**CREATE: Documentație**

**1. Introducere**

**Descriere generală**

Proiectul sprijină tranziția urbană către economia circulară prin inventarierea stocurilor de materiale din construcții, dezvoltarea scenariilor pentru fluxurile de materiale și furnizarea de aranjamente de guvernanță. Se concentrează pe infrastructurile urbane majore precum clădiri și drumuri municipale. Un consorțiu transdisciplinar combină modelarea cantitativă cu studiul calitativ. Proiectul îmbunătățește instrumente existente și integrează noi tehnologii de digitalizare, implicând diverse părți interesate prin laboratoare urbane și orașe partenere din cinci țări. Analiza celor mai bune practici și noi intervenții de guvernanță va duce la propuneri concrete pentru orașele participante și o strategie de extindere pentru Europa.

**Scopul aplicației**

Proiectul vizează cuantificarea și vizualizarea stocurilor și fluxurilor de materiale din mediul urban construit, precum betonul și oțelul, atât în stocurile acumulate, cât și în fluxurile necesare pentru construcții noi și renovări. Evaluarea bazată pe KPI va identifica potențialul de reciclare și reutilizare, comparând impactul utilizării resurselor primare și secundare. Se va adopta o abordare centrată pe părțile interesate pentru a vizualiza stocurile și fluxurile de materiale. Laboratoarele urbane vor experimenta noi aranjamente de guvernanță. Sub-obiectivele includ definirea metodologiei comune, dezvoltarea soluțiilor inovatoare de vizualizare, co-crearea aranjamentelor de guvernanță, evaluarea impactului în laboratoarele urbane și co-difuzarea celor mai bune practici pentru extinderea capacității de guvernanță în Europa.

**2. Instalare și configurare**

**Cerințe de sistem**

* **Node.js**
* **npm**
* **Browser**: Chrome, Firefox, Edge sau Safari.

**Pași pentru instalare**

1. **Descărcare cod sursă**: Descarcă și extrage arhiva codului sursă pe mașina locală.
2. **Instalare dependențe**: Deschide un terminal și navighează la directorul principal al aplicației, apoi rulează comanda:

npm install

Această comandă va instala toate modulele și pachetele necesare pentru rularea aplicației.

1. **Rulare aplicație**: După instalarea tuturor dependențelor, pornește aplicația cu comanda:

npm start

Aplicația va fi disponibilă la <http://localhost:3000>. (posibil la <http://localhost:5000>)

**3. Arhitectura aplicației**

**Structura directorilor și fișierelor**

Aplicația este organizată într-o structură de directoare bine definită, care include fișierele sursă, resursele statice și fișierele de configurare. Iată o prezentare generală a structurii principale:

* **public/**: Conține fișierele statice și resursele publice.
  + index.html: Punctul de intrare HTML al aplicației. Acest fișier este servit de serverul web și încarcă aplicația React.
  + city-images/: Imagini utilizate în diverse părți ale aplicației, cum ar fi vizualizarea proiectelor și afișarea detaliilor clădirilor.
* **src/**: Directorul principal al codului sursă.
  + App.js: Componenta principală a aplicației. Acest fișier gestionează rutele principale și structura generală a aplicației.
  + index.js: Punctul de intrare al aplicației React. Aici se inițializează aplicația și se montează componenta principală în DOM.
  + [Create\_project.js](#_Create_projects.js_/_Create_project): Gestionarea creării proiectelor noi. Această componentă include formularul pentru adăugarea detaliilor proiectului și logica aferentă.
  + [Dashboard.js](#_Dashboard.js_/_Dashboard.module.scs): Vizualizarea tabloului de bord. Afișează o privire de ansamblu asupra proiectelor și activităților curente.
  + [Forgot-password.js](#_Forgot-password.js_/_Forgot-passwor): Recuperarea parolei. Permite utilizatorilor să reseteze parola în caz de uitare.
  + [Login.js](#_Login.js_/_Login.module.scss:): Autentificare utilizator. Gestionarea procesului de autentificare a utilizatorilor în aplicație.
  + [Projects.js](#_Projects.js_/_Projects.module.scss:): Managementul proiectelor. Afișează lista de proiecte și permite interacțiunea cu acestea.
  + [Select\_perimeter.js](#_SelectPerimeter.js_/_SelectPerimete): Selectarea perimetrului. Permite utilizatorilor să definească și să selecteze perimetre pentru proiecte.
  + [TutorialUserManual.js](#_TutorialUserManual.js_/_TutorialUse): Manual de utilizare pentru utilizatori. Oferă ghiduri și instrucțiuni pentru utilizarea aplicației.
  + [**building-info-tabs/**:](#_BuildingInformation.js_/_BuildingIn) Componente pentru afișarea informațiilor detaliate despre clădiri. Include secțiuni pentru calitatea materialelor și conținutul acestora.
  + [**map**](#_Map.js:)**/**: Componente pentru hărți și vizualizarea datelor geospațiale. Permite utilizatorilor să vizualizeze proiectele pe hartă și să interacționeze cu datele spațiale.
* **styles**: Fișiere SCSS pentru stilizarea aplicației. Aceste fișiere controlează aspectul vizual al componentelor și al interfeței utilizator.

**Descrierea componentelor principale**

* **App.js**: Componenta de bază care structurează aplicația și gestionează rutele principale. Este punctul central de coordonare a aplicației.
* **Dashboard.js**: Oferă o privire de ansamblu asupra proiectelor și activităților curente. Este componenta principală afișată după autentificare.
* **Login.js**: Permite utilizatorilor să se autentifice în aplicație. Include formularele de login și logica de autentificare.
* **Create\_project.js**: Formular și funcționalitate pentru adăugarea de noi proiecte. Utilizatorii pot completa detaliile proiectului și pot salva informațiile.

**4. Ghid de utilizare**

**Fluxuri principale de utilizare**

1. **Autentificare**
   * Utilizatorii se pot autentifica folosind credențialele lor pentru a accesa funcționalitățile aplicației. Formularul de autentificare este afișat la accesarea paginii principale.
2. **Crearea unui proiect**
   * Navighează la secțiunea "Create Project".
   * Completează detaliile proiectului, cum ar fi numele, descrierea și locația.
   * Salvează proiectul pentru a-l adăuga în lista de proiecte.
3. **Vizualizarea proiectelor**
   * Accesează secțiunea "Projects" pentru a vedea lista completă a proiectelor.
   * Selectează un proiect pentru a vizualiza detaliile acestuia și pentru a efectua acțiuni suplimentare, cum ar fi editarea sau ștergerea proiectului.
4. **Informații despre clădiri**
   * Selectează un proiect pentru a accesa detalii despre clădiri.
   * Navighează prin tab-urile de informații pentru a vedea calitatea materialelor, conținutul materialelor și alte informații relevante despre clădiri.

**Funcționalități detaliate**

* **Tabloul de bord**: Vizualizarea rapidă a tuturor proiectelor și a stării acestora. Utilizatorii pot vedea o privire de ansamblu asupra activităților curente și a progresului proiectelor.
* **Secțiunea de proiecte**: Gestionarea proiectelor existente și adăugarea de noi proiecte. Utilizatorii pot adăuga, edita și șterge proiecte din această secțiune.
* **Informații despre clădiri**: Acces la detalii specifice despre fiecare clădire, inclusiv materialele utilizate și scenariile de utilizare. Utilizatorii pot naviga prin diferitele tab-uri pentru a obține informații detaliate.

**5. Personalizare și extensibilitate**

**Modificarea stilurilor și temelor**

Pentru a personaliza aspectul aplicației, editează fișierele .scss. Aceste fișiere controlează stilurile pentru componente individuale și pentru aplicație în general. De exemplu, pentru a schimba culorile temei sau stilurile butoanelor, poți modifica variabilele și clasele CSS din aceste fișiere.

**Adăugarea de noi funcționalități**

Pentru a extinde funcționalitatea aplicației:

1. **Crearea de noi componente**: Creează noi fișiere de componentă în directorul src/. Fiecare componentă nouă ar trebui să aibă propriul fișier JS și, dacă este necesar, un fișier CSS pentru stiluri specifice.
2. **Actualizarea rutelor**: Modifică App.js pentru a include noile componente în sistemul de rute. Asigură-te că noile rute sunt corect definite și conectate la componentele relevante.
3. **Adăugarea stilurilor**: Adaugă stiluri pentru noile componente în fișierele .scss corespunzătoare. Asigură-te că noile stiluri sunt incluse și aplicate corect.

**6. Depanare și suport**

**Probleme comune și soluții**

* **Dependențe lipsă**: Asigură-te că ai rulat npm install pentru a instala toate modulele necesare.
* **Probleme de rulare**: Verifică dacă portul 3000/5000 este liber sau specifică un alt port în configurare.
* **Erori de compilare**: Asigură-te că folosești versiuni actualizate de Node.js și npm. Verifică mesajele de eroare pentru indicații specifice și rezolvă problemele menționate.

**Resurse suplimentare**

* [Documentația oficială React](https://react.dev/learn)
* [Documentația Node.js](https://nodejs.org/en/learn/getting-started/introduction-to-nodejs)
* [Documentația npm](https://docs.npmjs.com/)

## Create\_projects.js / Create\_project.module.scss:

 **Importuri**:

* React și useState din React pentru gestionarea stării.
* styles dintr-un modul CSS pentru stilizarea componentei.
* Diverse componente Material-UI (Radio, RadioGroup, FormControlLabel, FormControl, Button) pentru controale de formă și butoane.
* useNavigate din react-router-dom pentru navigare.

 **Definiția componentei**:

* SelectCity este o componentă funcțională care acceptă mai multe props, dintre care doar setWizardStep, setStep și wizardStep sunt explicit destructurate. Props-urile suplimentare pot fi transmise prin ...props.

 **Gestionarea stării**:

* selectedValue este o variabilă de stare inițializată cu "gothenburg" pentru a ține evidența orașului selectat.

 **Funcții de manipulare a evenimentelor**:

* handleChange actualizează starea selectedValue atunci când este selectat un buton radio.
* handleBack utilizează navigate pentru a redirecționa utilizatorul la ruta "/projects".
* handleNext utilizează navigate pentru a redirecționa utilizatorul la ruta "/select\_perim".

 **Structura JSX**:

* JSX-ul componentei include un titlu, o descriere și un formular cu butoane radio pentru selectarea unui oraș.
* Fiecare opțiune de oraș este reprezentată printr-o imagine și un FormControlLabel cu un buton radio.
* Două butoane ("Înapoi" și "Înainte") la partea de jos permit navigarea între pași.

**Componenta de scss asigura ca interfata este organizata, fiecare element fiind descris pentru o experienta mai usoara.**

## Dashboard.js / Dashboard.module.scss:

 **Importuri**:

* React și useState din React pentru gestionarea stării.
* styles dintr-un modul CSS pentru stilizare.
* Diverse componente Material-UI pentru a construi interfața utilizatorului.
* useNavigate din react-router-dom pentru navigare între pagini.

 **Definiția componentei**:

* Dashboard este o componentă funcțională care gestionează mai multe state pentru a stoca valorile introduse în formular.

 **Gestionarea stării**:

* Hook-uri useState pentru a gestiona starea câmpurilor de formular, vizibilitatea parolei și acceptarea termenilor și condițiilor.

 **Funcții de manipulare a evenimentelor**:

* handleSubmit previne comportamentul implicit al formularului și poate include logica pentru trimiterea datelor.
* handleClickShowPassword inversează starea vizibilității parolei.
* handleSwitchToLogin redirecționează utilizatorul la pagina de login utilizând navigate.

 **Structura JSX**:

* JSX-ul componentei include un titlu, o descriere și un formular cu câmpuri de text pentru introducerea datelor utilizatorului.
* Câmpurile de text sunt însoțite de iconițe și butoane pentru a îmbunătăți interacțiunea cu utilizatorul.
* Două butoane la partea de jos permit trimiterea formularului și redirecționarea către pagina de login.

## Forgot-password.js / Forgot-password.module.scss:

 **Importuri**: Componenta importă diverse funcționalități și componente din React, React Router și Material-UI pentru a construi interfața și a gestiona starea.

 **Starea componentei**:

* email: stochează adresa de email introdusă de utilizator.
* isEmailSent: indică dacă email-ul de resetare a parolei a fost trimis.
* isLoading: indică dacă procesul de trimitere a email-ului este în curs.

 **Navigare**: useNavigate este utilizat pentru a permite redirecționarea utilizatorului către alte pagini.

 **Funcția handleSubmit**:

* Previne comportamentul implicit al formularului (reîncărcarea paginii).
* Simulează un proces de încărcare prin setarea stării isLoading la true.
* După 2 secunde, simulează trimiterea unei cereri de resetare a parolei și setează starea isEmailSent la true și isLoading la false.

 **Funcția handleBack**: Redirecționează utilizatorul către pagina de login atunci când butonul "Back to Login" este apăsat.

 **Renderizare**:

* Dacă isEmailSent este false, se afișează formularul de resetare a parolei cu un câmp pentru introducerea adresei de email și un buton pentru trimiterea cererii.
* Dacă isEmailSent este true, se afișează un mesaj care informează utilizatorul că un email cu instrucțiuni de resetare a parolei a fost trimis și un buton pentru a se întoarce la pagina de login.

## Login.js / Login.module.scss:

 **Importuri**:

* Importă diverse funcționalități și componente din React, React Router și Material-UI pentru a construi interfața și a gestiona starea.
* Importă module CSS pentru stilizarea componentei.

 **Starea componentei**:

* username: stochează numele de utilizator sau email-ul introdus de utilizator.
* password: stochează parola introdusă de utilizator.
* showPassword: indică dacă parola este vizibilă sau ascunsă.
* isSignUp: indică dacă utilizatorul este în modul de înregistrare.
* isForgotPassword: indică dacă utilizatorul este în modul de resetare a parolei.

 **Navigare**: useNavigate este utilizat pentru a permite redirecționarea utilizatorului către alte pagini.

 **Funcția handleClickShowPassword**: Comută între vizualizarea și ascunderea parolei.

 **Funcția handleSubmit**:

* Previne comportamentul implicit al formularului (reîncărcarea paginii).
* Dacă utilizatorul este în modul de înregistrare, ar trebui să se adauge logica pentru înregistrare (comentată momentan).
* Verifică dacă numele de utilizator și parola sunt corecte (în acest caz, doar un exemplu simplu cu "1" și "2").
* Dacă sunt corecte, setează autentificarea în localStorage și redirecționează către pagina "Projects".
* Dacă sunt incorecte, afișează un mesaj de eroare.

 **Funcția handleSwitchToSignUp**:

* Comută la modul de înregistrare și setează autentificarea în localStorage pentru a simula autentificarea după înregistrare.
* Redirecționează către pagina "Dashboard".

 **Funcția handleForgotPassword**: Redirecționează utilizatorul către pagina de resetare a parolei.

 **Funcția handleSwitchToLogin**: Comută la modul de login.

 **Renderizare**:

* Afișează un titlu și o descriere.
* Afișează un formular cu câmpuri pentru introducerea numelui de utilizator/email-ului și parolei.
* Include un buton pentru vizualizarea/ascunderea parolei.
* Include un link pentru resetarea parolei și butoane pentru autentificare și înregistrare.

## Projects.js / Projects.module.scss:

 **Importuri**:

* Importă React și hooks precum useState pentru gestionarea stării și useNavigate pentru navigare.
* Importă componentele de la Material-UI pentru interfața utilizatorului, precum Button, IconButton, Menu, MenuItem, etc.
* Importă modulele CSS pentru stilizarea componentei.

 **Componenta ActionsDropdown**:

* Gestionează afișarea și ascunderea unui meniu dropdown pentru fiecare proiect.
* anchorEl stochează referința la elementul care declanșează meniul.
* handleClick setează referința anchorEl la elementul curent și deschide meniul.
* handleClose resetează anchorEl și închide meniul.
* Afișează un buton pentru dropdown și meniul cu opțiuni: Overview/Edit, Duplicate, Unpin, Rename, Delete.

 **Componenta ProjectItem**:

* Primește proprietăți (city, name, creationDate) pentru a afișa informațiile fiecărui proiect.
* getBackgroundImage returnează calea către imaginea de fundal în funcție de orașul specificat.
* Renderizează un card pentru fiecare proiect cu o imagine de fundal și informațiile proiectului, inclusiv dropdown-ul de acțiuni.

 **Componenta principală Projects**:

* Utilizează useNavigate pentru a permite navigarea către alte pagini.
* handleBack redirecționează utilizatorul către pagina de login.
* handleCreateNewProject redirecționează utilizatorul către pagina de creare a unui nou proiect.
* handleTutorial redirecționează utilizatorul către pagina de tutorial/manual de utilizare.

 **Renderizare**:

* Afișează un titlu și o descriere.
* Afișează o secțiune cu proiectele utilizatorului, fiecare proiect fiind un ProjectItem.
* Afișează informații suplimentare despre instrumentul CREATE și cum poate fi utilizat.
* Afișează butoane pentru navigare: Back, Create a new project și Tutorial/User manual.

## SelectPerimeter.js / SelectPerimeter.module.js:

 **Importuri**:

* Importă React și hooks precum useState pentru gestionarea stării și useNavigate pentru navigare.
* Importă componentele de la Material-UI pentru interfața utilizatorului, precum Button, Tooltip, IconButton, HelpOutlineIcon.
* Importă stilurile specifice pentru această componentă din SelectPerimeter.module.scss.
* Importă componenta WMSMap pentru afișarea hărții.

 **Starea componentei**:

* smallMapBounds este un array utilizat pentru a stoca limitele hărții selectate.

 **Navigare**:

* Utilizează useNavigate pentru a permite navigarea către alte pagini.
* handleBack redirecționează utilizatorul către pagina de creare a proiectului (/create-project).
* handleNext redirecționează utilizatorul către pagina buildingTabs (/buildingTabs).

 **Renderizarea componentei**:

* Afișează un titlu și o descriere folosind stilurile definite în SelectPerimeter.module.scss.
* Afișează informații generale despre datele inventarului și folosește un tooltip pentru a oferi detalii suplimentare. Tooltip-ul este asociat cu un IconButton care afișează o pictogramă de ajutor (HelpOutlineIcon).
* Include o hartă (WMSMap) care permite utilizatorului să selecteze un perimetru. Harta este redimensionată la o înălțime de 400px și utilizează funcția setSmallMapBounds pentru a actualiza limitele selectate.
* Afișează butoane pentru navigare (Back și Next) care declanșează funcțiile handleBack și handleNext.

 **Export**:

* Exportă componenta SelectPerimeter pentru a putea fi utilizată în alte părți ale aplicației.

## TutorialUserManual.js / TutorialUserManual.module.scss:

Work in progress!

## BuildingInfoTabs.js / BuildingInfoTabs.module.scss:

 **Importuri**:

* React, useState pentru gestionarea stării în React.
* Tabs, Tab de la Material-UI pentru a crea tab-uri.
* Stiluri specifice din BuildingInfoTabs.module.scss.
* Componentele BuildingInformation, MaterialContent, și MaterialQuality care vor fi afișate în tab-uri.

 **Starea Componentelor**:

* value (stare) reține indexul tab-ului activ curent. Inițializat cu 0 pentru a arăta primul tab.

 **Funcția handleChange**:

* Actualizează starea value când utilizatorul schimbă tab-urile, permițând comutarea între diferitele secțiuni.

 **Tab-uri**:

* Tabs se folosește pentru a crea un grup de tab-uri. Se setează value cu starea curentă și onChange cu funcția handleChange pentru a răspunde la schimbările tab-urilor.
* Tab creează fiecare tab individual. label specifică textul tab-ului, iar value definește indexul tab-ului.

 **Contenutul Tab-urilor**:

* buildingInfoTabs este un array care conține componentele BuildingInformation, MaterialContent, și MaterialQuality.
* În funcție de valoarea stării value, se afișează componenta corespunzătoare din array.

 **Renderizare**:

* Se afișează tab-urile în partea superioară folosind Tabs și Tab.
* Sub tab-uri, se afișează componenta asociată cu tab-ul activ folosind indexul din value.

## BuildingInformation.js / BuildingInformation.module.scss:

 **Importuri**:

* Importă React și hooks precum useState pentru gestionarea stării.
* Importă componentele de la Material-UI pentru interfața utilizatorului, precum Button, Select, MenuItem, Slider, Tooltip, IconButton.
* Importă stilurile specifice pentru această componentă din BuildingInformation.module.scss.
* Importă componenta WMSMap pentru afișarea hărții.
* Importă useNavigate pentru navigare între pagini.

 **Starea componentei**:

* value pentru intervalul anilor de construcție.
* buildingUse, buildingType, buildingStructure, buildingSize, ownershipType pentru diversele filtre aplicate clădirilor.
* demolitionYear pentru intervalul anilor de demolare.

 **Navigare**:

* Utilizează useNavigate pentru a permite navigarea către alte pagini.
* handleBack redirecționează utilizatorul către pagina de selectare a perimetrului (/select\_perim).

 **Metode de gestionare a schimbărilor**:

* handleBuildingUseChange, handleBuildingTypeChange, handleBuildingStructureChange, handleBuildingSizeChange, handleOwnershipTypeChange actualizează starea componentelor respective pe baza valorilor selectate de utilizator.
* handleConstructionYearChange și handleDemolitionYearChange actualizează intervalele anilor pentru construcție și demolare.
* handleApplyFilters colectează toate filtrele și le afişează în consolă (poți adăuga logica de trimis datele către un server sau pentru a actualiza harta).

 **Renderizarea componentei**:

* Afișează filtre pentru utilizatori, inclusiv utilizarea clădirii, tipul clădirii, structura clădirii, dimensiunea clădirii, tipul de proprietate, anul construcției și anul demolării.
* Utilizează Select și Slider pentru a permite utilizatorilor să selecteze și să ajusteze filtrele.
* Include butoane pentru resetarea și aplicarea filtrelor.
* Afișează harta (WMSMap) cu o înălțime de 550px.
* Afișează informații sumare despre numărul clădirilor selectate și suprafața brută totală.
* Include butoane pentru navigare (Back și Save and continue).

 **Export**:

* Exportă componenta BuildingInformation pentru a putea fi utilizată în alte părți ale aplicației.

## MaterialContent.js / MaterialContent.module.scss:

 **Importuri**:

* React, useState pentru gestionarea stării.
* Componentele și stilurile de la Material-UI, inclusiv Button, Select, MenuItem, Slider, Tooltip, IconButton, TableContainer, Table, TableHead, TableBody, TableRow, TableCell, Paper.
* WMSMap pentru a afișa o hartă.
* useNavigate din react-router-dom pentru navigare între pagini.

 **MockData**:

* MockData este un array care conține datele de test pentru tabel. Acesta este folosit pentru a popula tabelul cu informații despre diferite tipuri de materiale.

 **Componente**:

* **DataTable**:
  + Creează un tabel folosind Material-UI. Tabelul afișează datele din MockData într-un format tabelar, cu coloane pentru Skin, Structure, Space, și Total.
* **MaterialContent**:
  + Utilizează starea pentru a gestiona filtrele și afișează diferite secțiuni și componente.

 **Funcționalitate**:

* **Filtre**:
  + Include selectoare pentru Category, Layer, și Element care permit utilizatorului să filtreze datele. Aceste selectoare sunt momentan nefuncționale (folosesc funcții goale onChange).
* **Mapă**:
  + Include un component WMSMap care afișează o hartă cu înălțimea specificată.
* **Informații**:
  + Afișează informații precum Total mass și Total volume și include selectoare suplimentare pentru clasificarea materialelor.
* **Tabel de date**:
  + DataTable este utilizat pentru a afișa datele în format tabelar.
* **Buton de navigare**:
  + handleBack navighează înapoi la pagina anterioară (/select\_perim).

 **Stilizare**:

* Stilurile sunt importate din MaterialContent.module.scss și aplicate pentru a aranja și stiliza elementele componentei.

## MaterialQuality.js / MaterialQuality.module.scss:

 **Importuri**:

* React, useState pentru gestionarea stării.
* Componentele și stilurile de la Material-UI, inclusiv Button, Select, MenuItem, Slider, Tooltip, IconButton.
* WMSMap pentru a afișa o hartă.
* MaterialScenariosModal pentru a afișa un modal cu scenarii de material.
* useNavigate din react-router-dom pentru navigare între pagini.

 **Stare**:

* openScenarios: Starea pentru a controla dacă modalul cu scenarii de material este deschis sau închis.

 **Funcționalitate**:

* **handleBack**:
  + Funcția pentru a naviga înapoi la pagina anterioară (/select\_perim).

 **Markup**:

* **Filtre**:
  + Include selectoare pentru Category, Layer, și Element care permit utilizatorului să filtreze informațiile. Aceste selectoare sunt momentan nefuncționale (folosesc funcții goale onChange).
* **Mapă**:
  + Include un component WMSMap care afișează o hartă cu înălțimea specificată.
* **Informații**:
  + Afișează mesaje și butoane pentru a accesa un formular de calitate a materialelor și pentru a deschide modalul cu scenarii de material.
* **Modal**:
  + MaterialScenariosModal este utilizat pentru a afișa scenarii de material și este controlat de starea openScenarios.

 **Butoane**:

* **Material Quality Form**: Buton pentru a accesa un formular de calitate a materialelor (funcționalitate neimplementată în acest cod).
* **Chosen Material**: Buton pentru a deschide modalul cu scenarii de material. Când este apăsat, setează openScenarios la true.
* **Save and continue**: Buton pentru a salva și continua (funcționalitate neimplementată în acest cod).
* **Export results**: Buton pentru a exporta rezultatele (funcționalitate neimplementată în acest cod).
* **Back**: Buton pentru a naviga înapoi la pagina anterioară.

 **Stilizare**:

* Stilurile sunt importate din MaterialQuality.module.scss și aplicate pentru a aranja și stiliza elementele componentei.

## Map.js:

 **Importuri**:

* React, useState, useRef pentru gestionarea stării și referințelor.
* Componentele și modulele necesare de la react-leaflet și leaflet.
* proj4 pentru transformarea coordonatelor între diferite sisteme de referință.
* data dintr-un fișier JSON pentru a încărca datele GeoJSON.
* Stilurile necesare pentru Leaflet.

 **Transformarea coordonatelor**:

* proj4.defs definește un sistem de coordonate personalizat (EPSG:3948).
* convertToGPS transformă coordonatele din EPSG:3948 în coordonate GPS (EPSG:4326).

 **Limitarea dimensiunii dreptunghiului**:

* MAX\_RECTANGLE\_WIDTH și MAX\_RECTANGLE\_HEIGHT definesc dimensiunile maxime ale dreptunghiului care poate fi desenat pe hartă.
* Extinde comportamentul implicit al desenării dreptunghiurilor (L.Draw.Rectangle) pentru a respecta aceste limite.

 **Componenta Map**:

* **Stări și referințe**:
  + editableFG: Starea pentru a gestiona grupul de caracteristici editabile.
  + editRef: Referința la controlul de editare.
* **Funcții**:
  + onCreated: Se apelează atunci când un nou obiect este creat. Stabilește limitele hărții și se asigură că există un singur obiect în grupul de caracteristici editabile.
  + parsedGeoJson: Transformă datele GeoJSON folosind convertToGPS pentru a le adapta sistemului de coordonate GPS.
  + onFeatureGroupReady: Se apelează atunci când grupul de caracteristici este pregătit și setează starea editableFG.

 **Markup**:

* **MapContainer**:
  + Contine hartă centrata la coordonatele specificate cu zoom-ul definit.
* **TileLayer**:
  + Sursa hărții utilizată (OpenStreetMap în acest caz).
* **FeatureGroup**:
  + Grup pentru a adăuga și edita obiectele desenate.
* **EditControl**:
  + Controlul pentru a activa/desactiva desenarea și editarea obiectelor pe hartă.
* **GeoJSON**:
  + Afișează datele GeoJSON pe hartă.

## WMSMap.js:

 **Importuri**:

* React: Importă biblioteca React necesară pentru crearea componentelor.
* MapContainer, TileLayer, WMSTileLayer din react-leaflet: Aceste componente sunt utilizate pentru a construi și personaliza harta.
* leaflet/dist/leaflet.css: Include stilurile CSS necesare pentru Leaflet.

 **Componenta WMSMap**:

* **Props**:
  + height: Înălțimea hartii care va fi setată prin stilul inline al MapContainer.
* **Markup**:
  + MapContainer: Componenta principală care încadrează harta. Primește center, zoom și stilul height și width ca prop-uri.
    - center: Setează centrul inițial al hărții (în coordonate GPS).
    - zoom: Nivelul de zoom inițial.
    - style: Aplică stiluri CSS pentru a seta înălțimea și lățimea hărții.
  + TileLayer: Adaugă un strat de fundal pentru hartă utilizând TileLayer de la OpenStreetMap.
    - url: URL-ul pentru descărcarea imaginilor de fundal ale hărții.
    - attribution: Atribuția necesară pentru OpenStreetMap.
  + WMSTileLayer: Adaugă un strat WMS de la un server GeoServer.
    - url: URL-ul serverului WMS.
    - layers: Numele stratului de date pe serverul WMS.
    - format: Formatul imaginii pentru cererile WMS.
    - transparent: Setează transparența stratului WMS, utilă pentru suprapunerea acestuia peste alte straturi.