



Ricerca di Sistema elettrico

Implementazione piattaforma raccolta di dati energetici residenziali e progettazione piattaforma raccolta dati energetici di un distretto universitario

Chiara Foglietta, Dario Masucci, Federica Pascucci, Stefano
Panzieri.

IMPLEMENTAZIONE PIATTAFORMA RACCOLTA DI DATI ENERGETICI RESIDENZIALI E PROGETTAZIONE PIATTAFORMA RACCOLTA DATI ENERGETICI DI UN DISTRETTO UNIVERSITARIO

Chiara Foglietta, Dario Masucci, Federica Pascucci, Stefano Panzieri (Università Roma Tre)

Aprile 2021

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico - ENEA

Piano Triennale di Realizzazione 2019-2021 - II annualità

Obiettivo: Tecnologie

Progetto: Tecnologie per la penetrazione efficiente del vettore elettrico negli usi finali

Work package: Local Energy District

Linea di attività: "Implementazione piattaforma raccolta di dati energetici residenziali e progettazione piattaforma raccolta dati energetici di un distretto universitario"

Responsabile del Progetto: Claudia Meloni, ENEA

Responsabile del Work package: Claudia Meloni, ENEA

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione *"Sviluppo di algoritmi, soluzioni tecnologiche ed informatiche per gli edifici esistenti e di nuova generazione"*

Responsabile scientifico ENEA: Fabio Moretti

Responsabile scientifico: Prof. Stefano Panzieri (Università Roma Tre)

Indice

SOMMARIO.....	5
1 INTRODUZIONE	6
2 DISCRIT DATABASE.....	7
2.1 IL DATABASE ENERGETICO DI ROMA TRE.....	7
2.2 LA SMART CITY PLATFORM	9
2.3 LA SOLUZIONE PROPOSTA.....	11
2.4 REQUISITI DI PROGETTAZIONE	15
3 AUDIT ENERGETICO.....	19
3.1 ELEMENTO STORAGE	20
3.2 ELEMENTO APPLICATION	23
3.3 SOLUZIONE REALIZZATA	27
3.3.1 <i>Percorso di benvenuto e accesso alla piattaforma</i>	27
3.3.2 <i>AuditScheda1 – Ubicazione ed utilizzo dell’abitazione</i>	28
3.3.3 <i>AuditScheda2 – Caratteristiche architettoniche</i>	29
3.3.4 <i>AuditScheda3 – Impianti</i>	30
3.3.5 <i>AuditScheda4 – Elettrodomestici</i>	30
3.3.6 <i>AuditScheda5 – Consumi e costi</i>	31
3.3.7 <i>AuditOutput – Visualizzazione risultati</i>	32
4 CONCLUSIONI.....	32
5 RIFERIMENTI BIBLIOGRAFICI	34

Elenco delle figure

Figura 1 - Diagramma EER database energetico Roma Tre	8
Figura 2 - Ecosistema Smart City Platform.....	11
Figura 3 - Schema struttura di memorizzazione	14
Figura 4 - Schema accessi ai database	15
Figura 5 - Database della piattaforma	21
Figura 6 - Schema degli accessi ai database	22
Figura 7 - Processo per il popolamento dei database.....	23
Figura 8 - Le funzioni del Web Service	24
Figura 9 - Elementi grafici di inserimento dati.....	25
Figura 10 - Layout delle pagine.....	25
Figura 11 - Script per l'implementazione del Web Service	27
Figura 12 - Screenshot pagina di benvenuto	28
Figura 13 - Screenshot pagina di accesso	28
Figura 14 - Screenshot prima pagina del questionario.....	29
Figura 15 - Screenshot seconda pagina del questionario	29
Figura 16 - Screenshot terza pagina del questionario	30
Figura 17 - Screenshot quarta pagina del questionario.....	31
Figura 18 - Screenshot quinta pagina del questionario	32
Figura 19 - Moke-up pagina di output del questionario.....	32
Figura 20 - Fasi logiche del DEM	33
Figura 21 - Ecosistema District Database.....	33
Figura 22 - Flusso logico District Database	34

Sommario

Le attività portate avanti durante questa annualità sono principalmente due. La prima riguarda la progettazione di un sistema informatico, chiamato *District Database*, per la raccolta dei consumi energetici che caratterizzano gli edifici afferenti al sistema di gestione energetica dell'Università Roma Tre. La seconda attività riguarda l'implementazione di un dimostrativo, chiamato Audit Energetico, derivante dalla LA6, relativa alla progettazione di una piattaforma informatica per l'acquisizione dei dati sui consumi energetici degli utenti residenziali. La raccolta di entrambe le tipologie di dati di consumo permettono la loro successiva elaborazione ad alto livello per fornire indicazioni e benchmark di riferimento.

Per quanto riguarda l'attività concernente il *District Database*, al termine di questa annualità si è conclusa la fase di progettazione dell'applicativo che dovrà gestire lo scambio di informazioni dei consumi energetici tra il sistema di memorizzazione attualmente attivo a Roma Tre e il sistema *Smart City Platform* (SCP) implementato in ENEA. Le specifiche riguardano:

- l'acquisizione dei dati di consumo degli edifici dal sistema energetico dell'Ateneo;
- l'organizzazione del messaggio in un formato adatto per l'inserimento nel sistema SCP;
- la creazione del messaggio secondo gli standard di comunicazione previsti dal sistema SCP.

Perciò gli output ottenuti riguardano le specifiche di progetto preliminari alla fase di implementazione delle metodologie di allaccio alla SCP, prevista nella LA8. Queste specifiche sono necessarie per una corretta implementazione del processo informatico che gestirà la comunicazione tra il sistema energetico di Roma Tre e il sistema SCP. In particolare, l'applicativo dovrà accedere al database riguardante i consumi energetici di Roma Tre, organizzarli secondo le modalità di immissione nel sistema SCP e convertirli in un file che abbia il formato JSON. Quindi dovrà inviare i dati al sistema SCP secondo gli standard adottati.

Al termine di questa annualità si è anche conclusa la fase di implementazione della parte di back-end e quella di front-end della piattaforma informatica associata all'Audit Energetico le cui specifiche risultano dall'output dell'annualità precedente. Le funzionalità realizzate riguardano:

- la corretta scrittura e lettura dei dati in una struttura di archiviazione;
- il processo di acquisizione delle informazioni inserite dall'utente e la visualizzazione dei risultati elaborati.

Quindi, partendo dalle specifiche di progetto derivanti dalla LA6 e che riguardano l'attività relativa alla piattaforma per la raccolta dati energetici residenziali, è stato implementato il processo informatico per l'acquisizione delle informazioni immesse dall'utente, l'elaborazione automatica delle informazioni e produzione dei risultati, e la memorizzazione di tutti i dati e la loro fruibilità da parte degli utenti.

1 Introduzione

La collaborazione in atto tra l'Università Roma Tre e la Divisione Energia del Centro Ricerche ENEA "La Casaccia" è incentrata sullo sviluppo di tecnologie innovative e soluzioni informatiche volte alla pianificazione strategica degli interventi di riqualificazione energetica degli edifici, siano questi residenziali o ad uso ufficio. Quindi, ad una prima fase di mappatura e raccolta standardizzata ed omogenea dei consumi, si vuole creare un catasto nazionale, strutturato in un DB interoperabile e in un tool di monitoraggio e valutazione delle prestazioni dei servizi. In particolare, la ricerca è stata suddivisa in due linee progettuali principali, ossia il District Database e l'Audit Energetico, i cui contributi verranno presentati, separatamente.

Il primo obiettivo considerato è quello di gestire l'interscambio in tempo reale dei dati riguardanti i dispositivi energivori che compongono le *essential utilities* che caratterizzano l'ambiente urbano. In particolare, si vuole creare una piattaforma informatica per l'acquisizione pseudo real time dei dati energetici degli edifici appartenenti al cluster di utilities di Roma Tre sottoposte a monitoraggio. È stato quindi progettato e realizzato, nei laboratori ENEA, un ecosistema tecnologico in grado di effettuare un'azione di diagnostica e di predizione dei potenziali di danno per un'infrastruttura e un'azione di monitoraggio dell'ambiente metropolitano. Il sistema informatico centrale per la gestione di questo ecosistema è la Smart City Platform (SCP) che, all'interno del più ampio progetto *Smart Urban Evolution*, si pone l'obiettivo su scala nazionale di avviare e supportare una riorganizzazione dei processi gestionali delle infrastrutture pubbliche energivore. In questo ambito progettuale l'attività su cui si concentra il contributo di Roma Tre è di definire ed implementare un sistema di acquisizione e mantenimento in memoria dei consumi energetici orari, aggregati per edificio, che caratterizzano il complesso universitario di Roma Tre. Si deve quindi fornire uno strumento software in grado di acquisire i dati di consumo degli edifici dal sistema energetico di Roma Tre, manipolare i dati per organizzarli in un formato che sia adatto per l'inserimento nel sistema SCP dell'ENEA, e creare il messaggio che verrà trasmesso utilizzando il protocollo di comunicazione *Message Queue Telemetry Transport* (MQTT) [1] attualmente adottato come standard ISO di messaggistica. Operativamente, deve essere stato progettato un applicativo informatico in grado di accedere al database che contiene i consumi energetici di Roma Tre, organizzare i dati secondo le modalità di immissione nel sistema SCP, e convertirli in un file che abbia il formato JSON. Il programma, quindi, dovrà ciclicamente avviare un *publisher* (server) MQTT che, dopo aver organizzato i messaggi in diversi topic, li potrà inviare al sistema SCP sul quale risiede un *subscriber* (client) MQTT [2]. Una descrizione più accurata della soluzione proposta e dei requisiti operativi e funzionali individuati è riportata nella sezione 2 del presente documento.

L'altra necessità emersa è quella di acquisire ed elaborare i dati energetici riguardanti le abitazioni private. Utilizzando un portale web, gli utenti, su base volontaria, potrebbero agevolmente inserire tutte le informazioni utili e necessarie per individuare e segnalare le zone geografiche o gli edifici con maggiore necessità di interventi di riqualificazione energetica. In questo modo sarebbe più semplice fornire un feedback mirato all'utente tramite statistiche e benchmark di riferimento, consigliando dove poter intervenire per migliorare il profilo energetico della sua abitazione. In questo modo si rendono omogenei i dati provenienti da tutti gli utenti e si consente una loro corretta elaborazione e valutazione statistica. Partendo dalle linee guida e dai requisiti di progettazione definiti nella precedente annualità, è stata realizzata, in versione beta, una *web application* che prevede due livelli distinti:

- la sezione di gestione, *back-end*, che elabora e supervisiona la produzione dei contenuti;
- la sezione applicativa, *front-end*, che gestisce il processo di immissione e visualizzazione dei contenuti da parte dell'utente.

Da un punto di vista funzionale gli elementi logici che compongono la piattaforma, ossia *Engine*, *Application*, *Orchestrator*, *Storage*, sono rimasti inalterati rispetto alla fase di progettazione, tuttavia dal punto di vista implementativo hanno subito alcune modifiche che ne migliorano la funzionalità, la

modularità e la portabilità. Una descrizione più accurata della soluzione implementata è riportata nella sezione 3 del presente elaborato.

2 District Database

L'obiettivo di questa attività è di realizzare un meccanismo per poter storicizzare i dati energetici degli edifici appartenenti all'Ateneo di Roma Tre all'interno di un database comune situato nei server ENEA del Centro Ricerche "La Casaccia". Per fare ciò devono essere utilizzati strumenti e linguaggi compliant con gli standard definiti dalla Smart City Platform. Analizzando il caso di studio si è deciso di adottare una strategia che comportasse un intervento minimamente invasivo sul preesistente sistema di gestione energetica operativo sul cluster di edifici di Roma Tre, e che mantenesse un grado di complessità di implementazione relativamente basso. Durante questa annualità si sono definiti l'approccio da utilizzare e i requisiti funzionali e operativi che devono caratterizzare il processo informatico che verrà realizzato.

La soluzione proposta consiste quindi nell'implementare un processo ciclico in grado di accedere al database energetico di Roma Tre, estrarre le informazioni di interesse, organizzare le informazioni secondo gli standard della Smart City Platform, e inviare il messaggio contenente le informazioni al database di ENEA. Per motivi di sicurezza e di disponibilità del dato, si è deciso di memorizzare le informazioni anche all'interno di un database ausiliario che risiede nello stesso *host* del processo. In questo modo si realizza il giusto compromesso tra invasività e complessità di implementazione. Per poter fare questo è necessario che il processo informatico, che chiameremo Data Exchange Manager (DEM), risieda all'interno di una macchina appartenente all'intranet di Roma Tre. È possibile riassumere le azioni che deve effettuare il DEM come segue:

1. Si attiva e acquisisce i dati di interesse dal database energetico di Roma Tre, utilizzando una connessione e un accesso sicuro;
2. Organizza i dati secondo il formato JSON e li memorizza all'interno del database ausiliario;
3. Incapsula in dati nel formato JSON in un pacchetto MQTT;
4. Invia il messaggio al database ENEA

Per realizzare un applicativo software capace di effettuare queste funzionalità si è deciso di procedere secondo step successivi. Per prima cosa è necessario definire e analizzare i dati energetici disponibili riguardanti gli edifici di Roma Tre per capire quali sono le grandezze misurate e tempi di campionamento utilizzati. Quindi, è opportuno allinearsi agli standard di sicurezza della rete di monitoraggio energetico preesistente e formalizzare le procedure di accesso al database in cui sono mantenuti i dati di interesse. Terminato lo studio di questi aspetti, si deve approfondire il funzionamento della Smart City Platform individuandone gli standard e le specifiche utilizzate. Ciò è cruciale per definire come popolare i messaggi che dovranno essere inviati al database ENEA e come questi verranno memorizzati nella struttura di archiviazione. Una volta terminata questa fase preliminare di analisi del contesto applicativo si può procedere a definire nel dettaglio il comportamento che dovrà assumere il processo informatico.

2.1 Il database energetico di Roma Tre

L'obiettivo è di analizzare dal punto di vista logico-strutturale il database che contiene i dati di consumo energetico. Si devono quindi definire le informazioni disponibili e le modalità con cui è possibile accedervi. A valle dell'incontro con l'Energy Manager che si occupa della gestione dei dati energetici di Roma Tre, si sono potute definire le informazioni messe a disposizione dal sistema di monitoraggio attivo sul cluster di edifici. Saranno, quindi, fruibili, per ogni edificio:

- Le informazioni anagrafiche dell'edificio, ossia il nome identificativo e le sue coordinate geografiche in termini di latitudine, longitudine e altitudine slm;
- Le informazioni anagrafiche delle grandezze monitorate, ossia il nome identificativo della grandezza corredata da unità di misura e tempo di campionamento utilizzato;
- Le Informazioni riguardanti il campione acquisito, ossia il valore registrato dal sensore e i riferimenti temporali (inizio e fine dell'acquisizione).

Gli edifici che appartengono al cluster, monitorati con tempo di campionamento di 15 minuti, sono 25 su 28 e di essi si può acquisire a livello aggregato:

- la potenza attiva P_{att} , ossia la capacità di un circuito di fornire un lavoro meccanico nell'unità di tempo t , è pari al prodotto scalare tra i vettori tensione V e corrente I .
- la potenza reattiva P_{rat} , ossia il prodotto algebrico della tensione V e della corrente I circolanti in un circuito preso in valore assoluto, rappresenta la potenza elettrica effettivamente circolante.
- Il fattore di potenza FP , in cui si scompone il vettore potenza su un piano cartesiano in termini di seno e coseno

$$FP = \frac{P_{att}}{P_{rea}} = \frac{V * I}{|V * I|} * \cos \varphi$$

dove φ è indica l'angolo di sfasamento, compreso tra il vettore tensione V e il vettore corrente I .

Il database ha una struttura classica SQL composta da tre tabelle, come riportato nella figura che segue.

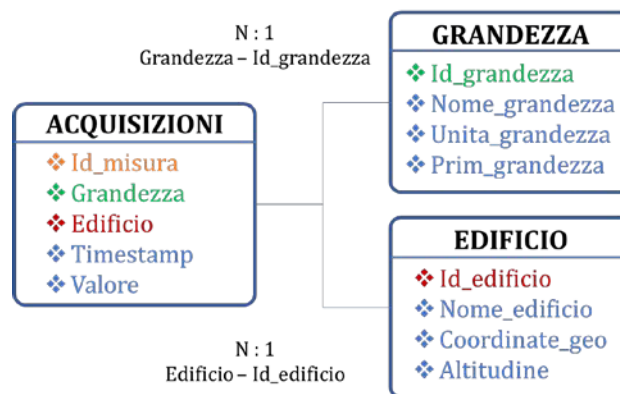


Figura 1 - Diagramma EER database energetico Roma Tre

I campi di interesse sono:

- *Edificio*, che contiene tutte le informazioni anagrafiche di ogni edificio, in particolare i campi che compongono questa tabella sono:
 - *Id_edificio*, codice univoco identificativo dell'edificio;
 - *Nome_edificio*, ossia il nome di riconoscimento dell'edificio;
 - *Coordinate_geo*, in cui si riportano la latitudine e la longitudine espresse in coordinate;
 - *Altitudine_geo*, ossia l'altitudine dell'edificio rispetto al livello del mare.
- *Grandezza*, contiene le informazioni riguardanti la grandezza misurata:
 - *Id_grandezza*, codice univoco identificativo della grandezza misurata;
 - *Nome_grandezza*, il nome di riconoscimento della grandezza misurata;

- *Unita_grandezza*, unità di misura della grandezza misurata;
- *Primitiva_grandezza*, tipo di dato primitivo con cui è memorizzata la grandezza misurata.
- *Acquisizioni*, contiene i dati acquisiti dal sistema di monitoraggio:
 - *Id_misura*, codice univoco identificativo dell'acquisizione;
 - *Grandezza*, codice identificativo della misura acquisita;
 - *Edificio*, codice identificativo dell'edificio;
 - *Timestamp*, riferimento temporale in cui è avvenuta l'acquisizione
 - *Valore*, valore acquisito.

Poiché l'host su cui si vuole implementare il programma informatico per la gestione del processo è fisicamente collocato all'interno dell'intranet di Roma Tre, l'accesso al database è consentito tramite l'utilizzo di username e password previa registrazione alla *white list* dei dispositivi considerati *trustable*.

2.2 La Smart City Platform

Per poter coordinare al meglio le funzionalità che deve svolgere il DEM è necessario analizzare la struttura informatica e logica che caratterizza la SCP implementata in ENEA. Lo studio della documentazione associata al rilascio [3], emerge che si tratta di una piattaforma orizzontale che aggrega e integra i dati di piattaforme verticali indipendenti tra loro e che si occupano del monitoraggio di uno specificato asset. La piattaforma, quindi, risulta non solo uno strumento per il monitoraggio dell'intero distretto, ma anche un broker per l'interscambio di dati e informazioni tra piattaforme verticali tra le quali si evidenzia un'interdipendenza. I componenti chiave relativi alla SCP, di interesse per questa attività sono riassunti di seguito.

Una *Solution verticale* corrisponde a una piattaforma locale, ossia ad un particolare contesto applicativo nella città o quartiere. Ad esempio, nel contesto in oggetto, la Smart Building Network, si può considerare Solution verticale di una Smart Building Platform orizzontale che permette di gestire il monitoraggio e la diagnostica su reti di edifici. Il modello di *Solution* descrive un insieme di caratteristiche comuni che permettono di individuare processi di gestione dei dati, che sono identici nei diversi contesti applicativi, per definire un approccio sistematico basato su una comunicazione interoperabile. I dati possono essere inviati e utilizzati a livello di quartiere o città, per avere una panoramica il più possibile completa sullo stesso tessuto urbano da diversi punti di vista. Questo modello di Solution è fondamentale per la definizione di un meccanismo di integrazione standard per la comunicazione interoperabile tra Solution e la Smart City Platform e che può essere condiviso e utilizzato per tutte le Solution dei diversi casi applicativi.

La *comunicazione interoperabile* tra sistemi è basata sullo scambio di dati eterogenei in ambito Smart City. Benché questi dati siano diversi tra loro sono comunque sempre caratterizzati da una collocazione spazio-temporale ben precisa e che li distingue dagli altri.

Un *Urban Dataset* (UD) è la rappresentazione del dato scambiato tra Solution della città e la Smart City Platform, secondo le specifiche definite. In particolare, un UD è caratterizzato da:

- Una descrizione semantica univoca e centralizzata, tramite l'Ontologia;
- Un modello dati astratto che definisce il contenuto in modo indipendente dalla sintassi;
- Un'implementazione sintattica in JSON o in XML. Sebbene XML sia uno strumento più potente e rigoroso per la rappresentazione formale dei dati, il JSON risulta più leggero.
- Un servizio di trasporto dati, l'*Urban Dataset Getaway* (UDG), basato su Web Service, REST e SOAP.

Le specifiche operative della SCP sono contenute nel documento pubblico Smart City Platform Specification (SCPS) per l'interoperabilità della comunicazione. In particolare, consta di cinque specifiche:

1. Comunicazione – standard utilizzabili per stabilire la comunicazione con l'UDG;
2. Ontologia – standard per la costruzione semantica del messaggio da scambiare;
3. Formato – standard per la costruzione sintattica del messaggio da scambiare;
4. Collaborazione – standard per la collaborazione tra UD e UDG;
5. Trasporto – standard di interfaccia per lo scambio dei dati con l'UDG;

Inoltre, secondo la documentazione, il processo di configurazione di una Smart City Platform si può riassumere come segue:

1. Creazione di un'istanza della SCP inserendo o modificando le informazioni presenti nella tabella `smart_city_administrator`;
2. L'utente Developer definisce l'utente Administrator;
3. Configurazione della Solution:
 - Individuazione della Solution;
 - Individuazione di almeno un UD;
 - Definizione della collaborazione tra la Solution e l'UD.
4. Abilitazione sulla Solution di un particolare UD, chiamato Resource;
5. Definizione di tre parametri:
 - `Resource_id` che esprime l'identificazione di un particolare Urban Dataset, prodotto da una particolare Solution, in una specifica Smart City Platform, in un preciso intervallo temporale;
 - `TimestampStart`, esprime l'istante di tempo in cui si vuole avviare la produzione del dataset e quindi la comunicazione con la SCP;
 - `TimestampEnd`, esprime l'istante di tempo in cui si vuole arrestare la produzione del dataset.
6. Avvio della fase di comunicazione interoperabile tramite l'UDG.

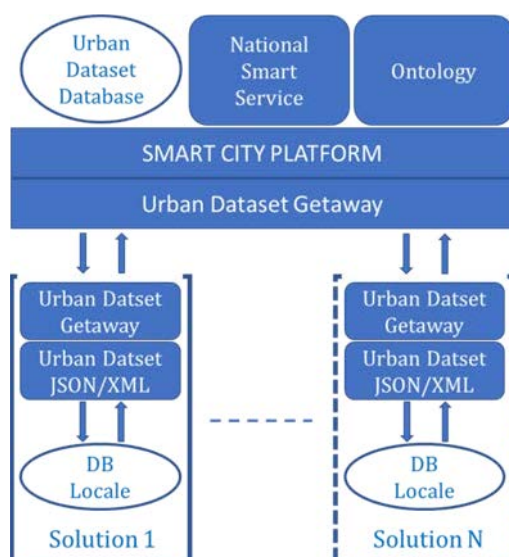


Figura 2 - Ecosistema Smart City Platform

Partendo dallo schema generale della piattaforma riportato sopra, la sequenza di azioni da intraprendere per produrre un Urban Dataset a partire da una Solution sono:

1. Esportare dei dati dal proprio database locale per la loro aggregazione nell'UD configurato;
2. Rappresentare l'UD secondo il formato JSON;
3. Utilizzare il Web Service UDG per inviare l'UD alla SCP;
4. Ricezione da parte della SCP dell'UD e immagazzinamento nella banca dati.

2.3 La soluzione proposta

Per questa attività, in base alla quantità e alla tipologia di dati che si devono inviare, si può prevedere una Solution verticale unica per tutti gli edifici di Roma Tre. Per realizzare un applicativo software capace di effettuare lo scambio di Urban Dataset tra il contesto applicativo di Roma Tre e la Smart City Platform si è deciso di procedere secondo step successivi:

1. Definire le informazioni disponibili – Individuare e analizzare i dati che vengono resi disponibili dal Web Service Energetico di Roma Tre in termini di grandezze misurate e tempi di campionamento utilizzati.
2. Definire la specifica dell'UB che si vuole implementare – Si ricerca tra gli UD esistenti in Ontologia per individuare un UD che risulta adeguato agli scopi prefissati. In alternativa si può definire una nuova specifica per l'UD.
3. Analizzare il Modello Dati Astratto (MDA) e della specifica individuata – L'MDA definisce il contenuto che devono avere i messaggi per essere scambiati tramite UD. Si deve quindi analizzare per stabilire quali proprietà specificare.
4. Definire la struttura di archiviazione – Progettare la struttura del database SQL, delle sue tabelle e del contenuto che queste devono avere in modo tale che le latenze dovute agli accessi al database in lettura e scrittura siano trascurabili.
5. Mapping tra Modello Dati e sistema – Definire come si popoleranno i messaggi creati a partire dal Modello Dati scelto e mappare ogni elemento del Modello con un campo della sorgente dati. Questa deve essere facilmente rintracciabile all'interno del sistema di archiviazione.

6. Progettare il processo informatico – Definire le funzionalità che dovranno essere svolte dal DEM che verrà implementato. Questo, ciclicamente, dovrà interrogare la struttura di archiviazione, creare gli UD e inviare il messaggio alla SCP.

Con lo scopo di rendere il più possibile conforme la struttura dell'UD agli obiettivi di monitoraggio prefissati, si è deciso di definire una nuova specifica prendendo come riferimento quella disponibile nella documentazione che più rispecchi la struttura che si vuole adottare. Per realizzare la nuova specifica ci si è quindi basati sul *Modello Dati Astratto* (MDA) e sull'UD *Building Electric Consumption* (BEC) che sono, evidentemente, conformi con i nostri obiettivi e sono stati già utilizzati con successo in contesti analoghi. L'MDA definisce il contenuto ed il significato dei messaggi che compongono l'UD. Il suo obiettivo è di facilitare la mappatura tra le implementazioni sintattiche utilizzate e supportate dalla SCP. In particolare, può essere suddiviso in tre parti:

- *Specification*: contiene le informazioni che descrivono l'UrbanDataset utilizzato, ad esempio il riferimento alla specifica a cui aderisce e le proprietà che lo compongono. Questo blocco è caratterizzato dalle seguenti informazioni:
 - *specificationReferences*, ossia l'insieme di riferimenti che consentono l'identificazione della specifica. Questi riferimenti sono la versione della specifica (*version*), il codice identificativo della specifica (*id*), il nome associato all'UB (*name*), l'*url* associato alla specifica.
 - *propertyDefinition*. Il modello prevede la possibilità di definire sia proprietà elementari, come ad esempio il consumo elettrico di una unità abitativa, sia proprietà aggregate, ovvero proprietà composte da più proprietà elementari. La definizione di una proprietà elementare deve essere data dalle seguenti informazioni: il nome della proprietà (*propertyName*), la descrizione della proprietà (*propertyDescription*), il tipo di dato primitivo con cui viene espresso il valore (*dataType*), l'unità di misura con cui è espresso il valore (*unitOfMeasure*).
- *Context*: fornisce le informazioni che contestualizzano i valori trasmessi. In particolare, il blocco prevede le seguenti informazioni:
 - *producer*, riferimento alla piattaforma che produce i dati tramite l'identificatore univoco del sistema (*id*), e l'identificativo dello schema di riferimento (*schemaID*).
 - *timeZone*, fuso orario della zona di interesse;
 - *timeStamp*, istante di generazione del particolare UD;
 - *coordinates*, coordinate WGS84 [4] del centro geometrico del sistema di monitoraggio, espresse tramite tre campi, ossia *latitude*, *longitude* e *height*.
- *Values*: è composto da una o più righe (1 riga = 1 gruppo di proprietà), ognuna contenente le seguenti informazioni:
 - *id*, l'identificatore della riga;
 - *description*, la descrizione testuale dei valori monitorati;
 - *timestamp*, data e ora di generazione dei dati riportati nella particolare riga (può essere omissa se coincide con il timestamp di generazione dell'UD);
 - *coordinates*, le coordinate dell'oggetto a cui sono riferiti i dati rilevati;
 - *period*, il periodo temporale di acquisizione dei dati
 - *property*, stabilisce come definire i valori delle proprietà introdotte, ossia il nome identificativo della proprietà (*name*) e il valore acquisito (*value*).

Osservando la struttura dell'MDA e ricordando le informazioni che saranno messe a disposizione dal sistema di monitoraggio energetico di Roma Tre, si è deciso di realizzare un modello di UD che sarà caratterizzato dai seguenti dati:

- **Specification**, che conterrà le informazioni riguardanti:
 - Il codice identificativo e il nome dell'UD;
 - Il codice identificativo e il nome dell'edificio da cui provengono le acquisizioni;
 - Il tipo e l'unità di misura della grandezza acquisita.
- **Context** che conterrà le informazioni riguardanti latitudine, longitudine e altitudine geografiche dell'edificio.
- **Value**: i riferimenti temporali di inizio e fine dell'acquisizione e il valore assunto dalla grandezza.

In base a queste valutazioni, si decide quindi di prevedere una struttura di archiviazione ausiliaria che risiederà a bordo del server messo a disposizione da Roma Tre e collocato negli ambienti dell'MCIP Lab. Questo database dovrà essere organizzato per mantenere in memoria tutte le informazioni scambiate ed entrerà in azione nell'eventualità che siano riscontrati dei problemi di comunicazione con la SCP. Infatti, il processo informatico che si vuole sviluppare potrà memorizzare nel database ausiliario le misurazioni acquisite dal database energetico di Roma Tre anche se poi queste non vengono ritrasmesse alla SCP a causa di eventuali malfunzionamenti e instabilità della comunicazione. Le informazioni verranno quindi memorizzate e trasmesse alla SCP una volta ristabilita la comunicazione, ottimizzando i tempi di accesso e di recupero delle informazioni di interesse. Per questo si prevede l'implementazione del database ausiliario tramite l'uso di strutture non relazionali in cui ogni documento sarà caratterizzato dalle informazioni riportate di seguito:

- **Building**, contiene tutte le informazioni anagrafiche di ogni edificio:
 - *building_id* [specification-properties], codice univoco identificativo dell'edificio;
 - *building_name* [specification-properties], nome dell'edificio;
 - *latitude* [context-coordinates], latitudine geografica dell'edificio;
 - *longitude* [context-coordinates], longitudine geografica dell'edificio;
 - *height* [context-coordinates], altitudine sul livello del mare dell'edificio.
- **Measurement**, contiene le informazioni riguardanti la grandezza misurata:
 - *measurement_id*, codice univoco identificativo di ogni grandezza misurata;
 - *measurement_description* [specification-properties], tipo di grandezza misurata;
 - *measurement_type* [specification-properties], unità di misura della grandezza misurata;
 - *measurement_primitive*, tipo di dato primitivo con cui è memorizzata.
- **Data**, contiene le informazioni riguardanti i dati acquisiti:
 - *data_id* [specification-id], codice univoco identificativo dell'UD;
 - *data_name* [specification-name], nome dell'UD;
 - *start_acquisition* [value-period], inizio del periodo di acquisizione dei dati;
 - *end_acquisition* [value-period], fine del periodo di acquisizione dei dati;
 - *data_value* [value-property], valore acquisito.

Tra parentesi quadre sono riportati i campi del MDA che trovano diretta corrispondenza con le informazioni mantenute in memoria. Per questo tipo di database si prevede di utilizzare un sistema di archiviazione basato su architettura non relazionale [5] caratterizzata da una strategia di memorizzazione *schemeless*. Questa permette non solo un accesso più veloce in lettura e scrittura dei dati, ma anche una gestione automatica dello *sharding* e della ridondanza della memoria che aumenta la persistenza dell'informazione. Per ogni edificio di Roma Tre deve essere creato un documento, ossia un record del database, all'interno del quale sono riconoscibili due tipi di informazioni:

- Immutabili, ossia i dati anagrafici dell'edificio e delle grandezze misurate;
- Real time, ossia il valore assunto dalla grandezza e il periodo temporale associato ad ogni acquisizione.

Consigliando l'utilizzo dell'applicativo MongoDB [6] per la realizzazione del database NO-SQL ausiliario, la struttura di ogni documento può essere rappresentata come riportato nella figura che segue.



Figura 3 - Schema struttura di memorizzazione

A valle di quanto detto, il processo informatico da realizzare deve essere in grado di:

- Al primo ciclo di attivazione:
 1. Creare una *Solution* e un *Urban Dataset* associato;
 2. Configurare i parametri per la trasmissione dei dati secondo le specifiche utilizzate dall'Urban Dataset Gateway.
- Nei cicli successivi:
 1. Acquisire i dati sui consumi energetici dal database di Roma Tre (RM3-DB);
 2. Parserizzare i dati in formato JSON per essere compliant con gli standard della Smart City Platform;
 3. Aggregare i dati all'interno di un Urban Dataset associato alla Solution;
 4. Configurare i parametri per la trasmissione dati all'Urban Dataset Gateway;
 5. Inviare l'Urban Dataset creato al sistema Smart City Platform tramite il Web Service Urban Dataset Gateway utilizzando il protocollo MQTT.
 6. Memorizzare i dati acquisiti all'interno del database ausiliario (Aux-DB), discriminando tramite un flag i dati trasmessi con successo all'Urban Dataset Gateway.

Il Data Exchange Manager (DEM) dovrà quindi accedere in lettura al database energetico di Roma Tre e in lettura e scrittura al database ausiliario, come raffigurato di seguito.

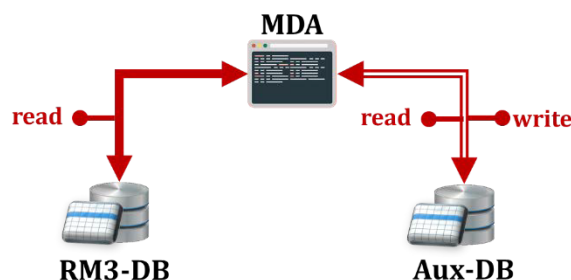


Figura 4 - Schema accessi ai database

2.4 Requisiti di progettazione

Lo scopo di questa sezione è di elencare e raggruppare una serie di caratteristiche che si rendono utili o necessarie per poter attuare quanto progettato. Queste caratteristiche vengono dette requisiti e possono essere principalmente i due tipi: una descrizione astratta e generica di un servizio o un vincolo del sistema, oppure una specifica concreta e dettagliata di una funzionalità del sistema.

Sulla base di questi contenuti sarà possibile indirizzare meglio la successiva fase di realizzazione e implementazione di un prototipo funzionale dell'ecosistema informatico proposto. Quest'ultimo, come detto, deve includere:

- *Local DataBase* (LDB), il database di Roma Tre nel quale sono memorizzati i dati energetici degli edifici dell'Ateneo;
- *Auxiliary DataBase* (ADB), il database ausiliario che consente una memorizzazione ridondante dei dati energetici;
- *Urban Dataset Gateway* (UDG), il Web Service al quale devono essere inviati i dati energetici
- *Data Exchange Manager* (DEM), il processo informatico che si occupa di gestire il flusso di dati all'interno dell'ecosistema.

I requisiti individuati sono divisi in base all'elemento dell'ecosistema a cui si riferiscono, e seguono un template predefinito, come riportato nella figura che segue.

Tabella 1 - Template descrizione requisito

ID:	Nome:	Importanza:
Descrizione:		

Si può osservare che il template è composto dai seguenti campi:

- *ID*, il codice univoco di identificazione del particolare requisito. Il codice si costruisce considerando il formato Progetto_Classificazione_numero, in cui:
 - Il progetto di District Database si abbrevia come DD;
 - I codici inerenti alla classificazione sono: ADB (database ausiliario), DEM (processo informatico), ECO (ecosistema informatico);
 - Il numero progressivo per definire univocamente il requisito.

I differenti codici per identificare i requisiti sono illustrati nella figura X.

- *Nome*, il nome assegnato al requisito;
- *Importanza*, che descrive il livello di importanza del requisito, e può essere:
 - *Richiesto* (RI), ossia obbligatorio e assolutamente richiesto;
 - *Raccomandato* (RA), quando può esistere un ragionevole motivo per ignorarlo ma è necessario valutare attentamente le conseguenze che si possono generare;
 - *Opzionale* (OP), ossia un requisito consigliato ma non obbligatorio.

Tabella 2 - Classificazione dei requisiti

Classificazione	Codice categoria
Generale	AE_GEN_<num>
Applicazione	AE_APP_<num>
Storage	AE_STO_<num>

Per riconoscere il livello di importanza del requisito si può fare anche riferimento al verbo che viene utilizzato nella sua descrizione. Infatti, verranno utilizzati: *Deve*, per riferirsi ad un requisito Richiesto e quindi obbligatorio; *Dovrebbe*, per riferirsi ad un requisito Raccomandato; *Potrebbe*, per riferirsi ad un requisito Opzionale.

Di seguito verranno riportati i requisiti funzionali della piattaforma, raggruppati per differenti macro-funzionalità individuate.

ID: DD_ECO_01	Nome: Main Authentication	Importanza: RI
Descrizione: Il sistema <i>deve</i> permettere l'accesso solo a seguito di autenticazione.		

ID: DD_ECO_02	Nome: Data Encryption	Importanza: RA
Descrizione: Il sistema <i>dovrebbe</i> supportare l'uso di protocolli criptati per la connessione da remoto (HTTPS e SSH).		

ID: DD_ECO_04	Nome: System Availability	Importanza: RI
Descrizione: Il sistema <i>deve</i> garantire un'alta disponibilità del servizio 24/7.		

ID: DD_ECO_05	Nome: Data Integrity Standard	Importanza: RI
Descrizione: Il sistema <i>deve</i> utilizzare standard e protocolli di comunicazione che garantiscono l'integrità dell'informazione.		

ID:	Nome:	Importanza:
------------	--------------	--------------------

DD_ECO_06	Prototyping Documentation	RA
Descrizione: Il prototipo che verrà realizzato in una fase successiva <i>dovrebbe</i> essere adeguatamente documentato.		

ID: DD_ECO_7	Nome: Prototyping Testability	Importanza: RI
Descrizione: Sul prototipo che verrà realizzato <i>devono</i> essere effettuati test di validazione inerenti ai requisiti funzionali.		

ID: DD_DEM_01	Nome: DEM-LDB Data Exchange	Importanza: RI
Descrizione: Il DEM <i>deve</i> essere in grado di strutturare un flusso massivo di lettura dati con la struttura di archiviazione LDB.		

ID: DD_DEM_02	Nome: DEM-ADB Data Exchange	Importanza: RI
Descrizione: Il DEM <i>deve</i> essere in grado di strutturare un flusso massivo di lettura e scrittura dati con la struttura di archiviazione ADB.		

ID: DD_DEM_03	Nome: DEM-UDG Data Exchange	Importanza: RI
Descrizione: Il DEM <i>deve</i> essere in grado di strutturare un flusso massivo di invio dati con la componente UDG.		

ID: DD_DEM_04	Nome: DEM-LDB Login Credentials	Importanza: RI
Descrizione: Il DEM <i>deve</i> essere in possesso delle credenziali di accesso alla struttura di archiviazione LDB.		

ID: DD_DEM_05	Nome: DEM-ADB Login Credentials	Importanza: RI
Descrizione: Il DEM <i>deve</i> essere in possesso delle credenziali di accesso alla struttura di archiviazione ADB.		

ID: DD_DEM_06	Nome: LDB Data Acquisition	Importanza: RI
Descrizione: Il DEM <i>deve</i> acquisire i dati energetici dalla struttura di archiviazione LDB utilizzando query di selezione strutturate in base all'ambiente di sviluppo utilizzato.		

ID:	Nome:	Importanza:
------------	--------------	--------------------

DD_DEM_07	JSON Translation	RI
Descrizione: Il DEM <i>deve</i> organizzare i dati acquisiti dal LDB secondo il formato JSON.		

ID: DD_DEM_08	Nome: ADB Data Insert	Importanza: RI
Descrizione: Il DEM <i>deve</i> memorizzare i dati all'interno della struttura di archiviazione ADB utilizzando query di inserimento strutturate in base all'ambiente di sviluppo utilizzato.		

ID: DD_DEM_09	Nome: Message Preparation	Importanza: RI
Descrizione: Il DEM <i>deve</i> incapsulare i dati energetici in messaggi compliant con lo standard MQTT.		

ID: DD_DEM_10	Nome: Publisher Implementation	Importanza: RI
Descrizione: Il DEM <i>deve</i> implementare un publisher MQTT in grado di comunicare con il broker di messaggi (UDG).		

ID: DD_DEM_11	Nome: DEM Network	Importanza: RI
Descrizione: Il DEM <i>deve</i> essere implementato a bordo di un host che risiede nell'intranet di Roma Tre.		

ID: DD_DEM_12	Nome: DEM Location	Importanza: RA
Descrizione: Il DEM <i>dovrebbe</i> essere implementato a bordo dello stesso host su cui risiede l'ADB.		

ID: DD_DEM_13	Nome: DEM Programming	Importanza: OP
Descrizione: Il DEM <i>potrebbe</i> essere implementato utilizzando il linguaggio di programmazione Python [7].		

ID: DD_DEM_14	Nome: DEM Protection	Importanza: RI
Descrizione: Il DEM <i>deve</i> prevedere un processo di autenticazione in scrittura per essere protetto da eventuali modifiche non desiderate.		

ID:	Nome:	Importanza:
------------	--------------	--------------------

DD_DEM_15	Unsent Data Recognition	RI
Descrizione: Il DEM <i>deve</i> riconoscere la presenza di un set di dati all'interno dell'ADB che non sono stati ancora inviati all'UDG.		

ID: DD_DEM_16	Nome: Communication Unavailable	Importanza: RI
Descrizione: Il DEM <i>deve</i> riconoscere la presenza di eventuali problemi di comunicazione con l'UDG.		

ID: DD_ADB_01	Nome: ADB Login Authentication	Importanza: RI
Descrizione: L'ADB <i>deve</i> prevedere un processo di autenticazione degli accessi basato sul controllo di username e password.		

ID: DD_ADB_02	Nome: ADB Login Authentication List	Importanza: RA
Descrizione: L'ADB <i>dovrebbe</i> prevedere un processo di autenticazione degli accessi basato su <i>white list</i> degli indirizzi IP associati agli <i>host</i> a cui è permesso l'accesso.		

ID: DD_ADB_03	Nome: ADB Network	Importanza: RI
Descrizione: L'ADB <i>deve</i> essere implementato a bordo di un host che risiede nell'intranet di Roma Tre.		

ID: DD_ADB_04	Nome: ADB Location	Importanza: RA
Descrizione: L'ADB <i>dovrebbe</i> essere implementato a bordo dello stesso host su cui risiede l'ADB.		

ID: DD_ADB_05	Nome: ADB Programming	Importanza: RI
Descrizione: L'ADB <i>deve</i> essere implementato secondo la logica dei database non relazionali basati su documenti.		

ID: DD_ADB_06	Nome: ADB Implementation	Importanza: OP
Descrizione: L'ADB <i>può</i> essere implementato utilizzando l'applicativo MongoDB.		

3 Audit Energetico

L'obiettivo di questa attività è implementare una piattaforma informatica in grado di acquisire, memorizzare ed elaborare i dati energetici riguardanti le unità abitative private che influiscono largamente sui consumi totali di un distretto energetico. La soluzione proposta è basata su una architettura modulare che dal punto di vista strutturale è suddivisa in di macro-elementi interoperabili, come emerge dalle specifiche funzionali e di progettazione definite nell'annualità precedente, osservabili in figura X. In particolare, si possono distinguere:

- Il macro-elemento *Engine* si occupa di elaborare i dati e produrre simulazioni energetiche per fornire benchmark e suggerimenti di riqualificazione energetica. Questo elemento, ampiamente documentato nel report RdS/2019/002 a cui si rimanda per maggiori approfondimenti, consiste in un complesso foglio di calcolo XLSM.
- Il macro-elemento *Orchestrator* si occupa di gestire e supervisionare il flusso di informazioni che devono essere inserite ed utilizzate all'interno del blocco *Engine* e di recuperare da esso i risultati delle elaborazioni effettuate. Questo elemento, progettato durante la precedente annualità, è stato implementato dal gruppo di ricerca di ENEA e ampiamente documentato nel RdS/2019/002.
- Il macro-elemento *Storage* si occupa di mantenere in memoria i dati energetici degli utenti e i risultati delle elaborazioni effettuate dall'*Engine*.
- Il macro-elemento *Application* si occupa di fornire un'interfaccia semplice e intuitiva che accompagna l'utente nel processo di immissione dei dati energetici e nella visualizzazione dei risultati ottenuti a seguito delle elaborazioni del blocco *Engine*.

Il presente elaborato fornisce le informazioni tecniche di implementazione e una descrizione delle funzionalità realizzate per gli ultimi due macro-elementi presentati sopra, ossia il blocco *Storage* ed il blocco *Application* di cui si è occupato il gruppo di ricerca di Roma Tre.

3.1 Elemento Storage

È stata effettuata un'analisi delle informazioni da archiviare all'interno dello *Storage* dalla quale è emerso che, nonostante i tipi di dato possono essere associati alle stesse primitive (numero intero, numero reale, stringa), le informazioni si riferiscono a grandezze molto differenti tra loro. Ovviamente, ognuna di queste informazioni è necessaria a caratterizzare la medesima unità operativa. Per questo motivo abbiamo ritenuto opportuno contravvenire a quelli che erano i requisiti progettuali definiti nella precedente annualità, che prevedevano l'utilizzo di un database relazionale (SQL) [8], adottando una soluzione più attuale basata su database non relazionale (NO-SQL) [5]. In particolare, è stato utilizzato MongoDB [6], un database non relazionale basato su documenti, che attualmente risulta esser molto utilizzato nelle soluzioni di *Big Data* e di *Cloud Computing*. Questo strumento costituisce una soluzione flessibile e affidabile per la gestione delle informazioni all'interno dell'ecosistema implementato. Tramite le interrogazioni, o *query*, scritte nel linguaggio associato a questo strumento di sviluppo, è possibile interagire con il database e compiere tutte le operazioni di lettura scrittura e aggiornamento dei dati. Ovviamente, le query sono state scritte seguendo pedissequamente regole ben precise di sintassi e testate per valutarne il corretto funzionamento creando delle connessioni con il database tramite indirizzo IP.

Per mantenere sicure, compatte e di facile accesso le informazioni memorizzate, è stata definita una struttura di archiviazione composta da due database distinti che risiedono su due diverse macchine. La scelta di adottare questa soluzione è stata dettata dalla necessità di ridurre al minimo le latenze dovute al sovraccarico di questa risorsa condivisa dovute agli accessi da parte del processo informatico *Engine* e delle routine associate al web service *Application*. Sviluppare il tutto in un unico database, come previsto dalla progettazione della precedente annualità, porterebbe sicuramente ad un sensibile aumento del tempo computazionale nelle fasi di lettura e scrittura dati una volta che il processo di *Audit Energetico* sarà utilizzato massivamente.

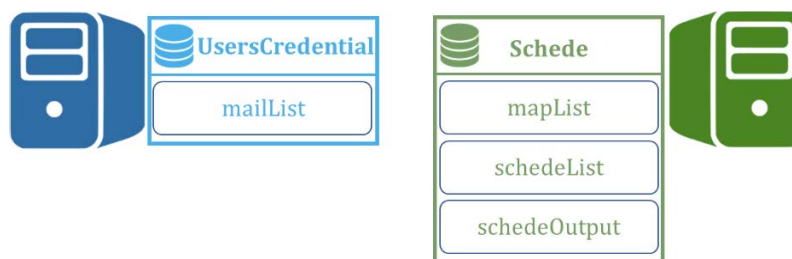


Figura 5 - Database della piattaforma

UsersCredential è un database all'interno del quale vengono memorizzate le informazioni anagrafiche, attualmente l'indirizzo email, di ogni utente che si avvale del servizio di *Audit Energetico*. Per evitare che eventuali malintenzionati possano accedere facilmente a questi dati tramite degli attacchi cyber, il database è stato realizzato con due gradi di protezione. Infatti, l'accesso al database è consentito solo a client che risiedono sulla stessa intranet, ossia la rete di comunicazioni interna ad ENEA. Inoltre, si è applicata una gestione degli accessi tramite username e password. L'unico elemento che ha la possibilità di interagire in lettura e scrittura con questo database è l'*Application* che ha il compito di aggiungere gli utenti registrati.

In questo database è stata implementata un'unica collezione, *mailList*, i cui documenti sono caratterizzati da due campi:

- *_id* , codice univoco identificativo di ogni documento all'interno del database;
- *Email* , indirizzo email inserito dall'utente in fase di registrazione dell'audit.

Per realizzare un'architettura che fosse compliant con il GDPR [9], in un altro database, *Schede*, sono memorizzati sia i dati inseriti dall'utente dell'audit che verranno utilizzati per l'elaborazione da parte dell'*Engine*, sia le informazioni di output da fornire all'utente a seguito dell'elaborazione dell'*Engine*. Perciò, a questo database possono accedere in lettura e scrittura sia l'elemento *Orchestrator* che l'elemento *Application*. In questo caso, la gestione degli accessi viene regolata tramite riconoscimento di username e password. Per poter associare le informazioni energetiche con gli utenti che si sono registrati all'audit, si sfrutta una collezione di appoggio per realizzare l'operazione di mapping. In particolare, l'indirizzo email dell'utente viene crittografato, ossia riorganizzato in una sequenza apparentemente senza senso denominata *testo cifrato*. In particolare, per gli scopi di questa attività, si è deciso di ricorrere ad uno dei più comuni metodi di crittografia dei dati, ossia il metodo *hash* [10]. La particolare tecnica di cifratura utilizzata viene detta *asimmetrica*, o *a senso unico*. Infatti, grazie alla funzione di *hash*, è possibile ricavare una stringa di lunghezza fissa e apparentemente incomprensibile, a partire da una stringa qualsiasi, facendo in modo però che l'operazione inversa sia impossibile.

Le collezioni implementate all'interno di questo database sono:

- *mapList* contiene le informazioni utili per mappare le informazioni energetiche con gli utenti registrati. In particolare, ogni documento della collezione è caratterizzato dai campi:
 - *_id*, codice univoco identificativo di ogni documento all'interno del database;
 - *id_registrazione* , corrisponde al valore contenuto campo *_id* presente nei documenti della collezione *mailList*;
 - *id_utente* , corrisponde al valore contenuto nel campo *Email* presente nei documenti della collezione *mailList* al quale è stata applicata la crittografia *hash*.
- *schedeList* contiene le informazioni inserite dall'utente. Ogni documento della collezione corrisponde ad un particolare utente ed è stato strutturato tenendo presente le macro-tematiche

ubicazione edificio, utilizzo casa, caratteristiche architettoniche, impianti, elettrodomestici, consumi e costi. Gli altri tre campi sono:

- *_id* , codice univoco identificativo di ogni documento all'interno del database;
 - *userID* , codice identificativo di ogni utente che ha iniziato il processo di audit;
 - *avanzamento* , per definire lo stato di completamento del processo di inserimento dei dati da parte dell'utente.
- *schedeOutput* contiene i dati elaborati dall'*Engine* e che devono essere forniti all'utente tramite l'elemento *Application*. Ogni documento della collezione contiene tutte le informazioni di output associate ad un particolare utente e suddivise secondo le macro-tematiche *dati geometrici, consumi elettrici, consumi gas, consumi energia primaria, elettrificazione, riqualificazione*. In questo caso sono presenti altri due campi:
 - *_id* , codice univoco identificativo di ogni documento all'interno del database;
 - *userID* , codice identificativo di ogni utente che ha iniziato, e terminato, il processo di audit.

Gli accessi consentiti in lettura e/o scrittura ai due diversi database e, più in particolare, alle loro collezioni sono schematizzati in figura X.

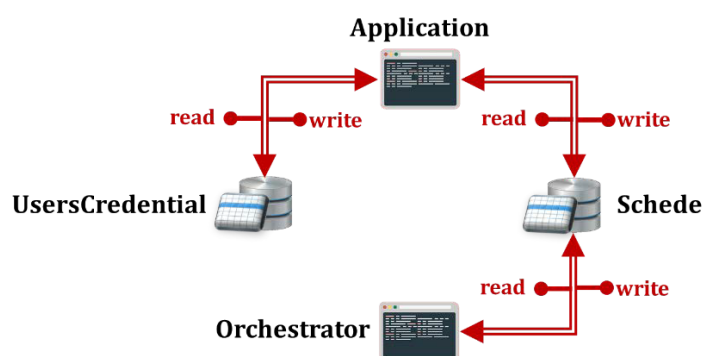


Figura 6 - Schema degli accessi ai database

In particolare, si può comprendere più chiaramente il processo che viene effettuato per popolare i database considerando i seguenti passi:

- L'utente si registra all'Audit Energetico inserendo l'indirizzo email.
- L'elemento *Application* crea un documento all'interno della collezione *mailList* memorizzando l'indirizzo dell'utente nel campo *Email*.
- L'applicativo MongoDB assegna di default, se non forzato manualmente, un codice identificativo univoco, *_id*, al documento appena inserito.
- L'elemento *Application* crea un documento all'interno della collezione *mapList* memorizzando:
 - il codice ottenuto al passo precedente nel campo *id_registrazione*.
 - nel campo *id_utente* la stringa che rappresenta l'email dell'utente crittografata utilizzando il metodo *hash*.
- L'elemento *Application* crea un documento all'interno della collezione *schedeList* memorizzando:
 - Nel campo *userID* il valore contenuto nel campo *id_utente* della collezione *mapList*.
 - Il valore 0 nel campo *avanzamento*.

A questo punto l'utente può iniziare la compilazione del questionario energetico.

- Ogni qualvolta che l'utente termina l'inserimento dei dati associati ad un macro-argomento, l'elemento *Application* memorizza i dati nei campi associati all'interno della collezione *schedeList*, e aggiorna il campo *avanzamento* secondo un ordine progressivo da 1 a 5.
- Terminato l'inserimento dei dati, l'elemento *Application* invia una richiesta *POST* all'elemento *Orchestrator* segnalando l'avvenuta conclusione del questionario.
- L'elemento *Application* crea un documento all'interno della collezione *schedeOutput* memorizzando nel campo *userID* il valore contenuto nel campo *id_utente* della collezione *mapList*.

L'elemento *Orchestrator* raccoglie i dati memorizzati all'interno della collezione *schedeList* e li inserisce nelle celle dell'elemento *Engine*. Quindi attende il termine dell'elaborazione, legge i dati dalle celle dell'elemento *Engine* e li memorizza nella collezione *schedeOutput* all'interno del documento associato all'utente.

- L'elemento *Application* legge i dati inseriti nella collezione *schedeOutput* e le riporta all'utente.

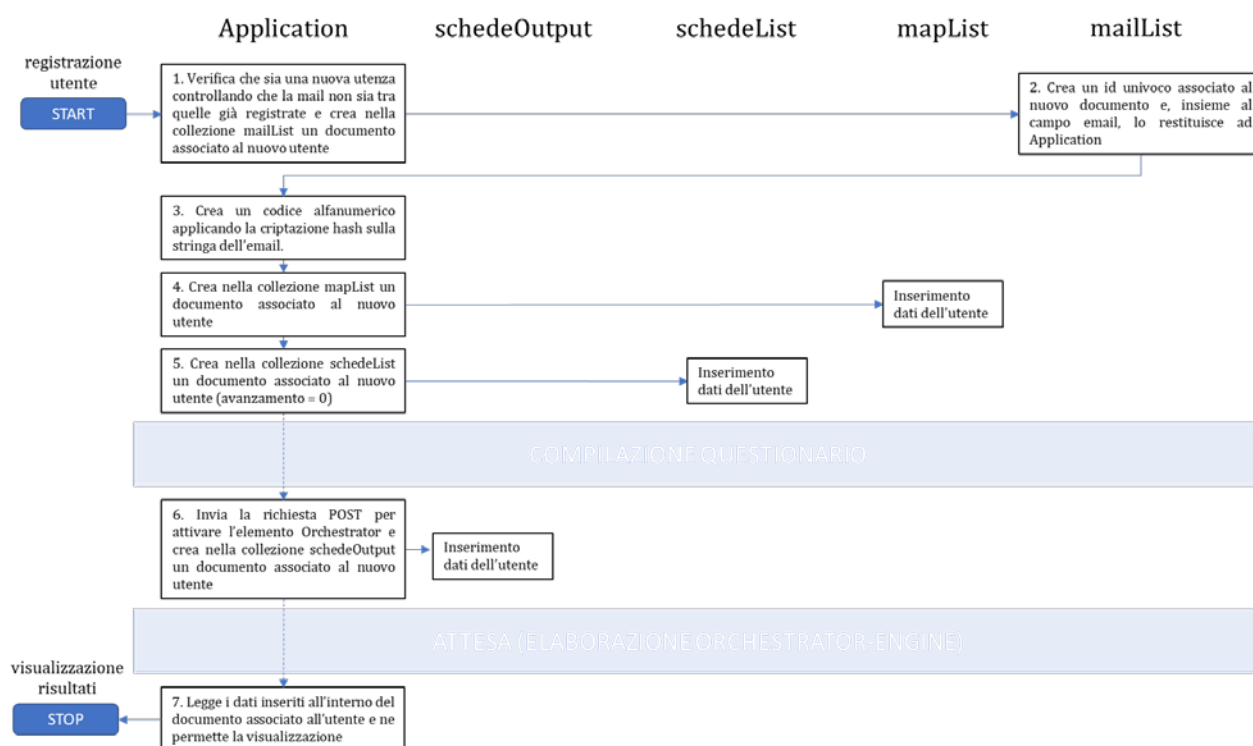


Figura 7 - Processo per il popolamento dei database

3.2 Elemento Application

Per realizzare l'elemento che si occupa della gestione della web application e della memorizzazione dei dati all'interno della struttura di archiviazione si è proceduto tenendo ben presente la fondamentale distinzione tra ciò che riguarda il front-end ed il back-end. Per front-end, infatti, si intende tutta quella parte di web application che permette l'interazione con l'utente, ossia tutte le pagine e le funzionalità ed esse associate che vengono rese disponibili all'utente. La parte di back-end, invece, rappresenta tutta la struttura della web application che si occupa della gestione e della realizzazione delle funzionalità dell'interfaccia. Questa distinzione permette di stabilire i linguaggi di programmazione che devono essere utilizzati per implementare le funzionalità richieste dalla piattaforma.

In particolare, si è deciso di utilizzare tre linguaggi classici di programmazione, molto utilizzati, e che, se ben maneggiati e coordinati consentono di definire l'aspetto e il comportamento dell'interfaccia grafica e, più in generale, della web application. Quindi è stata programmata in HTML [11] la struttura statica delle pagine, personalizzandola tramite il CSS [12] grazie al quale è possibile dotare la pagina di particolari colori, font, immagini di sfondo, ecc. Mentre, utilizzando il linguaggio Javascript [13] si è fornita la pagina di tutti quei meccanismi e di tutte quelle azioni che permettono l'interazione dinamica da parte dell'utente.

Tenendo ben presente l'obiettivo di questa attività, è stata realizzata un'interfaccia utente in grado di gestire i processi riportati di seguito:

- Introdurre l'utente alla compilazione del questionario e registrare una nuova utenza;
- Controllare gli accessi alla piattaforma;
- Guidare l'utente durante la compilazione del questionario;
- Scrivere sul database i dati inseriti dall'utente e leggere dal database i risultati dell'elaborazione dell'elemento *Engine*;
- Mostrare all'utente i risultati di output ottenuti.



Figura 8 - Le funzioni del Web Service

Per rendere più comprensibili le funzionalità a disposizione dell'utente e le scelte che questo può effettuare, si è deciso di adottare una rappresentazione orientata all'utilizzatore, traducendo i meccanismi di inserimento dei dati proposti dal componente *Engine* in elementi grafici che permettono un notevole miglioramento dell'accessibilità alla web application. In particolare, sono stati utilizzati:

- Elementi per l'inserimento dati: campi di testo, liste dropdown, checkbox, steppers, sliders, bottoni con icona;
- Elementi per l'auto compilazione (o compilazione facilitata): bottoni con testo, tooltip;
- Elementi che consentono di attivare ulteriori selezioni: interruttori toggle, checkbox.



Figura 9 - Elementi grafici di inserimento dati

Dopo aver analizzato le informazioni presenti nel questionario si è deciso di suddividere il processo di compilazione in cinque step consecutivi che corrispondono ai cinque macro-argomenti con cui è possibile classificare le domande proposte all'utente. Di conseguenza, ogni macro-argomento ha una pagina dedicata, che viene gestita attraverso tre diversi script:

- *AuditSchedaX.html* per la parte statica. Viene definita la struttura della pagina ed il contenuto grafico e testuale che deve essere visualizzato. Dal punto di vista della gestione dello spazio, l'interfaccia è suddivisa in aree funzionali. In particolare, è possibile distinguere:
 - Scheda domande, all'interno della quale sono riportate le domande del questionario;
 - Barra di progressione, per tenere traccia visiva sullo stato d'avanzamento del questionario;
 - Logo, che contiene il logo del capoprogetto ENEA, oppure del progetto stesso, a seconda dei casi;
 - Home/login, all'interno della quale è stato implementato un pulsante di *shortcut* verso la pagina iniziale oppure la pagina di login, a seconda dei casi;
 - Tab Prec, all'interno della quale è stato implementato un pulsante che consente il reindirizzamento alla pagina associata al macro-elemento appena completato;
 - Tab Succ, all'interno della quale è stato implementato un pulsante che permette di essere reindirizzati al macro-argomento successivo, se quello attuale è stato correttamente completato.

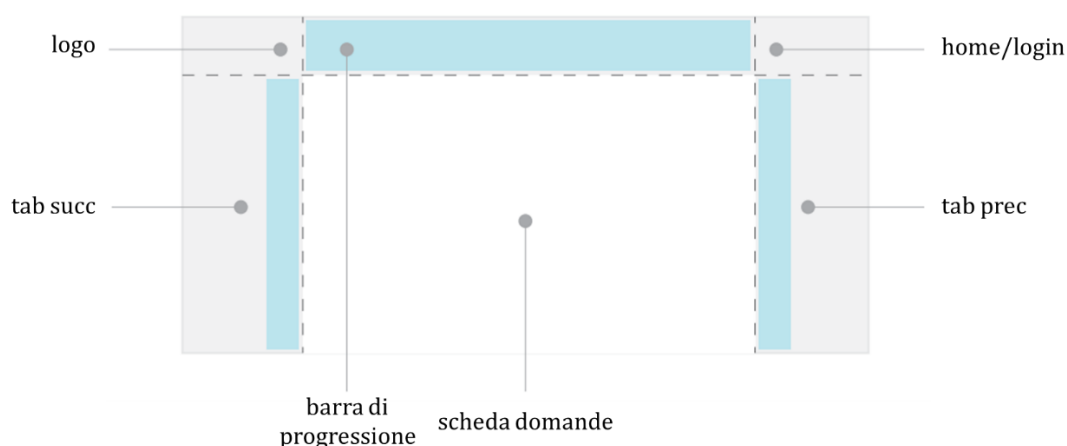


Figura 10 - Layout delle pagine

- *AuditSchedaX.js* per la parte dinamica. Implementa le dinamiche di controllo e di coerenza dei valori inseriti dall'utente. In particolare:
 - Nel caso che un dato necessario per l'elaborazione non sia stato inserito il processo non permette di proseguire alla pagina successiva.
 - Nel caso un valore non sia coerente con le informazioni rese in precedenza, o conforme con i valori limiti previsti per quel particolare dato, il processo non permette di proseguire alla pagina successiva.

In entrambi i casi viene segnalato l'errore e si riporta la visualizzazione sull'elemento di input associato. Tramite questo script si gestiscono anche:

- La definizione delle funzionalità di auto completamento che sono previste in parti diverse del questionario;
- L'attivazione di parti del questionario associate alla dinamica degli interruttori toggle;
- Gli indirizzamenti alle altre pagine del questionario, mantenendo le informazioni necessarie per garantire le funzioni di auto completamento;
- L'invocazione, tramite chiamate POST, delle *ajax* per poter accedere in lettura e scrittura all'elemento *Storage*.

E' possibile passare alla sezione successiva del questionario utilizzando il bottone *proseguì* presente al termine di ogni sezione. Se non sono rilevati errori o incongruenze nella compilazione, l'utente viene indirizzato alla pagina successiva e vengono effettuate le chiamate POST per memorizzare i dati appena inseriti.

- *AuditSchedaX_ajax.php* per stabilire la comunicazione con la struttura di memoria *Storage* ed effettuare le operazioni di lettura e scrittura dei dati energetici dell'utente. I dati riguardanti la particolare sezione vengono memorizzati solo quando il back-end ha validato la loro presenza e coerenza.

Terminata l'elaborazione dei dati, è necessario che i risultati ottenuti siano presentati all'utente che ha compilato il questionario. Per questo è stata implementata la pagina *AuditResult*, anch'essa composta dai tre diversi files caratterizzati rispettivamente dall'estensione *.html* per la parte statica, *.js* per l'aspetto dinamico, *.php* per la gestione delle comunicazioni con l'elemento *Storage*.

Per gestire gli accessi alla piattaforma di audit energetico, distinguendo gli utenti al primo accesso dagli utenti già registrati, è stata implementata una pagina di login chiamata *AuditAccesso*. La combinazione di files per realizzare le funzionalità di questa pagina è la stessa di quella utilizzata per le altre ma, in questo caso, lo script con estensione *.js* si occupa di memorizzare l'email dell'utente nel database *UsersCredential* e di generare il codice alfanumerico identificativo per ogni singolo utente e che verrà utilizzato all'interno del database *Schede*.

Nella figura che segue è riportato lo schema logico delle interazioni e delle associazioni esistenti tra i files di script implementati.

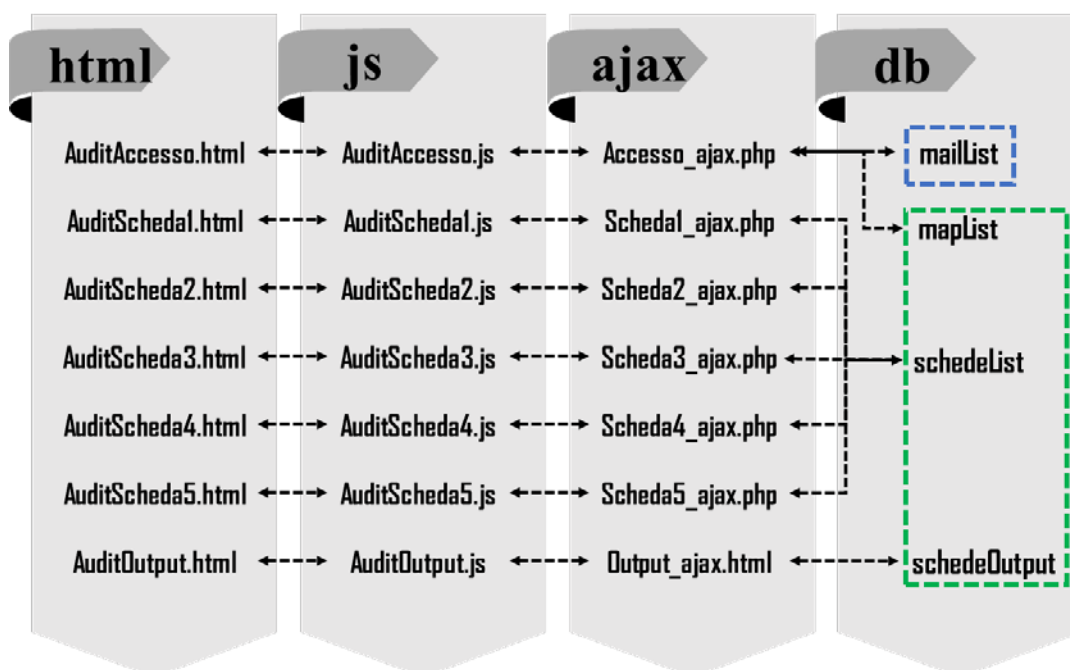


Figura 11 - Script per l'implementazione del Web Service

3.3 Soluzione realizzata

In questa sezione si vuole presentare la web application implementata per realizzare le funzionalità di audit energetico per le utenze residenziali private. Per chiarezza di esposizione, viene descritto di seguito il processo di accesso e compilazione messo a disposizione dell'utente.

3.3.1 Percorso di benvenuto e accesso alla piattaforma

Ogni utente interessato ad effettuare l'Audit Energetico ha la possibilità di usufruire di questo servizio in forma anonima fornendo solamente un indirizzo e-mail di riferimento. Sulla pagina principale dell'Agenzia (www.enea.it) verrà prevista una sezione dedicata a contenere un collegamento diretto shortcut alla pagina *AuditBenvenuto*. Infatti, prima di procedere con l'accesso, vengono fornite all'utente tutta una serie di informazioni utili per la compilazione del questionario come, ad esempio, la necessaria disponibilità dei consumi mensili su arco annuale riportati in bolletta, oppure la conoscenza delle dimensioni approssimative dei muri perimetrali dell'abitazione. In questo modo l'utente avrà chiaro quali sono le possibili azioni da effettuare durante l'inserimento dei dati e le informazioni necessarie per ottenere un'elaborazione corretta da parte dell'elemento *Engine*. Viene comunque lasciata la possibilità di saltare il percorso di benvenuto premendo il pulsante nell'area *home/login* ed essere indirizzati direttamente alla pagina di accesso.

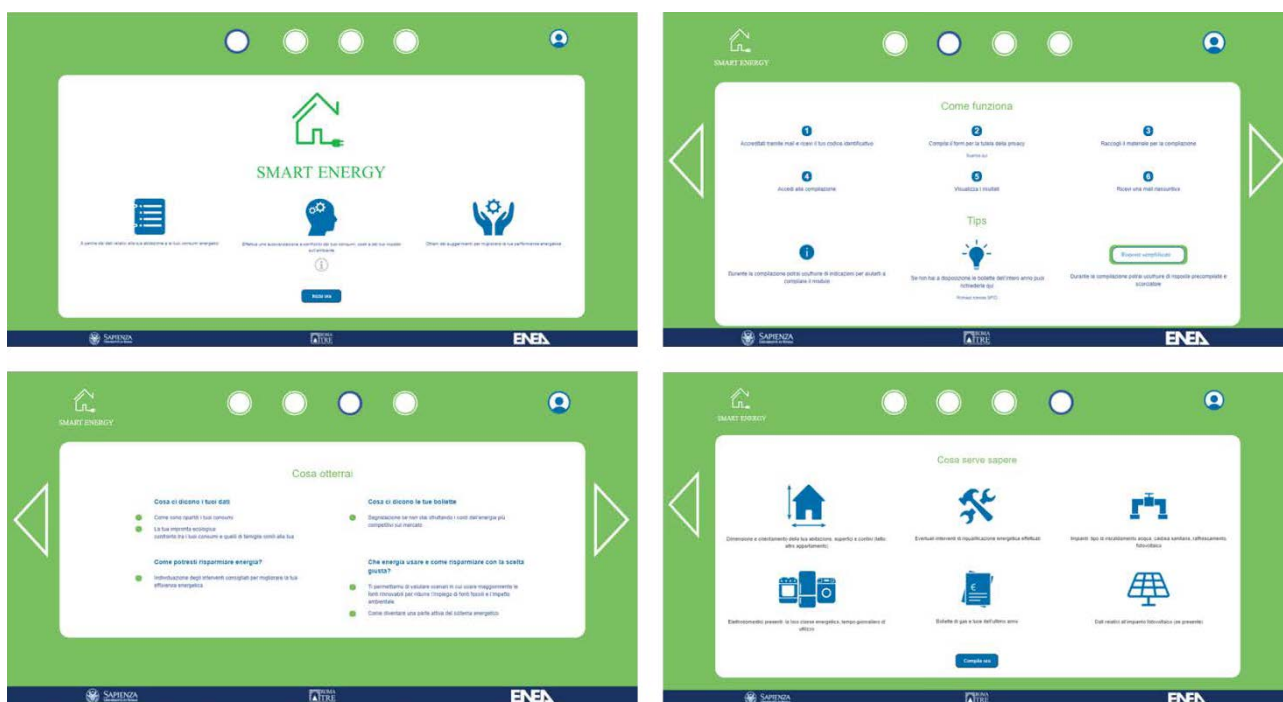


Figura 12 - Screenshot pagina di benvenuto

Una volta terminati il percorso di benvenuto, si è indirizzati alla pagina *AuditAccesso* all'interno della quale è possibile creare una nuova utenza oppure accedere come un utente registrato, come indicato testualmente.



Figura 13 - Screenshot pagina di accesso

- Per creare una nuova utenza è sufficiente inserire una e-mail di riferimento e premere il tasto invio. Se l'indirizzo è già presente tra gli utenti registrati viene riportato un messaggio testuale che invita ad effettuare l'accesso tramite password, altrimenti viene fornito un codice alfanumerico che l'utente dovrà appuntare e inserire nel campo password. Una volta premuto il pulsante di invio corrispondente, si viene indirizzati alla pagina *AuditScheda1* riguardante il primo macro-argomento del questionario.
- Per accedere come utente registrato è sufficiente inserire nel campo password il codice alfanumerico che è stato visualizzato durante la procedura di registrazione e, dopo aver premuto sul pulsante di invio corrispondente, si viene indirizzati alla pagina associata al primo macro-argomento in ordine logico-sequenziale per cui non si è ancora compilato il questionario.

3.3.2 AuditScheda1 – Ubicazione ed utilizzo dell'abitazione

REACT ENERGY

Impostazioni

Impostazioni generali

Localizzazione

Indirizzo

Città

Categorie

Utilizzo della casa

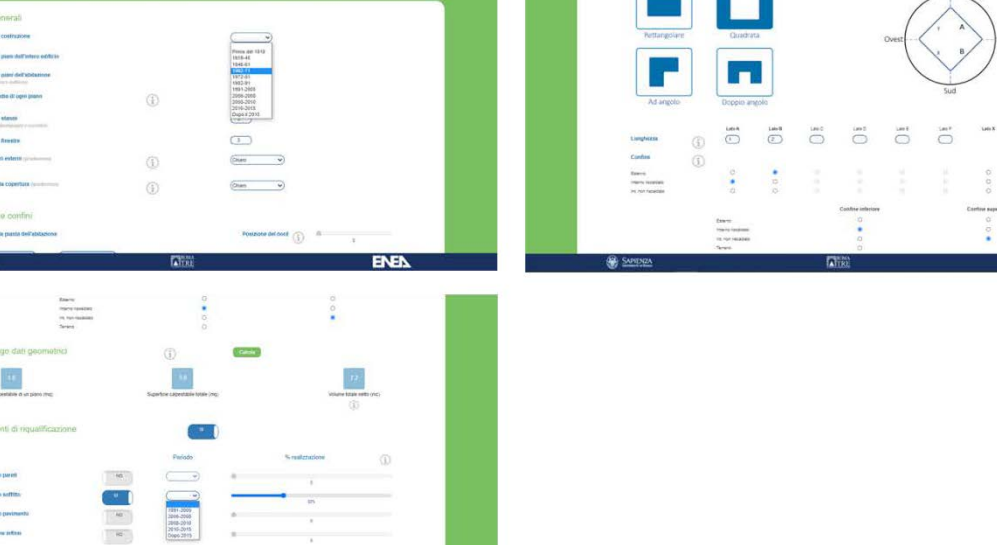
Numero di occupanti

Data di nascita

Impostazioni avanzate

REACT ENERGY

3.3.3 AuditScheda2 – Caratteristiche architettoniche



- I dati generali dell'edificio, come il periodo di costruzione, il numero di piani da cui è composto, il colore dei muri, ecc.
- I dati sulla pianta e i confini dell'abitazione, come la forma, la dimensione di ogni lato e il tipo di locali con cui è confinante, ecc.
- I lavori di isolamento dell'abitazione effettuati ed il loro stato di avanzamento, come ad esempio l'utilizzo di infissi termoisolanti, ecc.

Viene mantenuta traccia di alcune di queste informazioni poiché verranno sfruttate nelle schede successive per permettere, ove possibile, la compilazione automatica di alcuni campi.

3.3.4 AuditScheda3 – Impianti

In questa parte del questionario si raccolgono le informazioni riguardanti la tipologia di impianti che caratterizzano l'abitazione e la modalità di utilizzo. In particolare, si considerano:

- Impianto di riscaldamento, impianto di raffrescamento e impianto di erogazione dell'acqua sanitaria, definendone tipologia, caratteristiche e utilizzo medio;
- Impianti di generazione energetica se presenti, ovvero la tipologia e le caratteristiche dell'impianto solare-termico e dell'impianto fotovoltaico.

Figura 16 - Screenshot terza pagina del questionario

3.3.5 AuditScheda4 – Elettrodomestici

In questa parte del questionario si raccolgono le informazioni riguardanti la tipologia e quantità di elettrodomestici presenti nell'abitazione e una stima sul loro utilizzo medio settimanale o giornaliero. In particolare, i dispositivi energivori domestici vengono suddivisi in base al loro utilizzo per facilitare l'operazione di inserimento da parte dell'utente. Viene, quindi, tenuta traccia dei dispositivi utilizzati per:

- Le operazioni di cottura del cibo, ossia il forno, il piano cottura e il microonde;
- La refrigerazione e il riscaldamento elettrici, perciò si fa riferimento ai dispositivi di condizionamento dell'aria;
- Il lavaggio e la stiratura di vestiti, ossia lavatrice, asciugatrice, ecc., e anche la lavastoviglie;
- L'illuminazione dell'abitazione, ossia lampade e punti luce;
- L'igiene personale, come l'asciugacapelli, oppure per il lavoro, come il computer fisso e quello portatile.

Inoltre, viene data la possibilità all'utente di inserire altri tre apparecchi che non sono presenti nel questionario, fornendo le stesse indicazioni richieste per quelli presenti, ossia indicandone il nome, la quantità e l'utilizzo medio giornaliero o settimanale.

Figura 17 - Screenshot quarta pagina del questionario

3.3.6 AuditScheda5 – Consumi e costi

In questa parte del questionario si raccolgono le informazioni, desumibili dai dati contenuti nella bolletta dell'utente, riguardanti due servizi essenziali principali, ossia l'elettricità ed il gas. Per entrambi, è stata strutturata una form per facilitare l'inserimento di:

- Consumi elettrici mensili e spesa elettrica annuale;
- Consumi del gas mensili e spesa di consumo gas annuale.

Viene lasciata all'utente la scelta del mese dal quale iniziare l'inserimento dei dati, l'importante è che questi siano basati su un arco di tempo annuale.

Una volta terminato il questionario, l'utente viene avvisato tramite messaggio testuale dell'avvenuto salvataggio di tutte le informazioni, e viene indirizzato alla pagina di accesso alla piattaforma. Infatti, i risultati saranno disponibili solo dopo l'elaborazione da parte dell'elemento *Engine*, che solitamente richiede qualche minuto. Quindi l'utente, inserendo nuovamente la password, viene indirizzato alla pagina adibita alla visualizzazione degli output.

The screenshot shows a questionnaire interface with two main sections: 'Energia elettrica' (Electricity) and 'Gas naturale' (Natural Gas). Each section has a table for inputting consumption data for different months (Maggio, Aprile, Maggio, Giugno, Luglio, Agosto) and a 'Spesa energetica' (Energy cost) field. The interface includes a navigation bar at the top and logos for Sapienza, Roma Tre, and ENEA at the bottom.

Figura 18 - Screenshot quinta pagina del questionario

3.3.7 AuditOutput – Visualizzazione risultati

Tramite questa pagina l'utente è in grado di visualizzare le informazioni ottenute a valle dell'elaborazione dei dati energetici che ha inserito nel questionario. Poiché le funzionalità di questa pagina non sono ancora state definite, si propone di seguito un mock-up per fornire un'idea di quale sarà la visualizzazione che sarà disponibile all'utente.



Figura 19 - Mock-up pagina di output del questionario

4 Conclusioni

In questa sezione si vuole sintetizzare i risultati complessivi ottenuti durante la presente annualità.

Il primo obiettivo è stato quello di definire e progettare un sistema per acquisire i dati energetici degli edifici dell'Università Roma Tre e inviarli alla Smart City Platform realizzata all'interno dei server di ENEA. Lo scopo di questa piattaforma è realizzare un ecosistema tecnologico in grado di effettuare un'azione di diagnostica e di predizione dei potenziali di danno per un'infrastruttura e un'azione di monitoraggio dell'ambiente metropolitano. Quindi, durante la prima fase di analisi dello scenario è stato indispensabile

individuare i dati messi a disposizione dal database energetico di Roma Tre e le caratteristiche richieste dalla SCP per poter memorizzare le informazioni al suo interno. Per organizzare, controllare e gestire la comunicazione tra questi due elementi è stato progettato un processo informatico in grado di:

- Configurare la Solution che verrà popolata di dati e di creare l'istanza della Smart City Platform che verrà usata;
- Acquisire i dati energetici dal database di Roma Tre;
- Aggregare i dati nel Urban Dataset;
- Formattare e i dati rappresentando l'Urban Dataset in formato JSON;
- Inviare i dati alla Smart City Platform tramite la Urban Dataset Getaway.

La rappresentazione schematica delle fasi logiche introdotte è riportata di seguito.



Figura 20 - Fasi logiche del DEM

E' possibile che durante il normale funzionamento del processo avvengano degli eventi imprevisti che possono causare l'interruzione della comunicazione tra il processo informatico DEM e l'UDG. Ciò comporterebbe la perdita dei dati non ancora trasmessi. Per ovviare a questo problema deve essere implementato un database ausiliario per mantenere in memoria tutte le informazioni da inviare alla SCP. L'ecosistema informatico progettato è quindi composto da quattro macro-elementi come riportato in figura.

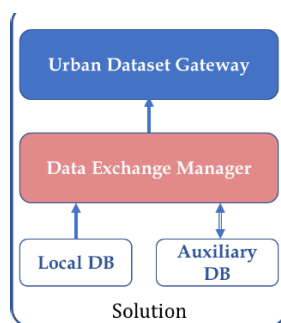


Figura 21 - Ecosistema District Database

Il percorso che i dati energetici effettuano può essere riassunto nei seguenti passi:

- Acquisizione: i dati di interesse nel database di Roma Tre vengono prelevati dal DEM;
- Formattazione: I dati vengono formattati secondo gli standard della SCP;
- Trasmissione: I dati vengono trasmessi all'UDG;
- Storage: I dati vengono memorizzati all'interno del database ausiliario.

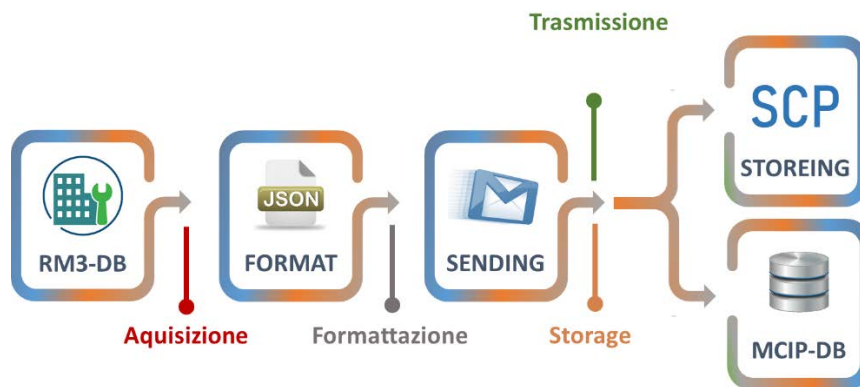


Figura 22 - Flusso logico District Database

L'obiettivo della prossima annualità sarà quello di implementare il processo informatico di invio dei dati energetici alla SCP partendo dalle specifiche e dalle linee guida definite in questa annualità.

Il secondo obiettivo di questa annualità è stato quello di implementare una piattaforma informatica di supporto al processo di *Audit Energetico* volto a raccogliere i dati energetici degli utenti residenziali. L'obiettivo finale è sempre quello di elaborare i dati ottenuti e fornire all'utente dei benchmark di riferimento rispetto ai suoi consumi e dei suggerimenti sugli interventi di riqualificazione energetica che può attuare per migliorare il suo profilo energetico.

In particolare, ci si è occupati di implementare l'interfaccia utente e la struttura di memorizzazione dati che sono stati progettati nella precedente annualità, e verranno ingegnerizzati nella prossima. E' stata quindi realizzata una versione in *beta test* della web application *Audit Energetico* alla quale si può accedere tramite il link http://192.107.92.38/audit/Smart_Sim/AuditBenvenuto.

La piattaforma così progettata presenta una struttura non troppo articolata ma estremamente disaccoppiata presentando una spiccata modularità. Questa caratteristica assicura una buona portabilità e riusabilità della soluzione proposta. Infatti, si può notare che è possibile inserire, modificare, o eliminare qualsiasi sezione del questionario in modo chiaramente poco invasivo, senza pregiudicare la consistenza e la disponibilità dell'audit energetico stesso. In questa annualità sono state effettuate delle scelte progettuali e tecniche leggermente diverse rispetto ai requisiti tecnici definiti durante la precedente annualità. In corso d'opera, infatti, è stato opportuno introdurre alcuni aspetti funzionali che non erano stati tenuti in giusta considerazione.

A valle di quanto riportato, gli obiettivi prestabiliti per questa annualità sono stati raggiunti, fermo restando che su entrambe le progettualità devono essere realizzati numerosi test di robustezza e di utilizzo massivo prima di poter entrare a pieno regime.

5 Riferimenti bibliografici

- [1] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks," in *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE '08)*, Jan. 2008, pp. 791–798, doi: 10.1109/COMSWA.2008.4554519.
- [2] M. B. Yassein, M. Q. Shatnawi, S. Aljwarneh, and R. Al-Hatmi, "Internet of Things: Survey and open issues of MQTT protocol," in *Proceedings - 2017 International Conference on Engineering and MIS, ICEMIS 2017*, Jan. 2018, vol. 2018-January, pp. 1–6, doi: 10.1109/ICEMIS.2017.8273112.
- [3] C. Novelli, "Smart City Platform Specification Functional Level 1.0 - RdS/PAR2017/104," 2017.

- [4] United States Department of Defense, "World Geodetic System 1984: Its Definition and Relationships with Local Geodetic Systems," 2000.
- [5] J. Han, E. Haihong, G. Le, and J. Du, "Survey on NoSQL database," in *Proceedings - 2011 6th International Conference on Pervasive Computing and Applications, ICPCA 2011*, 2011, pp. 363–366, doi: 10.1109/ICPCA.2011.6106531.
- [6] K. Chodorow, *MongoDB : the definitive guide*. .
- [7] M. F. Sanner, "PYTHON: A PROGRAMMING LANGUAGE FOR SOFTWARE INTEGRATION AND DEVELOPMENT." Accessed: Feb. 22, 2021. [Online]. Available: <http://www.python.org/doc/Comparisons.html>.
- [8] J. Melton, "Database Language SQL," in *Handbook on Architectures of Information Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 105–132.
- [9] "EUR-Lex - Document 32016R0679." https://eur-lex.europa.eu/legal-content/IT/TXT/?uri=uriserv:OJ.L_.2016.119.01.0001.01.ITA&toc=OJ:L:2016:119:TOC.
- [10] D. R. Stinson and Paterson Maura, *Cryptography: theory and practice*. CRC press, 2018.
- [11] C. Musciano, B. Kennedy, and G. M. Novak, "HTML: The Definitive Guide, Second Edition," *Computers in Physics*, vol. 12, no. 1, p. 41, 1998, doi: 10.1063/1.168646.
- [12] E. A. Mayer, *CSS: The Definitive Guide*. O'Reilly Media Inc, 2006.
- [13] D. Flanagan and W. S. Like, *JavaScript: The Definitive Guide, 5th*. 2006.

Dario Masucci è un collaboratore di ricerca presso l'Università degli Studi "Roma Tre" dal 2015, anno in cui ha conseguito la laurea magistrale in Automazione. I suoi interessi di ricerca includono lo sviluppo di strumenti di supporto alle decisioni (DSS) incentrati sull'utilizzo di algoritmi di ottimizzazione multi-obiettivo, e soluzioni basate sull'Internet of Things (IoT) e la Big Data Analysis. Negli ultimi anni ha lavorato a diversi progetti europei ideando e realizzando algoritmi decisionali multicriterio, modelli di interdipendenze per le Infrastrutture Critiche (CI), e strumenti ad alto livello di gestione delle emergenze e degli asset coinvolti.