**React Project Folder Structure Documentation**

In this documentation, I will provide a detailed explanation of the folder structure for the React project named "our project." This structure aims to promote a well-organized and maintainable codebase. Each folder and file within the project will be described along with its purpose and responsibilities.

**Project Structure Overview:**

[**Link**](https://tree.nathanfriend.io/?s=(%27optiJs!(%27fancy!true~fullPath!fwX~trailYgSlash!true~rootDot!fwX)~sOrce!(%27sOrce!%27KmplaKs6deploymFts0docker-compoX.yml0Dockerfile6FvVKstVdevelopmFtVpGiJ6public0asXt5layOtIpage5sas5staticItheme0locwe5ar_AE4*F_US4*tr_TR4C.html6src0asXts0compJFt5basicILayOtIorgYizim0cJfig5Axios3*i183*MFuqxQyO%20can%20addNoNhZmFu%20from%20herZIregex3cJKxt5rtl-cJKxt3page5HomeILogYIPG0rOK5C3*PrivaKROK3*PublicROK3Xrvice5Home0W9B5pG0W9B5AuthI*LogYIW*9*Bs0util5cJtant5*theme7hemZvwue5type5*baeMFu7ypescripttypesNhat%20cOld%20uXd%20Y%20globw0App3Cqx6package.jsJ%27)~versiJ!%271%27)*%20%20%20%2006*2**C.3qx04I*translaK.jsJ05sI6%5Cn*7qQwlN92FdpoYt3B2model.tCYdexFenGroductI0*JonKteN%20tOouQ%20%2F%2FV0.Fv.W2api3XseYinZe%20q.tswal%01wqZYXWVQONKJIGFCB97654320*)

templates/

├── deployments/

│ ├── docker-compose.yml

│ └── Dockerfile

├── env/

│ ├── .env.test

│ ├── .env.development

│ └── .env.production

├── public/

│ ├── assets/

│ │ ├── layout

│ │ ├── pages

│ │ ├── sass

│ │ ├── static

│ │ └── theme

│ ├── locales/

│ │ ├── ar\_AE/

│ │ │ └── translate.json

│ │ ├── en\_US/

│ │ │ └── translate.json

│ │ └── tr\_TR/

│ │ └── translate.json

│ └── index.html

├── src/

│ ├── assets

│ ├── components/

│ │ ├── basic

│ │ ├── Layout

│ │ └── orginizim

│ ├── configs/

│ │ ├── Axios.tsx

│ │ ├── i18.tsx

│ │ ├── Menu.tsx //you can add to the menu from here

│ │ └── regex.tsx

│ ├── contexts/

│ │ └── rtl-context.tsx

│ ├── pages/

│ │ ├── Home

│ │ ├── Login

│ │ └── Product

│ ├── routes/

│ │ ├── index.tsx

│ │ ├── PrivateRoute.tsx

│ │ └── PublicRoute.tsx

│ ├── services/

│ │ ├── Home/

│ │ │ ├── index.api.tsx

│ │ │ ├── index.endpoint.tsx

│ │ │ └── index.model.ts

│ │ ├── product/

│ │ │ ├── index.api.tsx

│ │ │ ├── index.endpoint.tsx

│ │ │ └── index.model.ts

│ │ └── Auth/

│ │ └── Login/

│ │ ├── index.api.tsx

│ │ ├── index.endpoint.tsx

│ │ └── index.model.ts

│ ├── utils/

│ │ ├── contants/

│ │ │ └── theme.ts //all theme values

│ │ └── types/

│ │ └── baeMenu.ts //all typescripttypes that could used in global

│ ├── App.tsx

│ └── index.tsx

└── package.json

Now, let's dive into the details of each folder and file:

**deployments/**

This folder contains deployment-related configuration files. It includes:

* **docker-compose.yml**: A YAML file that defines the services, networks, and volumes for Docker container orchestration.
* **Dockerfile**: A configuration file used to build a Docker image for the React application.

**env/**

This folder holds environment-specific configuration files. It includes:

* **.env.test**: Environment variables for the test environment.
* **.env.development**: Environment variables for the development environment.
* **.env.production**: Environment variables for the production environment.

**public/**

This folder contains public assets and resources served by the application. It includes:

* **assets/**: A directory for storing static assets like images, videos, etc.
* **locales/**: Contains translation files for internationalization purposes.
* **index.html**: The main HTML file that serves as the entry point to the React application.

**src/**

This is the heart of the React application, containing the source code. It includes the following folders:

**assets/**

This folder is used to store any project-specific assets required for the application.

**components/**

This folder holds all reusable React components, organized into subdirectories:

* **basic/**: Contains basic and generic components that can be reused across the application.
* **Layout/**: Components related to the overall layout of the application.
* **orginizim/**: Components specific to the organization of the project.

**configs/**

This folder houses configuration files, utilities, and constants:

* **Axios.tsx**: Configuration for Axios, a popular HTTP client used for making API requests.
* **i18.tsx**: Internationalization configuration for translations using i18next or similar libraries.
* **Menu.tsx**: Menu configuration, allowing easy customization of the application's navigation menu.
* **regex.tsx**: Regular expressions or patterns used throughout the application.

**contexts/**

This folder contains React context providers or hooks that facilitate global state management:

* **rtl-context.tsx**: A context provider for right-to-left language support.

**pages/**

This folder is where individual pages of the application are organized:

* **Home/**: Components and logic specific to the home page.
* **Login/**: Components and logic related to the login page.
* **Product/**: Components and logic for the product page.

**routes/**

This folder holds routing-related components:

* **index.tsx**: Centralized routing configuration, defining all the application's routes.
* **PrivateRoute.tsx**: A higher-order component for creating private routes that require authentication.
* **PublicRoute.tsx**: A higher-order component for creating public routes that are accessible to all users.

**services/**

This folder contains service-related files and API-related logic:

* **Home/**: Components related to the home page's services, organized by feature.
  + **index.api.tsx**: Service file for handling API calls related to the "Home" feature.
  + **index.endpoint.tsx**: Endpoint configuration for the "Home" API calls.
  + **index.model.ts**: Data models for the "Home" feature.
* **product/**: Components related to product services, organized by feature.
* **Auth/**: Components related to authentication services, organized by feature.
  + **Login/**: Components and logic specific to the login service.

**utils/**

This folder contains utility files that are not directly related to components or services:

* **constants/**: Contains constant values, such as theme configurations.
* **types/**: Holds TypeScript type definitions used globally throughout the application.

**App.tsx**

The main component that serves as the root of the React application. It handles routing and provides a container for all other components.

**index.tsx**

The entry point of the application, which renders the **App** component and sets up the React application.

**package.json**

The standard Node.js package configuration file that includes dependencies, scripts, and other project metadata.

This concludes the documentation for the React project folder structure. It is designed to promote modularity, organization, and scalability, making it easier for developers to work collaboratively and maintain the application over time.