

Dokumentasi Project Mobile IF-Kelas5

Nama : Lorensius Firngadi

NPM : 24382001P

Judul : Aplikasi Cetak Resi

Deskripsi Aplikasi :

Belajar informatika jadi lebih seru dengan “Informatika Kelas 5”! Aplikasi ini membantu siswa memahami materi lewat pembelajaran interaktif, latihan soal, dan fitur akun guru-siswa untuk memantau perkembangan belajar.

A. Membuat Project Backend (API)

1. Inisialisasi Project Next.js

Ketik kode di Terminal :

```
npx create-next-app@latest api --typescript
```

Pilih:

- App Router
- TypeScript
- ESLint
- src/ directory

2. Menjalankan Server Awal

Ketik kode di Terminal :

```
npm run dev
```

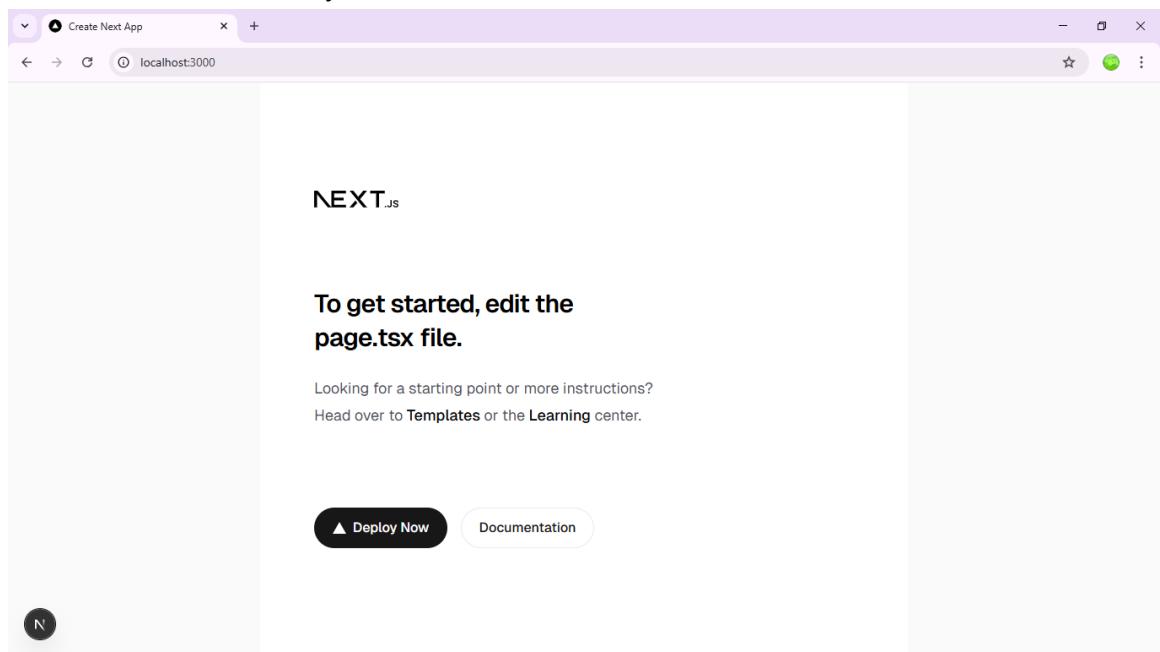
The screenshot shows the Visual Studio Code interface with the following details:

- EXPLORER:** Shows the project structure for "IF-KELAS5". It includes an "api" folder containing ".next", "node_modules", and "prisma" (with "migrations" and "schema.prisma"). It also has a "src" folder with "app" (containing "api", "favicon.ico", "globals.css", "layout.tsx", and "page.tsx"), "lib" (containing "jwt.ts" and "prisma.ts"), and an ".env" file.
- TERMINAL:** Shows the command "npm run dev" being run, which starts a Next.js development server at port 3000. The output indicates the server is ready on both local and network ports, and shows a successful GET request to the root URL.

Buka di Chrome atau Browser lainnya :

<http://localhost:3000>

Jika muncul halaman Next.js → berhasil



3. Install Prisma & Library Pendukung

Ketik di terminal kode ini :

```
npm install prisma @prisma/client  
dan  
npm install bcryptjs jsonwebtoken
```

Untuk TypeScript:

```
npm install -D @types/jsonwebtoken
```

4. Setup Prisma & Database

Inisialisasi Prisma

Kode :

```
npx prisma init
```

Setelah selesai akan dibuatkan folder : prisma/schema.prisma dan .env

5. Konfigurasi .env

The screenshot shows the Visual Studio Code (VS Code) interface. On the left, the code editor displays the contents of a `.env` file:

```
api > .env
1 // This file does not import or use any Prisma Client code - it's just environment variables.
2 # To load environment variables from .env files: https://pris.ly/prisma-config-env-vars.
3
4
5 # Prisma supports the native connection string format for PostgreSQL, MySQL, SQLite, SQL Server, MongoDB and
6 # See the documentation for all the connection string options: https://pris.ly/d/connection-strings
7
8 # The following `prisma:postgres` URL is similar to the URL produced by running a local Prisma Postgres
9 # server with the `prisma dev` CLI command, when not choosing any non-default ports or settings. The API key
10 # one found in a remote Prisma Postgres URL, does not contain any sensitive information.
11
12 DATABASE_URL="postgresql://postgres:postgres@localhost:5432/if_kelas5"
13 JWT_SECRET="rahasia_super_kelas5"
14
15
```

The Explorer sidebar on the right shows the project structure for "IF-KELAS5". The `.env` file is selected in the tree view.

The bottom status bar indicates the file is `Ln 12, Col 71`, has `Spaces: 4`, and is in `UTF-8` encoding.

Untuk menyimpan data sensitif dan konfigurasi aplikasi secara terpisah dari kode sumber (source code). Menggunakan format untuk keamanan, Fleksibilitas dan Standar dalam pengembangan perangkat lunak modern.

6. Membuat Schema Database

Kode :

```
npx prisma migrate dev --name init
```

7. Migrasi Database

8. Membuka Prisma Studio

Kode :

```
npx prisma studio
```

9. Setup Prisma Client

```
src/lib/prisma.ts
```

```
TS prisma.ts
api > lib > TS prisma.ts > ...
1 import { PrismaClient } from "@prisma/client";
2
3 const globalForPrisma = globalThis as unknown as {
4   prisma: PrismaClient | undefined;
5 };
6
7 export const prisma =
8   globalForPrisma.prisma ??
9   new PrismaClient({
10   log: ["query"],
11 });
12
13 if (process.env.NODE_ENV !== "production") {
14   globalForPrisma.prisma = prisma;
15 }
16
```

File Edit Selection View Go Run ... 🔍 IF-Kelas5

prisma.ts

api > lib > TS prisma.ts > ...

1 import { PrismaClient } from "@prisma/client";

2

3 const globalForPrisma = globalThis as unknown as {

4 prisma: PrismaClient | undefined;

5 };

6

7 export const prisma =

8 globalForPrisma.prisma ??

9 new PrismaClient({

10 log: ["query"],

11 });

12

13 if (process.env.NODE_ENV !== "production") {

14 globalForPrisma.prisma = prisma;

15 }

16

IF-KELAS5

api

prisma

ts seed.ts

public

src

app

api

favicon.ico

globals.css

layout.tsx

page.tsx

lib

ts jwt.ts

TS prisma.ts

.env

.gitignore

JS eslint.config.mjs

TS next-env.d.ts

TS next.config.ts

(1) package-lock.json

(1) package.json

JS postcss.config.mjs

TS tsconfig.json

Dokumentasi Project IF-Kelas5

~Skumtasi_Mobile_Lorenzus Firdausi.docx

~WRL2321tmp

DOC Dokumentasi_Mobile_Lorenzus Firdausi_M

OUTLINE

TERMINAL

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

powerShell - api

GET / 200 in 10.6s (compile: 6.2s, render: 4.5s)

GET / 200 in 314ms (compile: 36ms, render: 277ms)

PS C:\PA-Mobile\IF-kelas5\api>

Tanitech (3 weeks ago) Ln 11, Col 6 Spaces: 2 UTF-8 CRLF ⚡ TypeScript

Menginisialisasi Prisma Client untuk menghubungkan kode program dengan database. Berfungsi : Membuat variabel yang bisa diimpor di file lain untuk melakukan query, dan Menghindari pembuatan instance baru setiap kali refresh saat proses development.

10. Setup JWT

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the right, it shows the project structure for "Project IF-Kelas5". The "prisma" folder contains "seed.ts", "public", "src" (which includes "app", "api", "favicon.ico", "# globals.css", "layout.tsx", "page.tsx"), "lib" (which includes "jwt.ts"), and "prisma.ts". Other files like ".env", ".gitignore", "eslint.config.mjs", "next-env.dts", "next.config.ts", "package-lock.json", "package.json", "postcss.config.mjs", "README.md", "tsconfig.json", and "Dokumentasi_Mobile_Lorenzus Firmqadi.docx" are also listed.
- Code Editor:** The main area displays the "prisma.ts" file content, which defines a PrismaClient instance named "prisma" and sets it as the globalForPrisma variable if the environment is not production.
- Bottom Status Bar:** Shows the current file path as "C:\PA-Mobile\IF-Kelas5\api\prisma.ts", the file name as "prisma.ts", and the status "master".

Sebuah Utility Module yang berfungsi untuk menangani sistem keamanan login menggunakan JSON Web Token (JWT). Mengambil kunci rahasia dari variabel lingkungan (.env).

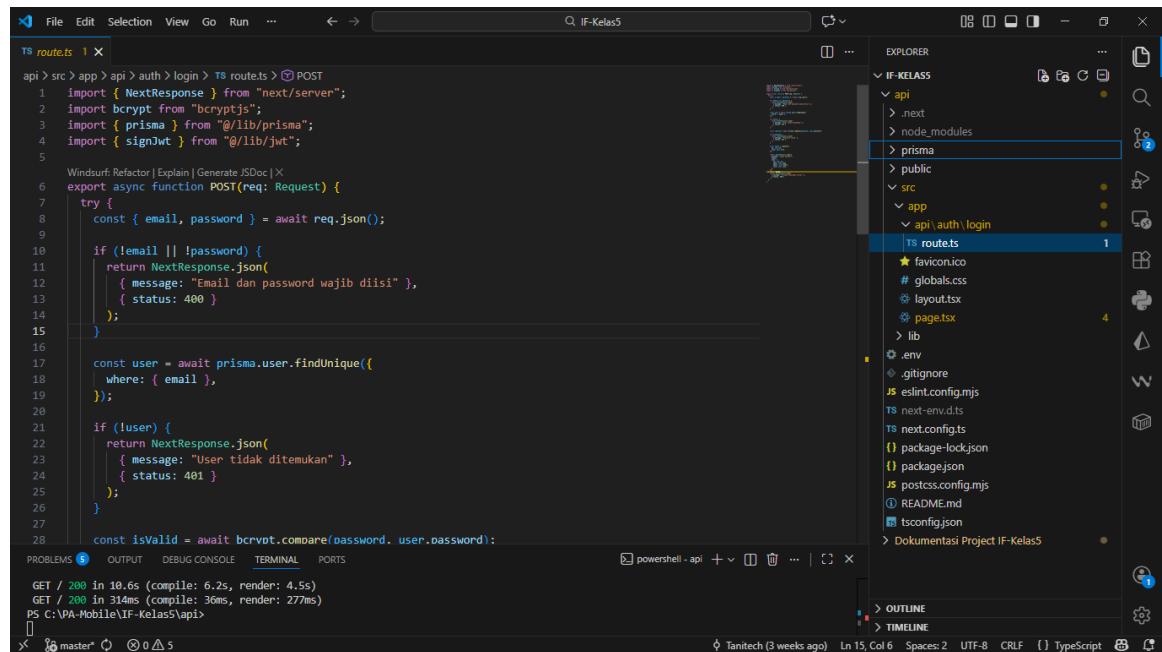
11. Membuat API Login

Struktur Folder API :

`src/app/api/auth/login/route.ts`.

Kode API Login

`route.ts`



The screenshot shows the Visual Studio Code interface with the following details:

- File Path:** api > src > app > api > auth > login > route.ts
- Code Content:** The code is a TypeScript file for a POST endpoint. It imports NextResponse, bcryptjs, prisma, and signJwt. It checks if email and password are provided, then finds a user by email. If no user is found, it returns a 401 error. It then compares the provided password with the hashed one in the database using bcrypt.compare.
- Terminal:** Shows recent API requests and responses.
- Explorer:** Shows the project structure with files like .next, node_modules, prisma, public, src, app, and api\auth\login.
- Status Bar:** Shows the current branch (master), commit count (0), and other development details.

Logika Backend untuk Proses Login fungsi utamanya :

1. Validasi & Keamanan: Memastikan email/password sudah diisi dan memeriksa apakah password yang dimasukkan cocok dengan yang ada di database (menggunakan enkripsi bcrypt).
2. Identifikasi User: Mencari data pengguna di database menggunakan Prisma berdasarkan email yang dimasukkan.
3. Pemberian Izin (Token): Jika data cocok, fungsi ini membuatkan JWT (JSON Web Token) sebagai "kartu akses" digital agar user bisa mengakses fitur lain tanpa harus login ulang terus-menerus.

Alurnya: Input Data > Cek Database > Verifikasi Password > Kirim Token Akses.

12. Testing API dengan Apidog

The screenshot shows the Apidog interface. On the left sidebar, there are icons for APIs, Tests, Publish Docs, History, Settings, and Invite. The main area shows a POST request to 'http://localhost:3000 /api/auth/login'. The request body is set to JSON with the following content:

```

1 {
2   "email": "guru@ifkelas5.com",
3   "password": "123456"
4 }
5

```

The response tab shows a status of 500 Internal Server Error. The response body contains a large amount of HTML code, indicating a server-side error.

Untuk sementara apidog masih error.

B. Membuat Project Frontend (Folder mobile)

1. Membuat Project React Native (Expo)

Jalankan pada terminal folder mobile :

npx create-expo-app . -t expo-template-blank-typescript

The screenshot shows the VS Code interface with a file named 'route.ts' open in the editor. The terminal at the bottom shows the command 'npx create-expo-app . -t expo-template-blank-typescript' being run, followed by a list of npm warnings about deprecated glob versions. The output ends with 'Your project is ready!' and instructions to run the project using npm commands like 'npm run android', 'npm run ios', or 'npm run web'. The Explorer sidebar on the right shows a project structure with a 'mobile' folder containing files like 'App.tsx', 'index.tsx', 'package.json', and 'tsconfig.json'.

Proses install expo selesai.

2. Install Dependency

Navigasi & API

Kode :

npm install axios

npm install @react-navigation/native

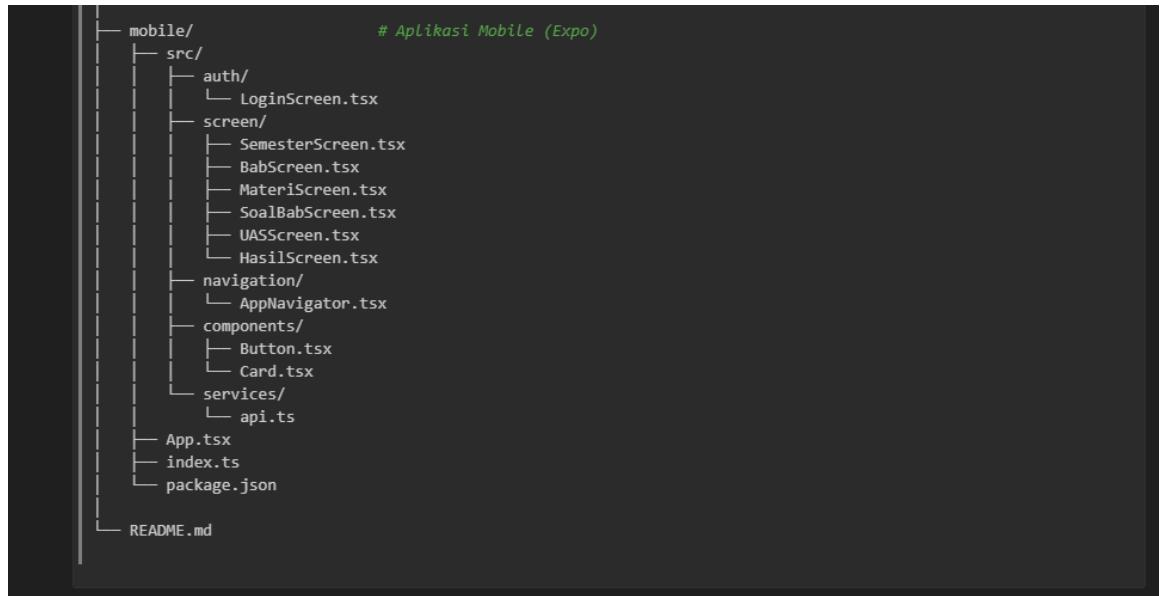
npm install @react-navigation/native-stack

Dependency Expo Navigation

npx expo install react-native-screens react-native-safe-area-context

3. Susunan Struktur Folder

```
Struktur Folder Project
IF-Kelas5/
  api/                               # Backend API (Next.js + Prisma)
    prisma/
      schema.prisma
      seed.ts
  src/
    app/
      api/
        auth/
          login/
            route.ts
        guru/
          semester/
            route.ts
          bab/
            route.ts
          materi/
            route.ts
        siswa/
          semester/
            route.ts
          bab/
            route.ts
      lib/
        prisma.ts
        jwt.ts
        auth.ts
    .env
    package.json
```



4. Konfigurasi API (services/api.ts)

```

TS route.ts 1  TS page.tsx 4  TS apits x
mobile > src > services > apits > api
1 import axios from "axios";
2
3 const api = axios.create({
4   baseURL: "http://localhost:3000/api", // ganti IP jika pakai HP
5   headers: {
6     "Content-Type": "application/json",
7   },
8 });
9
10 export default api;

```

EXPLORER

- IF-KELAS5
- > api
- > Dokumentasi Project IF-Kelas5
- mobile
- > assets
- > node_modules
- > src\services
- TS apits**
- .gitignore
- app.json
- App.tsx
- index.ts
- package-lock.json
- package.json
- tsconfig.json

PROBLEMS 0 OUTPUT DEBUG CONSOLE TERMINAL PORTS

> npm install

changed 1 package, and audited 756 packages in 13s

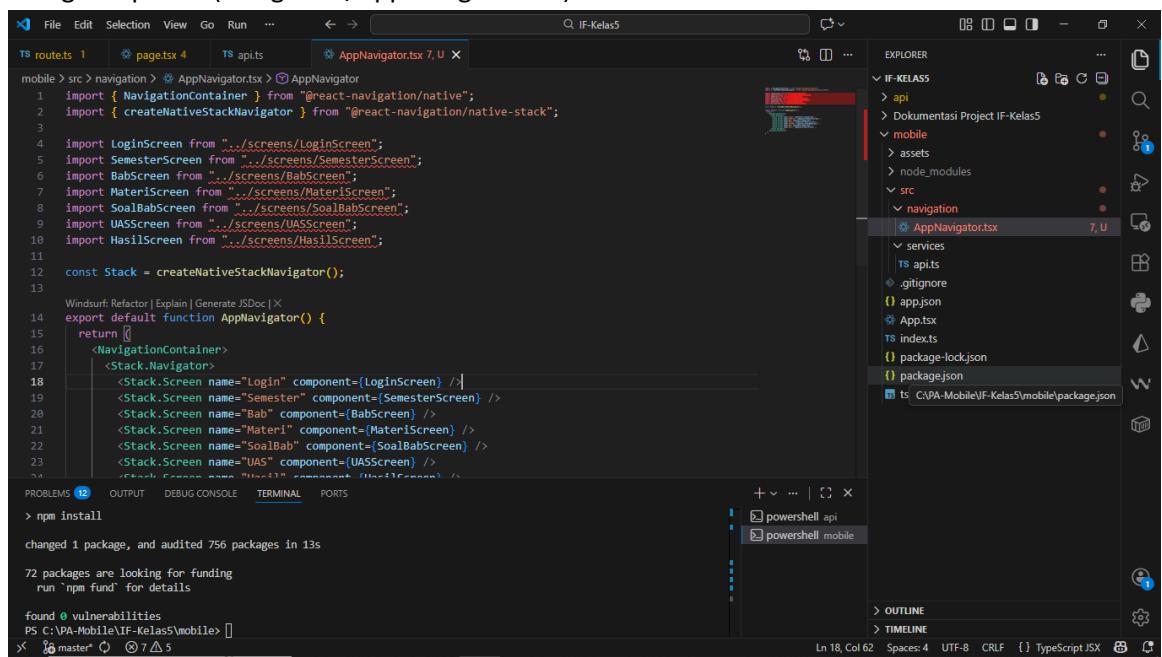
72 packages are looking for funding
run 'npm fund' for details

found 0 vulnerabilities

PS C:\PA-Mobile\IF-Kelas5\mobile>

Berfungsi sebagai konfigurasi terpusat (instance) untuk melakukan HTTP request (seperti GET, POST, DELETE) dari aplikasi ke server backend.

5. Navigasi Aplikasi (navigation/AppNavigator.tsx)



```

mobile > src > navigation > AppNavigator.tsx > AppNavigator
1 import { NavigationContainer } from "@react-navigation/native";
2 import { createNativeStackNavigator } from "@react-navigation/native-stack";
3
4 import LoginScreen from "../screens/LoginScreen";
5 import SemesterScreen from "../screens/SemesterScreen";
6 import BabScreen from "../screens/BabScreen";
7 import MateriScreen from "../screens/MateriScreen";
8 import SoalBabScreen from "../screens/SoalBabScreen";
9 import UASScreen from "../screens/UASScreen";
10 import HasilScreen from "../screens/HasilScreen";
11
12 const Stack = createNativeStackNavigator();
13
Windsurf: Refactor | Explain | Generate JSDoc | X
14 export default function AppNavigator() {
15   return (
16     <NavigationContainer>
17       <Stack.Navigator>
18         <Stack.Screen name="Login" component={LoginScreen} />
19         <Stack.Screen name="Semester" component={SemesterScreen} />
20         <Stack.Screen name="Bab" component={BabScreen} />
21         <Stack.Screen name="Materi" component={MateriScreen} />
22         <Stack.Screen name="SoalBab" component={SoalBabScreen} />
23         <Stack.Screen name="UAS" component={UASScreen} />
24       </Stack.Navigator>
25     <Stack.Screen name="Hasil" component={HasilScreen} />
26   )
27 }

```

PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS

> npm install

changed 1 package, and audited 756 packages in 13s

72 packages are looking for funding
run `npm fund` for details

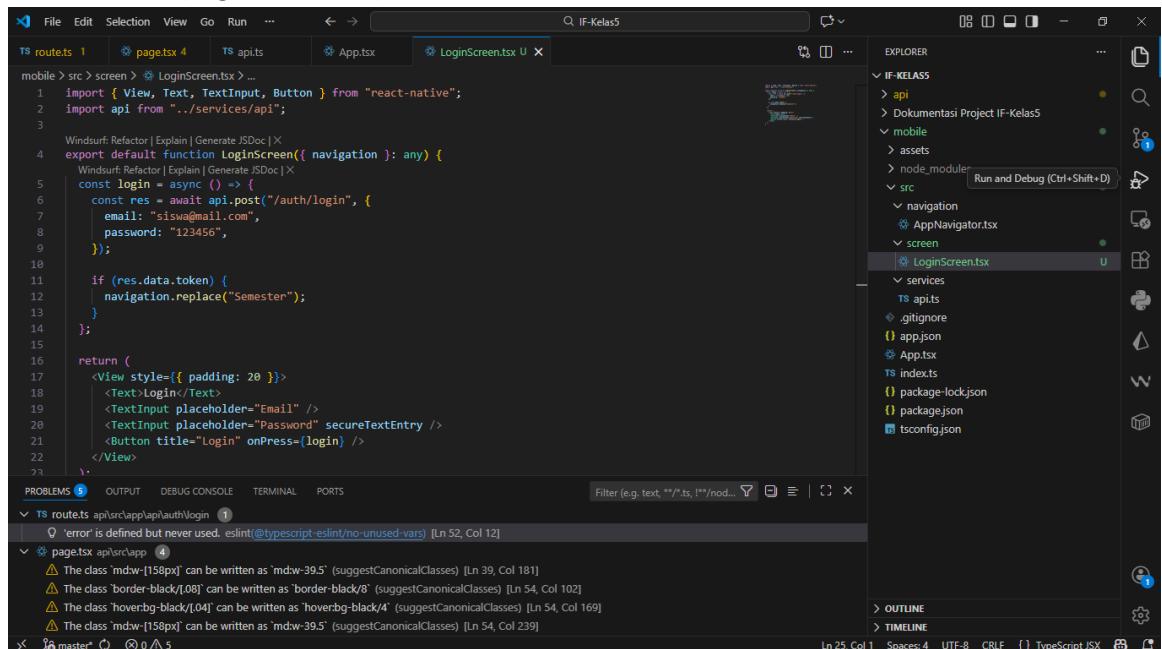
found 0 vulnerabilities

PS C:\PA-Mobile\IF-Kelas5\mobile> []

× master* ⌂ ⑧ 7 5

Sistem Navigasi Utama (App Navigator) pada aplikasi. Berfungsi sebagai pengatur alur perpindahan halaman (routing) di dalam aplikasi React Native menggunakan library React Navigation.

6. Screen LOGIN (LoginScreen.tsx)



```

mobile > screen > LoginScreen.tsx ...
1 import { View, Text, TextInput, Button } from "react-native";
2 import api from "./services/api";
3
Windsurf: Refactor | Explain | Generate JSDoc | X
4 export default function LoginScreen({ navigation }: any) {
5   const login = async () => {
6     const res = await api.post("/auth/login", {
7       email: "siswa@mail.com",
8       password: "123456",
9     });
10
11     if (res.data.token) {
12       navigation.replace("Semester");
13     }
14   };
15
16   return (
17     <View style={{ padding: 20 }}>
18       <Text>Login</Text>
19       <TextInput placeholder="Email" />
20       <TextInput placeholder="Password" secureTextEntry />
21       <Button title="Login" onPress={login}>/>
22     </View>
23   );
}

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

TS route.ts api\src\app\api\auth\login 1

error is defined but never used. eslint(@typescript-eslint/no-unused-vars) [ln 52, Col 12]

page.tsx api\src\app 4

- The class `mdw-[158px]` can be written as `mdw-39.5` (suggestCanonicalClasses) [ln 39, Col 181]
- The class `border-black/[08]` can be written as `border-black/8` (suggestCanonicalClasses) [ln 54, Col 102]
- The class `hoverbg-black/[04]` can be written as `hoverbg-black/4` (suggestCanonicalClasses) [ln 54, Col 169]
- The class `mdw-[158px]` can be written as `mdw-39.5` (suggestCanonicalClasses) [ln 54, Col 239]

× master* ⌂ ⑧ 0 5

Berfungsi sebagai gerbang masuk (autentikasi) pengguna ke dalam aplikasi. Alurnya : Proses Autentikasi (login function) dan Antarmuka Pengguna (UI).

7. Ambil Data Semester (SemesterScreen.tsx)

```
mobile > src > screen > SemesterScreen.tsx > ...
1 import { View, Text, Button } from "react-native";
2 import api from "./services/api";
3 import { useEffect, useState } from "react";
4
5 WurstRefactor | Explain | Generate JSONDoc | X
6 export default function SemesterScreen({ navigation }: any) {
7   const [data, setData] = useState([ ]);
8
9   useEffect(() => {
10     api.get("/semester").then(res => setData(res.data));
11   }, [ ]);
12
13   return (
14     <View>
15       <Text>Daftar Semester</Text>
16       {data.map((s: any) => (
17         <Button
18           key={s.id}
19           title={s.nama}
20           onPress={() => navigation.navigate("Bab", { id: s.id })}
21       )})
22     </View>
23   );
24 }
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Filter (e.g. text, **/*ts, !**/node...) ✖ ☰

VS Code routes app/src/app/api/auth/login 1

⚠ 'error' is defined but never used. eslint[@typescript-eslint/no-unused-vars] [Ln 52, Col 12]

VS page.tsx app/src/app 4

⚠ The class 'mdw-[15px]' can be written as 'mdw-39.5' (suggestCanonicalClasses) [Ln 39, Col 181]
⚠ The class 'border-black/[0]' can be written as 'border-black/8' (suggestCanonicalClasses) [Ln 54, Col 102]
⚠ The class 'hover:rgb-black/[0.04]' can be written as 'hover:rgb-black/4' (suggestCanonicalClasses) [Ln 54, Col 169]
⚠ The class 'mdw-[15px]' can be written as 'mdw-39.5' (suggestCanonicalClasses) [Ln 54, Col 239]

LN 25, Col 1 SPACES: 4 UTF-8 CRLF [] TypeScript JSX ✖

EXPLORER

IF-KELAS

- api
- Dokumentasi Project IF-Kelas5
- mobile
- assets
- node_modules
- src
 - navigation
 - AppNavigator.tsx
 - screen
 - LoginScreen.tsx
 - SemesterScreen.tsx
- services
 - TS apits
 - .gitignore
 - app.json
 - App.tsx
 - TS index.ts
 - package-lock.json
 - package.json
 - tsconfig.json

OUTLINE

TIMELINE

Untuk menampilkan daftar pilihan semester yang datanya diambil langsung dari database melalui API. Pengambilan Data (Data Fetching), Rendering Dinamis (Mapping), dan Navigasi dengan Parameter.

8. BabScreen.tsx

```
mobile > src > screen > BabScreen.tsx ...
1 import { View, Text, Button } from "react-native";
2 import {useEffect, useState} from "react";
3 import api from "../../services/api";
4
5 WurstRefactor | Explain | GenerateJSDoc | X
6 export default function BabScreen({ route, navigation }: any) {
7   const { semesterId } = route.params;
8   const [bab, setBab] = useState([]);
9
10  useEffect(() => {
11    api.get(`/semester/${semesterId}/bab`)
12      .then(res => setBab(res.data));
13  }, []);
14
15  return (
16    <View>
17      <Text>Daftar Bab</Text>
18      {bab.map((b: any) => (
19        <Button
20          key={b.id}
21          title={b.judul}
22          onPress={() => navigation.navigate("Materi", { babId: b.id })}
23        />
24      ))}
25    </View>
26  );
27}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\PA-Mobile\IF-Kelas5>

Ln 27, Col 1 Spaces: 4 UTF-8 CRLF {} TypeScript JSX

Screen Component pada React Native bernama BabScreen yang berfungsi untuk menampilkan daftar bab berdasarkan semester yang dipilih.

Berikut adalah penjelasan poin-poin utamanya:

- Penerimaan Data (route): Kode ini mengambil semesterId dari parameter rute (route.params). Ini berarti layar ini diekspektasikan menerima data dari layar sebelumnya.
 - Pengambilan Data API: Menggunakan useEffect untuk memicu pengambilan data (fetching) dari API segera setelah layar dimuat. Endpoint yang diakses adalah /semester/\${semesterId}/bab.
 - Manajemen State: Menggunakan useState untuk menyimpan daftar bab yang diterima dari server ke dalam variabel bab.
 - Iterasi Data (Mapping): Data bab yang berupa array diulang menggunakan fungsi .map() untuk merender komponen Button bagi setiap itemnya.
 - Navigasi: Setiap tombol memiliki fungsi onPress yang akan mengarahkan pengguna ke layar "Materi" sambil membawa data babId.

9. MateriScreen.tsx

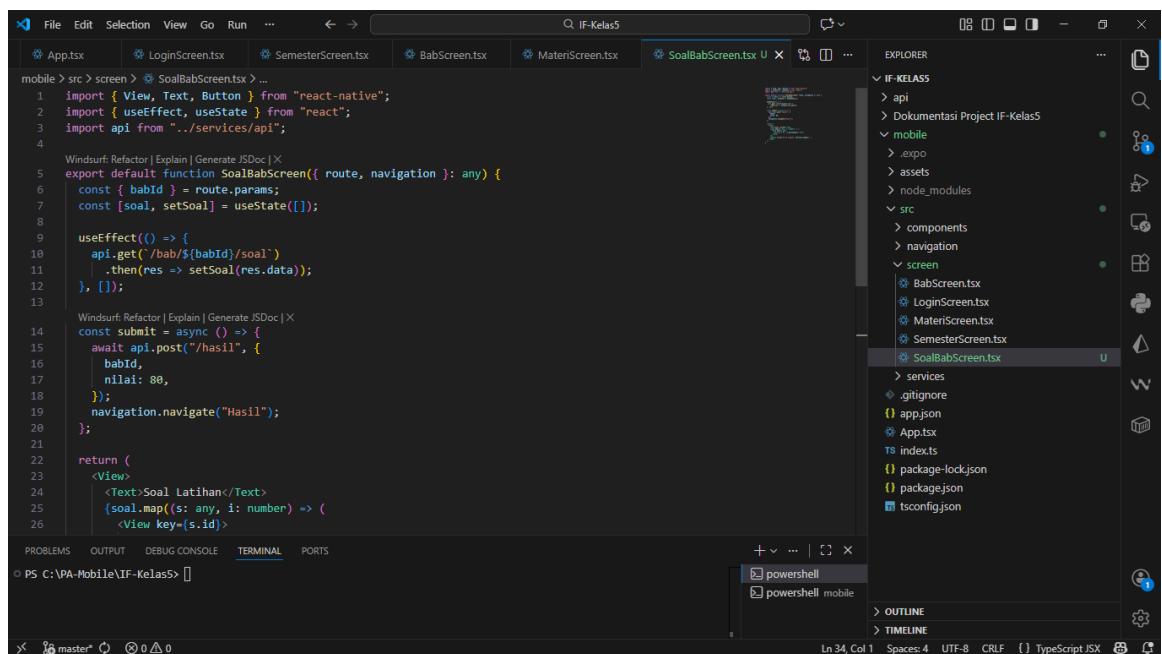
MateriScreen, yaitu layar yang berfungsi untuk menampilkan isi materi pembelajaran berdasarkan bab yang dipilih dan menyediakan akses ke latihan soal.

Berikut adalah deskripsi alur kerianya:

- Integrasi Data: Mengambil babId dari layar sebelumnya via route.params untuk memastikan materi yang muncul sesuai dengan bab yang diklik pengguna.

- Pengambilan Konten: Menggunakan useEffect untuk melakukan fetch data dari endpoint API /bab/\${babId}/materi. Data tersebut kemudian disimpan dalam state materi.
- Penyajian Informasi: Menggunakan metode .map() untuk menampilkan daftar materi. Berbeda dengan layar sebelumnya, di sini aplikasi menampilkan judul dan isi materi secara detail di dalam komponen View.
- Navigasi Lanjutan: Di bagian bawah daftar materi, terdapat tombol "Mulai Latihan" yang akan mengarahkan pengguna ke layar SoalBab dengan membawa babId sebagai referensi soal yang harus dimuat.

10. SoalBabScreen.tsx



```

1  import { View, Text, Button } from "react-native";
2  import { useEffect, useState } from "react";
3  import api from "../services/api";
4
5  Windsurf: Refactor | Explain | Generate JSDoc | X
6  export default function SoalBabScreen({ route, navigation }: any) {
7    const { babId } = route.params;
8    const [soal, setSoal] = useState([]);
9
10   useEffect(() => {
11     api.get(`/bab/${babId}/soal`)
12       .then(res => setSoal(res.data));
13     }, []);
14
15   const submit = async () => {
16     await api.post("/hasil", {
17       babId,
18       nilai: 80,
19     });
20     navigation.navigate("Hasil");
21   };
22
23   return (
24     <View>
25       <Text>Soal Latihan</Text>
26       {soal.map((s: any, i: number) => (
27         <View key={s.id}>

```

SoalBabScreen sebagai modul evaluasi atau latihan soal untuk menguji pemahaman pengguna terhadap bab yang baru saja dipelajari.

Berikut adalah rincian fungsionalitasnya:

- Pemuatan Soal: Mengambil babId dari parameter navigasi dan menggunakan useEffect untuk memanggil API guna mendapatkan daftar pertanyaan terkait bab tersebut.
- Tampilan Dinamis: Menggunakan fungsi .map() untuk merender daftar pertanyaan. Teks pertanyaan ditampilkan dengan nomor urut otomatis menggunakan indeks array (i + 1).
- Proses Pengiriman (submit):
 - ✓ Fungsi ini bersifat asynchronous (async/await).

- ✓ Mengirimkan data hasil latihan ke endpoint /hasil melalui metode POST.
- ✓ Dalam contoh ini, nilai dikirim secara statis (nilai: 80) sebagai placeholder.
- Alur Navigasi: Setelah proses pengiriman data ke server selesai, aplikasi secara otomatis mengarahkan pengguna ke layar "Hasil".

11. HasilScreen.tsx

The screenshot shows the Visual Studio Code interface with the following details:

- Code Editor:** Displays the `HasilScreen.tsx` file content. The code is a React Native component that fetches data from an API, processes it, and then posts the results back to another endpoint. It uses hooks like `useEffect` and `useState`.
- File Explorer:** Shows the project structure for "IF-Kelas5". The `src` folder contains components, navigation, and screens. The `HasilScreen.tsx` file is selected in the list.
- Terminal:** Shows the command `PS C:\PA-Mobile\IF-Kelas5>`, indicating the current working directory.
- Status Bar:** Shows the line number (Ln 132), column number (Col 4), and file encoding (UTF-8 CRLF). It also indicates the file is a TypeScript file.

HasilScreen, yang berfungsi sebagai layar penyelesaian atau ringkasan skor setelah pengguna menyelesaikan latihan soal pada bab tertentu.

12. UASScreen.tsx

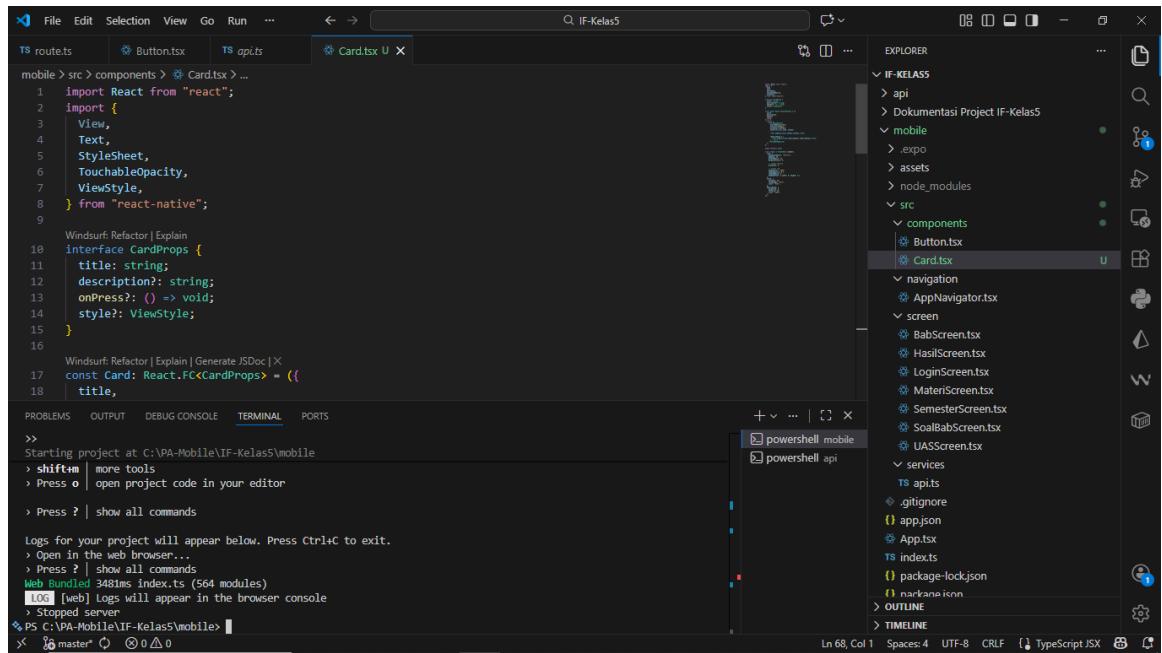
UASScreen, yang merupakan modul ujian akhir untuk mengevaluasi seluruh materi dalam satu semester.

alur kerjanya sebagai berikut :

- Pengambilan Data Terpusat: Berbeda dengan latihan per bab, layar ini memanggil endpoint `/semester/1/uas`, yang berarti sistem mengambil kumpulan soal ujian khusus untuk semester 1.
- State Management: Daftar soal disimpan dalam state soal dan akan diperbarui secara otomatis setelah data berhasil ditarik dari API menggunakan `useEffect`.
- Penyajian Soal: Menggunakan metode `.map()` untuk menampilkan daftar pertanyaan secara berurutan dengan nomor otomatis ($i + 1$).

- Aksi Selesai: Saat ini, tombol "Selesai" hanya memicu fungsi alert sederhana sebagai penanda bahwa ujian telah berakhir.

13. Card.tsx



```

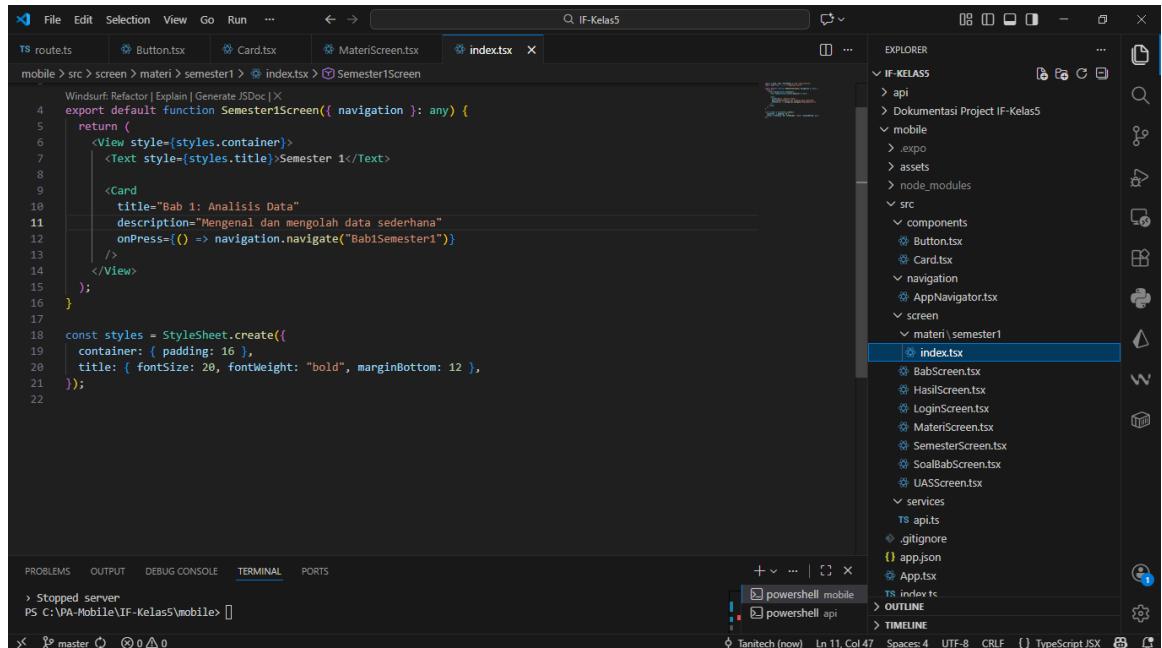
TS route.ts  Button.tsx  TS apि.ts  Card.tsx U
mobile > src > components > Card.tsx > ...
1 import React from "react";
2 import {
3   View,
4   Text,
5   StyleSheet,
6   Touchableopacity,
7   Viewstyle,
8 } from "react-native";
9
10 Windsurf: Refactor | Explain
11 interface CardProps {
12   title: string;
13   description?: string;
14   onPress?: () => void;
15 }
16
17 const Card: React.FC<CardProps> = ({title,
18
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
>>> Starting project at C:\PA-Mobile\IF-Kelas5\mobile
> shift+& more tools
> Press o open project code in your editor
> Press ? | show all commands
Logs for your project will appear below. Press Ctrl+C to exit.
> Open in the web browser...
> Press ? | show all commands
Web Bundled 348ms index.ts (564 modules)
LOG [web] Logs will appear in the browser console
> Stopped server
PS C:\PA-Mobile\IF-Kelas5\mobile>

```

Ln 68, Col 1 Spaces: 4 UTF-8 CRLF { TypeScript JSX

Komponen ini berfungsi untuk menampilkan sebuah kotak informasi (kartu) yang memiliki judul, deskripsi opsional, dan bisa diklik (interaktif).

14. Bab Semester 1 (semester1/index.tsx)



```

TS route.ts  Button.tsx  Card.tsx  MateriScreen.tsx  index.tsx
mobile > src > screen > materi > semester1 > index.tsx > Semester1Screen
1 Windsurf: Refactor | Explain | Generate JSDoc | X
2 export default function Semester1Screen({ navigation }: any) {
3   return (
4     <View style={styles.container}>
5       <Text style={styles.title}>Semester 1</Text>
6
7       <Card
8         title="Bab 1: Analisis Data"
9         description="Mengenal dan mengolah data sederhana"
10        onPress={() => navigation.navigate("Bab1Semester1")}
11      />
12    </View>
13  );
14}
15
16 const styles = StyleSheet.create({
17   container: { padding: 16 },
18   title: { fontSize: 20, fontWeight: "bold", marginBottom: 12 },
19 });
20
21
22
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
> Stopped server
PS C:\PA-Mobile\IF-Kelas5\mobile>

```

Tanitech (now) Ln 11, Col 47 Spaces: 4 UTF-8 CRLF { TypeScript JSX

Berfungsi untuk menampilkan daftar materi pembelajaran pada Semester 1.

- Implementasi Komponen Reusable: Layar ini menggunakan komponen kustom `<Card />` yang dipanggil dari folder komponen luar untuk menampilkan materi bab secara konsisten.
 - Navigasi Interaktif: Terdapat fungsi `onPress` pada kartu yang akan mengarahkan pengguna ke layar detail materi `Bab1Semester1` menggunakan `navigation.navigate`.
 - Struktur Sederhana: * Judul: Menampilkan teks Semester 1 dengan gaya tebal (`bold`).
 - Konten: Kartu berisi informasi spesifik seperti Bab 1: Analisis Data beserta ringkasan materinya.
 - Styling: Menggunakan `StyleSheet` untuk memberikan ruang (`padding`) di sekeliling layar dan mengatur jarak antar elemen agar terlihat rapi.

15. Bab Semester 1 (semester1/bab1.tsx)

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the right, it displays the project structure under "IF-Kelas5". The "mobile" folder contains "src", "assets", "node_modules", and "services". The "src" folder has "components", "navigation", and "screen" subfolders. The "screen" folder contains "materi", which in turn contains "semester1", "bab1.tsx", and "index.tsx". Other files in "screen" include "BabScreen.tsx", "HasilScreen.tsx", "LoginScreen.tsx", "MateriScreen.tsx", "SemesterScreen.tsx", "SoalbabScreen.tsx", and "UASScreen.tsx".
- Code Editor:** The main area shows the content of "bab1.tsx". The code uses React Native components like `ScrollView`, `Text`, `Card`, and `Button`. It includes styling with `StyleSheet.create` and navigation logic with `navigate`.
- Terminal:** At the bottom, the terminal shows the command "Stopped server" and the path "PS C:\PA-Mobile\IF-Kelas5\mobile>".

16. Komponen Reusable

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left pane displays a code editor with a file named 'Button.tsx' containing the following TypeScript code:

```
mobile > src > components > Button.tsx > ...
1 import { TouchableOpacity, Text } from "react-native";
2
3 Windsurf Refactor | Explain | Generate JSDoc | X
4 export default function Button({ title, onPress }: any) {
5   return (
6     <TouchableOpacity onPress={onPress} style={{ padding: 10 }}>
7       <Text>{title}</Text>
8     </TouchableOpacity>
9   );
10 }
```

The right pane shows the 'EXPLORER' view, which lists the project structure for 'IF-KELAS5'. The 'src/components' folder contains a file named 'Button.tsx'. Other files listed include 'api', 'assets', 'node_modules', 'navigation', 'screen', 'services', '.gitignore', 'app.json', 'App.tsx', 'index.ts', 'package-lock.json', 'package.json', and 'tsconfig.json'. The bottom status bar indicates the file is in 'In 10, Col 1' with 'Spaces: 4' and 'UTF-8' encoding.

Komponen Button kustom yang bersifat reusable (dapat digunakan kembali) dalam aplikasi React Native.

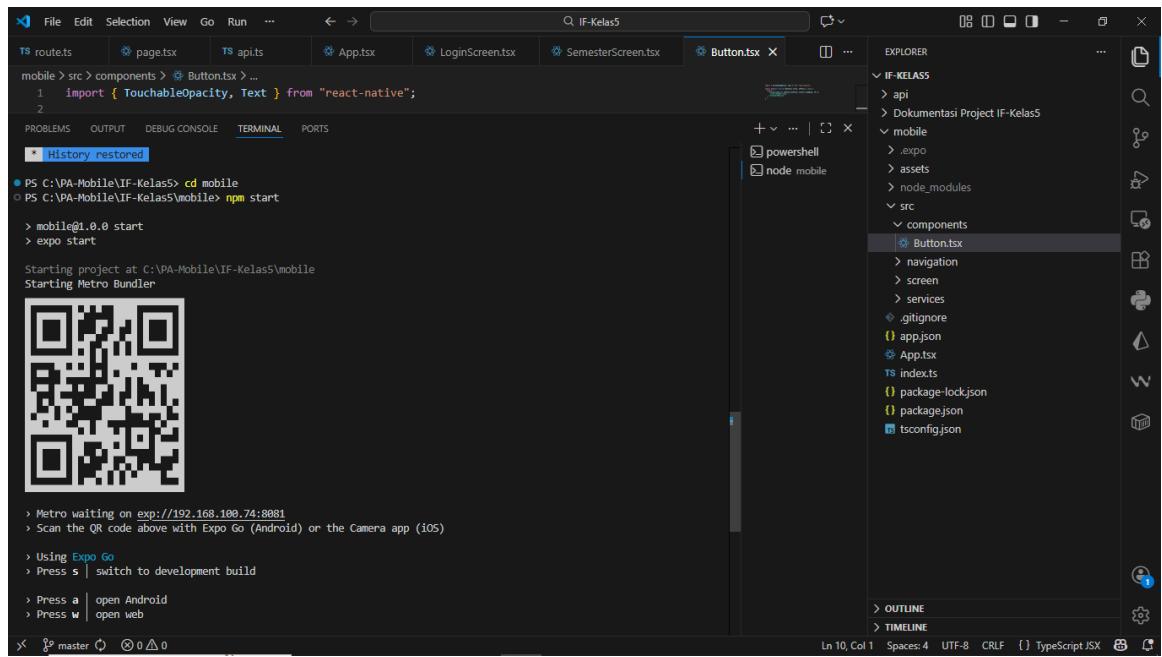
Struktur Komponen:

- Impor: Mengambil komponen dasar dari library react-native.
- Definisi: Menggunakan functional component dengan TypeScript (ditandai dengan tipe : any).
- Eksport: Menggunakan export default agar komponen ini bisa diimpor dan digunakan dengan mudah di file lain.

17. Menjalankan aplikasinya

Kode :

npm start



```
PS C:\PA-Mobile\IF-Kelas5> cd mobile
PS C:\PA-Mobile\IF-Kelas5\mobile> npm start
> mobile@1.0.0 start
> expo start
Starting project at C:\PA-Mobile\IF-Kelas5\mobile
Starting Metro Bundler
QR code to scan with Expo Go (Android) or the Camera app (iOS)
Metro waiting on exp://192.168.100.74:8081
Scan the QR code above with Expo Go (Android) or the Camera app (iOS)
Using Expo Go
Press s | switch to development build
Press a | open Android
Press w | open web
```

Lalu:

Tekan a → Android Emulator

Scan QR → Expo Go

Download aplikasi Expo Go di PlayStore lalu install aplikasi dan scan barcodenya.

18. Alur Aplikasi

