke dalam list Artikel pada state. componentDidMount pada Gambar 3.2 untuk memudahkan dan membuat fungsi sendiri.

4.2 Pertanyaan Praktikum 4

- a. Jelaskan apa yang terjadi pada file listArtikel.json sebelum dan setelah melakukan penambahan data?
- b. Data yang ditampilkan di browser adalah data terbaru berada di posisi atas dan data lama berada di bawah, sedangkan pada file listArtikel.json data terbaru malah berada di bawah. Jelaskan mengapa demikian?

Jawaban

- a. apabila melakukan penambahan data maka secara otomatis di file listAstikel.json tambah data sendiri, apabila melakukan pengurangan data maka secara otomatis di file listAstikel.json kurang data sendiri
- b. dikarenakan kita menulis script "http://localhost:3001/posts?_sort=id&_order=desc" yang artinya data apabila kita insert maka secara otomatis di list artikel yang terbaru diatas

TUGAS PRAKTIKUM Modul 4

Buatlah program menggunakan Fake API (JSON Server) tentang pendataan Mahasiswa aktif/cuti/lulus di Jurusan Teknologi Informasi. Atribut-atribut yang ada dari mahasiswa adalah NIM, nama, alamat, no hp, tahun Angkatan, dan status. Buatlah aplikasi yang menggunakan API dengan method GET, DELETE, dan POST.

The screenshot shows a web application running on localhost:3000. On the left is a cyan sidebar with the word "idebar". The main area contains a form with the following fields: "nama" (filled with "bagus123"), "hp" (filled with "3123131"), "status" (filled with "aktif"), "angkatan" (filled with "2018"), and "alamat" (filled with "Sawojajar"). Below the form is a blue "Simpan" button. Underneath the form is a section titled "Daftar Mahasiswa" containing a list of students. The first student is "bagus123" with NIM "3123131", year "2018", status "aktif", and address "Sawojajar". There is a yellow "Hapus" button next to this entry. The second student is "bagus satria".

Nama	NIM	Tahun Angkatan	Status	Alamat
bagus123	3123131	2018	aktif	Sawojajar
bagus satria				

This screenshot shows the same web application after a second student has been added. The form fields are now empty, with "angkatan" containing "2018" and "alamat" containing "Sawojajar". The "Daftar Mahasiswa" section now lists two students. The first student is "bagus123" with NIM "3123131", year "2018", status "aktif", and address "Sawojajar". The second student is "bagus satria" with NIM "1234", year "2021", status "lulus", and address "sawojajar malang". A yellow "Hapus" button is visible next to the first student's entry.

Nama	NIM	Tahun Angkatan	Status	Alamat
bagus123	3123131	2018	aktif	Sawojajar
bagus satria	1234	2021	lulus	sawojajar malang

Link Github dan youtube

1. Session 1 Pemrograman Berbasis Framework
<https://youtu.be/RlfXRslwKos>
<https://github.com/BagusSatria123/PemrogramanFrameworkReact/tree/master/Modul%201>
2. Session 2 Modern JavaScript Pemrograman Berbasis Framework
<https://youtu.be/POzDchFJAU4>
<https://github.com/BagusSatria123/PemrogramanFrameworkReact/tree/master/Pertemuan%202>
3. Session 3: React Component Pemrograman Berbasis Framework
<https://youtu.be/BlxF-eVq28I>
<https://github.com/BagusSatria123/PemrogramanFrameworkReact/tree/master/Pertemuan%203>
4. Session 4 API Pemrograman Berbasis Framework
<https://youtu.be/J-fZymTnQU4>
<https://github.com/BagusSatria123/PemrogramanFrameworkReact/tree/master/Pertemuan%204>

--Alhamdulillah--