**RegisterForm Component**

**Purpose**:  
The RegisterForm component manages the user registration process, dividing it into multiple steps. Each step is loaded lazily, ensuring optimized performance. The form data is persisted across sessions using localStorage, enabling users to resume the registration process if interrupted.

**Props**:

* orgId (string, optional): The ID of the organization associated with the user. Defaults to "ORG\_ID".
* createdBy (string, optional): The entity that created the user. Defaults to "000000".

**State**:

* formData: An object containing the user's registration details. Persisted in localStorage.
* formFillStep: An integer representing the current step of the registration process. Persisted in localStorage.
* alertContainer: An array managing the alert messages displayed to the user.

**Key Methods**:

* showAlert(message, type): Displays an alert with the given message and type.
* nextStep(Element): Renders the current form section based on the active step.
* wrapperNextStep(id, element): Conditionally renders the next form section or redirects based on the current step.

**Routing**:

* The component uses react-router-dom to navigate between form sections.
* Routes:
  + /register/: Loads FormSection1.
  + /register/0: Loads FormSection1.
  + /register/1: Loads FormSection2.
  + /register/2: Loads FormSection3.
  + /\*: Loads an ErrorPage for any undefined routes.

**Usage**: Import and render the RegisterForm component in the appropriate view. Ensure that the necessary form sections (FormSection1, FormSection2, FormSection3) and utility components (AlertContainer, Loading, ErrorPage) are available.

**formData Object**

**Purpose**: The formData object holds all the user input data during the registration process. It is persisted across sessions using localStorage to allow users to continue filling out the form even if they leave and return later.

**Attributes**:

1. **userId** (string): The Six alphanumeric charecters unique identifier for the user. This is generated when the fill all the required field and verified from the server.
2. **name** (string): The full name of the user as entered in the registration form.
3. **mobile** (string): The user's mobile phone number. This should be validated for proper formatting (e.g., not including country code).
4. **email** (string): The user's email address. This should be validated for proper email format.
5. **profilePic** (string): The URL or path to the user's profile picture. This may be an uploaded image or a default avatar.
6. **status** (string): The current status of the user's registration or account. This typically indicates whether the account is active, inactive, or pending verification.
7. **designation** (string): The role or title of the user within the organization. This could be pre-set or selected by the user during registration.
8. **orgId** (string): The identifier for the organization associated with the user. This is typically passed as a prop to the component.
9. **createdBy** (string): Indicates the entity or application that created the user record. This could be the name of the application or an administrator's identifier.
10. **updatedBy** (string): Tracks the entity that last updated the user record. Typically, this is the same as createdBy unless the record is updated by another entity later.

List of the dependencies used in the RegisterForm component:

**1. React :** useState, useRef, lazy, Suspense, useEffect

* useState: Manages state variables like formData, alertContainer, and formFillStep.
* useRef: Creates a reference (alertRef) to interact with the AlertContainer component.
* lazy: Lazy loads components (FormSection1, FormSection2, FormSection3) to optimize performance.
* Suspense: Wraps routes to provide a fallback UI (Loading) while lazy-loaded components are being fetched.
* useEffect: Persists formData and formFillStep to localStorage whenever they change.

**2. React Router DOM:** Route, Routes, Navigate

* Routes and Route: Define the different routes (/, 0, 1, 2, \*) for the registration form.
* Navigate: Redirects the user to the appropriate step in the form based on formFillStep.

**3. CSS Module**

* **Import**: style from "../css/RegisterForm.module.css"
* **Purpose**: Provides scoped and modular CSS classes for styling the component, such as formRapper, userForm, progress, circle, and currentSec.

**4. Custom Components**

* **Imports**:
  + AlertContainer (from ./AlertContainer)
  + ErrorPage (from ../../SpecialPages/js/ErrorPage)
  + Loading (from ../../SpecialPages/js/Loading)
* **Purpose**:
  + AlertContainer: Manages and displays alert messages in the form.
  + ErrorPage: Renders an error page for any invalid or non-existent routes.
  + Loading: Displays a loading spinner while the form sections are being lazy-loaded.

**5. Dynamic Imports**

* **Imports**: FormSection1, FormSection2, FormSection3
* These components are lazy-loaded to improve the initial load time of the registration form. The actual components are loaded only when the user navigates to the corresponding step.

**Additional Dependencies (Indirect):**

* **LocalStorage**: Used to store and retrieve formData and formFillStep to persist form progress across sessions.
* **Browser APIs**: The component relies on browser features such as localStorage for data persistence.