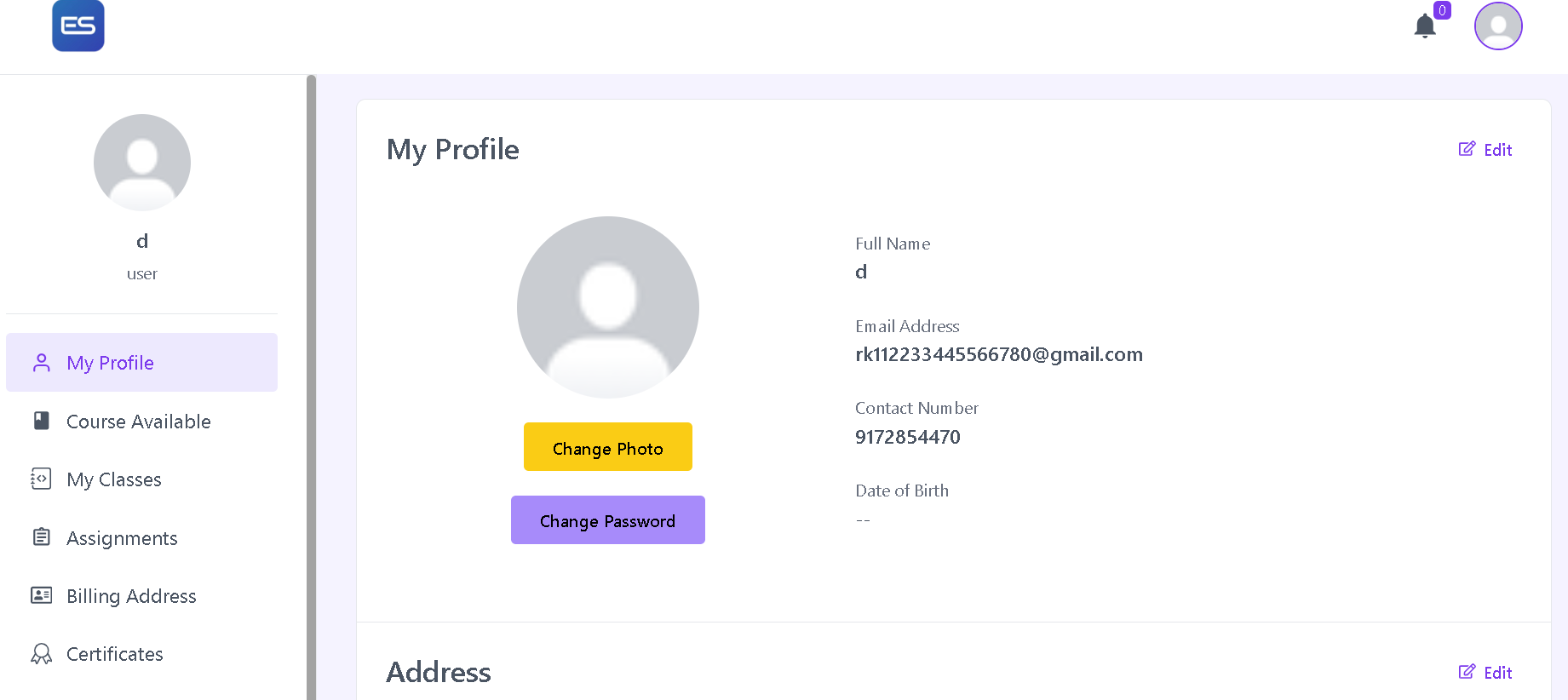
**MyProfile** [**https://citria.org/profile**](https://citria.org/profile)



This React component, named MyProfile, is a part of a larger web application, presumably developed using the React.js library. Let's break down the key elements of this component:

Import Statements:

It imports necessary dependencies and components, including useState from 'react', the FaRegEdit icon from 'react-icons/fa', PageTitle, useGetProfile, Spinner, ProfileForm, AddressForm, and Link from 'react-router-dom'.

State Initialization:

The component uses the useState hook to manage states for profileData, loading, editProfile, and editAddress.

Fetching Profile Data:

The useGetProfile hook is used to fetch profile data, and the loading state is utilized to display a loading spinner until the data is fetched.

Profile and Address Editing:

Conditional rendering is employed based on whether the user is in edit mode for the profile or address. Buttons with the option to edit or cancel the edit are provided.

Profile Display Section:

The profile section displays user information such as name, avatar, email, contact number, and date of birth. Additionally, there are options to change the user's photo and password.

Address Display Section:

The address section shows country, city/state, address, and zip code. Similar to the profile section, there's an option to edit or cancel editing the address.

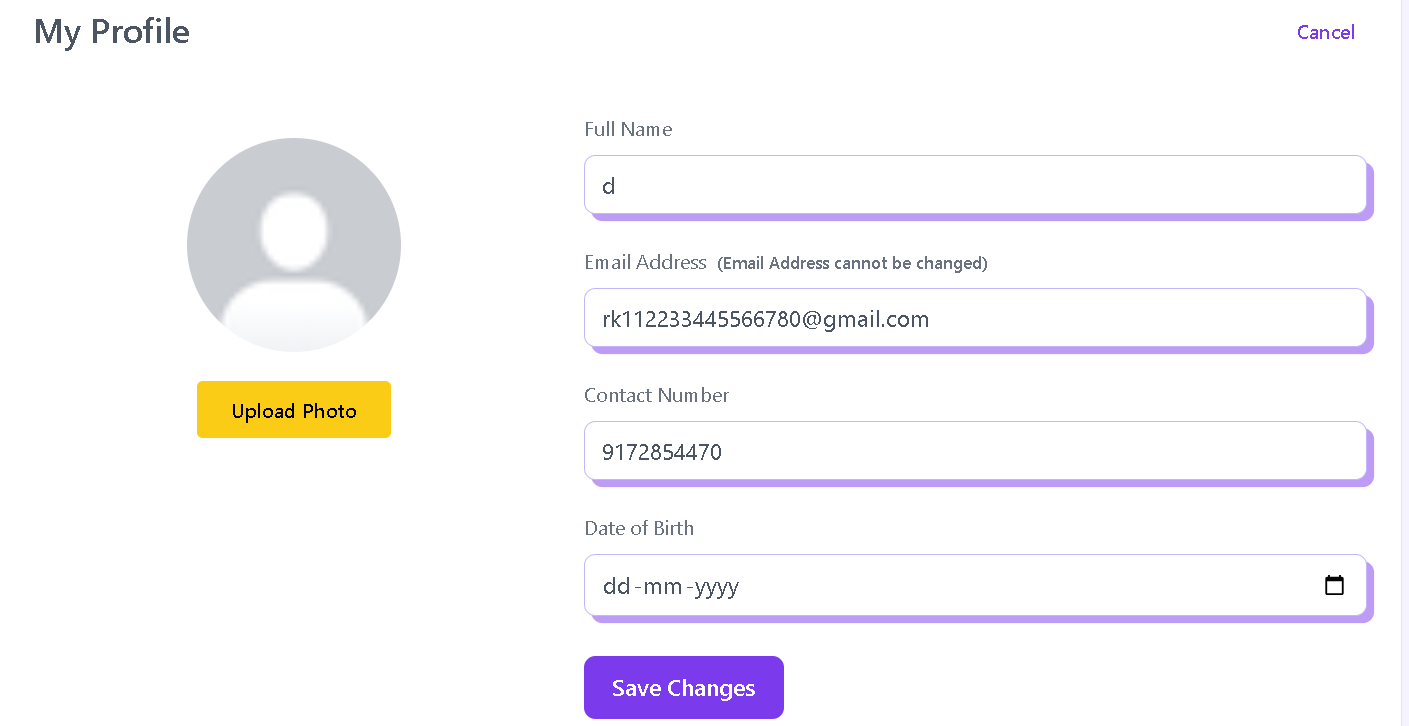
Styling and Layout:

Tailwind CSS classes are extensively used for styling, including responsiveness for different screen sizes.

Routing:

The Link component from 'react-router-dom' is used to navigate to the '/update-password' route.

**Profileform**

****

Component Declaration:

The ProfileForm component receives profileData and setEditProfile as props, which are used to manage the user's profile information and control the editing state.

const ProfileForm = ({ profileData, setEditProfile }) => {

// Component logic...

};

State Management:

The component uses the useState hook to manage various states, including form data, form errors, the selected profile photo, and loading status.

const [formData, setFormData] = useState({

name: profileData.name,

contactNumber: profileData.userId.contactNumber,

birthday: profileData.birthday,

});

const [formErrors, setFormErrors] = useState({});

const [profilePhoto, setProfilePhoto] = useState('');

const [loading, setLoading] = useState(false);

Form Handling Functions:

Functions like captureAvatar, handleChange, and validateForm handle capturing the profile photo, updating form data on input change, and validating the form data, respectively.

const captureAvatar = (e) => {

// Logic to capture and set the selected profile photo...

};

const handleChange = (event) => {

// Logic to update form data on input change...

};

const validateForm = (data) => {

// Logic to validate form data and return errors...

};

Form Submission:

The handleSubmit function handles the form submission. It sends a PATCH request to update the user's profile information using the axios library. Success or error messages are displayed using the showToast function from the context.

const handleSubmit = async (event) => {

// Logic to handle form submission...

};

Form JSX Structure:

The JSX structure represents the layout of the profile editing form, including input fields for name, email (read-only), contact number, birthday, and an option to upload a new profile photo.

return (

<>

{/\* JSX structure for the profile editing form... \*/}

</>

);

Loading Spinner:

A loading spinner (<Spinner />) is displayed while the form submission is in progress.

{

loading && <Spinner />

}

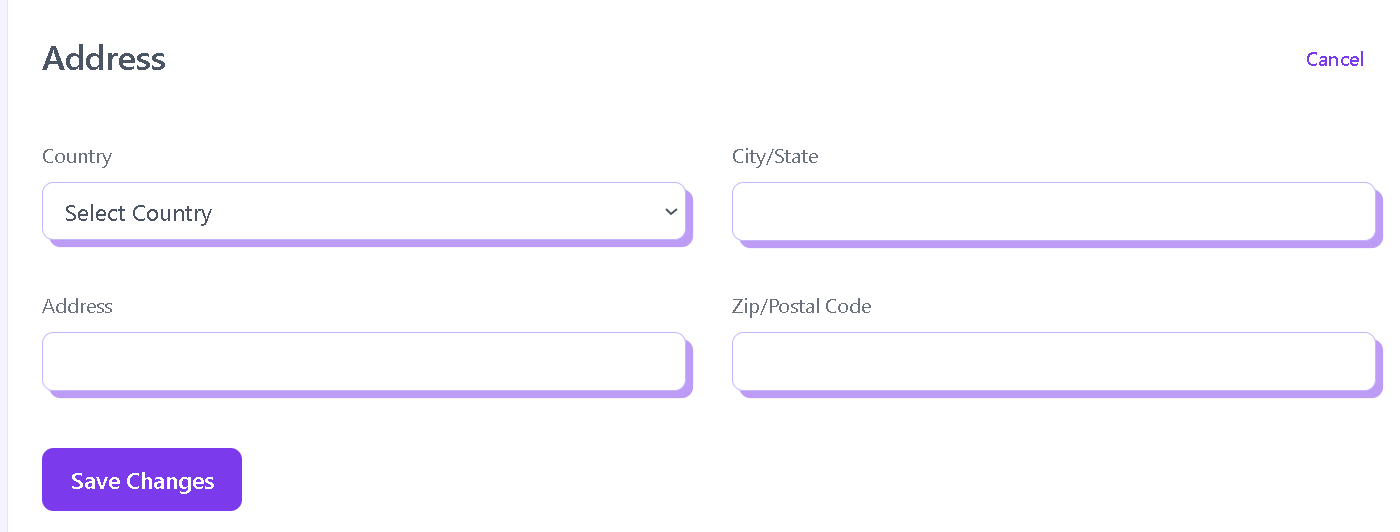
Export Statement:

The component is exported for use in other parts of the application.

export default ProfileForm;

This ProfileForm component is designed for use in a larger React application, providing a user-friendly interface for editing and updating profile information

**AddressForm**

****

The provided code is a React component named **AddressForm** that represents a form for editing address information. Let's break down the key aspects of the code:

Imports:

The component imports necessary modules from React, including useContext and useState hooks for state management. Additionally, it imports icons (BsExclamationCircleFill) and external components (CountryList and Spinner).

Functional Component:

AddressForm is a functional component that receives two props, profileData and setEditAddress. The profileData contains information about the user's address, and setEditAddress is a function to control the editing state.

State Management:

The component uses the useState hook to manage the form data (formData), form errors (formErrors), and a loading state (loading).

Form Handling Functions:

handleChange: Updates the form data when input values change.

validateForm: Checks for errors in the form data, such as missing values.

handleSubmit: Handles the form submission by making a PATCH request to update the user's address using Axios. It includes error handling and displays success or error messages using the showToast function from the context.

JSX Structure:

The JSX structure represents a form with input fields for country, city, address, and zip/postal code. It also includes dropdowns and error messages for validation. The form has a submit button triggering the handleSubmit function.

Styling:

The component applies styles using Tailwind CSS classes for layout, spacing, and appearance.

Loading Spinner:

If the form submission is in progress (loading is true), a spinner component (<Spinner />) is displayed.

Export:

The component is exported for use in other parts of the application.

Overall, this component provides a user-friendly interface for editing and updating address information within a larger React application.