the system is "Provincial Product Accounts" project created by Philippine Statistics Authority , the data needed to compile the Provincial Product Accounts (PPA) is from the Local Government Unit (LGU) namely

1. Annex 1: Revenue of Establishments from BPLO

2. Annex 2: Local Government Construction Projects from Engineering

3. Annex 3: Inventory of Government Human Resources from HR

4. Annex 4: Designated Focal Person for PPA from C/MPDC

Each annexes have a template or tables

I want to have a CRUD system for each annexes to put in the database make a dropdown list of which the user where he/she want to in-code, each user have specific user and their LGU address so that what they in-code only the data of their LGU

After logging in the home page should have the process status of the LGUs of the Province

In the Navbar should have a logo of PSA and title Provincial Product Accounts, only in the navbar have the links to that annexes with their own pages

'use client'

import { useEffect, useState } from 'react'

import { Button } from "/components/ui/button"

import { Input } from "/components/ui/input"

import { Card, CardContent, CardHeader, CardTitle, CardDescription } from "/components/ui/card"

import { Table, TableBody, TableCell, TableHead, TableHeader, TableRow } from "/components/ui/table"

import { Home, Settings, LogOut, FileText, ChevronRight, AlertCircle } from "lucide-react"

type LGUStatus = {

id: string

name: string

annex1: 'pending' | 'completed' | 'not-started'

annex2: 'pending' | 'completed' | 'not-started'

annex3: 'pending' | 'completed' | 'not-started'

annex4: 'pending' | 'completed' | 'not-started'

lastUpdated: string

}

type User = {

username: string

lgu: string

role: 'admin' | 'user'

}

export default function PPASystem() {

const [user, setUser] = useState<User | null>(null)

const [username, setUsername] = useState('')

const [password, setPassword] = useState('')

const [error, setError] = useState('')

const [activeTab, setActiveTab] = useState<'dashboard' | 'annex1' | 'annex2' | 'annex3' | 'annex4'>('dashboard')

const [lgus] = useState<LGUStatus[]>([

{

id: '1',

name: 'Quezon City',

annex1: 'completed',

annex2: 'completed',

annex3: 'pending',

annex4: 'not-started',

lastUpdated: '2023-06-15'

},

{

id: '2',

name: 'Makati City',

annex1: 'completed',

annex2: 'pending',

annex3: 'not-started',

annex4: 'not-started',

lastUpdated: '2023-06-10'

},

{

id: '3',

name: 'Manila City',

annex1: 'completed',

annex2: 'completed',

annex3: 'completed',

annex4: 'pending',

lastUpdated: '2023-06-18'

},

{

id: '4',

name: 'Taguig City',

annex1: 'pending',

annex2: 'not-started',

annex3: 'not-started',

annex4: 'not-started',

lastUpdated: '2023-06-05'

}

])

// Check for existing session on initial load

useEffect(() => {

const storedUser = localStorage.getItem('user')

if (storedUser) {

setUser(JSON.parse(storedUser))

}

}, [])

const handleLogin = (e: React.FormEvent) => {

e.preventDefault()

// Simple authentication - replace with real API call in production

if (username === 'admin' && password === 'ppa123') {

const userData = {

username,

lgu: 'Quezon City',

role: 'admin' as const

}

localStorage.setItem('user', JSON.stringify(userData))

setUser(userData)

setError('')

} else if (username === 'user' && password === 'ppa123') {

const userData = {

username,

lgu: 'Makati City',

role: 'user' as const

}

localStorage.setItem('user', JSON.stringify(userData))

setUser(userData)

setError('')

} else {

setError('Invalid credentials. Please try again.')

}

}

const handleLogout = () => {

localStorage.removeItem('user')

setUser(null)

setUsername('')

setPassword('')

setActiveTab('dashboard')

}

const getStatusBadge = (status: string) => {

const baseClasses = "px-2 py-1 text-xs rounded-full"

switch (status) {

case 'completed':

return <span className={`${baseClasses} bg-green-100 text-green-800`}>Completed</span>

case 'pending':

return <span className={`${baseClasses} bg-yellow-100 text-yellow-800`}>Pending</span>

default:

return <span className={`${baseClasses} bg-gray-100 text-gray-800`}>Not Started</span>

}

}

// Show login screen if no user is authenticated

if (!user) {

return (

<div className="min-h-screen flex items-center justify-center bg-gray-50 px-4">

<Card className="w-full max-w-md shadow-lg">

<CardHeader className="text-center">

<CardTitle className="text-2xl font-bold">Provincial Product Accounts</CardTitle>

<CardDescription>Login to your LGU account</CardDescription>

</CardHeader>

<CardContent>

<form onSubmit={handleLogin} className="space-y-4">

<div className="space-y-2">

<Label htmlFor="username">Username</Label>

<Input

id="username"

type="text"

value={username}

onChange={(e) => setUsername(e.target.value)}

required

/>

</div>

<div className="space-y-2">

<Label htmlFor="password">Password</Label>

<Input

id="password"

type="password"

value={password}

onChange={(e) => setPassword(e.target.value)}

required

/>

</div>

{error && (

<div className="flex items-center text-red-600 text-sm">

<AlertCircle className="mr-1 h-4 w-4" />

{error}

</div>

)}

<Button type="submit" className="w-full">

Login

</Button>

</form>

</CardContent>

</Card>

</div>

)

}

// Main application after login

return (

<div className="min-h-screen bg-gray-50">

{/\* Navigation Header \*/}

<nav className="bg-white shadow-sm">

<div className="container mx-auto px-4 py-3 flex justify-between items-center">

<div className="flex items-center space-x-4">

<div className="bg-blue-600 text-white p-2 rounded">

<span className="font-bold">PSA</span>

</div>

<h1 className="text-xl font-bold">Provincial Product Accounts</h1>

</div>

<div className="flex items-center space-x-6">

{['annex1', 'annex2', 'annex3', 'annex4'].map((annex) => (

<button

key={annex}

onClick={() => setActiveTab(annex as any)}

className={`flex items-center space-x-1 ${

activeTab === annex ? 'text-blue-600' : 'text-gray-600 hover:text-blue-500'

}`}

>

<FileText className="h-4 w-4" />

<span>{annex.replace('annex', 'Annex ')}</span>

</button>

))}

<div className="flex items-center space-x-4">

<button className="text-gray-600 hover:text-blue-500">

<Settings className="h-5 w-5" />

</button>

<button

onClick={handleLogout}

className="text-gray-600 hover:text-red-500 flex items-center space-x-1"

>

<LogOut className="h-5 w-5" />

<span>Logout</span>

</button>

</div>

</div>

</div>

</nav>

{/\* Main Content Area \*/}

<main className="container mx-auto px-4 py-8">

{activeTab === 'dashboard' ? (

<div className="space-y-6">

<div className="flex justify-between items-center">

<h2 className="text-2xl font-bold">LGU Process Status</h2>

<div className="flex space-x-2">

<Button variant="outline">Export Data</Button>

<Button>Refresh Data</Button>

</div>

</div>

<Card>

<CardContent className="p-0">

<Table>

<TableHeader>

<TableRow>

<TableHead className="w-[200px]">LGU Name</TableHead>

<TableHead>Annex 1</TableHead>

<TableHead>Annex 2</TableHead>

<TableHead>Annex 3</TableHead>

<TableHead>Annex 4</TableHead>

<TableHead>Last Updated</TableHead>

<TableHead>Actions</TableHead>

</TableRow>

</TableHeader>

<TableBody>

{lgus

.filter(lgu => user.role === 'admin' || lgu.name === user.lgu)

.map((lgu) => (

<TableRow key={lgu.id}>

<TableCell className="font-medium">{lgu.name}</TableCell>

<TableCell>{getStatusBadge(lgu.annex1)}</TableCell>

<TableCell>{getStatusBadge(lgu.annex2)}</TableCell>

<TableCell>{getStatusBadge(lgu.annex3)}</TableCell>

<TableCell>{getStatusBadge(lgu.annex4)}</TableCell>

<TableCell>{lgu.lastUpdated}</TableCell>

<TableCell>

<Button variant="ghost" size="sm">

View Details <ChevronRight className="ml-1 h-4 w-4" />

</Button>

</TableCell>

</TableRow>

))}

</TableBody>

</Table>

</CardContent>

</Card>

</div>

) : (

<Card>

<CardHeader>

<CardTitle className="text-2xl">

{activeTab.toUpperCase()} Form

</CardTitle>

<CardDescription>

{user.lgu} - {activeTab.replace('annex', 'Annex ')} Data Entry

</CardDescription>

</CardHeader>

<CardContent>

<div className="bg-gray-100 rounded-lg p-8 text-center">

<FileText className="mx-auto h-12 w-12 text-gray-400" />

<h3 className="mt-2 text-lg font-medium text-gray-900">

{activeTab.toUpperCase()} Data Entry

</h3>

<p className="mt-1 text-gray-500">

This form would collect data for {activeTab.replace('annex', 'Annex ')} from {user.lgu}.

</p>

<div className="mt-6">

<Button onClick={() => setActiveTab('dashboard')}>

<Home className="mr-2 h-4 w-4" /> Back to Dashboard

</Button>

</div>

</div>

</CardContent>

</Card>

)}

</main>

<footer className="bg-white border-t mt-8">

<div className="container mx-auto px-4 py-6 text-center text-sm text-gray-500">

© {new Date().getFullYear()} Provincial Product Accounts System. All rights reserved.

</div>

</footer>

</div>

)

}

// Missing Label component definition - adding it here

function Label({ htmlFor, children }: { htmlFor: string; children: React.ReactNode }) {

return (

<label htmlFor={htmlFor} className="block text-sm font-medium text-gray-700">

{children}

</label>

)

}

make a step by step process in creating it into a web base application, what tools needed, what IDE, what language, what software needed?, the system must have UI interface with CRUD features and login system for client and admin access

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| business\_id\_number | VARCHAR(20) or INT | Unique ID for each business (can be auto-generated or user-input) |
| business\_name | VARCHAR(255) | Trade name / storefront name |
| registered\_name | VARCHAR(255) | Legal name of business (per DTI/SEC/CDA) |
| complete\_address | TEXT | Full business address |
| gross\_revenue | DECIMAL(15,2) | Annual gross revenue |
| tin (Tax Identification Number) | VARCHAR(15) | Official BIR-registered TIN |
| lgu\_psic\_code | VARCHAR(5) | PSIC class code (foreign key to psic\_codes table) |
| main\_activity | TEXT | Main economic activity of the business |
| major\_products\_services | TEXT | Core goods or services offered |
| business\_type | ENUM('Single Proprietorship', 'Partnership', 'Corporation', 'Cooperative') | Business structure |
| total\_employees | INT | Total number of employees |
| male\_employees | INT | Number of male employees |
| female\_employees | INT | Number of female employees |
| telephone\_number | VARCHAR(20) | Landline number |
| mobile\_number | VARCHAR(20) | Mobile contact |
| email\_address | VARCHAR(255) | Official business email |
| social\_media\_accounts | TEXT | Facebook, Instagram, etc. (can use JSON array if storing multiple) |
| website | VARCHAR(255) | Business website URL |
| registration\_authority | ENUM('DTI', 'SEC', 'CDA') | Type of registration |
| registration\_number | VARCHAR(50) | Number assigned by DTI/SEC/CDA |
| year\_started | YEAR or INT | Year the business was started |
| capital | DECIMAL(15,2) | Declared capital |
| assets | DECIMAL(15,2) | Total assets |
| remarks | TEXT | Any special notes |

**psic\_codes Table (Reference)**

**Sql**

CREATE TABLE psic\_codes (

class\_code VARCHAR(5) PRIMARY KEY,

section\_code VARCHAR(2),

section\_description TEXT,

division\_code VARCHAR(2),

division\_description TEXT,

group\_code VARCHAR(3),

group\_description TEXT,

class\_description TEXT

);

**businesses Table**

**sql**

CREATE TABLE businesses (

business\_id\_number VARCHAR(20) PRIMARY KEY,

business\_name VARCHAR(255) NOT NULL,

registered\_name VARCHAR(255),

complete\_address TEXT,

gross\_revenue DECIMAL(15,2),

tin VARCHAR(15),

lgu\_psic\_code VARCHAR(5),

main\_activity TEXT,

major\_products\_services TEXT,

business\_type ENUM('Single Proprietorship', 'Partnership', 'Corporation', 'Cooperative'),

total\_employees INT,

male\_employees INT,

female\_employees INT,

telephone\_number VARCHAR(20),

mobile\_number VARCHAR(20),

email\_address VARCHAR(255),

social\_media\_accounts TEXT,

website VARCHAR(255),

registration\_authority ENUM('DTI', 'SEC', 'CDA'),

registration\_number VARCHAR(50),

year\_started YEAR,

capital DECIMAL(15,2),

assets DECIMAL(15,2),

remarks TEXT,

-- Foreign key to PSIC codes

FOREIGN KEY (lgu\_psic\_code) REFERENCES psic\_codes(class\_code)

);

this is the CRUD input of Annex 1. - BUSINESS ID NUMBER, BUSINESS NAME ,REGISTERED NAME, COMPLETE BUSINESS ADDRESS, GROSS REVENUE, TAX IDENTIFICATION NUMBER, LGU PSIC, MAIN ECONOMIC ACTIVITY, MAJOR PRODUCT/ SERVICES, TYPE OF BUSINESS, TOTAL EMPLOYEES, TOTAL EMPLOYEES MALE, TOTAL EMPLOYEES FEMALE, TELEPHONE NUMBER, MOBILE NUMBER, EMAIL ADDRESS, SOCIAL MEDIA ACCOUNTS, WEBSITE, DTI/SEC/CDA, REGISTRATION NUMBER, STARTED YEAR REGISTERED, CAPITAL , ASSETS, REMARKS

ANNEX 1

import { useState } from 'react'

import { Button } from "/components/ui/button"

import { Input } from "/components/ui/input"

import { Label } from "/components/ui/label"

import { Card, CardContent, CardHeader, CardTitle, CardDescription } from "/components/ui/card"

import { Table, TableBody, TableCell, TableHead, TableHeader, TableRow } from "/components/ui/table"

import { Plus, Trash2, Edit, Save, X } from "lucide-react"

import { Select, SelectContent, SelectItem, SelectTrigger, SelectValue } from "/components/ui/select"

type BusinessRecord = {

id: string

businessIdNumber: string

businessName: string

registeredName: string

completeAddress: string

grossRevenue: string

taxIdNumber: string

lguPsic: string

mainActivity: string

majorProduct: string

businessType: string

totalEmployees: string

maleEmployees: string

femaleEmployees: string

telephone: string

mobile: string

email: string

socialMedia: string

website: string

registrationNumber: string

registeredYear: string

capital: string

assets: string

remarks: string

}

export default function Annex1Form() {

const [records, setRecords] = useState<BusinessRecord[]>([])

const [editingId, setEditingId] = useState<string | null>(null)

const [formData, setFormData] = useState<Omit<BusinessRecord, 'id'>>({

businessIdNumber: '',

businessName: '',

registeredName: '',

completeAddress: '',

grossRevenue: '',

taxIdNumber: '',

lguPsic: '',

mainActivity: '',

majorProduct: '',

businessType: '',

totalEmployees: '',

maleEmployees: '',

femaleEmployees: '',

telephone: '',

mobile: '',

email: '',

socialMedia: '',

website: '',

registrationNumber: '',

registeredYear: '',

capital: '',

assets: '',

remarks: ''

})

const businessTypes = [

'Sole Proprietorship',

'Partnership',

'Corporation',

'Cooperative',

'Others'

]

const psicCodes = [

{ code: '5510', description: 'Hotels and similar accommodation' },

{ code: '5610', description: 'Restaurants and mobile food service' },

{ code: '6201', description: 'Computer programming activities' },

{ code: '6820', description: 'Real estate activities' },

{ code: '9609', description: 'Other personal service activities' }

]

const handleInputChange = (e: React.ChangeEvent<HTMLInputElement>) => {

const { name, value } = e.target

setFormData(prev => ({ ...prev, [name]: value }))

}

const handleAddRecord = () => {

if (editingId) {

setRecords(records.map(record =>

record.id === editingId ? { ...formData, id: editingId } : record

))

setEditingId(null)

} else {

const newRecord = { ...formData, id: Date.now().toString() }

setRecords([...records, newRecord])

}

setFormData({

businessIdNumber: '',

businessName: '',

registeredName: '',

completeAddress: '',

grossRevenue: '',

taxIdNumber: '',

lguPsic: '',

mainActivity: '',

majorProduct: '',

businessType: '',

totalEmployees: '',

maleEmployees: '',

femaleEmployees: '',

telephone: '',

mobile: '',

email: '',

socialMedia: '',

website: '',

registrationNumber: '',

registeredYear: '',

capital: '',

assets: '',

remarks: ''

})

}

const handleEditRecord = (record: BusinessRecord) => {

setFormData({

businessIdNumber: record.businessIdNumber,

businessName: record.businessName,

registeredName: record.registeredName,

completeAddress: record.completeAddress,

grossRevenue: record.grossRevenue,

taxIdNumber: record.taxIdNumber,

lguPsic: record.lguPsic,

mainActivity: record.mainActivity,

majorProduct: record.majorProduct,

businessType: record.businessType,

totalEmployees: record.totalEmployees,

maleEmployees: record.maleEmployees,

femaleEmployees: record.femaleEmployees,

telephone: record.telephone,

mobile: record.mobile,

email: record.email,

socialMedia: record.socialMedia,

website: record.website,

registrationNumber: record.registrationNumber,

registeredYear: record.registeredYear,

capital: record.capital,

assets: record.assets,

remarks: record.remarks

})

setEditingId(record.id)

}

const handleDeleteRecord = (id: string) => {

setRecords(records.filter(record => record.id !== id))

}

const handleCancelEdit = () => {

setEditingId(null)

setFormData({

businessIdNumber: '',

businessName: '',

registeredName: '',

completeAddress: '',

grossRevenue: '',

taxIdNumber: '',

lguPsic: '',

mainActivity: '',

majorProduct: '',

businessType: '',

totalEmployees: '',

maleEmployees: '',

femaleEmployees: '',

telephone: '',

mobile: '',

email: '',

socialMedia: '',

website: '',

registrationNumber: '',

registeredYear: '',

capital: '',

assets: '',

remarks: ''

})

}

return (

<div className="space-y-6">

<Card>

<CardHeader>

<CardTitle>Annex 1 - Business Establishment Form</CardTitle>

<CardDescription>

Register and manage business establishments in your LGU

</CardDescription>

</CardHeader>

<CardContent>

<div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4">

{/\* Basic Information \*/}

<div className="space-y-2">

<Label>Business ID Number</Label>

<Input

name="businessIdNumber"

value={formData.businessIdNumber}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Business Name</Label>

<Input

name="businessName"

value={formData.businessName}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Registered Name</Label>

<Input

name="registeredName"

value={formData.registeredName}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2 md:col-span-2">

<Label>Complete Business Address</Label>

<Input

name="completeAddress"

value={formData.completeAddress}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Gross Revenue</Label>

<Input

name="grossRevenue"

type="number"

value={formData.grossRevenue}

onChange={handleInputChange}

/>

</div>

{/\* Tax and Registration \*/}

<div className="space-y-2">

<Label>Tax Identification Number</Label>

<Input

name="taxIdNumber"

value={formData.taxIdNumber}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>LGU PSIC</Label>

<Select

value={formData.lguPsic}

onValueChange={(value) => setFormData({...formData, lguPsic: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select PSIC Code" />

</SelectTrigger>

<SelectContent>

{psicCodes.map((code) => (

<SelectItem key={code.code} value={code.code}>

{code.code} - {code.description}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

<div className="space-y-2">

<Label>Main Economic Activity</Label>

<Input

name="mainActivity"

value={formData.mainActivity}

onChange={handleInputChange}

/>

</div>

{/\* Business Details \*/}

<div className="space-y-2">

<Label>Major Product/Services</Label>

<Input

name="majorProduct"

value={formData.majorProduct}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Type of Business</Label>

<Select

value={formData.businessType}

onValueChange={(value) => setFormData({...formData, businessType: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select Business Type" />

</SelectTrigger>

<SelectContent>

{businessTypes.map((type) => (

<SelectItem key={type} value={type}>

{type}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

<div className="space-y-2">

<Label>Total Employees</Label>

<Input

name="totalEmployees"

type="number"

value={formData.totalEmployees}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Male Employees</Label>

<Input

name="maleEmployees"

type="number"

value={formData.maleEmployees}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Female Employees</Label>

<Input

name="femaleEmployees"

type="number"

value={formData.femaleEmployees}

onChange={handleInputChange}

/>

</div>

{/\* Contact Information \*/}

<div className="space-y-2">

<Label>Telephone Number</Label>

<Input

name="telephone"

value={formData.telephone}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Mobile Number</Label>

<Input

name="mobile"

value={formData.mobile}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Email Address</Label>

<Input

name="email"

type="email"

value={formData.email}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Social Media Accounts</Label>

<Input

name="socialMedia"

value={formData.socialMedia}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Website</Label>

<Input

name="website"

value={formData.website}

onChange={handleInputChange}

/>

</div>

{/\* Registration Details \*/}

<div className="space-y-2">

<Label>DTI/SEC/CDA Registration Number</Label>

<Input

name="registrationNumber"

value={formData.registrationNumber}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Started Year Registered</Label>

<Input

name="registeredYear"

type="number"

value={formData.registeredYear}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Capital</Label>

<Input

name="capital"

type="number"

value={formData.capital}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Assets</Label>

<Input

name="assets"

type="number"

value={formData.assets}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2 md:col-span-2">

<Label>Remarks</Label>

<Input

name="remarks"

value={formData.remarks}

onChange={handleInputChange}

/>

</div>

</div>

<div className="flex justify-end mt-6 space-x-2">

{editingId && (

<Button variant="outline" onClick={handleCancelEdit}>

<X className="mr-2 h-4 w-4" /> Cancel

</Button>

)}

<Button onClick={handleAddRecord}>

{editingId ? (

<>

<Save className="mr-2 h-4 w-4" /> Update Record

</>

) : (

<>

<Plus className="mr-2 h-4 w-4" /> Add Record

</>

)}

</Button>

</div>

</CardContent>

</Card>

{/\* Records Table \*/}

{records.length > 0 && (

<Card>

<CardHeader>

<CardTitle>Registered Businesses</CardTitle>

<CardDescription>

List of all business establishments in the system

</CardDescription>

</CardHeader>

<CardContent>

<Table>

<TableHeader>

<TableRow>

<TableHead>Business ID</TableHead>

<TableHead>Business Name</TableHead>

<TableHead>Registered Name</TableHead>

<TableHead>Address</TableHead>

<TableHead>Actions</TableHead>

</TableRow>

</TableHeader>

<TableBody>

{records.map((record) => (

<TableRow key={record.id}>

<TableCell>{record.businessIdNumber}</TableCell>

<TableCell>{record.businessName}</TableCell>

<TableCell>{record.registeredName}</TableCell>

<TableCell>{record.completeAddress}</TableCell>

<TableCell>

<div className="flex space-x-2">

<Button

variant="outline"

size="sm"

onClick={() => handleEditRecord(record)}

>

<Edit className="h-4 w-4" />

</Button>

<Button

variant="destructive"

size="sm"

onClick={() => handleDeleteRecord(record.id)}

>

<Trash2 className="h-4 w-4" />

</Button>

</div>

</TableCell>

</TableRow>

))}

</TableBody>

</Table>

</CardContent>

</Card>

)}

</div>

)

}

this is the CRUD input of Annex 2. - Project Name, Total Project Cost/Amount (In PhP),Fund Source,Year Started, Target Year of Completion, Physical Accomplishment (In PhP), Estimated Physical Accomplishment (In Percent), Physical Accomplishment (In PhP),REMARKS

import { useState } from 'react'

import { Button } from "/components/ui/button"

import { Input } from "/components/ui/input"

import { Label } from "/components/ui/label"

import { Card, CardContent, CardHeader, CardTitle, CardDescription } from "/components/ui/card"

import { Table, TableBody, TableCell, TableHead, TableHeader, TableRow } from "/components/ui/table"

import { Plus, Trash2, Edit, Save, X } from "lucide-react"

import { Select, SelectContent, SelectItem, SelectTrigger, SelectValue } from "/components/ui/select"

type ProjectRecord = {

id: string

projectName: string

totalCost: string

fundSource: string

yearStarted: string

targetCompletionYear: string

physicalAccomplishmentPhp: string

estimatedPhysicalAccomplishmentPercent: string

remarks: string

}

export default function Annex2Form() {

const [records, setRecords] = useState<ProjectRecord[]>([])

const [editingId, setEditingId] = useState<string | null>(null)

const [formData, setFormData] = useState<Omit<ProjectRecord, 'id'>>({

projectName: '',

totalCost: '',

fundSource: '',

yearStarted: '',

targetCompletionYear: '',

physicalAccomplishmentPhp: '',

estimatedPhysicalAccomplishmentPercent: '',

remarks: ''

})

const fundSources = [

'Local Government Unit (LGU)',

'National Government',

'Official Development Assistance (ODA)',

'Private Sector',

'Public-Private Partnership (PPP)',

'Others'

]

const currentYear = new Date().getFullYear()

const yearOptions = Array.from({ length: 10 }, (\_, i) => currentYear - 5 + i)

const handleInputChange = (e: React.ChangeEvent<HTMLInputElement>) => {

const { name, value } = e.target

setFormData(prev => ({ ...prev, [name]: value }))

}

const handleAddRecord = () => {

if (editingId) {

setRecords(records.map(record =>

record.id === editingId ? { ...formData, id: editingId } : record

))

setEditingId(null)

} else {

const newRecord = { ...formData, id: Date.now().toString() }

setRecords([...records, newRecord])

}

setFormData({

projectName: '',

totalCost: '',

fundSource: '',

yearStarted: '',

targetCompletionYear: '',

physicalAccomplishmentPhp: '',

estimatedPhysicalAccomplishmentPercent: '',

remarks: ''

})

}

const handleEditRecord = (record: ProjectRecord) => {

setFormData({

projectName: record.projectName,

totalCost: record.totalCost,

fundSource: record.fundSource,

yearStarted: record.yearStarted,

targetCompletionYear: record.targetCompletionYear,

physicalAccomplishmentPhp: record.physicalAccomplishmentPhp,

estimatedPhysicalAccomplishmentPercent: record.estimatedPhysicalAccomplishmentPercent,

remarks: record.remarks

})

setEditingId(record.id)

}

const handleDeleteRecord = (id: string) => {

setRecords(records.filter(record => record.id !== id))

}

const handleCancelEdit = () => {

setEditingId(null)

setFormData({

projectName: '',

totalCost: '',

fundSource: '',

yearStarted: '',

targetCompletionYear: '',

physicalAccomplishmentPhp: '',

estimatedPhysicalAccomplishmentPercent: '',

remarks: ''

})

}

const calculateCompletionPercentage = (cost: string, accomplishment: string) => {

if (!cost || !accomplishment) return '0%'

const costNum = parseFloat(cost.replace(/,/g, ''))

const accomplishmentNum = parseFloat(accomplishment.replace(/,/g, ''))

if (costNum === 0) return '0%'

return `${((accomplishmentNum / costNum) \* 100).toFixed(2)}%`

}

return (

<div className="space-y-6">

<Card>

<CardHeader>

<CardTitle>Annex 2 - Projects Form</CardTitle>

<CardDescription>

Register and manage projects in your LGU

</CardDescription>

</CardHeader>

<CardContent>

<div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4">

{/\* Project Information \*/}

<div className="space-y-2 md:col-span-2">

<Label>Project Name</Label>

<Input

name="projectName"

value={formData.projectName}

onChange={handleInputChange}

required

/>

</div>

<div className="space-y-2">

<Label>Total Project Cost (₱)</Label>

<Input

name="totalCost"

type="number"

value={formData.totalCost}

onChange={handleInputChange}

required

/>

</div>

<div className="space-y-2">

<Label>Fund Source</Label>

<Select

value={formData.fundSource}

onValueChange={(value) => setFormData({...formData, fundSource: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select Fund Source" />

</SelectTrigger>

<SelectContent>

{fundSources.map((source) => (

<SelectItem key={source} value={source}>

{source}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

{/\* Timeline \*/}

<div className="space-y-2">

<Label>Year Started</Label>

<Select

value={formData.yearStarted}

onValueChange={(value) => setFormData({...formData, yearStarted: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select Year" />

</SelectTrigger>

<SelectContent>

{yearOptions.map((year) => (

<SelectItem key={year} value={year.toString()}>

{year}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

<div className="space-y-2">

<Label>Target Year of Completion</Label>

<Select

value={formData.targetCompletionYear}

onValueChange={(value) => setFormData({...formData, targetCompletionYear: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select Year" />

</SelectTrigger>

<SelectContent>

{yearOptions.map((year) => (

<SelectItem key={year} value={year.toString()}>

{year}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

{/\* Accomplishment \*/}

<div className="space-y-2">

<Label>Physical Accomplishment (₱)</Label>

<Input

name="physicalAccomplishmentPhp"

type="number"

value={formData.physicalAccomplishmentPhp}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Estimated Physical Accomplishment (%)</Label>

<Input

name="estimatedPhysicalAccomplishmentPercent"

type="number"

min="0"

max="100"

value={formData.estimatedPhysicalAccomplishmentPercent}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2 md:col-span-2">

<Label>Remarks</Label>

<Input

name="remarks"

value={formData.remarks}

onChange={handleInputChange}

/>

</div>

</div>

<div className="flex justify-end mt-6 space-x-2">

{editingId && (

<Button variant="outline" onClick={handleCancelEdit}>

<X className="mr-2 h-4 w-4" /> Cancel

</Button>

)}

<Button onClick={handleAddRecord}>

{editingId ? (

<>

<Save className="mr-2 h-4 w-4" /> Update Project

</>

) : (

<>

<Plus className="mr-2 h-4 w-4" /> Add Project

</>

)}

</Button>

</div>

</CardContent>

</Card>

{/\* Records Table \*/}

{records.length > 0 && (

<Card>

<CardHeader>

<CardTitle>Registered Projects</CardTitle>

<CardDescription>

List of all projects in the system

</CardDescription>

</CardHeader>

<CardContent>

<Table>

<TableHeader>

<TableRow>

<TableHead>Project Name</TableHead>

<TableHead>Total Cost (₱)</TableHead>

<TableHead>Fund Source</TableHead>

<TableHead>Years</TableHead>

<TableHead>Accomplishment</TableHead>

<TableHead>Actions</TableHead>

</TableRow>

</TableHeader>

<TableBody>

{records.map((record) => (

<TableRow key={record.id}>

<TableCell className="font-medium">{record.projectName}</TableCell>

<TableCell>

{parseFloat(record.totalCost).toLocaleString('en-PH', {

style: 'currency',

currency: 'PHP',

minimumFractionDigits: 2

})}

</TableCell>

<TableCell>{record.fundSource}</TableCell>

<TableCell>

{record.yearStarted} - {record.targetCompletionYear}

</TableCell>

<TableCell>

<div className="flex flex-col">

<span>

{parseFloat(record.physicalAccomplishmentPhp).toLocaleString('en-PH', {

style: 'currency',

currency: 'PHP',

minimumFractionDigits: 2

})}

</span>

<span className="text-sm text-gray-500">

{calculateCompletionPercentage(

record.totalCost,

record.physicalAccomplishmentPhp

)}

</span>

</div>

</TableCell>

<TableCell>

<div className="flex space-x-2">

<Button

variant="outline"

size="sm"

onClick={() => handleEditRecord(record)}

>

<Edit className="h-4 w-4" />

</Button>

<Button

variant="destructive"

size="sm"

onClick={() => handleDeleteRecord(record.id)}

>

<Trash2 className="h-4 w-4" />

</Button>

</div>

</TableCell>

</TableRow>

))}

</TableBody>

</Table>

</CardContent>

</Card>

)}

</div>

)

}

this is the CRUD input of Annex 3. -

number of permanent employee who are involve in Education , Health, Nutrition and Population Control Social Services ,Other LGU Units

number of Co-Terminus to the Official employee who are involve in Education , Health, Nutrition and Population Control Social Services ,Other LGU Units

number of Others -Plantilla, J.O, Elected employee who are involve in Education , Health, Nutrition and Population Control Social Services ,Other LGU Units

import { useState } from 'react'

import { Button } from "/components/ui/button"

import { Input } from "/components/ui/input"

import { Label } from "/components/ui/label"

import { Card, CardContent, CardHeader, CardTitle, CardDescription } from "/components/ui/card"

import { Table, TableBody, TableCell, TableHead, TableHeader, TableRow } from "/components/ui/table"

import { Plus, Trash2, Edit, Save, X } from "lucide-react"

import {

Select,

SelectContent,

SelectItem,

SelectTrigger,

SelectValue,

} from "/components/ui/select"

type EmployeeRecord = {

id: string

department: string

permanentEducation: string

permanentHealth: string

permanentNutrition: string

permanentOtherUnits: string

coterminusEducation: string

coterminusHealth: string

coterminusNutrition: string

coterminusOtherUnits: string

otherEducation: string

otherHealth: string

otherNutrition: string

otherOtherUnits: string

year: string

remarks: string

}

export default function Annex3Form() {

const [records, setRecords] = useState<EmployeeRecord[]>([])

const [editingId, setEditingId] = useState<string | null>(null)

const [formData, setFormData] = useState<Omit<EmployeeRecord, 'id'>>({

department: '',

permanentEducation: '',

permanentHealth: '',

permanentNutrition: '',

permanentOtherUnits: '',

coterminusEducation: '',

coterminusHealth: '',

coterminusNutrition: '',

coterminusOtherUnits: '',

otherEducation: '',

otherHealth: '',

otherNutrition: '',

otherOtherUnits: '',

year: '',

remarks: ''

})

const departments = [

'Mayor\'s Office',

'Health Office',

'Education Office',

'Social Welfare',

'Engineering Office',

'Treasurer\'s Office',

'Assessor\'s Office',

'Planning Office',

'Others'

]

const currentYear = new Date().getFullYear()

const yearOptions = Array.from({ length: 5 }, (\_, i) => currentYear - i)

const handleInputChange = (e: React.ChangeEvent<HTMLInputElement>) => {

const { name, value } = e.target

setFormData(prev => ({ ...prev, [name]: value }))

}

const handleAddRecord = () => {

if (editingId) {

setRecords(records.map(record =>

record.id === editingId ? { ...formData, id: editingId } : record

))

setEditingId(null)

} else {

const newRecord = { ...formData, id: Date.now().toString() }

setRecords([...records, newRecord])

}

setFormData({

department: '',

permanentEducation: '',

permanentHealth: '',

permanentNutrition: '',

permanentOtherUnits: '',

coterminusEducation: '',

coterminusHealth: '',

coterminusNutrition: '',

coterminusOtherUnits: '',

otherEducation: '',

otherHealth: '',

otherNutrition: '',

otherOtherUnits: '',

year: '',

remarks: ''

})

}

const handleEditRecord = (record: EmployeeRecord) => {

setFormData({

department: record.department,

permanentEducation: record.permanentEducation,

permanentHealth: record.permanentHealth,

permanentNutrition: record.permanentNutrition,

permanentOtherUnits: record.permanentOtherUnits,

coterminusEducation: record.coterminusEducation,

coterminusHealth: record.coterminusHealth,

coterminusNutrition: record.coterminusNutrition,

coterminusOtherUnits: record.coterminusOtherUnits,

otherEducation: record.otherEducation,

otherHealth: record.otherHealth,

otherNutrition: record.otherNutrition,

otherOtherUnits: record.otherOtherUnits,

year: record.year,

remarks: record.remarks

})

setEditingId(record.id)

}

const handleDeleteRecord = (id: string) => {

setRecords(records.filter(record => record.id !== id))

}

const handleCancelEdit = () => {

setEditingId(null)

setFormData({

department: '',

permanentEducation: '',

permanentHealth: '',

permanentNutrition: '',

permanentOtherUnits: '',

coterminusEducation: '',

coterminusHealth: '',

coterminusNutrition: '',

coterminusOtherUnits: '',

otherEducation: '',

otherHealth: '',

otherNutrition: '',

otherOtherUnits: '',

year: '',

remarks: ''

})

}

const calculateTotal = (record: EmployeeRecord, category: 'permanent' | 'coterminus' | 'other') => {

const fields = category === 'permanent'

? [record.permanentEducation, record.permanentHealth, record.permanentNutrition, record.permanentOtherUnits]

: category === 'coterminus'

? [record.coterminusEducation, record.coterminusHealth, record.coterminusNutrition, record.coterminusOtherUnits]

: [record.otherEducation, record.otherHealth, record.otherNutrition, record.otherOtherUnits]

return fields.reduce((sum, val) => sum + (parseInt(val) || 0), 0)

}

return (

<div className="space-y-6">

<Card>

<CardHeader>

<CardTitle>Annex 3 - Human Resource Inventory</CardTitle>

<CardDescription>

Track employees across different LGU units and employment types

</CardDescription>

</CardHeader>

<CardContent>

<div className="grid grid-cols-1 md:grid-cols-2 gap-4">

{/\* Department and Year \*/}

<div className="space-y-2">

<Label>Department/Office</Label>

<Select

value={formData.department}

onValueChange={(value) => setFormData({...formData, department: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select Department" />

</SelectTrigger>

<SelectContent>

{departments.map((dept) => (

<SelectItem key={dept} value={dept}>

{dept}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

<div className="space-y-2">

<Label>Year</Label>

<Select

value={formData.year}

onValueChange={(value) => setFormData({...formData, year: value})}

>

<SelectTrigger>

<SelectValue placeholder="Select Year" />

</SelectTrigger>

<SelectContent>

{yearOptions.map((year) => (

<SelectItem key={year} value={year.toString()}>

{year}

</SelectItem>

))}

</SelectContent>

</Select>

</div>

</div>

{/\* Permanent Employees \*/}

<div className="mt-6">

<h3 className="font-medium mb-4">Permanent Employees</h3>

<div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-4">

<div className="space-y-2">

<Label>Education</Label>

<Input

name="permanentEducation"

type="number"

min="0"

value={formData.permanentEducation}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Health</Label>

<Input

name="permanentHealth"

type="number"

min="0"

value={formData.permanentHealth}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Nutrition & Population</Label>

<Input

name="permanentNutrition"

type="number"

min="0"

value={formData.permanentNutrition}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Other LGU Units</Label>

<Input

name="permanentOtherUnits"

type="number"

min="0"

value={formData.permanentOtherUnits}

onChange={handleInputChange}

/>

</div>

</div>

</div>

{/\* Co-Terminus Employees \*/}

<div className="mt-6">

<h3 className="font-medium mb-4">Co-Terminus Employees</h3>

<div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-4">

<div className="space-y-2">

<Label>Education</Label>

<Input

name="coterminusEducation"

type="number"

min="0"

value={formData.coterminusEducation}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Health</Label>

<Input

name="coterminusHealth"

type="number"

min="0"

value={formData.coterminusHealth}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Nutrition & Population</Label>

<Input

name="coterminusNutrition"

type="number"

min="0"

value={formData.coterminusNutrition}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Other LGU Units</Label>

<Input

name="coterminusOtherUnits"

type="number"

min="0"

value={formData.coterminusOtherUnits}

onChange={handleInputChange}

/>

</div>

</div>

</div>

{/\* Other Employees \*/}

<div className="mt-6">

<h3 className="font-medium mb-4">Other Employees (Plantilla, J.O., Elected)</h3>

<div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-4">

<div className="space-y-2">

<Label>Education</Label>

<Input

name="otherEducation"

type="number"

min="0"

value={formData.otherEducation}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Health</Label>

<Input

name="otherHealth"

type="number"

min="0"

value={formData.otherHealth}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Nutrition & Population</Label>

<Input

name="otherNutrition"

type="number"

min="0"

value={formData.otherNutrition}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Other LGU Units</Label>

<Input

name="otherOtherUnits"

type="number"

min="0"

value={formData.otherOtherUnits}

onChange={handleInputChange}

/>

</div>

</div>

</div>

{/\* Remarks \*/}

<div className="mt-6">

<Label>Remarks</Label>

<Input

name="remarks"

value={formData.remarks}

onChange={handleInputChange}

/>

</div>

<div className="flex justify-end mt-6 space-x-2">

{editingId && (

<Button variant="outline" onClick={handleCancelEdit}>

<X className="mr-2 h-4 w-4" /> Cancel

</Button>

)}

<Button onClick={handleAddRecord}>

{editingId ? (

<>

<Save className="mr-2 h-4 w-4" /> Update Record

</>

) : (

<>

<Plus className="mr-2 h-4 w-4" /> Add Record

</>

)}

</Button>

</div>

</CardContent>

</Card>

{/\* Records Table \*/}

{records.length > 0 && (

<Card>

<CardHeader>

<CardTitle>Employee Records</CardTitle>

<CardDescription>

Summary of employees by department and employment type

</CardDescription>

</CardHeader>

<CardContent>

<Table>

<TableHeader>

<TableRow>

<TableHead>Department</TableHead>

<TableHead>Year</TableHead>

<TableHead>Permanent</TableHead>

<TableHead>Co-Terminus</TableHead>

<TableHead>Other</TableHead>

<TableHead>Total</TableHead>

<TableHead>Actions</TableHead>

</TableRow>

</TableHeader>

<TableBody>

{records.map((record) => (

<TableRow key={record.id}>

<TableCell className="font-medium">{record.department}</TableCell>

<TableCell>{record.year}</TableCell>

<TableCell>{calculateTotal(record, 'permanent')}</TableCell>

<TableCell>{calculateTotal(record, 'coterminus')}</TableCell>

<TableCell>{calculateTotal(record, 'other')}</TableCell>

<TableCell>

{calculateTotal(record, 'permanent') +

calculateTotal(record, 'coterminus') +

calculateTotal(record, 'other')}

</TableCell>

<TableCell>

<div className="flex space-x-2">

<Button

variant="outline"

size="sm"

onClick={() => handleEditRecord(record)}

>

<Edit className="h-4 w-4" />

</Button>

<Button

variant="destructive"

size="sm"

onClick={() => handleDeleteRecord(record.id)}

>

<Trash2 className="h-4 w-4" />

</Button>

</div>

</TableCell>

</TableRow>

))}

</TableBody>

</Table>

</CardContent>

</Card>

)}

</div>

)

}

this is the CRUD input of Annex 4. –

Coordinator:

Name,

Designation,

Office/Unit,

Mailing address,

Telephone Number,

Fax Number,

Email address

Alternate Coordinator:

Name,

Designation,

Office/Unit,

Mailing address,

Telephone Number,

Fax Number,

Email address

import { useState } from 'react'

import { Button } from "/components/ui/button"

import { Input } from "/components/ui/input"

import { Label } from "/components/ui/label"

import { Card, CardContent, CardHeader, CardTitle, CardDescription } from "/components/ui/card"

import { Table, TableBody, TableCell, TableHead, TableHeader, TableRow } from "/components/ui/table"

import { Plus, Trash2, Edit, Save, X } from "lucide-react"

type CoordinatorRecord = {

id: string

type: 'coordinator' | 'alternate'

name: string

designation: string

officeUnit: string

mailingAddress: string

telephoneNumber: string

faxNumber: string

email: string

}

export default function Annex4Form() {

const [records, setRecords] = useState<CoordinatorRecord[]>([])

const [editingId, setEditingId] = useState<string | null>(null)

const [formData, setFormData] = useState<Omit<CoordinatorRecord, 'id'>>({

type: 'coordinator',

name: '',

designation: '',

officeUnit: '',

mailingAddress: '',

telephoneNumber: '',

faxNumber: '',

email: ''

})

const handleInputChange = (e: React.ChangeEvent<HTMLInputElement>) => {

const { name, value } = e.target

setFormData(prev => ({ ...prev, [name]: value }))

}

const handleTypeChange = (type: 'coordinator' | 'alternate') => {

setFormData(prev => ({ ...prev, type }))

}

const handleAddRecord = () => {

if (editingId) {

setRecords(records.map(record =>

record.id === editingId ? { ...formData, id: editingId } : record

))

setEditingId(null)

} else {

const newRecord = { ...formData, id: Date.now().toString() }

setRecords([...records, newRecord])

}

setFormData({

type: 'coordinator',

name: '',

designation: '',

officeUnit: '',

mailingAddress: '',

telephoneNumber: '',

faxNumber: '',

email: ''

})

}

const handleEditRecord = (record: CoordinatorRecord) => {

setFormData({

type: record.type,

name: record.name,

designation: record.designation,

officeUnit: record.officeUnit,

mailingAddress: record.mailingAddress,

telephoneNumber: record.telephoneNumber,

faxNumber: record.faxNumber,

email: record.email

})

setEditingId(record.id)

}

const handleDeleteRecord = (id: string) => {

setRecords(records.filter(record => record.id !== id))

}

const handleCancelEdit = () => {

setEditingId(null)

setFormData({

type: 'coordinator',

name: '',

designation: '',

officeUnit: '',

mailingAddress: '',

telephoneNumber: '',

faxNumber: '',

email: ''

})

}

const coordinators = records.filter(r => r.type === 'coordinator')

const alternates = records.filter(r => r.type === 'alternate')

return (

<div className="space-y-6">

<Card>

<CardHeader>

<CardTitle>Annex 4 - Coordinator Information</CardTitle>

<CardDescription>

Manage contact information for coordinators and alternate coordinators

</CardDescription>

</CardHeader>

<CardContent>

<div className="flex space-x-4 mb-6">

<Button

variant={formData.type === 'coordinator' ? 'default' : 'outline'}

onClick={() => handleTypeChange('coordinator')}

>

Coordinator

</Button>

<Button

variant={formData.type === 'alternate' ? 'default' : 'outline'}

onClick={() => handleTypeChange('alternate')}

>

Alternate Coordinator

</Button>

</div>

<div className="grid grid-cols-1 md:grid-cols-2 gap-4">

<div className="space-y-2">

<Label>Full Name</Label>

<Input

name="name"

value={formData.name}

onChange={handleInputChange}

required

/>

</div>

<div className="space-y-2">

<Label>Designation</Label>

<Input

name="designation"

value={formData.designation}

onChange={handleInputChange}

required

/>

</div>

<div className="space-y-2">

<Label>Office/Unit</Label>

<Input

name="officeUnit"

value={formData.officeUnit}

onChange={handleInputChange}

required

/>

</div>

<div className="space-y-2">

<Label>Mailing Address</Label>

<Input

name="mailingAddress"

value={formData.mailingAddress}

onChange={handleInputChange}

required

/>

</div>

<div className="space-y-2">

<Label>Telephone Number</Label>

<Input

name="telephoneNumber"

type="tel"

value={formData.telephoneNumber}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2">

<Label>Fax Number</Label>

<Input

name="faxNumber"

type="tel"

value={formData.faxNumber}

onChange={handleInputChange}

/>

</div>

<div className="space-y-2 md:col-span-2">

<Label>Email Address</Label>

<Input

name="email"

type="email"

value={formData.email}

onChange={handleInputChange}

required

/>

</div>

</div>

<div className="flex justify-end mt-6 space-x-2">

{editingId && (

<Button variant="outline" onClick={handleCancelEdit}>

<X className="mr-2 h-4 w-4" /> Cancel

</Button>

)}

<Button onClick={handleAddRecord}>

{editingId ? (

<>

<Save className="mr-2 h-4 w-4" /> Update Record

</>

) : (

<>

<Plus className="mr-2 h-4 w-4" /> Add Record

</>

)}

</Button>

</div>

</CardContent>

</Card>

{/\* Records Tables \*/}

{coordinators.length > 0 && (

<Card>

<CardHeader>

<CardTitle>Coordinators</CardTitle>

<CardDescription>

List of registered coordinators

</CardDescription>

</CardHeader>

<CardContent>

<Table>

<TableHeader>

<TableRow>

<TableHead>Name</TableHead>

<TableHead>Designation</TableHead>

<TableHead>Office/Unit</TableHead>

<TableHead>Contact</TableHead>

<TableHead>Actions</TableHead>

</TableRow>

</TableHeader>

<TableBody>

{coordinators.map((record) => (

<TableRow key={record.id}>

<TableCell className="font-medium">{record.name}</TableCell>

<TableCell>{record.designation}</TableCell>

<TableCell>{record.officeUnit}</TableCell>

<TableCell>

<div className="flex flex-col">

<span>{record.telephoneNumber}</span>

<span className="text-sm text-gray-500">{record.email}</span>

</div>

</TableCell>

<TableCell>

<div className="flex space-x-2">

<Button

variant="outline"

size="sm"

onClick={() => handleEditRecord(record)}

>

<Edit className="h-4 w-4" />

</Button>

<Button

variant="destructive"

size="sm"

onClick={() => handleDeleteRecord(record.id)}

>

<Trash2 className="h-4 w-4" />

</Button>

</div>

</TableCell>

</TableRow>

))}

</TableBody>

</Table>

</CardContent>

</Card>

)}

{alternates.length > 0 && (

<Card>

<CardHeader>

<CardTitle>Alternate Coordinators</CardTitle>

<CardDescription>

List of registered alternate coordinators

</CardDescription>

</CardHeader>

<CardContent>

<Table>

<TableHeader>

<TableRow>

<TableHead>Name</TableHead>

<TableHead>Designation</TableHead>

<TableHead>Office/Unit</TableHead>

<TableHead>Contact</TableHead>

<TableHead>Actions</TableHead>

</TableRow>

</TableHeader>

<TableBody>

{alternates.map((record) => (

<TableRow key={record.id}>

<TableCell className="font-medium">{record.name}</TableCell>

<TableCell>{record.designation}</TableCell>

<TableCell>{record.officeUnit}</TableCell>

<TableCell>

<div className="flex flex-col">

<span>{record.telephoneNumber}</span>

<span className="text-sm text-gray-500">{record.email}</span>

</div>

</TableCell>

<TableCell>

<div className="flex space-x-2">

<Button

variant="outline"

size="sm"

onClick={() => handleEditRecord(record)}

>

<Edit className="h-4 w-4" />

</Button>

<Button

variant="destructive"

size="sm"

onClick={() => handleDeleteRecord(record.id)}

>

<Trash2 className="h-4 w-4" />

</Button>

</div>

</TableCell>

</TableRow>

))}

</TableBody>

</Table>

</CardContent>

</Card>

)}

</div>

)

}