RdS
RICERCA DI SISTEMA

ENEA
AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE, L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

MINISTERO DELL'AMBIENTE E DELLA SICUREZZA ENERGETICA

Ricerca di Sistema elettrico

Sviluppo Smart Sim 2.0 (LA1.2)

G. Schisani

Nomos Consulting

RdS_PTR22-24_PR1.7_LA1.2_317

Sviluppo Smart SIM 2.0 (LA1.2)

Avvio CER: Perfezionamento di RECON v2.0, sviluppo di nuove funzionalità di Smart Sim e DHOMUS, implementazione dei servizi prototipali di Assisted Living (LA1.2)

G.Schisani (Nomos Consulting)

Dicembre 2024

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dell'Ambiente e della Sicurezza Energetica - ENEA Piano Triennale di Realizzazione 2022-2024

Obiettivo: Decarbonizzazione

Progetto: Tema di ricerca 1.7 – Tecnologie per la penetrazione efficiente del vettore elettrico negli usi finali

Linea di attività: *2.8*

Responsabile del Progetto: Claudia Meloni, ENEA

Responsabile del Work Package: Claudia Meloni, ENEA

Responsabile Linea di Attività: Matteo Caldera, ENEA

Mese inizio previsto: 18

Mese inizio effettivo: 18

Mese fine previsto: 36

Mese fine effettivo: 36

Il presente documento descrive le attività di ricerca svolte all'interno contratto dal titolo: "Sviluppo software per perfezionamento Smart Sim 2.0, rilascio in produzione"

Indice

Indice delle figure

1 Risultati attesi

Nell'attività in oggetto è prevista la realizzazione di una serie di task sulla piattaforma Smart SIM 2.0. Di seguito il dettaglio:

Smart Sim 2.0:

- Costruzione nuovo portale compatibile con altri prodotti software ENEA
- Integrazione nuovi dati di input per la creazione di una scheda
- Gestione nuovi dati di output con rielaborazione backing
- Studio di fattibilità versione semplificata: Easy SIM
- Dashboard amministratore
- Input dati ARERA
- Esportazione Report
- Studio di fattibilità della georeferenziazione Schede
- Studio di Fattibilità per l'Interoperabilità con RECON
- Studio di Fattibilità dell'Integrazione dell'Indice di Flessibilità

2 Risultati ottenuti

Sono stati completati con successo i seguenti task:

Smart SIM 2.0:

- Costruzione nuovo portale compatibile con altri prodotti software ENEA
- Integrazione nuovi dati di input
- Gestione nuovi dati di output con rielaborazione backend
- Studio di fattibilità versione semplificata: Easy SIM
- Dashboard amministratore
- Input dati ARERA
- Esportazione Report
- Studio di fattibilità della georeferenziazione Schede
- Studio di Fattibilità per l'Interoperabilità con RECON
- Studio di Fattibilità dell'Integrazione dell'Indice di Flessibilità

3 Prodotti attesi

Nuovo portale Smart SIM 2.0

Codice rilasciato nel repository ENEA

4 Prodotti sviluppati

Nuovo portale Smart SIM 2.0

Codice rilasciato nel repository ENEA

5 Analisi degli scostamenti su attività e risultati

N/A

6 Sintesi delle attività svolte

Per Smart SIM 2.0, sono state realizzate significative migliorie, partendo dall'aggiornamento del framework Laravel per una migliore gestione dei dati backend e l'integrazione di tecnologie come Python e MongoDB, che supportano operazioni avanzate su dati energetici. È stata rafforzata l'interfaccia utente grazie all'uso di jQuery, e implementati Highcharts e Leaflet per arricchire le funzionalità di visualizzazione grafica e mappatura geografica. In aggiunta, è stato creato un processo più raffinato di raccolta e analisi degli input degli utenti, che migliora la precisione e l'efficacia delle simulazioni energetiche.

Per la versione EasySIMdi Smart SIM, si è optato per l'integrazione di un modulo di calcolo Excel semplificato, gestito attraverso Laravel e automatizzato da Python, per fornire stime energetiche rapide ed efficienti. Le simulazioni di consumo energetico ora beneficiano di un'analisi statistica più profonda e di un sistema automatizzato che migliora l'interazione con i dati e ne aumenta l'accessibilità. Inoltre, è stata implementata una dashboard amministrativa avanzata per una gestione e monitoraggio efficace delle operazioni di simulazione.

7 Dettaglio delle attività svolte

7.1 SMART SIM

7.1.1 Implementazione nuovo portale e GUI

Smart SIM 2.0 rappresenta un'evoluzione significativa del precedente tool Smart SIM, progettato per offrire simulazioni dettagliate e personalizzate dei consumi energetici domestici. Questa versione introduce miglioramenti sostanziali nel processo di raccolta e analisi dei dati, garantendo agli utenti una maggiore flessibilità e precisione nelle simulazioni energetiche.

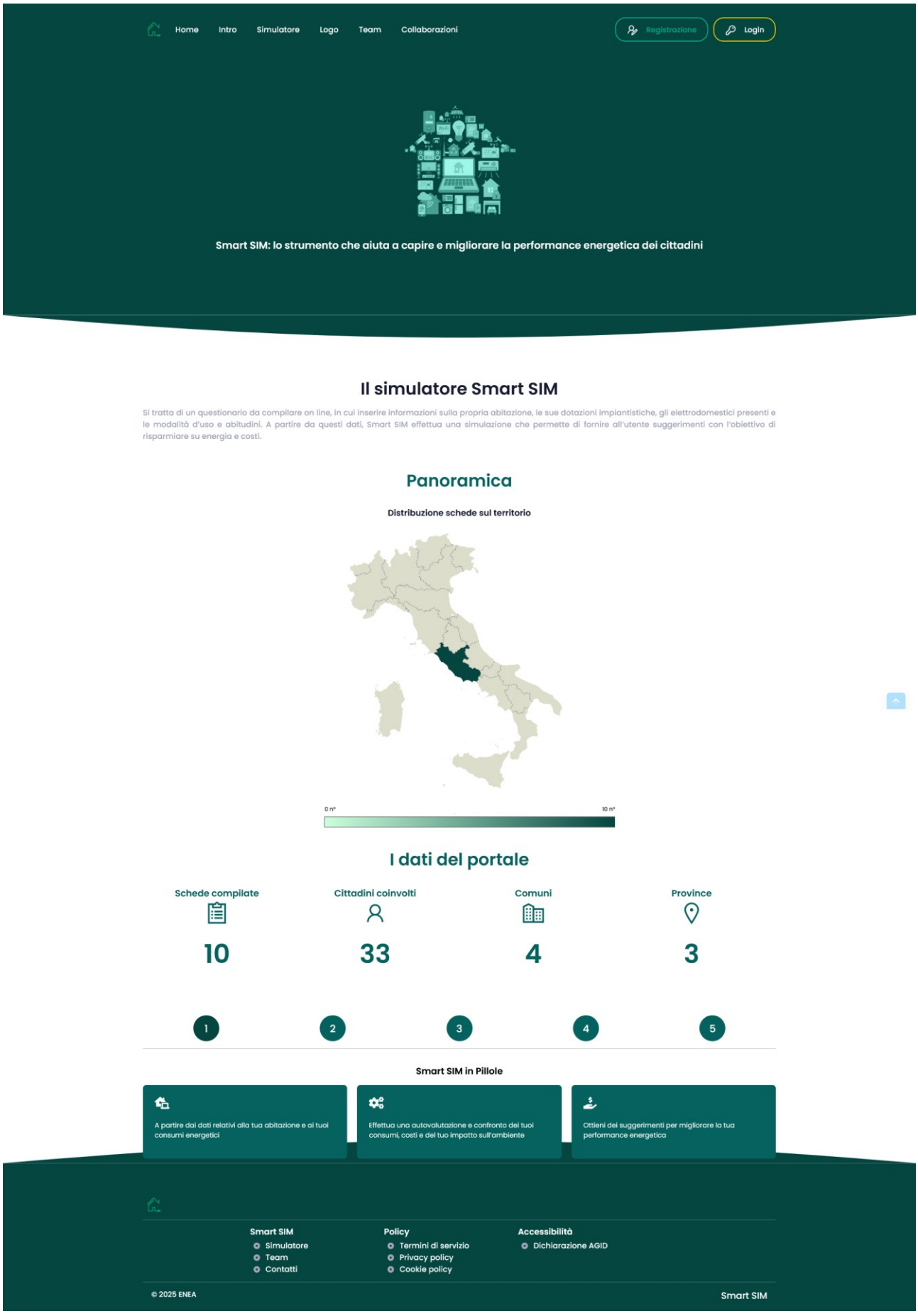

Figura 1- Home page nuovo portale

1. Panoramica Tecnologica

- **Laravel**: Il framework PHP Laravel è stato utilizzato come colonna vertebrale del backend, gestendo la logica di business, l'autenticazione degli utenti, e la comunicazione tra i diversi componenti del sistema.
- **JavaScript e Frameworks**: L'interfaccia utente si avvale di jQuery per dinamizzare la manipolazione dei DOM e migliorare l'esperienza utente. Inoltre, framework come Highcharts per la visualizzazione grafica dei dati e Leaflet per la mappatura geografica sono stati integrati per arricchire le funzionalità di visualizzazione.
- **Python**: Utilizzato per il parsing dei dati complessi e, nella prospettiva degli sviluppi EasySIM, per l'interazione con Google Sheets. Python supporta le operazioni di calcolo avanzato e la manipolazione dei dati energetici.
- **Database**: MySQL è impiegato per la gestione degli utenti e l'orchestrazione generale delle operazioni, mentre MongoDB è utilizzato per il salvataggio efficace e la gestione delle schede energetiche, con supporto per l'editing online.

2. Modifiche alla Schermata Home

- **Obiettivo**: Migliorare la comprensione dell'utente riguardo le novità di Smart SIM 2.0, illustrando le opzioni disponibili e guidando l'utente attraverso il nuovo workflow.
- **Implementazione**: Utilizzo di Laravel Blade templates per costruire una interfaccia utente accattivante. La dinamica dell'interfaccia è potenziata tramite l'uso di jQuery, che contribuisce a una navigazione fluida e intuitiva.

3. Integrazione degli Input

- **Nuovi Input**: Implementazione di campi per acquisire informazioni dettagliate come la, la superficie disponibile per gli impianti fotovoltaici e le opzioni di differenti tipi di generatori di calore.
- **Implementazione**: Sviluppo di form dinamici in Laravel e jQuery, che si adattano in base alle selezioni precedenti dell'utente. Questo approccio permette di esporre opzioni avanzate solo nel contesto appropriato, migliorando l'esperienza utente e la rilevanza dei dati raccolti.

7.1.2 Implementazioni specifiche

1. Dati di Input Dettagliati

- **Simulazione Generatori di Calore**: Possibilità di selezionare diverse fonti energetiche: gasolio, GPL, pellets, e biomassa.
- **Dettagli Impianti Fotovoltaici**: Abbiamo introdotto campi aggiuntivi per raccogliere dettagli sugli impianti fotovoltaici, quali le superfici disponibili.

2. Riallineamento dei Risultati: Per garantire che i risultati delle simulazioni siano il più possibile allineati ai dati reali forniti dagli utenti, come quelli derivanti dalle bollette

energetiche, abbiamo integrato una procedura di riallineamento. Questa procedura utilizza coefficienti di proporzionalità per aggiustare i risultati, creando un indice di attendibilità che riflette lo scostamento tra i consumi simulati e quelli effettivi.

Integrazione Tecnica

- **Form di Recupero Dati**: Abbiamo implementato una form dinamica in Laravel per la raccolta dei dati necessari alle simulazioni. Questa form utilizza condizioni logiche per esporre o nascondere campi in base alle selezioni precedenti, migliorando l'esperienza dell'utente e la precisione dei dati raccolti.
- **Controller Principale**: Il controller SmartsimSheetController gestisce l'incameramento delle schede. Questo controller è stato progettato per massimizzare l'efficienza nella manipolazione delle sessioni e nella gestione dei dati inviati attraverso la form.

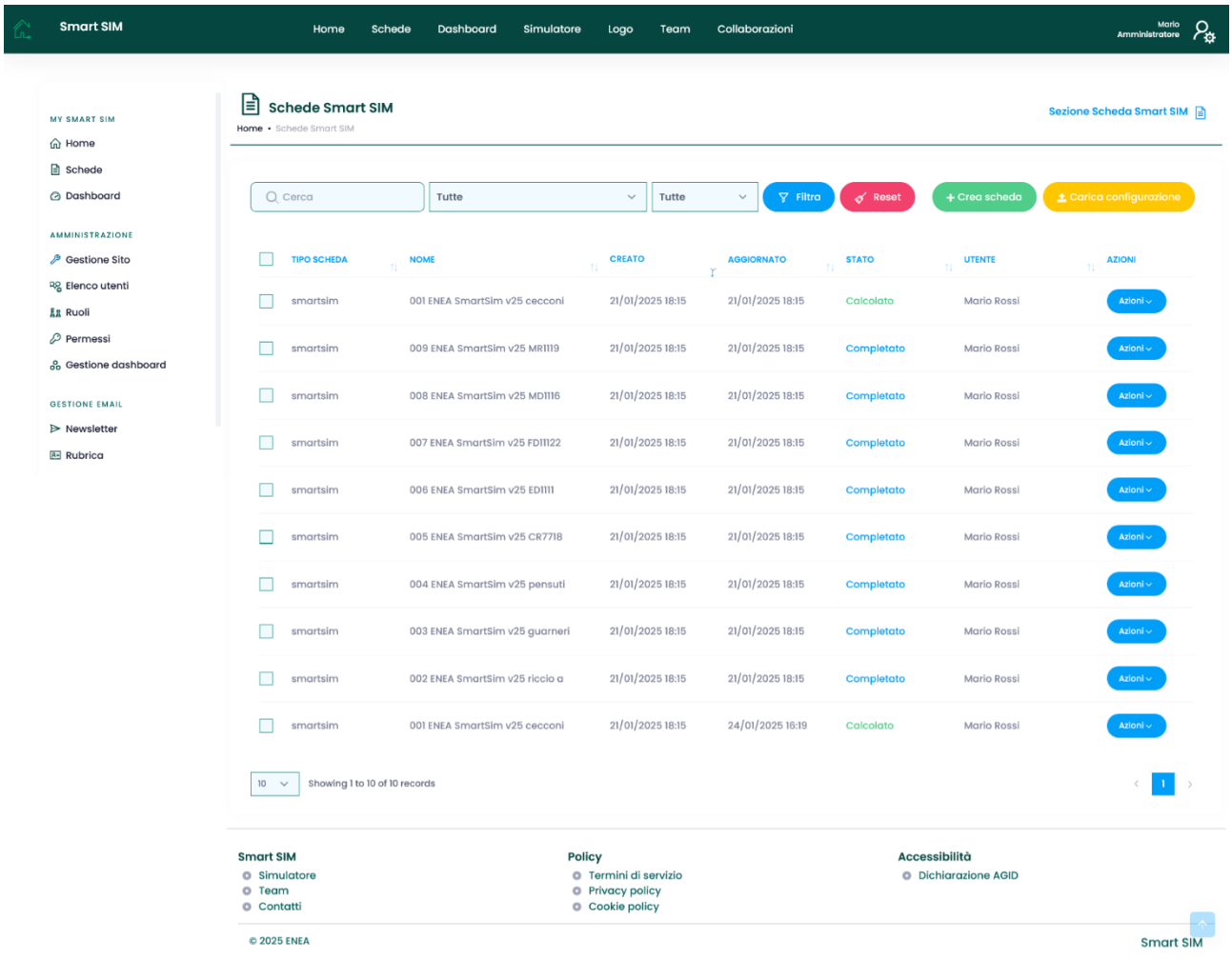

Figura 2- interfaccia per gestione schede

Importazione Schede Legacy

L'importazione delle schede legacy nel sistema Smart SIM 2.0 è una componente critica per assicurare la continuità operativa e l'integrità dei dati storici durante la transizione alla nuova piattaforma. Questa operazione è stata resa possibile attraverso lo sviluppo di un command specifico in Laravel, designato a gestire l'intero processo di migrazione dei dati. Il comando,

denominato ProcessSmartsimCSV, è stato implementato per trattare vari formati di dati e garantire che le informazioni siano correttamente trasferite nel nuovo sistema, mantenendo la consistenza e l'accessibilità dei dati.

Componenti Tecnici dell'Importazione

1. **Parsing del CSV**: Il primo passo nell'importazione delle schede legacy consiste nel parsing dei file CSV. Il command utilizza la libreria PHP per la manipolazione dei file CSV per leggere e separare i dati in base alle virgole. Durante questa fase, il sistema esegue controlli di validità sui dati per assicurarsi che rispettino i formati attesi e siano liberi da errori comuni come campi mancanti o formati di data non validi.
2. **Mappatura dei Dati**: Una volta letti, i dati vengono mappati alle strutture del database di Smart SIM 2.0. Questo processo richiede una trasformazione dei campi per allinearli ai nuovi schemi del database, utilizzando sia MySQL che MongoDB. Ad esempio, campi relativi a informazioni sull'utente o sui consumi energetici vengono trasformati e inseriti nelle nuove tabelle e documenti con relazioni appropriate.
3. **Gestione delle Incongruenze**: Nel caso in cui i dati legacy non siano direttamente compatibili con i nuovi modelli di dati, il sistema impiega una serie di regole e algoritmi di fallback per adattare o convertire i dati in un formato utilizzabile. Ciò include la conversione delle unità di misura, l'aggiustamento delle date e la trasformazione dei formati numerici.
4. **Testing e Verifica**: Dopo l'importazione dei dati, il sistema esegue una serie di test per verificare l'integrità dei dati importati. Questo include controlli per assicurarsi che tutte le schede siano state trasferite correttamente e che i dati nei campi chiave siano accurati e completi. Questi test sono essenziali per prevenire la perdita di dati o errori che potrebbero influenzare le simulazioni future.
5. **Interfaccia Utente per la Gestione delle Importazioni**: Per facilitare il monitoraggio e la gestione del processo di importazione, è stata implementata un'interfaccia utente che permette agli amministratori di avviare l'importazione, monitorare il suo stato e visualizzare i log degli eventi. Questa interfaccia è costruita con Laravel Blade e offre feedback in tempo reale sull'operazione di importazione, compresi eventuali errori o avvisi.

Sicurezza e Performance

La sicurezza e la performance sono state considerazioni fondamentali durante lo sviluppo del comando di importazione. Tutti i dati in transito e in riposo sono criptati, e sono state implementate misure per garantire che l'importazione non comprometta la performance del sistema esistente. Inoltre, il processo è stato ottimizzato per gestire grandi volumi di dati senza degradare le prestazioni dell'applicazione.

In sintesi, l'importazione delle schede legacy in Smart SIM 2.0 è stata attentamente progettata per garantire una transizione fluida e sicura dai sistemi precedenti, sfruttando tecnologie avanzate per migliorare la gestione dei dati e preparare il sistema per future integrazioni e scalabilità.

Implementazione dei Moduli

- **Highchart e Leaflet**: L'uso di Highcharts per la visualizzazione dei risultati delle simulazioni e di Leaflet per la mappatura geografica dei dati aggiunge un ulteriore livello di interattività e comprensione per l'utente. Questi strumenti sono integrati per fornire grafici dettagliati e mappe interattive che aiutano gli utenti a visualizzare i risultati in modi più intuitivi e utili.

Queste implementazioni non solo migliorano la precisione e la personalizzazione delle simulazioni in Smart SIM ma introducono anche una gestione dati più flessibile e adattiva, sfruttando le tecnologie moderne per superare le limitazioni dei sistemi precedenti.

7.1.3 Studio di fattibilità di una versione semplificata del motore di calcolo Easy SIM

Obiettivo

E' stato condotto uno studio di fattibilità per l'integrazione nel tool esistente di una versione semplificata del motore di calcolo, denominata Easy SIM, concepita per offrire una versione semplificata del tool di simulazione energetica Smart SIM, basata su Excel che sarà fornito dall'Università Roma1,

Tecnologie e Librerie da utilizzare

- **Excel**: Il motore di calcolo semplificato di EasySIM utilizza Excel per eseguire simulazioni basate su un modello energetico sviluppato.
- **Python**: Python è impiegato per automatizzare la lettura e l'analisi dei dati da Excel e per interfacciarsi con il database MySQL.
- **MongoDB**: Utilizzato come database NoSQL per registrare l'output delle simulazioni di EasySIM.
- **Laravel**: Framework utilizzato per integrare il motore di calcolo di EasySIM nell'applicazione web e per gestire l'interazione con MongoDB.

Metodologia di Sviluppo

1. **Integrazione del Modulo di Calcolo Excel**:
    - **Adattamento del Modulo Excel**: Il modulo di calcolo fornito dovrà essere adattato per operare efficacemente all'interno dell'ambiente web. Ciò include la conversione di alcune logiche di calcolo in Python e l'uso di librerie che permettono l'esecuzione di codice VBA direttamente dal server web.
    - **Automazione Python**: Script Python dovranno essere utilizzati per automatizzare la lettura e l'analisi dei dati dal modulo Excel. Questi script elaborano i dati inseriti dall'utente e calcolano le stime energetiche.
2. **Sviluppo di Analisi Statistiche**:
    - **Elaborazione dei Dati**: Python esegue analisi statistiche sui dati raccolti nel database MongoDB. Queste analisi aiutano a identificare pattern e tendenze che possono essere utilizzati per affinare le stime dei consumi energetici.

- **Integrazione delle Analisi nel Calcolo**: I risultati delle analisi statistiche verranno integrati nel processo di calcolo di EasySIM per migliorare l'accuratezza delle stime.

3. **Registrazione dei Risultati su MongoDB**:
    - **Struttura dei Dati**: Si definiscono e configurano le strutture di dati appropriate in MongoDB per memorizzare efficacemente i risultati delle simulazioni di EasySIM.
    - **Persistenza dei Dati**: Funzionalità CRUD (Create, Read, Update, Delete) saranno implementate in Laravel per gestire i dati di output di EasySIM su MongoDB, garantendo che i risultati siano facilmente accessibili e gestibili tramite l'interfaccia web.
4. **Integrazione nell'Applicazione Web**:
    - **Interfaccia Utente**: Modifica dell'interfaccia utente in Laravel per le modifiche conseguenti l'adozione del modulo EasySIM.
    - **Testing e Validazione**: Test approfonditi saranno condotti per assicurare che il modulo di calcolo di EasySIM funzioni correttamente all'interno dell'ambiente web e che i dati siano registrati correttamente in MongoDB.

Processo Dettagliato di Integrazione e Automazione in EasySIM

Il processo dettagliato di integrazione e automazione in EasySIM utilizza una serie di tecnologie e script per facilitare l'interazione tra i vari componenti del sistema, garantendo l'accuratezza e l'efficienza delle simulazioni energetiche. Ecco una descrizione più approfondita del flusso di lavoro, basata sull'analisi del file Python process_sheet.py che gestisce l'interazione tra il database MySQL, Google Sheets, e MongoDB.

1. **Estrazione dei Dati da MySQL**:
    - Il processo inizia con l'estrazione dei dati da una tabella MySQL. Utilizzando la funzione fetch_sheet_by_id, il sistema recupera le informazioni specifiche necessarie per la simulazione basate sull'ID della scheda fornito dall'utente. Questi dati comprendono parametri di configurazione e input utente che influenzano i risultati della simulazione.
2. **Caricamento e Gestione dei Dati su Google Sheets**:
    - Dopo l'estrazione, i dati vengono preparati per essere inseriti in un Google Sheet. Utilizzando le credenziali autenticate tramite load_credentials, il sistema accede a Google Sheets e cerca il file specifico tramite search_file_by_name.
    - Una volta identificato il foglio corretto, viene creato un clone di questo per mantenere l'integrità del template originale. Questa operazione è gestita dalla funzione clone_google_sheet.

- Successivamente, i dati vengono inseriti nel nuovo Google Sheet clonato utilizzando la funzione update_google_sheet_inputs, che si assicura che tutti gli input siano correttamente posizionati per permettere il calcolo delle simulazioni. Questo passaggio è fondamentale per garantire che il modello di calcolo riceva i dati corretti in formato e struttura adeguati.

3. **Elaborazione dei Calcoli su Google Sheets**:
    - Con i dati correttamente posizionati nel foglio clonato, il sistema invoca il calcolo automatizzato definito nel Google Sheet. Questo include la simulazione dei consumi energetici basati sulle configurazioni e gli input inseriti.
4. **Lettura dei Risultati di Output da Google Sheets**:
    - Una volta completati i calcoli, il sistema legge i risultati dal Google Sheet utilizzando la funzione read_google_sheet_outputs. Questi risultati sono estratti specificamente dalla colonna designata che contiene gli output della simulazione.
5. **Registrazione dei Risultati in MongoDB**:
    - Gli output raccolti vengono poi trasferiti a MongoDB, dove sono memorizzati per ulteriori analisi o per essere visualizzati agli utenti. Questa fase è gestita dalla funzione update_sheet_record, che aggiorna i record sia in MySQL che in MongoDB con i nuovi dati calcolati e l'identificativo del foglio di calcolo utilizzato.
6. **Gestione delle Traduzioni e Configurazioni**:
    - Il sistema utilizza configurazioni e traduzioni per gestire vari aspetti della localizzazione e della personalizzazione delle simulazioni. Queste configurazioni sono caricate e gestite nel corso del processo per assicurare che ogni aspetto della simulazione sia correttamente indirizzato in base alle preferenze dell'utente e alle impostazioni del sistema.

Questo flusso di lavoro dettagliato non solo garantirà l'integrazione fluida tra diverse tecnologie ma ottimizzerà anche l'efficienza del processo di simulazione in EasySIM, riducendo significativamente la possibilità di errori e migliorando l'esperienza dell'utente finale.

7.1.4 Dashboard Amministratore

La dashboard amministratore di Smart SIM è progettata per fornire una panoramica efficace e interattiva delle metriche chiave raccolte e aggregate da MongoDB predisposta per l'utente amministratore del sistema, basandosi strettamente sul codice fornito nei file DashboardController.php, WidgetBuilder.php, dashboard.php, e init.js. Ecco un'analisi dettagliata di come questi componenti interagiscono per costruire la dashboard.

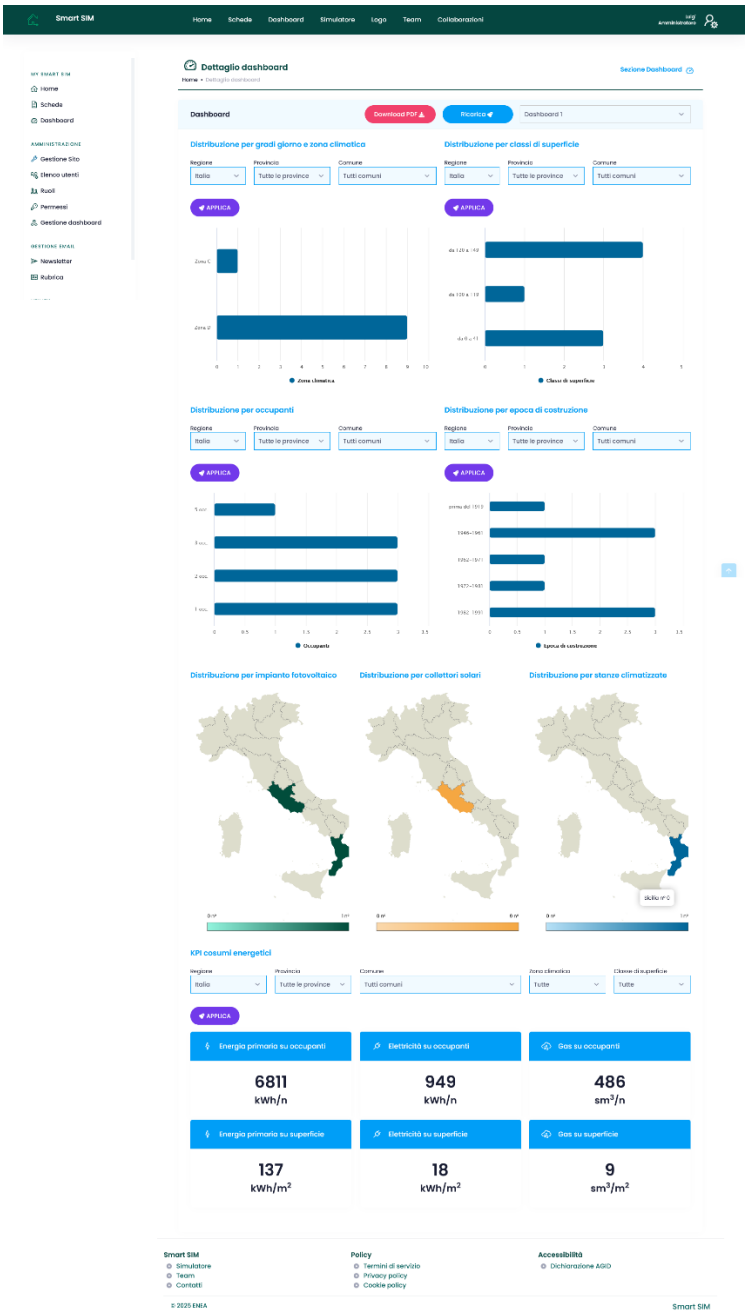

Figura 3: Dashboard amministratore

Backend - Logica di Estrazione e Gestione dei Dati:

- **DashboardController.php**: Questo controller in Laravel gestisce la logica di backend per recuperare i dati da MongoDB attraverso query complesse. Utilizza metodi specifici per estrarre dati aggregati basati su diversi criteri, come la distribuzione geografica delle schede e l'uso energetico per numero di occupanti. Il controller prepara questi dati per essere visualizzati nel frontend, strutturandoli in un formato che può essere facilmente interpretato e visualizzato dai widget della dashboard.
- **API RESTful**: Il DashboardController espone questi dati aggregati tramite un'API RESTful, consentendo al frontend di richiedere e ricevere dati in modo asincrono, il che è fondamentale per mantenere l'interfaccia utente reattiva e aggiornata senza ricaricare completamente la pagina.

Frontend - Configurazione e Visualizzazione dei Widget:

- **WidgetBuilder.php**: Questo file svolge un ruolo cardine nel definire e costruire i widget che saranno visualizzati sulla dashboard. Utilizza una serie di funzioni per generare dinamicamente componenti UI come grafici e mappe, sfruttando librerie JavaScript come Chart.js o D3.js. Il builder accetta parametri di configurazione che definiscono il tipo di grafico, la sorgente dei dati e altre opzioni visive, permettendo una personalizzazione flessibile in base alle esigenze del momento.
- **dashboard.php e init.js**: dashboard.php funge da punto di ingresso per il frontend della dashboard, caricando risorse essenziali e inizializzando la dashboard con configurazioni specifiche. init.js gestisce l'interattività della dashboard, inizializzando e aggiornando i widget in base agli input dell'utente e alle risposte dell'API. Questo script JavaScript è responsabile per la manipolazione del DOM per aggiornare dinamicamente i contenuti dei widget senza necessità di ricaricare la pagina.

Configurazione Modulare dei Widget e Interattività:

- **Modularità**: I widget sono configurati attraverso un file esterno (come indicato dalla struttura del codice, presumibilmente definito all'interno di WidgetBuilder.php o esternamente in un file di configurazione JSON o PHP), permettendo agli amministratori di modificare facilmente le metriche visualizzate o l'aspetto dei widget senza alterare il codice sorgente.
- **Dinamicità e Reattività**: Utilizzando AJAX chiamate dal file init.js, la dashboard può reagire in tempo reale agli input dell'utente, come selezioni di filtri o range di date, e aggiornare i widget di conseguenza. Questa interattività è essenziale per una dashboard amministrativa, poiché permette agli amministratori di esplorare attivamente i dati e ottenere insight immediati.

7.1.5 Input dati ARERA

Obiettivo

L'obiettivo è migliorare l'interfaccia di inserimento degli input in Smart SIM, rendendo il processo più intuitivo e informativo per gli utenti. Questo include l'opzione di caricare i dati di consumo energetico direttamente tramite file CSV, conformemente ai formati stabiliti da ARERA, oltre alla possibilità di inserimento manuale dei dati.

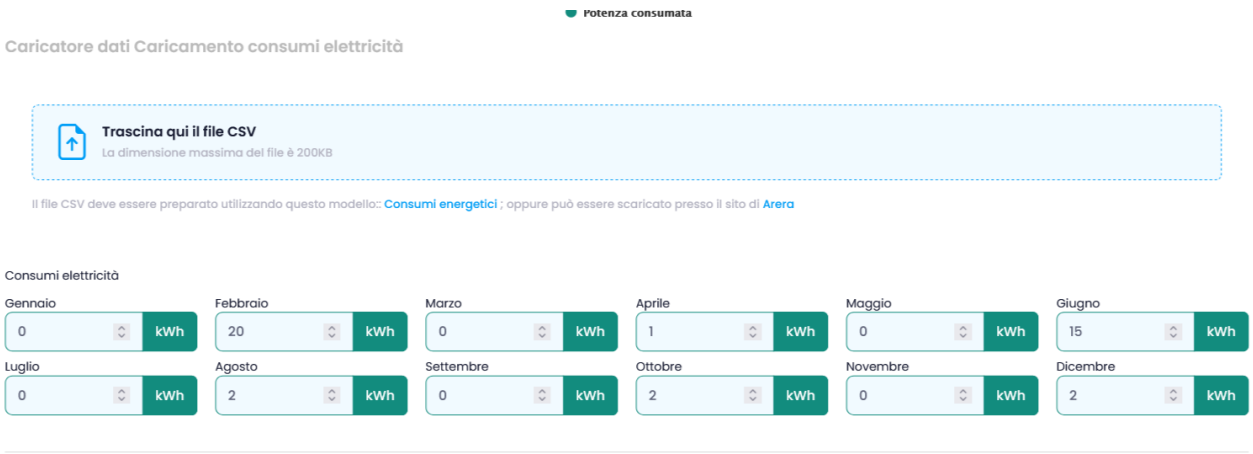

Figura 4_ interfaccia per inserimento csv

Tecnologie e Librerie Utilizzate

- **Laravel**: Gestione del backend e del processo di upload dei file.
- **JavaScript/jQuery**: Per l'interazione lato client durante il processo di inserimento dei dati.
- **Papa Parse**: Libreria JavaScript per il parsing di file CSV lato client.
- **Bootstrap File Input**: Plugin jQuery per migliorare l'interfaccia di input dei file.

Metodologia di Sviluppo

1. **Caricamento File CSV**:
    - **Interfaccia di Upload**: Implementazione di un'interfaccia utente intuitiva tramite Bootstrap File Input, che permette agli utenti di caricare file CSV direttamente nel sistema. Questa interfaccia offre feedback visivo immediato, come la progressione del caricamento e la conferma di upload completato.
    - **Parsing del CSV**: Uso della libreria Papa Parse per analizzare il contenuto del file CSV lato client prima dell'invio al server. Questo passaggio pre-valida i dati per formati e errori comuni, migliorando l'esperienza utente e riducendo il carico sul server.
    - **Elaborazione lato Server**: I dati del CSV sono inviati al server Laravel, dove sono ulteriormente elaborati per l'inserimento nel database. Le eccezioni e gli errori di formato sono gestiti in questa fase per assicurare che solo dati validi siano memorizzati.

2. **Inserimento Manuale dei Dati**:
    - **Form Dinamici**: Offerta continua della possibilità di inserimento manuale dei dati energetici attraverso form interattivi. jQuery è utilizzato per aggiungere o rimuovere dinamicamente campi del form in base alle selezioni dell'utente, migliorando l'interattività e la comprensibilità per l'utente.
    - **Validazione dei Dati**: Implementazione di validazioni sia lato client che server per garantire che tutti i dati inseriti manualmente rispettino i requisiti specificati, come i formati dei numeri, le date, e altro. Laravel gestisce le funzioni di validazione lato server.
3. **Integrazione con la Sezione "Costi e Consumi"**:
    - **Aggiornamento Automatico della Sezione**: Assicurazione che sia i dati caricati via CSV sia quelli inseriti manualmente aggiornino automaticamente la sezione "Costi e Consumi" di Smart SIM. Sono implementati trigger nel database o nel backend Laravel che rinfrescano i dati mostrati all'utente non appena vengono inseriti o modificati.

Integrazione Dettagliata Utilizzando il Codice JavaScript Fornito

Il file initUpsert.js ha una funzione nell'interazione lato client per il caricamento e la gestione dei dati energetici. Di seguito sono riportati alcuni punti chiave derivati dall'analisi del codice:

- **Dropzone Setup**: Utilizzo di Dropzone.js per creare un'area di trascinamento per i file CSV, facilitando l'upload diretto dei file nel sistema. La configurazione della Dropzone è personalizzata per accettare solo file CSV e limitare la dimensione massima del file.
- **Papa Parse Integration**: Il codice sfrutta Papa Parse direttamente all'interno dell'evento addedfile di Dropzone. Questo significa che ogni volta che un file è caricato tramite l'interfaccia, Papa Parse inizia immediatamente il parsing del file per verificarne il formato e la correttezza prima di procedere con l'upload definitivo al server.
- **Feedback Visivo**: Sono implementate notifiche e progress bar per fornire agli utenti un feedback immediato sullo stato del caricamento e dell'analisi dei file. Questo riduce la confusione e migliora l'esperienza generale dell'utente durante il processo di upload.
- **Elaborazione Errori**: Il codice gestisce attivamente gli errori di parsing e di upload, notificando gli utenti in caso di problemi con i file inviati, il che è essenziale per prevenire l'inserimento di dati errati o incompleti nel sistema.

Questo approccio tecnico non solo rende l'interfaccia più intuitiva ma assicura anche che i dati caricati siano precisi e pronti per l'ulteriore elaborazione, minimizzando così i potenziali errori nei calcoli di consumo energetico successivi.

7.1.6 Esportazione Report

Obiettivo L'obiettivo è fornire agli utenti di Smart SIM la capacità di esportare i dati delle loro simulazioni sia in formato PDF che JSON, offrendo una panoramica chiara ed esaustiva delle

caratteristiche dell'abitazione e dei risultati delle simulazioni, facilitando così la condivisione e l'analisi ulteriore dei dati.

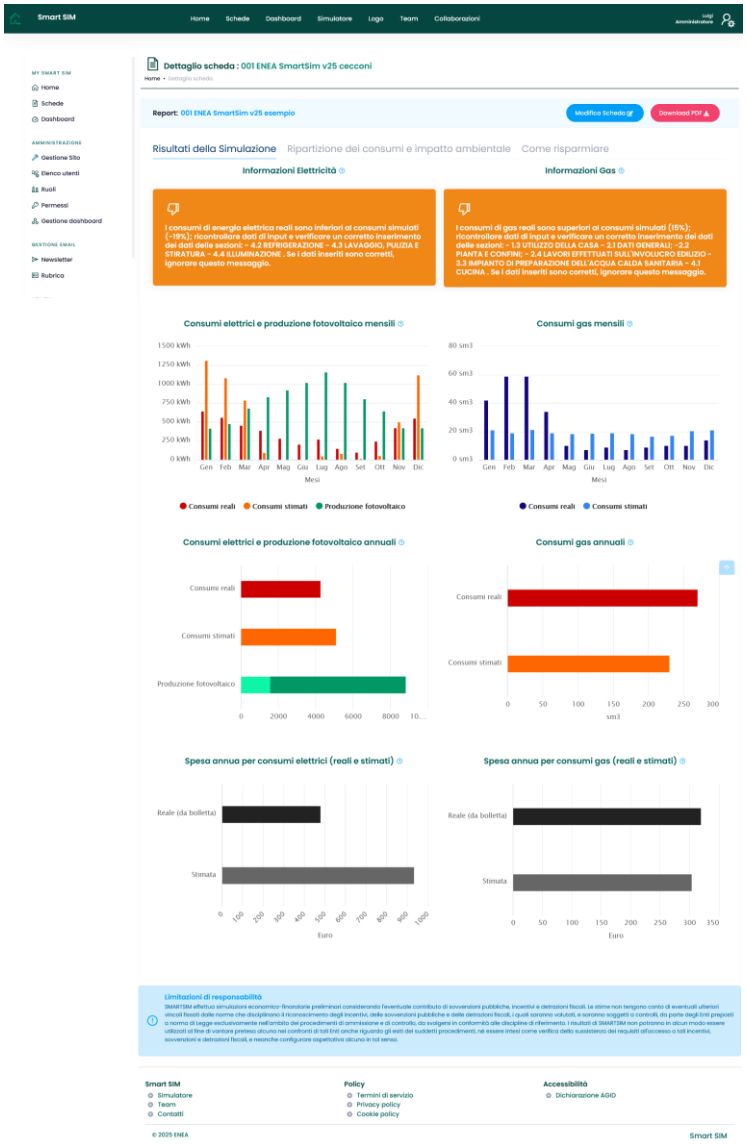

Figura 5: Report

Tecnologie e Librerie Utilizzate

- **Laravel**: Gestione del processo di esportazione dei dati e integrazione delle librerie necessarie.
- **dompdf/dompdf**: Libreria PHP per la generazione di file PDF.
- **Laravel Excel**: Libreria che supporta l'esportazione di dati in vari formati, incluso JSON.
- **JavaScript/jQuery**: Per gestire le richieste di esportazione lato client e migliorare l'interfaccia utente.

Metodologia di Sviluppo

1. **Definizione del Template di Report**:
    - **Progettazione del Template PDF**: Sviluppo di un layout concordato per il report PDF che include tutte le informazioni rilevanti dell'abitazione e i risultati della simulazione.
    - **Struttura del JSON**: Definizione della struttura dei dati JSON utilizzata per l'esportazione.
2. **Implementazione dell'Esportazione PDF**:
    - **Integrazione di dompdf**: Utilizzo della libreria dompdf per generare documenti PDF a partire dal template definito. Configurazione di dompdf nel progetto Laravel per gestire la creazione dei PDF basandosi sui dati delle simulazioni.
    - **Generazione dinamica dei PDF**: Sviluppo di una funzionalità in Laravel che raccolga i dati necessari dall'utente e dal risultato delle simulazioni, li formatti secondo il template PDF e generi il documento finale.
3. **Implementazione dell'Esportazione JSON**:
    - **Utilizzo di Laravel Excel**: Configurazione di Laravel Excel per esportare dati in formato JSON, sfruttando le sue capacità di manipolazione dei dati.
    - **Creazione di file JSON**: Implementazione di una funzione che compili i dati delle simulazioni in un oggetto JSON strutturato e li esporti come file scaricabile.
4. **Interfaccia Utente per la Richiesta di Esportazione**:
    - **Sviluppo UI**: Creazione di pulsanti o form nella GUI di Smart Sim che permettano agli utenti di selezionare il formato di esportazione desiderato e di iniziare il processo di esportazione.
    - **Gestione delle Richieste**: Utilizzo di JavaScript/jQuery per gestire le azioni dell'utente, inviare richieste asincrone al server e fornire feedback sull'avanzamento dell'operazione.
5. **Testing e Validazione**:
    - **Test dei Processi di Esportazione**: Verifica che i file PDF e JSON siano generati correttamente e contengano tutte le informazioni richieste. Assicurazione che i dati siano accurati e che il layout dei PDF rispetti il design concordato.
    - **Usabilità e Performance**: Test dell'interfaccia utente per assicurarsi che sia intuitiva e che i processi di esportazione non influenzino negativamente le performance della piattaforma.

Dettagli Tecnici dell'Implementazione Utilizzando il Codice Fornito

Il file generate_charts.py gioca un ruolo chiave nel processo di creazione dei report PDF e JSON. Il script Python è configurato per lavorare in sinergia con il backend Laravel per generare visualizzazioni di dati che verranno poi incorporate nei PDF. Ecco alcuni punti chiave del processo:

- **Elaborazione dei Dati**: Il codice Python utilizza variabili di ambiente per configurare la connessione a MongoDB, dove vengono memorizzati i dati delle simulazioni. Successivamente, recupera i dati specifici per l'ID fornito, che poi utilizza per generare grafici e mappe.
- **Generazione dei Grafici**: Utilizzo delle librerie matplotlib e plotly per creare diverse tipologie di grafici come linechart, piechart, barchart e heatmaps. Questi grafici sono personalizzati per adattarsi ai dati delle simulazioni specifiche di ogni utente.
- **Esportazione dei Grafici**: I grafici generati vengono salvati come immagini PNG o come dati serializzabili che possono essere incorporati direttamente nei report PDF o JSON.
- **Interazione con Laravel**: Il codice Python è progettato per essere chiamato da Laravel tramite una richiesta che passa parametri come l'ID della simulazione e la lingua per le traduzioni, permettendo così la dinamicità nel processo di generazione dei report.

Questo processo assicura che ogni report generato sia personalizzato in base ai dati specifici dell'utente e che le informazioni siano presentate in modo chiaro e professionale, sia nel formato PDF che JSON.

7.1.7 Georeferenziazione schede

Obiettivo E' stata studiata una modalità per implementare la georeferenziazione per le schede di Smart SIM, facilitando l'aggregazione e l'integrazione degli utenti nelle comunità energetiche. Questa funzionalità può consentire agli utenti di identificare potenziali membri per le comunità energetiche e di localizzare comunità esistenti a cui unirsi.

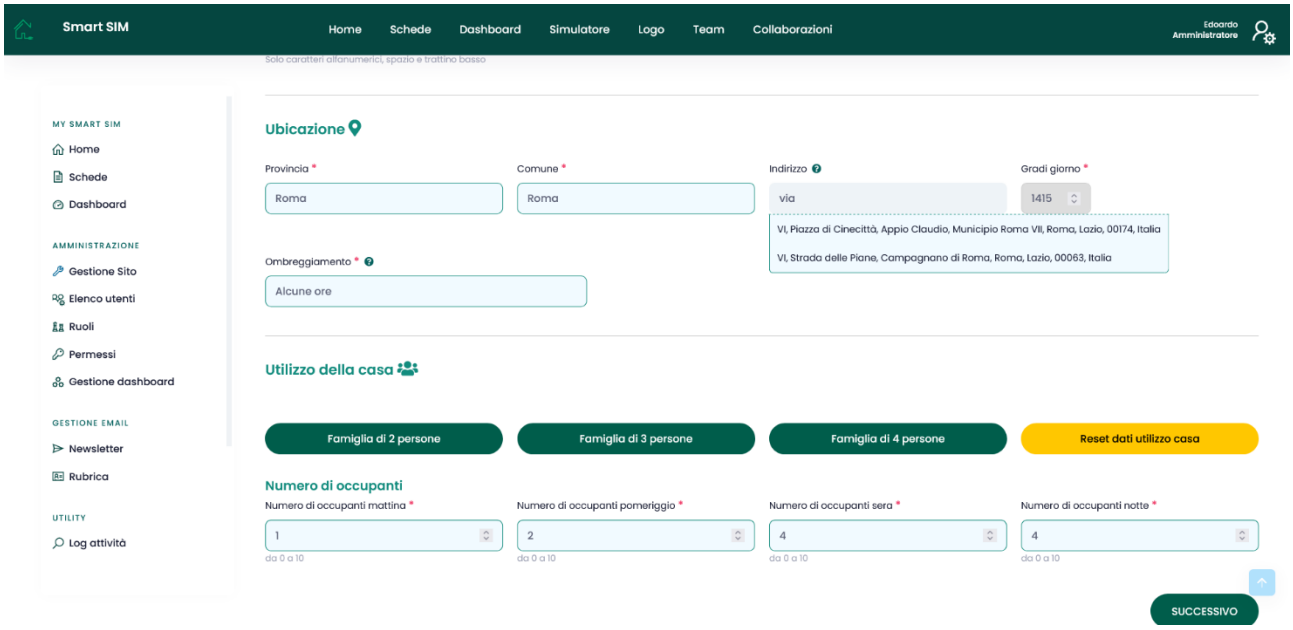

Figura 6 - Inserimnto dati geografici

Tecnologie e Librerie Utilizzate

- **Laravel**: Gestione del backend e integrazione con i servizi di georeferenziazione.
- **Google Maps API e OpenStreetMap**: Servizi di geocoding per convertire gli indirizzi in coordinate geografiche.
- **JavaScript/jQuery**: Per l'interfaccia utente e la gestione delle interazioni di georeferenziazione.

Metodologia di Sviluppo

1. **Raccolta dell'Indirizzo dell'Utente**:
    - **Form di Input**: Aggiunta di un campo di input nell'interfaccia utente di Smart Sim dove gli utenti possono inserire l'indirizzo della loro abitazione. Questo campo sarà integrato nelle form esistenti dove gli utenti inseriscono dati relativi al consumo energetico.
    - **Validazione dell'Indirizzo**: Implementazione di una validazione lato client usando JavaScript per assicurare che l'indirizzo inserito sia completo e corretto prima di inviarlo al server per il geocoding.
2. **Implementazione del Reverse Geocoding**:

- **Integrazione con API di Geocoding**: Utilizzo delle API di Google Maps o OpenStreetMap per convertire gli indirizzi in coordinate geografiche. Questo processo, gestito da un servizio Laravel, invia richieste alle API e gestisce le risposte.
- **Salvataggio delle Coordinate**: Le coordinate ottenute sono salvate nel database insieme ai dati della scheda dell'utente, permettendo loro di essere utilizzate per analisi future o visualizzazioni geografiche.

3. **Visualizzazione delle Comunità Energetiche**:
   - **Mappa Interattiva**: Implementazione di una mappa interattiva nel frontend di Smart SIM, utilizzando librerie come Leaflet.js per mostrare la posizione delle abitazioni degli utenti e delle comunità energetiche esistenti.
   - **Filtri e Ricerche sulla Mappa**: Fornitura di filtri e funzioni di ricerca sulla mappa per permettere agli utenti di trovare abitazioni o comunità basate su specifici criteri, come la vicinanza geografica o le caratteristiche energetiche.
4. **Testing e Verifiche**:
   - **Test di Accuratezza**: Verifica dell'accuratezza del processo di georeferenziazione per assicurare che le coordinate corrispondano agli indirizzi forniti dagli utenti.
   - **Test dell'Interfaccia Utente**: Verifica che la mappa interattiva e altri elementi dell'interfaccia siano facili da usare e funzionino correttamente su vari dispositivi.

Dettagli Tecnici dall'Integrazione del Codice JavaScript

Le funzioni Sheet.initGeoCodingNominatim() e Sheet.initGeoCodingGoogleMap() nel file initUpsert.js sono essenziali per l'implementazione del geocoding nel sistema Smart SIM. Ecco come queste funzioni contribuiscono al processo di georeferenziazione:

- **initGeoCodingNominatim()**: Questa funzione integra il servizio OpenStreetMap Nominatim per convertire gli indirizzi degli utenti in coordinate geografiche. È particolarmente utile per le implementazioni che richiedono una soluzione di geocoding gratuita e aperta.
- **initGeoCodingGoogleMap()**: Utilizza l'API di Google Maps per eseguire il geocoding degli indirizzi. Questa funzione è ideale per le implementazioni che necessitano di una maggiore precisione e di funzionalità aggiuntive offerte da Google, come il geocoding inverso e la ricerca di luoghi.

Queste funzioni saranno chiamate all'interno dell'interfaccia utente di Smart SIM quando un utente inserisce un indirizzo nel form dedicato. Una volta che l'indirizzo è validato e inviato, queste funzioni convertiranno l'indirizzo in coordinate geografiche che poi verranno visualizzate sulla mappa interattiva e salvate nel database. L'uso di jQuery assicura che queste richieste siano gestite in modo asincrono, migliorando l'esperienza dell'utente e minimizzando i tempi di attesa.

7.1.8 Studio fattibilità integrazione RECON

Obiettivo L'obiettivo è progettare un'interfaccia di interoperabilità tra Smart SIM e l'applicativo ENEA RECON per facilitare lo scambio di dati tra i due sistemi, mirando a ridurre la richiesta di dati ridondanti agli utenti e migliorare l'efficienza della gestione dati.

Tecnologie e Librerie Utilizzate

- **API RESTful**: Utilizzate per lo scambio di dati tra i due sistemi in maniera efficiente e sicura.
- **Laravel**: Framework scelto per l'implementazione delle API e la gestione della logica di integrazione.
- **GraphQL**: Considerata per permettere query più flessibili e ottimizzate rispetto a REST, riducendo la quantità di dati trasferiti.
- **MongoDB e SQL Server**: Utilizzo combinato per gestire i dati persistenti di entrambi i sistemi con sincronizzazione ottimale.

Metodologia di Sviluppo

1. **Analisi dei Requisiti di Dati**:
    - **Mappatura dei Dati**: Identificazione dei dati comuni tra Smart SIM e RECON attraverso una revisione approfondita delle strutture di database esistenti. Si valuteranno le entità dati e i loro attributi per determinare le migliori strategie di integrazione.
    - **Definizione dei Flussi di Dati**: Determinazione delle specifiche del flusso di dati, inclusi i dati da scambiare, la frequenza di scambio e i trigger per l'avvio della trasmissione.
2. **Progettazione dell'Architettura di Integrazione**:
    - **API RESTful con GraphQL**: Sviluppo di API RESTful arricchite da GraphQL per una gestione più agile delle richieste di dati, permettendo agli utenti di specificare esattamente quali dati ricevere, riducendo così la banda e il tempo di elaborazione.
    - **Minimizzazione della Ridondanza**: Implementazione di logiche che prevengano la duplicazione di dati, garantendo l'integrità e l'aggiornamento dei dati tra i sistemi senza sovrapposizioni non necessarie.

7.1.9 Studio fattibilità integrazione dell'Indice di Flessibilità

Obiettivo Esaminare la possibilità di integrare indici di flessibilità energetica nel sistema Smart SIM, utilizzando un modello di dati fornito da ENEA per arricchire le simulazioni energetiche e fornire analisi più dettagliate e personalizzate.

Tecnologie e Librerie Utilizzate

- **Modello Dati**: Studio approfondito del modello di dati fornito da ENEA per garantire una corretta integrazione.
- **Laravel**: Per gestire la logica di integrazione e aggiornamenti dell'interfaccia utente.
- **TensorFlow o PyTorch**: Valutazione dell'uso di algoritmi di apprendimento automatico per analizzare e prevedere i pattern di consumo energetico basati sugli indici di flessibilità.

Metodologia di Sviluppo

1. **Analisi dei Requisiti e dei Modelli di Dati**:
    - **Comprensione del Modello di Dati**: Analisi dettagliata del modello di dati degli indici di flessibilità per determinare come possono essere integrati efficacemente nel database di Smart SIM.
    - **Identificazione delle Intersezioni**: Mappatura delle intersezioni tra i dati esistenti di Smart SIM e gli indici di flessibilità per identificare le sinergie e le opportunità di miglioramento delle simulazioni energetiche.
2. **Progettazione del Processo di Integrazione**:
    - **Sviluppo del Flusso di Dati**: Progettazione di un flusso di dati efficiente tra gli indici di flessibilità e i moduli di Smart SIM, garantendo che i dati integrati siano facilmente accessibili per analisi e simulazioni.
    - **Interfaccia Utente**: Aggiornamento dell'interfaccia di Smart SIM per includere nuove metriche e visualizzazioni che riflettano l'integrazione degli indici di flessibilità, migliorando l'usabilità e l'applicabilità delle simulazioni.

8 Contributo delle eventuali consulenze alle attività sopra descritte

N/A

9 Pubblicazioni scientifiche

N/A

10 Eventi di disseminazione

N/A