

The screenshot shows a code editor with a dark theme. On the left is a file tree for a React project:

```

graph TD
    root[library root] --> graphql
    root --> hooks
    root --> lib
    lib --> node_modules
    node_modules --> pages
    pages --> admin
    pages --> api
    pages --> _app.tsx
    pages --> _document.tsx
    pages --> about.tsx
    pages --> index.tsx
    public
    tests
    utils
        theme.ts
        babelrc
        env.example
        env.local
        eslint.js
        .gitignore
        next-env.dts
        package.json
        README.md
        tsconfig.json
        yarn.lock
    External Libraries
    Scratches and Consoles

```

The main editor area contains a component named `App`:

```

import { ThemeProvider } from '@material-ui/core/styles';
import CssBaseline from '@material-ui/core/CssBaseline';
import { Container } from '@material-ui/core';
import { useApollo } from '../graphql/client';

import { lightTheme, darkTheme } from '../utils/theme';
import useLocalStorage from '../hooks/useLocalStorage';

import NavBar from '../components/NavBar';

function App({ Component, pageProps }: AppProps) {
  const [currentTheme, setCurrentTheme] = useLocalStorage('key.theme-value', initialValue: 'light');
  const apolloClient = useApollo(pageProps.initialApolloState);

  useEffect(() => {
    const jssStyles = document.querySelector('selector: #jss-server-side');
    if (jssStyles) {
      jssStyles.parentElement.removeChild(jssStyles);
    }
  }, [deps]);
}

return (
  <>
  <Head>
    <title>ECU-DEV</title>
    <meta name="viewport" content="minimum-scale=1, initial-scale=1, width=device-width"/>
  </Head>
  <Component {...pageProps} theme={currentTheme === 'light' ? lightTheme : darkTheme} />
)

```

Node.js API - Karishma

▼ /signup POST

Parameters:

body:

```
{
  "name": string,
  "email": string,
  "password": string,
}
```

Responses

1. 400:

```
{
  status:0,
  message:"user already Registered with this email."
}
```

2. 200:

```
{  
  status:1,  
  message:"User has been Registered Successfully !"  
}
```

▼ /login POST

Parameters:

body:

```
{  
  "email": string,  
  "password": string  
}
```

Responses

1. 500:

```
{  
  status: 0,  
  message: "password didnt match"  
}
```

2. 200:

```
{  
  status:1,  
  message:"user logged in successfully !",  
  token: userToken  
}
```

3. 501:

```
{  
  status: 0,
```

```
    message: "user not exist with this email address"
}
```

▼ /checkBalance POST

Parameters:

headers:

```
{
  "Authorization": userToken(string)
}
```

Responses

1. 500:

```
{
  status: 0,
  message: "failed to check balance",
  data: error
}
```

2. 200:

```
{
  status: 1,
  message: " User balance",
  data: user.balance
}
```

▼ /addBalance POST

Parameters:

headers:

```
{
  "Authorization": userToken(string)
}
```

```
}
```

body:

```
{
  "balance": [FLOAT]
}
```

Responses

1. 500:

```
{
  status: 0,
  message: "failed to add balance"
}
```

2. 200:

```
{
  status: 1,
  message: "balance added successfully",
  data: [FLOAT]
}
```

3. 501:

```
{
  status: 0,
  message: "failed to add balance",
  data: error
}
```

▼ /transferBalance POST

Parameters:

headers:

```
{
```

```
"Authorization": userToken(string)  
}  
  
body:  
{  
  "To": [INTEGER]  
  "amount": [FLOAT],  
  
}
```

Responses

1. 500:

```
{  
  status:0,  
  message:"Transaction Failed"  
}
```

2. 200:

```
{  
  status:1,  
  message:"Transaction Successfull"  
}
```

3. 501:

```
{  
  status:0,  
  message:"Unable to update Sender Balance",  
  error: error  
}
```

4. 404:

```
{  
  status:0,  
  message:"Reciever does not exist",  
}
```

▼ /transactionHistory POST

Parameters:

headers:

```
{  
  "Authorization": userToken(string)  
}
```

body:

```
{  
  
}
```

Responses

1. 500:

```
{  
  status:0,  
  message: "No transaction records found",  
}
```

2. 200:

```
{  
  message: "All Transactions Found",  
  data: allTransactions,  
}
```

▼ /withdrawBalance POST

Parameters:

headers:

```
{  
  "balance": [FLOAT]  
}
```

Responses

1. **500:**

```
{  
  status: 0,  
  message: "Failed to withdraw Balance "  
}
```

2. **200:**

```
{  
  status: 1,  
  message: "balance withdraw successfully"  
}
```

Middleware-checkToken()

Responses:

1. **400:**

```
{  
  status: 0,  
  message: "Please provide authorization token"  
}
```

2. **401:**

```
{  
  status:0,  
  message: "Token is not valid",  
  data: error  
}
```