



**MISSION
INNOVATION**

accelerating the clean energy revolution

POA MATERIALI AVANZATI PER L'ENERGIA

**PROGETTO IEMAP - Piattaforma Italiana Accelerata per i Materiali per
l'Energia**

Realizzazione di un prototipo di infrastruttura dati

Sergio Ferlito, Marco Puccini, Claudio Ronchetti,
Simone Giusepponi, Francesco Buonocore, Marialuisa Mongelli,
Massimo Celino



ENEA

D1.5, REALIZZAZIONE DI UN PROTOTIPO DI INFRASTRUTTURA DATI

Sergio Ferlito (ENEA), Marco Puccini (ENEA), Claudio Ronchetti (ENEA), Simone Giusepponi (ENEA),
Francesco Buonocore (ENEA), Marialuisa Mongelli (ENEA), Massimo Celino (ENEA)

Maggio 2023

Report MISSION INNOVATION

Ministero dell'Ambiente e della Sicurezza Energetica - ENEA
Mission Innovation 2021-2024 - II annualità
Progetto: Piattaforma accelerata per i Materiali per l'Energia
Work package: IEMAP: Italian Energy Materials Acceleration Platform
Linea di attività 1.5: Infrastruttura dati (DB ENEA)
Responsabile del Progetto: Massimo Celino, ENEA
Responsabile della LA: Marialuisa Mongelli, ENEA

INDICE

SOMMARIO	4
1 INTRODUZIONE	5
2 REST API	7
3 SERVIZI BACKEND	8
4 STORAGE METADATI (DATABASE MONGODB)	9
5 STORAGE FILE SU FILESYSTEM CEPH	13
6 FRONT-END	15
7 MODULO PYTHON IEMAP	29
8 SVILUPPI FUTURI	30
9 CONCLUSIONI	30
10 BIBLIOGRAFIA	31
11 ABBREVIAZIONI ED ACRONIMI	31

Sommario

Il presente documento riporta i dettagli tecnici dell'intero sistema/architettura software sviluppata nell'ambito del progetto IEMAP (area "Materials for Energy" del progetto "Mission Innovation") per realizzare un "portale web" per le attività di condivisione e collaborazione dei dati tra i partner del progetto. Indicazioni su quali sono le caratteristiche principali che deve avere il sistema software sono riportate nel rapporto D1.8. in cui si trovano gli schemi di base di progettazione per l'architettura del portale e sono descritti i principi base/funzionalità condivisi con i laboratori IEMAP. Qui di seguito sono indicati i dettagli della soluzione realizzata e già disponibile online (iemap.enea.it). Per ogni elemento componente la soluzione sono indicati sia i dettagli tecnici, con alcune note di base esplicative, nonché le immagini catturate a schermo che illustrano l'attuale layout del front-end (sito web).

1 Introduzione

La piattaforma informatica IEMAP è costituita essenzialmente dalle seguenti componenti:

- **REST API.** Ha lo scopo principale di consentire il trasferimento dei contenuti dinamici tra il front-end, ovvero l'interfaccia web, ed il server che fornisce/ospita il sito stesso. Inoltre, costituiscono l'elemento principale per le operazioni CRUD (Create, Read, Update e Delete) verso il database.
- **Servizi Backend.** Con tale termine s'intendono le funzionalità messe a disposizione dal server che non sono direttamente/immediatamente visibili all'utente finale, o più genericamente "client", dei servizi IEMAP. Tale elemento consta in una macchina virtuale VMware ospitata sui cluster CRESCO. Tali servizi includono: NGINX, un "reverse proxy server" che rende accessibile verso l'esterno i servizi REST API ed il front-end, un container Docker tramite il quale è messo a disposizione il servizio REST API, il servizio Fail2Ban per la protezione del server (connessioni ssh, NGINX, etc.)
- **Storage Metadati.** Il database MongoDB, in configurazione "sharding", disponibile presso il cluster CRESCO in ENEA C. R. Portici, ospita due "collezioni" per l'archiviazione rispettivamente delle credenziali utente e dei metadati dei progetti archiviati in piattaforma. La scelta di MongoDB come database per IEMAP è stata discussa nel rapporto tecnico D1.4.
- **Storage file.** L'archiviazione dei file di progetto avviene tramite file system CephFS. I file che l'utente IEMAP può caricare sulla piattaforma sono gestiti tramite le REST API che consentono sia l'upload che il download degli stessi verso uno storage remoto, ospitato dai server presso il Centro ENEA di Brindisi, tramite una soluzione di archiviazione CEPH.
- **Front-end,** ovvero sito web. L'interfaccia web sviluppata per IEMAP è accessibile al link <https://iemap.enea.it>. Si tratta di una applicazione SPA (Single Page Application) esposta tramite NGINX
- **Modul Python IEMAP.** Tramite il noto gestore di moduli Python *pip* è disponibile pubblicamente, all'indirizzo, <https://pypi.org/project/iemap/>, un modulo Python che consente di utilizzare i servizi IEMAP in maniera programmatica, ovvero senza dover passare attraverso l'interfaccia web. È questa sicuramente la modalità più efficiente e semplice di usufruire dei servizi IEMAP, che consente una semplice integrazione nel proprio flusso di lavoro. Ovviamente una conoscenza basilare di Python costituisce un requisito necessario.

Di seguito è riportato uno schema esplicativo dell'intera architettura mentre nelle sezioni seguenti sono riportati i dettagli di ciascun elemento sopra riportato.

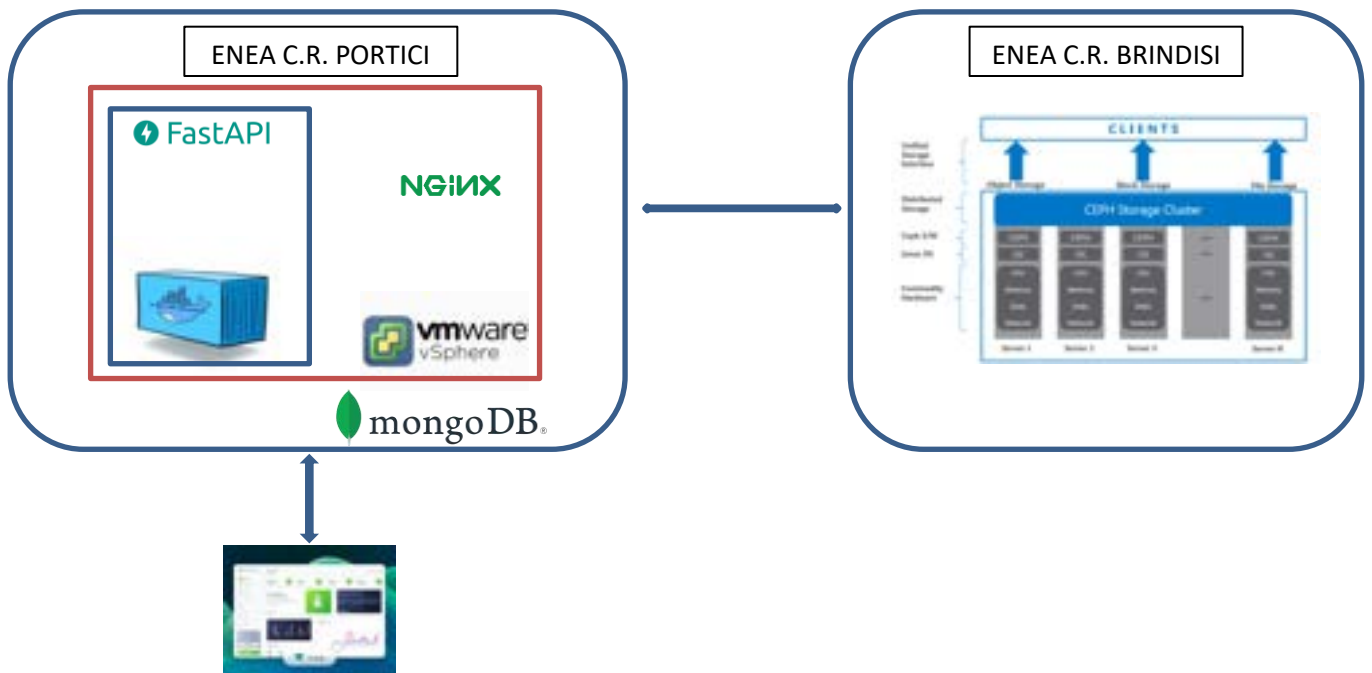


Fig. 1 Schema esplicativo dell'architettura della piattaforma IEMAP.

2 REST API

Con il termine REST API s'intende un'interfaccia di programmazione delle applicazioni, Application Programming Interface (API), conforme ai vincoli dello stile architetturale REST che consente l'interazione con servizi web RESTful. Il termine REST, coniato dall'informatico Roy Fielding, è l'acronimo di REpresentational State Transfer. Come detto REST non è un protocollo e neppure uno standard. Quando una richiesta client viene inviata tramite un'API RESTful, questa trasferisce al richiedente o all'endpoint uno stato rappresentativo della risorsa. L'informazione, o rappresentazione, viene consegnata in uno dei diversi formati tramite HTTP: JSON (Javascript Object Notation), TXT o XML. Pertanto, il servizio di REST API consente di trasferire in maniera programmatica un contenuto dal server (o backend) verso i client, mentre la forma con cui sono descritti tali informazioni/contenuti è tipicamente JSON. Il protocollo di comunicazione utilizzato è invece http e nel caso specifico https. Una richiesta REST API si realizza richiedendo una specifica URL o URI (Uniform Resource Identifier), anche detto endpoint, specificando nella richiesta il metodo o verbo con eventualmente informazioni aggiuntive, ad esempio i dati da inserire/trasferire al server. Nell'ambito del progetto IEMAP il servizio REST API costituisce l'elemento fondamentale dell'intera infrastruttura in quanto svolge tutte le principali funzionalità messe a disposizione.

Tramite le REST API è possibile:

- Effettuare tutte le operazioni d'interazione con il Database MongoDB, ovvero le operazioni CRUD in precedenza accennate.
- Autenticare gli utenti, restringendo alcune operazioni, come l'upload di file e metadati, ai soli utenti autenticati. Tale funzionalità, trattandosi di richieste client-server prive di stato (stateless) avviene secondo un meccanismo basato su un particolare JSON inserito nell'intestazione (header) delle richieste che prende il nome di JWT (JSON WEB TOKEN).
- Effettuare l'upload e il download di file. Tali file sono trasferiti grazie al servizio REST API con specifici endpoint con accesso autenticato (tramite JWT) e sono memorizzati su file system remoto CEPH. I file al momento dell'upload vengono momentaneamente memorizzati con nome ed estensione originale, si calcola l'Hash* del file temporaneo, in modo da ottenerne un identificativo univoco, dopodiché nel Database è memorizzato sia l'Hash che il nome originale del file con estensione ed alcuni dati sintetici come le dimensioni. Infine, il file viene rinominato con l'hash (+ estensione), calcolato dal sistema e memorizzato sul file system.

** L'impronta hash di un testo o di un file informatico è una sequenza di lettere e cifre, lunga solitamente 64 caratteri, ottenuta applicando un particolare algoritmo di calcolo alla sequenza di bit che formano il testo o il file.*

Il servizio REST API è stato realizzato in Python, in particolare si è utilizzato il framework FASTAPI, si tratta di un "web framework" in Python per realizzare servizi API basati su standard.

Principali caratteristiche di FASTAPI sono:

- Considerato uno dei framework più veloci in Python. Compatibile con ASGI per la scrittura di codice asincrono (richieste e risposte asincrone)
- Ampia e semplice documentazione è semplice
- La codifica con il supporto del suggerimento dei tipi (Python type hints) aiuta a ridurre circa il 40% dei bug indotti.
- Compatibile con standard aperti per API e schema JSON.
- Documentazione API interattiva e automatica tramite Swagger.

Come detto, il sistema di autenticazione è integrato nelle REST API tramite un modulo Python aggiuntivo FASTAPI-USERS, di seguito alcune funzionalità e/o caratteristiche messe a disposizione da FASTAPI-Users:

- Modello utente di base estensibile
- Endpoint per le funzionalità basilari come: registrati, accedi, reimposta la password e verifica account tramite email pronti all'uso
- Supporto per autenticazione social OAuth2
- Supporto per differenti database per lo storage delle credenziali (SQLAlchemy ORM e MongoDB tramite Beanie ODM)
- Molteplici back-end di autenticazione personalizzabili
 - Trasporti: autorizzazione tramite intestazione richiesta oppure cookie (HTTP-Only Cookie)
 - Strategie: JWT, Database, Redis
- Supporto completo dello schema OpenAPI, anche con diversi backend di autenticazione

Nell'ambito di IEMAP si è optato per una strategia di autenticazione basata su JWT con duplice opzione di trasporto, nell'intestazione delle richieste per ogni endpoint protetto richiamato tramite modulo python iemap o accesso diretto REST API e cookie (HTT-Only cookie) per l'accesso autenticato tramite interfaccia web. Non si esclude in un prossimo futuro l'integrazione di Oauth2.

Le gestione delle operazioni CRUD relative ai metadati avviene tramite modulo Python Motor, si tratta di driver MongoDB asincrono per applicazioni Python.

3 Servizi Backend

I servizi backend, come accennato nell'introduzione, sono resi disponibile tramite una macchina virtuale VMware, che ospita un sistema operativo Ubuntu 22.04 LTS, all'interno della infrastruttura CRESCO.

Il servizio principale, quello delle REST API, dettagliato nel precedente paragrafo è reso disponibile all'interno di un container Docker che consente un rapido e sicuro "deployment" del servizio REST API oltre a costituire l'elemento base per una futura soluzione scalabile basata su Kubernetes. Nell'ottica di una gestione tipo CI/CD (Continuous Integration/Continuous Development), ovvero l'integrazione e il deployment continui con tempi di inattività pari a zero, lo sviluppo avviene come segue:

- Sviluppo codice su desktop con integrazione GIT/GITHUB

- “PUSH” codice aggiornato su branch remoto GITHUB
- Aggiornamento del container Docker tramite script da shell che si occupa di fare il “PULL” da GitHub delle modifiche con “rebuild” dell’immagine Docker e avvio nuovo container

Nel prosieguo delle attività si conta di utilizzare GitHub Action per ulteriormente automatizzare l’intero workflow.

Il servizio REST API è reso disponibile verso l’esterno tramite NGINX (nella sua versione open-source), pronunciato “engine-ex”. Si tratta di un “reverse proxy server” con funzionalità di streaming e bilanciamento del carico che consente di esporre il servizio REST API messo a disposizione nel container Docker tramite l’indirizzo IP pubblico registrato su DNS dell’host. NGINX espone le REST API tramite https (connessione TLS) con certificato digitale rilasciato da LetsEncrypt e rinnovato automaticamente da certbot (script Python integrato con NGINX). In tal modo NGINX provvede all’inoltro del traffico https, recepito all’IP pubblico, verso il protocollo http del servizio REST API messo a disposizione da FASTAPI + Uvicorn. Sempre NGINX si occupa di rendere disponibile il front-end, si tratta, come detto, di una applicazione SPA (Single Page Application). Tale applicazione SPA funziona come segue: alla prima richiesta del client, il server, tramite NGINX, fornisce la pagina index.html che contiene tutto il codice Javascript per la navigazione (anche detta routing) da una pagina ad un’altra che avviene interamente sul client (nel browser). Per una maggiore protezione dei servizi resi disponibili dal server è stato installato e configurato anche il servizio Fail2Ban che, monitorando opportuni file di log, può bloccare il traffico proveniente da determinati indirizzi IP in base a “policy” configurabili.

4 Storage Metadati (database MongoDB)

I dati relativi ad ogni progetto, sia esso di calcolo o esperimento/simulazione, da qui in poi definiti come metadati, sono memorizzati in un database MongoDB (nella versione Community Edition) installato su cluster CRESCO tramite configurazione di “sharding”. L’accesso al DB avviene unicamente tramite REST API, pertanto il DB non è esposto direttamente all’esterno, è il servizio REST API che si connette al DB con un account con autorizzazioni opportunamente “ridotte” e svolge tutte le operazioni CRUD.

Anche i dati relativi agli account degli utenti sono memorizzati su tale database. Tali account sono gestiti dal modulo Python FASTAPI-USERS tramite ODM (Object Document Modeller) Beanie e sempre tramite REST API.

MongoDB è un database NoSQL disponibile in tre soluzioni:

1. Community Edition; si tratta della versione Open Source utilizzata in IEMAP
2. MongoDB ATLAS; è la versione Cloud di MongoDB, si tratta di una soluzione DAAS (Database As A Service) che costituisce la versione più completa ed immediatamente/facilmente implementabile
3. Enterprise Edition; la versione On-Premise, dotata di tutte le funzionalità Mission-Critical quali supporto protocollo autenticazione Kerberos, OPS Manager, etc., sebbene non sia una soluzione completa come ATLAS

MongoDB è un database non relazionale, in grado di elaborare dati strutturati, semi-strutturati e non strutturati. Utilizza un modello di dati non relazionale e orientato ai documenti e un linguaggio di query non strutturato che prende il nome di MQL (MongoDB Query Language), il cui scopo è essenzialmente di operare le principali funzionalità CRUD, a cui si aggiunge un modello di query basato su “pipeline” che prende il nome di Aggregation Framework per le funzionalità di Data Analytics.

MongoDB è altamente flessibile e consente di combinare e memorizzare più tipi di dati. Inoltre, memorizza e gestisce quantità di dati maggiori rispetto ai database relazionali tradizionali con eccellenti performance in particolare in scrittura. MongoDB utilizza un formato di storage dei documenti definito BSON, che è una forma binaria di JSON (JavaScript Object Notation) in grado di ospitare più tipi di dati i cui tipi di dati base va ad estendere quelli supportati dal semplice JSON. L'ambiente di query, MongoDB CLI, è un vero e proprio interprete Javascript, mentre l'interfaccia grafica MongoDB Compass consente di eseguire e validare pipeline scritte tramite Aggregation Framework, creare indici e validare lo schema dei dati.

In IEMAP si sono utilizzate due collezioni dati (la collezione è l'unità base di memorizzazione dei dati che corrisponde alla tabella in un DB relazionale) che sono:

1. UserAuth, per la gestione degli account, gestita tramite l'Object Document Modeller (ODM) Beanie supportato da FASTAPI-USERS
2. Iemap, per la memorizzazione dei metadati dei progetti archiviati in IEMAP

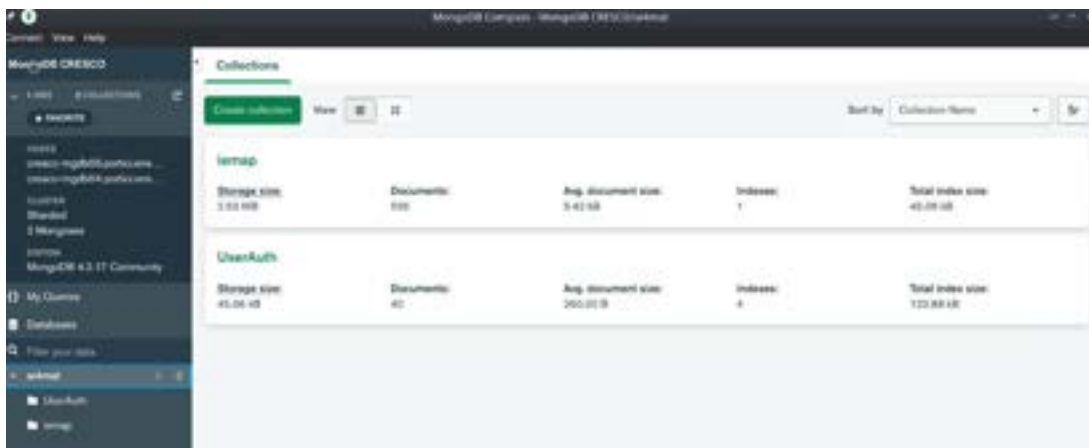


Fig. 2 Dettaglio collezioni MongoDB per IEMAP come appaiono nell'UI MongoDB Compass

Nell'immagine seguente (fig. 3) è riportato un esempio di dato archiviato in MongoDB (collezione iemap nel DA ai4mat creato per il progetto IEMAP).

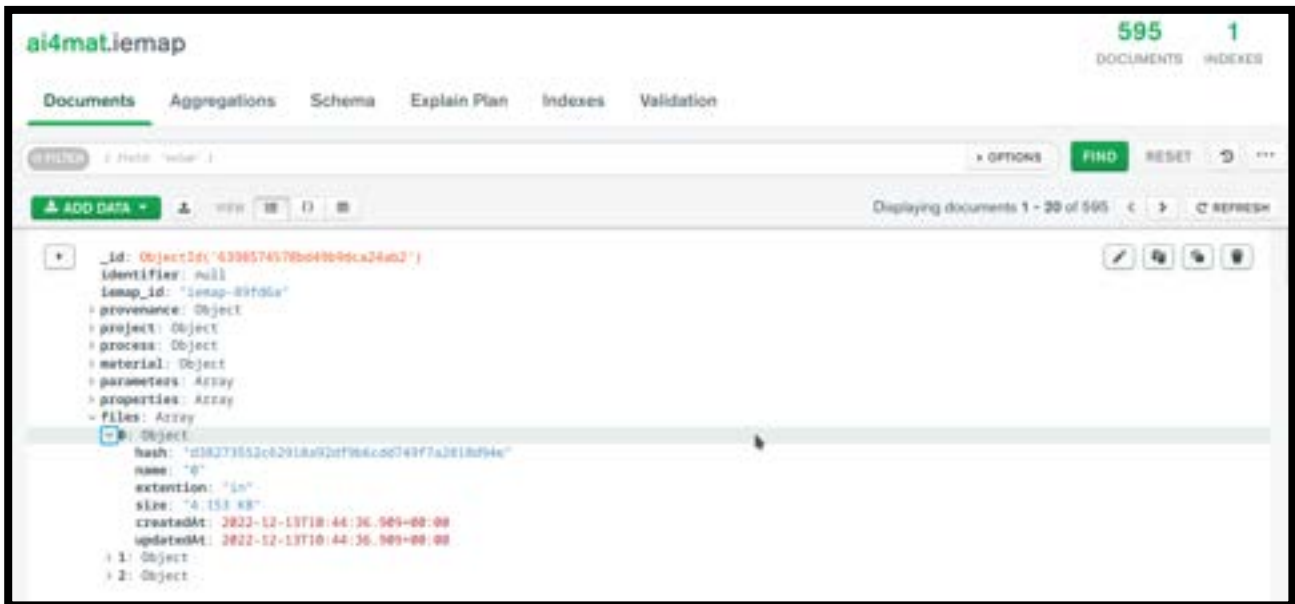


Fig.3. Dettaglio singolo documento MongoDB nella collezione per i metadati IEMAP

Con riferimento all'immagine riportata sopra, si evidenziano i seguenti elementi:

- Il singolo documento (l'equivalente del record in una tabella per DB relazionali) ha uno schema tipo JSON, come detto è un BSON, con possibilità di sotto-documenti (indicati con Object in fig. 3) e/o array di sotto-documenti (Array in fig. 3)
- Alcuni campi sono inseriti in automatico dal sistema, ad esempio *_id* per definire in maniera univoca il documento e utilizzato a fini "interni" da MongoDB, oppure *iemap_id* utilizzabile in fase di query ed assegnato automaticamente al momento dell'inserimento dei metadati del progetto
- Ogni progetto contiene uno o più file associati ed è per questo motivo presente tramite un *Array di Oggetti* dove il singolo oggetto contiene:
 - ◆ L'HASH del file caricato
 - ◆ Il nome del file
 - ◆ L'estensione del file
 - ◆ La dimensione del file
 - ◆ La data d'inserimento o modifica del file

Altri elementi in merito alla struttura dati verranno riportati nella sezione "front-end" più avanti nel documento corrente.

MongoDB offre i seguenti principali vantaggi:

1. **Flessibilità.** MongoDB ha un'architettura a schema dinamico che funziona con dati e storage non strutturati. Poiché i dati sono memorizzati in documenti flessibili simili a JSON, lo schema del database non deve essere predefinito e gli schemi possono essere modificati in modo dinamico senza causare interruzioni del servizio e/o inefficienze nello storage del dato. Con il formato di dati BSON di MongoDB, gli oggetti di una "collezione" possono avere diversi insiemi di campi e quasi ogni tipo di struttura di dati può essere modellata e manipolata. Per questo motivo, il modello di database

flessibile di MongoDB è particolarmente vantaggioso quando i requisiti di business e di dati cambiano nel tempo.

2. **Partizionamento.** MongoDB offre scalabilità orizzontale tramite un processo denominato partizionamento (sharding). Il partizionamento divide i dati da un set di dati di grandi dimensioni e li distribuisce su più server. Se un server non è in grado di gestire un carico elevato, i dati possono essere divisi e distribuiti automaticamente senza che la loro elaborazione venga interrotta.
3. **Ridondanza.** Oltre alla scalabilità orizzontale tramite sharding, è possibile definire uno schema di replica dei dati, detto replica-set, dove più istanze MongoDB, tipicamente su server separati, memorizzano gli stessi dati per garantire un'alta affidabilità del servizio (se viene meno un server l'accesso ai dati non si interrompe essendo questo automaticamente "rimpiazzato" da un altro server all'interno del replica-set)
4. **Performance elevate.** MongoDB memorizza i dati nella RAM per accelerarne l'accesso e ottimizzare le performance durante l'esecuzione delle query. Raccoglie i dati direttamente dalla RAM anziché dal disco rigido, rendendo più veloci le letture e le scritture dei dati. La struttura non relazionale dei dati di MongoDB richiede inoltre una minore potenza di elaborazione per la ricerca e il recupero dei dati rispetto a un database relazionale.
5. **Analytics in tempo reale.** In quanto database NoSQL, MongoDB è una scelta indicata per l'integrazione e l'elaborazione dei Big Data (che sono enormi quantità di dati eterogenei, che per le loro dimensioni non possono essere elaborate dai database relazionali tradizionali). Poiché MongoDB ha uno schema flessibile, è possibile effettuare l'accesso e la memorizzazione di vari tipi di dati all'istante. Il supporto integrato di MongoDB per il partizionamento consente inoltre di scalare i dati orizzontalmente su più server. Inoltre, offre la flessibilità necessaria per unire centinaia di origini di dati in un'unica vista per gli "analytics" in tempo reale e l'integrazione dei dati, grazie anche a funzionalità di creazione di collezioni con valori aggregati fornita dai Change Streams, funzionalità questa molto utile in scenari di analisi dati da sensori e/o esperimenti.
6. **Gestione dei contenuti.** In virtù del suo modello documentale non strutturato, MongoDB è un'opzione eccellente per la gestione dei contenuti e la distribuzione di siti web di e-commerce, pubblicazioni online e sistemi di gestione dei contenuti web. Il suo modello di dati flessibile consente di memorizzare facilmente diversi tipi di contenuti, tra cui immagini, testi e video, oltre ai metadati. Tutti i contenuti correlati sono memorizzati in un unico documento, il che facilita l'aggiunta di nuove funzionalità e attributi e spesso riduce i tempi di risposta per le query sui dati. MongoDB può anche essere utilizzato per archiviare contenuti generati dagli utenti, come i commenti, che possono essere analizzati e utilizzati per orientare lo sviluppo di contenuti futuri.

Ovviamente MongoDB è una delle N possibili soluzioni per l'archiviazione dati e come tutte offre vantaggi e svantaggi. Di seguito a titolo informativo si elencano altre soluzioni di storage maggiormente idonee per alcuni contesti specifici che potrebbero anche essere integrate con MongoDB qualora in un prossimo futuro dovessero emergere necessità non soddisfabili in maniera ottimale da MongoDB:

- **InfluxDB**, si tratta di un DB specifico per serie temporali come quelle originate da letture di sensori
- **Elasticsearch**, è un motore di ricerca e analisi distribuito e open source particolarmente idoneo per ricerche testuali

- **Apache Spark (Spark)** è un motore di elaborazione dati open source per grandi dataset. È progettato per offrire la velocità di elaborazione, la scalabilità e la programmabilità richieste per i Big Data - specificamente per dati in streaming, dati grafici, machine learning e applicazioni AI.

La figura seguente riassume sinteticamente alcune delle caratteristiche fondamentali di MongoDB di cui si è parlato sopra.



Fig.4 Schema caratteristiche MongoDB

5 Storage file su filesystem Ceph

Ogni utente registrato può inserire uno o più file per ogni singolo progetto, tali file sono trattati come segue:

- Il file viene caricato tramite REST API secondo due possibili opzioni:
 - Interfaccia web
 - Modulo Python iemap
- Il file viene temporaneamente salvato su filesystem, se l'operazione va a buon fine il back-end calcola l'hash del file
- Le informazioni: hash del file + estensione, con informazioni quali data creazione e dimensione file, sono aggiunte ai metadati già presenti nel DB
- Il file viene rinominato con l'hash precedentemente calcolato mantenendone l'estensione

L'operazione di associare uno o più file ad un dato progetto inserito in IEMAP è possibile solamente dopo aver caricati i metadati che descrivono il progetto stesso.

Lo storage effettivo dei file avviene su di un cluster di storage **Ceph** (*Introduction to Ceph - Better Tomorrow with Computer Science*, n.d.).

Ceph è una **soluzione di archiviazione completa**, che ha il proprio file system CephFS. Con Ceph è possibile archiviare dati su vari dispositivi nella propria rete. Inoltre, è possibile eseguire il backup dei dati più volte in aree di archiviazione fisicamente diverse. Ceph offre una grande flessibilità nella scelta dei dispositivi di archiviazione e un'elevata scalabilità.

Ceph funziona **solo su sistemi Linux**. Il supporto è fornito, ad esempio, da CentOS, Debian, Fedora, RedHat/RHEL, OpenSUSE e Ubuntu. L'accesso dai sistemi Windows non è diretto, ma è possibile utilizzando iSCSI (Internet Small Computer System Interface). Ceph è quindi particolarmente adatto per l'uso in data center, che rendono disponibile il proprio spazio di archiviazione tramite servizi server, nonché per soluzioni cloud di tutti i tipi, che utilizzano software per fornire memoria.

Di seguito un riepilogo delle caratteristiche salienti di Ceph:

- Open source
- Elevata scalabilità
- Sicurezza dei dati tramite archiviazione dati ridondante
- Affidabilità grazie all'archiviazione distribuita dei dati
- Aumento della disponibilità, basato su software, tramite algoritmo interno per la ricerca dei dati
- Rappresentazione come memoria contigua
- Bassi requisiti hardware

Per i dettagli relativi al funzionamento di Ceph si rimanda a (*Ceph: L'archiviazione Pratica per Tutte Le Aziende - IONOS*, n.d.)

In IEMAP si è utilizzato il cluster Ceph ospitato presso il centro ENEA di Brindisi, per i dettagli del cluster vedi (Puccini et al., n.d.)

La seguente figura sintetizza la geolocalizzazione dei servizi IEMAP, ovvero:

- Servizi Backend in macchina virtuale VMware nel VSphere presso CRESCO nel C.R. Portici
- Sharding MongoDB presso CRESCO nel C.R. Portici
- Storage dei file associati ai progetti archiviati in IEMAP presso il cluster Ceph nel C.R. Brindisi

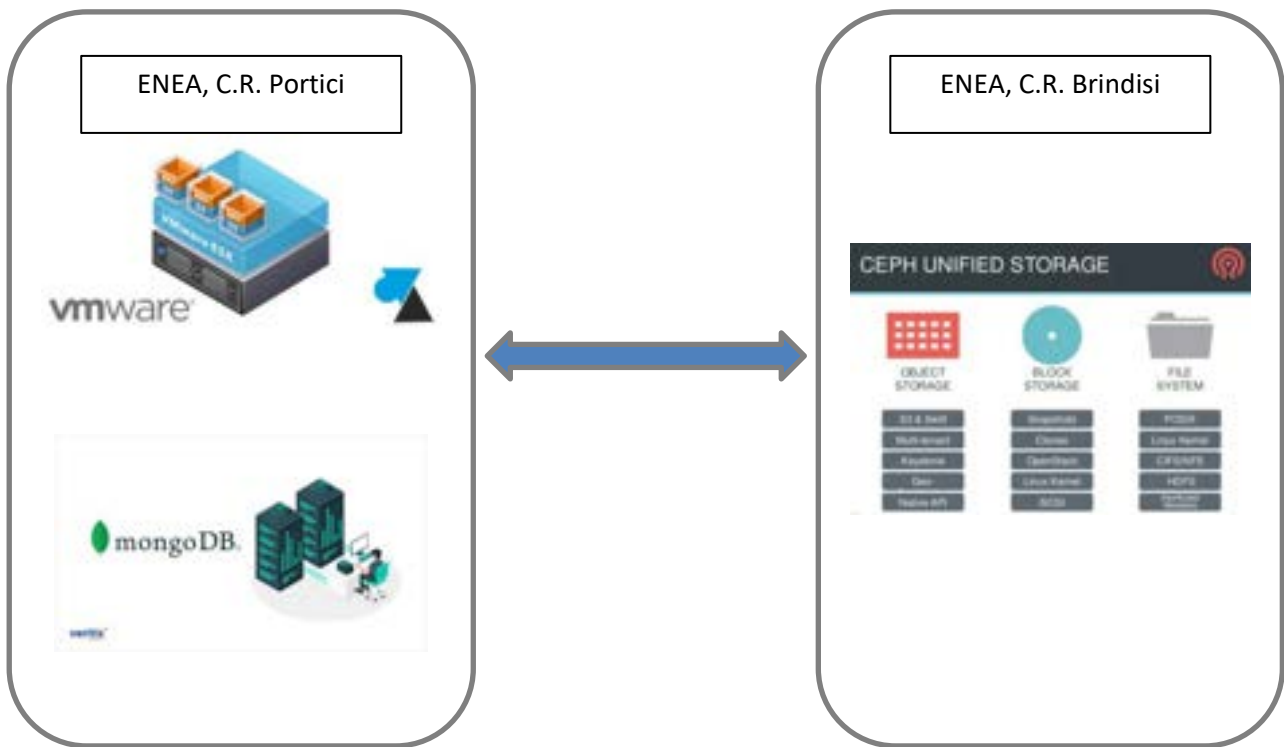


Fig.5 Schema dislocazione dei servizi IEMAP tra i centri ENEA di Portici e Brindisi

6 Front-End

Con in termine front-end s'intende l'interfaccia web (sito web) realizzato per usufruire di tutti i servizi messi a disposizione in IEMAP tramite un'interfaccia grafica "user-friendly" fruibile tramite un qualsiasi browser.

Si tratta di una applicazione web sviluppata tramite il framework Javascript VUEJS v.3.2 con una serie di moduli Javascript aggiuntivi per internazionalizzazione, validazione dati, etc..

La peculiarità di tale applicazione web (front-end) sta nell'essere una applicazione che, come detto, si definisce Single Page Application (SPA) e consente di realizzare applicazioni web performanti poiché la navigazione avviene esclusivamente client-side mentre l'interazione con il backend e la gestione di tutti gli elementi/dati dinamici è realizzata tramite richieste REST API che inviano contenuto JSON.

Il processo è schematizzato nelle due figure seguenti.



Fig.6 Schema flusso richieste al caricamento della pagina iniziale del sito web



Fig.7 Schema flusso richieste per navigazione tra pagine dello stesso sito

Il front-end IEMAP è raggiungibile all'indirizzo

<https://iemap.enea.it>

Alla prima richiesta da parte del client alla "url" su indicata il server, nel caso di IEMAP si tratta del servizio NGINX, risponde fornendo il file *index.html* (punto 2 nella fig. 6), da questo punto in poi il framework VUEJS si occupa di effettuare il rendering della pagina stessa. Nelle successive richieste (punto 4 fig. 6), ad esempio passando da una pagina a un'altra del sito, non si richiede più alcuna pagina a server web. Come detto, nel caso specifico, è NGINX che agisce da reverse proxy server e fornisce il contenuto statico *index.html* con tutti il codice Javascript suddiviso in "chunks", ovvero blocchi, per ottimizzare il tempo del rendering iniziale della pagina (oltre a contenuti allegati correlati quali immagini, file css/sass, etc.). In sostanza la navigazione da una pagina ad un'altra avviene tutto client-side, senza più interazioni con il server remoto, grazie a VUEJS che gestisce il rendering dei vari elementi visualizzati nel client. Il punto 4 nella fig. 6 sopra è relativo al solo contenuto dinamico e/o dati da DB o filesystem ed è realizzato tramite REST API ed invio di contenuto JSON e/o stream di dati (upload/download file).

Nella fig. 6 i servizi SPA file server e API server, sono separati ma nel caso specifico sono entrambi forniti dalla stessa macchina virtuale VMware con Ubuntu Linux 22.04; le funzionalità di SPA file server sono fornite da

NGINX mentre quelle di API server dal container Docker con il servizio REST API sviluppato tramite FASTAPI (container esposto all'esterno sempre tramite NGINX). La figura che illustra le interazioni successive con il sever, fig. 7, sta and indicare che la navigazione (detta "routing") avviene tutta client-side (nel browser) e solo i "dati" (API request/response), in formato JSON, sono trasferiti/richiesti al server.

La pagina iniziale del sito web IEMAP appare come illustrato nelle due pagine seguenti.

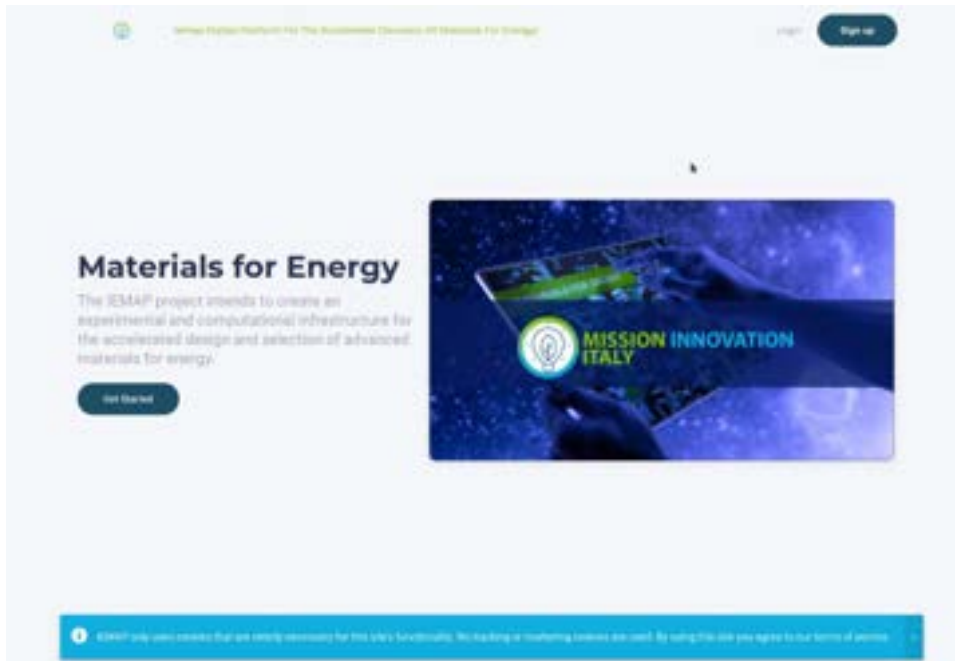


Fig.8 Home page IEMAP (<https://iemap.enea.it>), porzione superiore



Fig.8 Home page IEMAP, porzione inferiore

Nella “home page” è data una precisa contestualità in merito al progetto IEMAP e le sue finalità con link a Mission Innovation, ai sotto progetti facente parte di IEMAP, oltre al disclaimer per i cookie utilizzati nel sito web (nessun cookie di profilazione ma unicamente per preferenze utente o per consentire l’autenticazione tramite JWT memorizzato in un HTTP-Only cookie).

Da tale pagina iniziale è possibile, essere indirizzati al repository per il modulo sviluppato in Python (link “*install IEMAP Python module*”) oppure accedere all’area riservata o alla sezione di registrazione tramite la barra presente in alto (pulsanti “*Login*”/“*Sign up*”).

Con il pulsante “*Get started*” in home page si va alla pagina seguente (“*routing*” client side e nessun “*round-trip*” verso il server) riportata nella fig. 9 seguente.

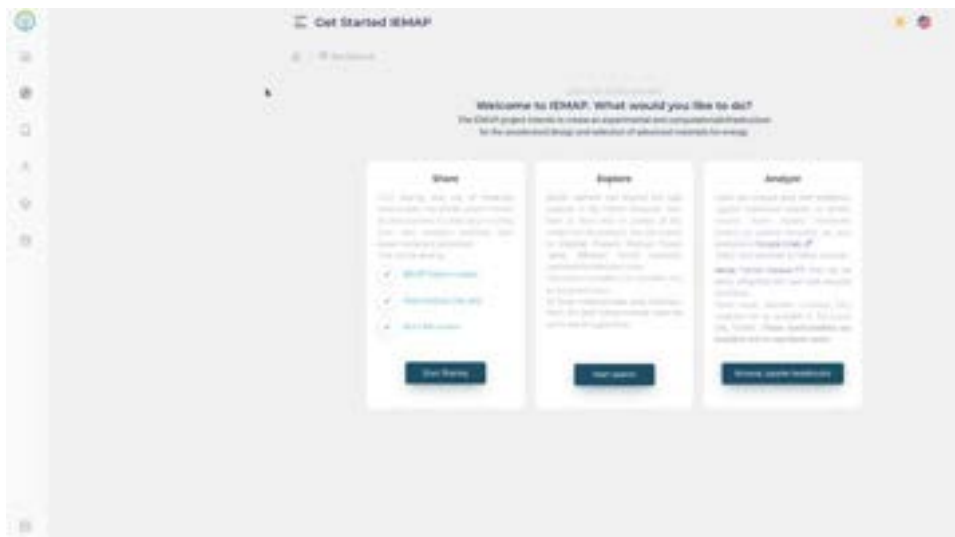


Fig.8 Get Started page IEMAP (<https://iemap.enea.it/getstarted>)

In tale pagina sono sintetizzati i servizi forniti dal “portale” IEMAP:

- 1 **Share**, ovvero la condivisione di metadati relativi ai vari progetti e/o file ad essi associati. Tale funzionalità, come dettagliato, è possibile secondo tre modalità che sono: interfaccia web, servizio diretto agli endpoint del servizio REST API, modulo Python iemap.
- 2 **Explore**, indirizza l’utente alla sezione per le “query” sui metadati.
- 3 **Analyze**, tale sezione non è al momento attiva e consentirà, in una prossima revisione, all’utente di accedere ad un’area con il servizio JupyterHub per l’esecuzione online (ovvero sempre all’interno del browser) di codice Python tramite dei notebook Python pre-condivisi.

Da notare che, tranne che nella pagina iniziale, in tutte le pagine successive del sito IEMAP vi è la barra di navigazione laterale che consente di navigare verso le seguenti sezioni (riportate di seguito come in fig.8 dal primo elemento di navigazione in alto all’ultimo in basso):

- Home
- Get started
- IEMAP documentation
- Login page

- Show user projects
- IEMAP research data
- IEMAP General architecture

Ognuna delle voci di navigazione su elencate è identificata da un'icona con un "tooltip" che compare allorquando si passa su con il mouse e che mostra la dicitura completa come su riportato.

Inoltre, è possibile cambiare lingua di visualizzazione tramite il pulsante in alto a destra nelle pagine che indica una bandiera Americana/Italiana per la lingua Inglese (scelta di default) o Italiano. Tale funzionalità è predisposta ma solamente in minima parte implementata, cosa che verrà completata nel prosieguo del progetto. Nel prosieguo si illustreranno tutte le sezioni pubbliche del sito, ovvero consultabili senza alcuna registrazione (come le due sin qui illustrate) per poi fornire qualche nota ulteriore di dettaglio sul sistema di autenticazione e passare ad esaminare le sezioni "protette" del sito, ovvero quelle consultabili solo previa registrazione di un account con validazione dello stesso tramite mail.

La pagina

IEMAP documentation (<https://iemap.enea.it/documentation>)

Riporta, per riquadri sintetici, tutto quanto è necessario all'utilizzo del portale IEMAP e consente anche l'invio di una mail ad un account specifico di supporto (il link apre automaticamente il client di posta con destinatario già inserito, ovvero iemap.support@enea.it) per eventuali chiarimenti o generiche richieste di supporto.

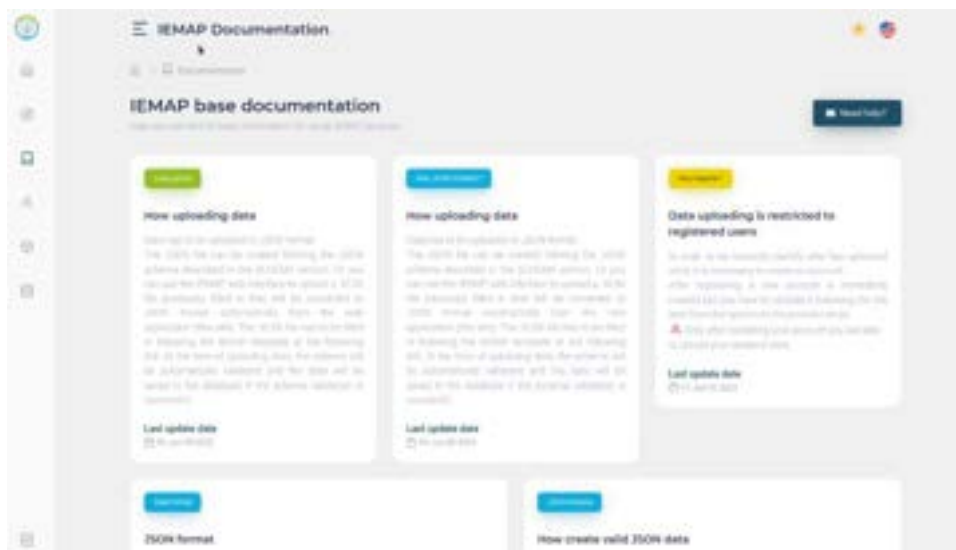


Fig.9 IEMAP documentation , porzione superiore, (<https://iemap.enea.it/documentation>)

In tale pagina sono riportate le informazioni necessarie per il corretto inserimento dei metadati di progetto tramite formato JSON (vedi figura sotto).

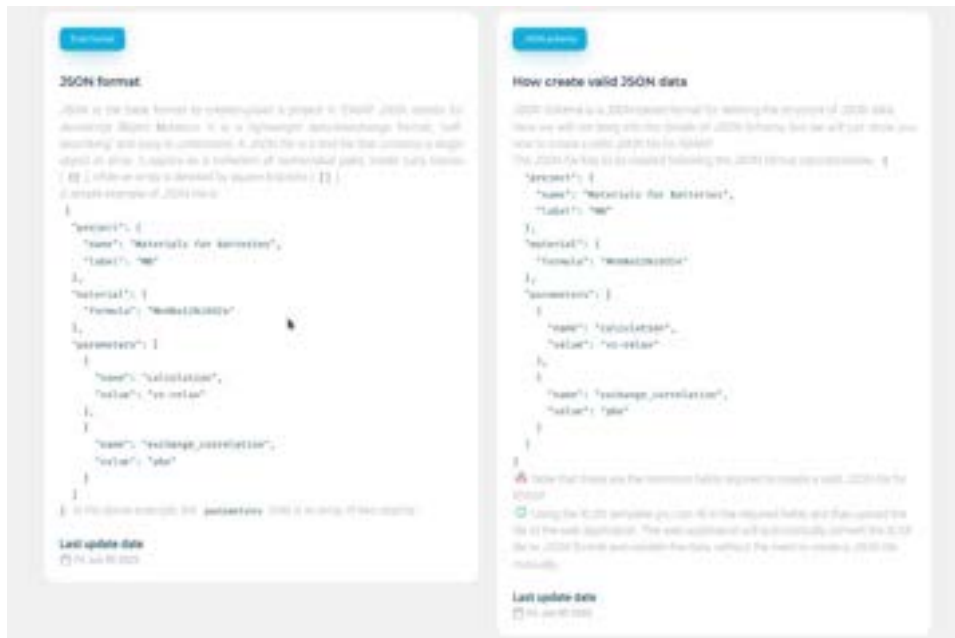


Fig.10 IEMAP documentation, porzione inferiore, (<https://iemap.enea.it/documentation>)

I metadati di progetto sono salvati su database MongoDB e trasferiti dal client al server in formato JSON.

Per le operazioni CRUD, vi sono degli endpoint “protetti”, accessibili tramite autenticazione, che ricevono i metadati in formato JSON, tali metadati sono validati sia in termini di correttezza del formato JSON che in termini di requisiti di schema JSON (presenza obbligatori ad alcuni campi, tipologia dato ammesso, etc.).

I metadati, come già indicato in precedenza, sono inseribili tramite tre modalità:

- REST API (a tal fine può risultare molto utile un client REST API quali Postman)
- Modulo Python iemap
- Interfaccia web. Nello specifico è possibile sia inserire direttamente un JSON con lo schema valido oppure compilare un semplice template .xlsx. In quest’ultimo caso sarà l’applicazione web (VUEJS) che trasformerà il contenuto Excel in un documento JSON che se supererà le validazioni indicate sarà inserito nel DB tramite l’utilizzo delle REST API

La pagina

IEMAP Research data (<https://iemap.enea.it/iemap/data>)

Consente a chiunque di accedere ai dati presenti nella piattaforma IEMAP e realizzare alcune query per filtrare i risultati disponibili.

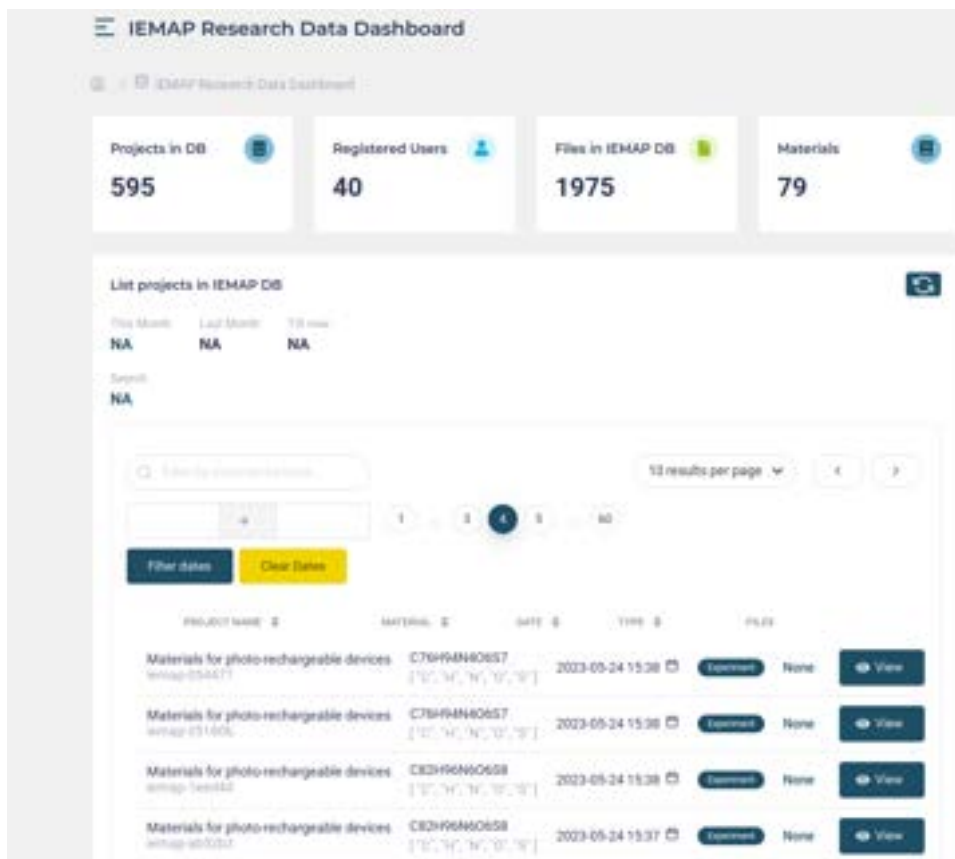


Fig.11 IEMAP Research data (<https://iemap.enea.it/iemap/data>)

Come si evince dalla figura su riportata sono indicati alcuni dati statistici compressivi quali:

- Numero totale progetti inseriti
- Numero utenti registrati
- Numero totale file caricati in piattaforma
- Numero totale materiali differenti tra tutti i progetti disponibili

Le funzionalità di query consentono di filtrare i dati per:

- Data inizio e fine
- Materiale

oltre a riportare i dati tramite pagine di lunghezza variabile dove per ogni progetto si indicano gli elementi di maggior rilievo.

Tramite il pulsante “View” è possibile mostrare ulteriori dettagli e, qualora presenti, scaricare i file collegati al progetto selezionato.

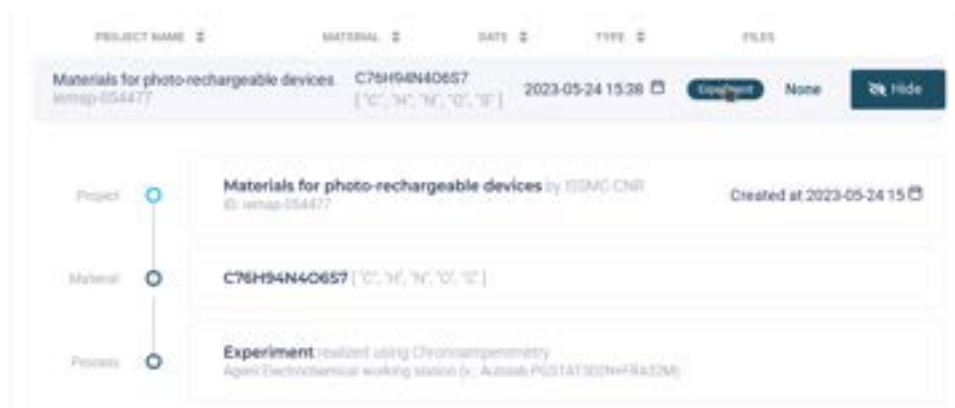


Fig.12 Dettagli singolo progetto in IEMAP Research data (<https://iemap.enea.it/iemap/data>)

La pagina

IEMAP project Architecture (<https://iemap.enea.it/architecture>)

Mostra in forma sintetica gli elementi costituenti l'architettura oggetto di questo documento, indicando anche link di approfondimento per ogni elemento descritto.

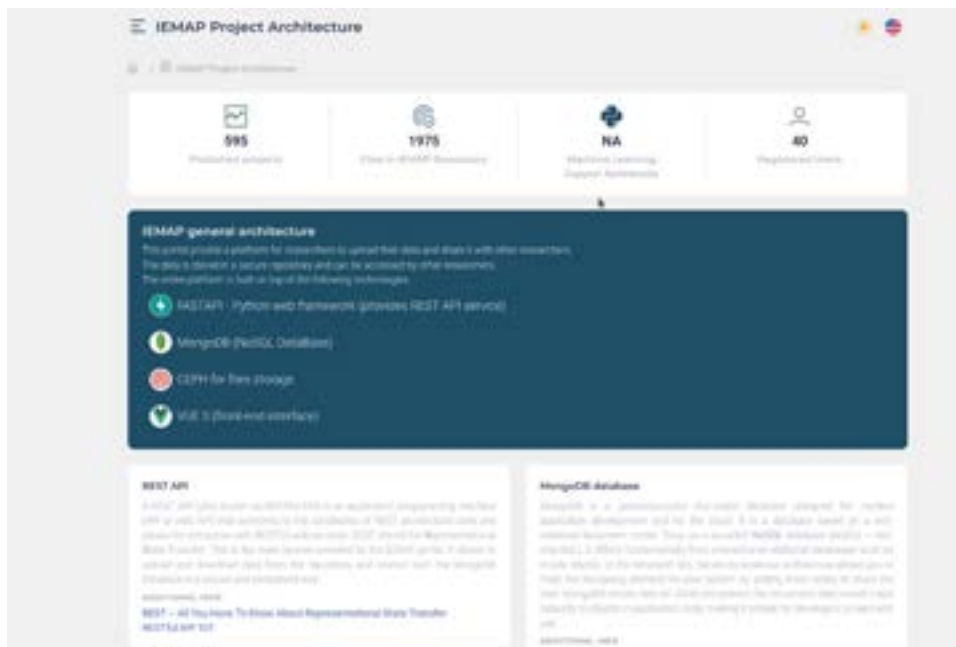


Fig. 13 IEMAP project Architecture (<https://iemap.enea.it/architecture>)

Fin qui si sono indicate tutte le pagine relative alla sezione “libera”, ovvero aperta a tutti, del sito IEMAP, mentre quelle che consentono di inserire metadati e/o file richiedono l'autenticazione. Lo scopo dell'autenticazione è quello di poter definire chi ha caricato cosa oltre ad evitare che chiunque possa caricare metadati o file che non sono di utilità per il progetto.

L'accesso alle aree riservate è gestito dal sistema di routing client disponibile in VUEJS, tramite l'interazione tra VUEJS e gli endpoint specifici per l'autenticazione del servizio REST API sono implementate le pagine di registrazione/login e l'accesso alla pagina dei progetti dell'utente.

L'implementazione attuale di autenticazione vede:

- JWT come strumento di comunicazione sicura delle credenziali
- MongoDB come storage delle credenziali (nel DB sono memorizzati email utente, data creazione, data ultimo login, affiliazione, stato validazione)

Il JWT contiene in maniera crittografata e le informazioni utili al backend per determinare se l'utente è autorizzato (ha un account verificato) e quale è la sua affiliazione (ente o istituto di appartenenza).

Ogni endpoint relativo alle operazioni CRUD richiede un JWT valido senza il quale il server REST API non porta a completamento l'operazione richiesta ma fornisce un messaggio di errore. Nel caso del modulo Python iemap tale JWT è inserito nell'intestazione della richiesta https per l'endpoint d'interesse, mentre nel caso di interfaccia web tale JWT è memorizzato come HTTP-Only cookie nel browser, tale tipologia di cookie è accessibile solamente al server che lo ha creato, il tutto ai fini di una maggiore sicurezza.

La sezione di registrazione del sito è accessibile in due modi:

1. Dalla Home page tramite la voce "Sign Up" nella barra in alto
2. Dalla barra laterale, in una qualsiasi delle pagine del sito, tramite la voce Login

Se si segue l'opzione 2 di cui sopra oppure tramite il pulsante login si ha la pagina seguente.

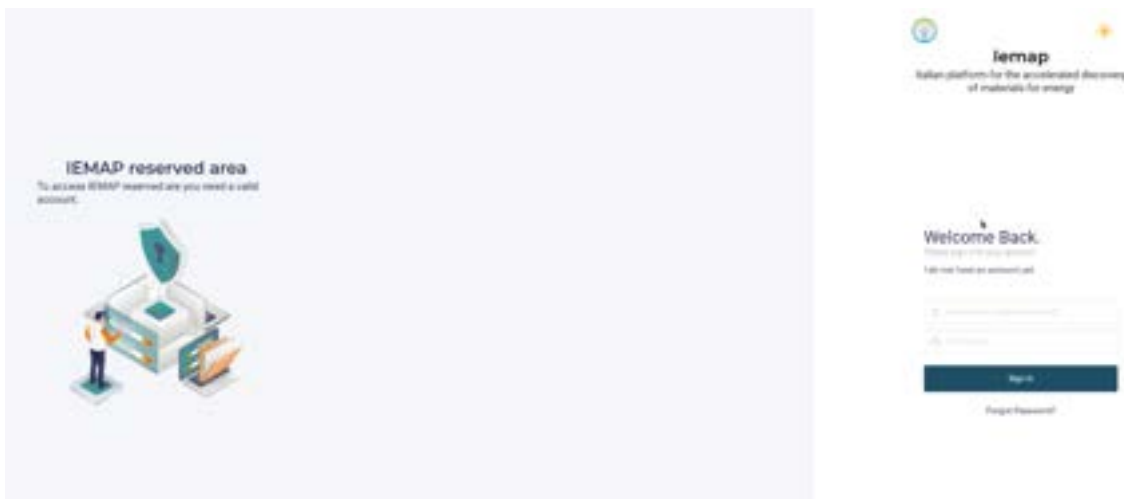


Fig. 14 Pagina a cui si è re-indirizzati qualora si richieda l'accesso ad una pagina "riservata" del sito (quali <https://iemap.enea.it/iemap/user/projects>)

Da tale pagina è possibile:

- Inserire le credenziali (email e password) e accedere all'area riservata (qualora si disponga già di un account validato)
- Accedere alla pagina di registrazione con click su "I do not have an account yet"
- Richiedere di reimpostare la password (link "Forgot Password?")

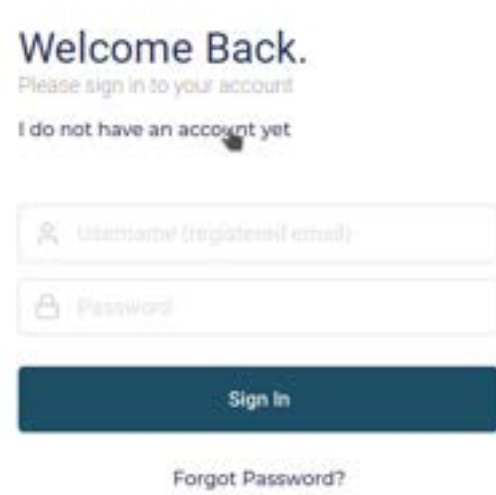


Fig. 15 Dettaglio modulo "login/sign up")

Le credenziali devono essere validate entro 24h, oltre tale termine non è più possibile validare la mail inserita e l'account viene rimosso automaticamente dal database.

L'operazione di validazione avviene tramite email, il backend (servizio REST API), dopo la registrazione di un account provvede ad inviare una mail all'indirizzo inserito.

Tale mail riporta quanto indicato di seguito.



Fig. 16 Esempio email per validazione account inviata dal server IEMAP

A questo punto l'utente deve completare la validazione semplicemente facendo "click" sul pulsante "Verify your registration". Una volta registrato e validato l'account è possibile accedere all'area riservata, dopo il login la barra di navigazione si modifica leggermente e viene mostrata automaticamente la pagina per l'inserimento dei progetti afferenti all'utente che ha effettuato il login.

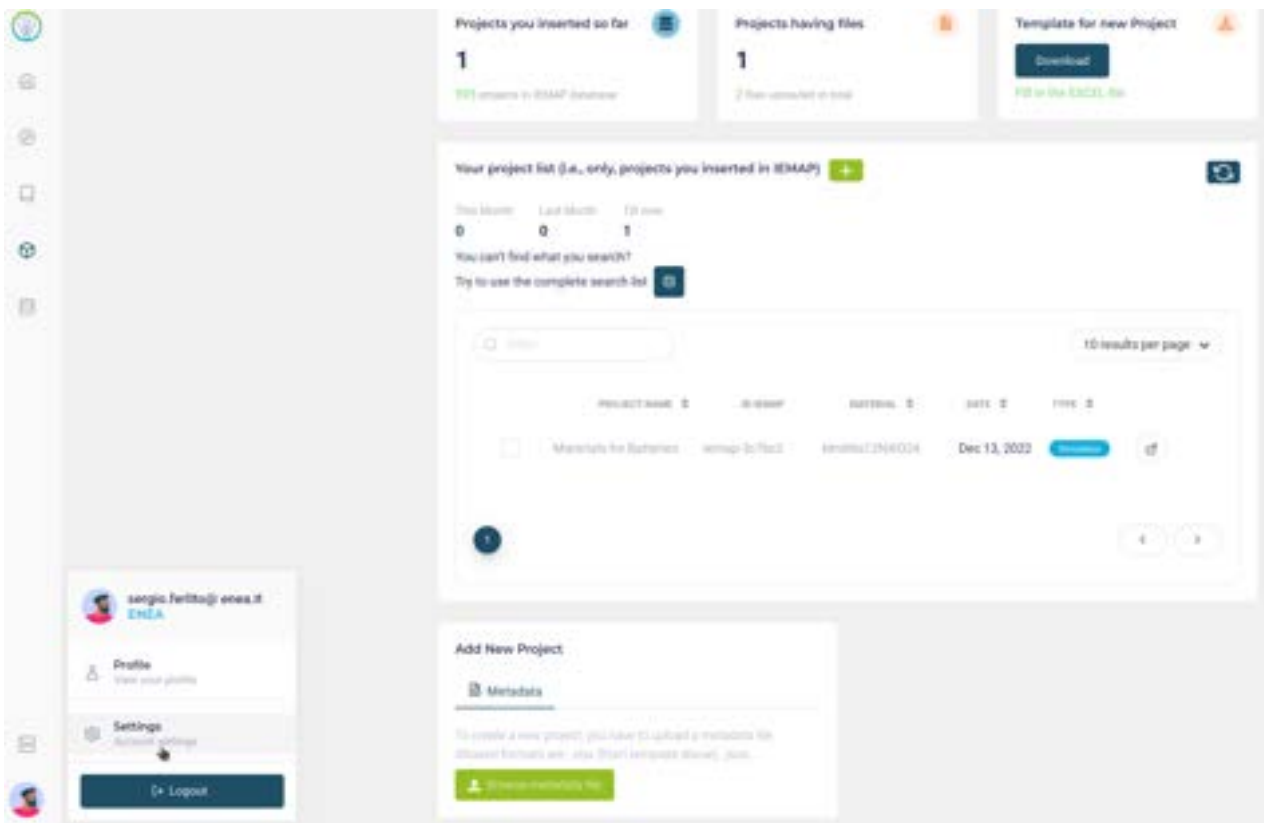


Fig. 17 Pagina con elenco progetti per uno specifico utente (<https://iemap.enea.it/iemap/user/projects>)

Riporto brevemente le funzionalità disponibili in tale pagina e le modalità d’inserimento dei metadati di progetto nonché l’upload dei file, omettendo alcune indicazioni/schermate per non allungare oltremodo quanto del resto è di per sé esplicativo.

Il menù aggiuntivo (riportato in fig. 17 in basso a sinistra) nella barra laterale sinistra compare solo a login effettuato con successo. Tale menù, oltre a indicare email ed affiliazione, consente di effettuare il logout e di andare ad una pagina con i dati del proprio profilo e d’impostazioni dello stesso. Nella pagina d’impostazione dell’account è possibile richiedere all’amministratore del sito la cancellazione dell’account stesso oppure modificare la password corrente.

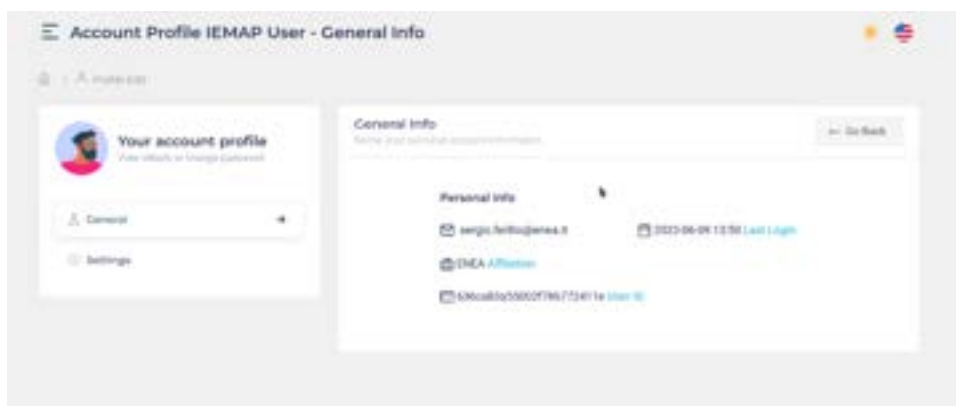


Fig. 18 Pagina con dettagli profilo account corrente

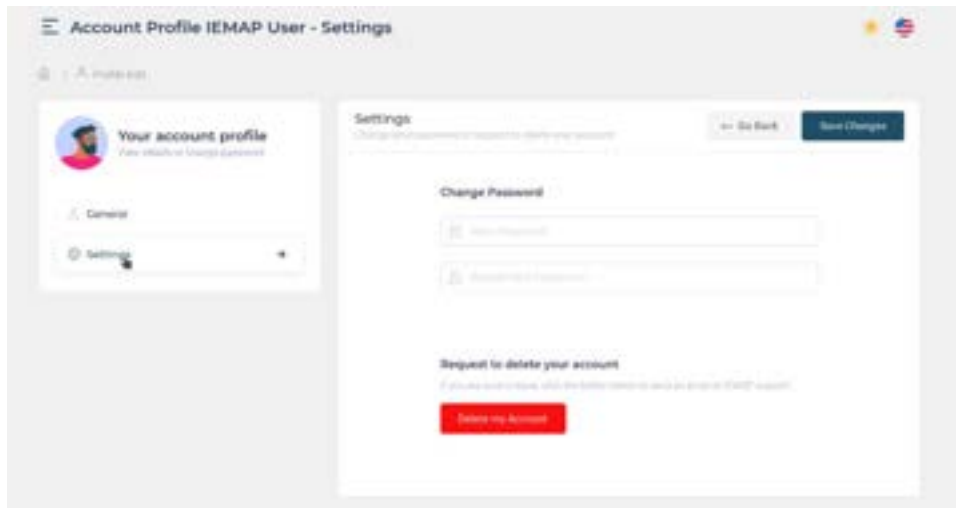


Fig. 19 Pagina con le impostazioni dell'account corrente

La pagina di gestione dei progetti utente riporta alcune informazioni generali quali progetti totali inseriti e progetti con file, nonché l'elenco dei progetti inseriti (con paginazione dei risultati). Tale elenco consente filtri su ogni elemento visualizzato:

- Nome
- Iemap-id
- Materiale
- Data
- Tipo

Facendo click sull'icona dei dettagli appare un tooltip con l'ID del documento memorizzato in MongoDB

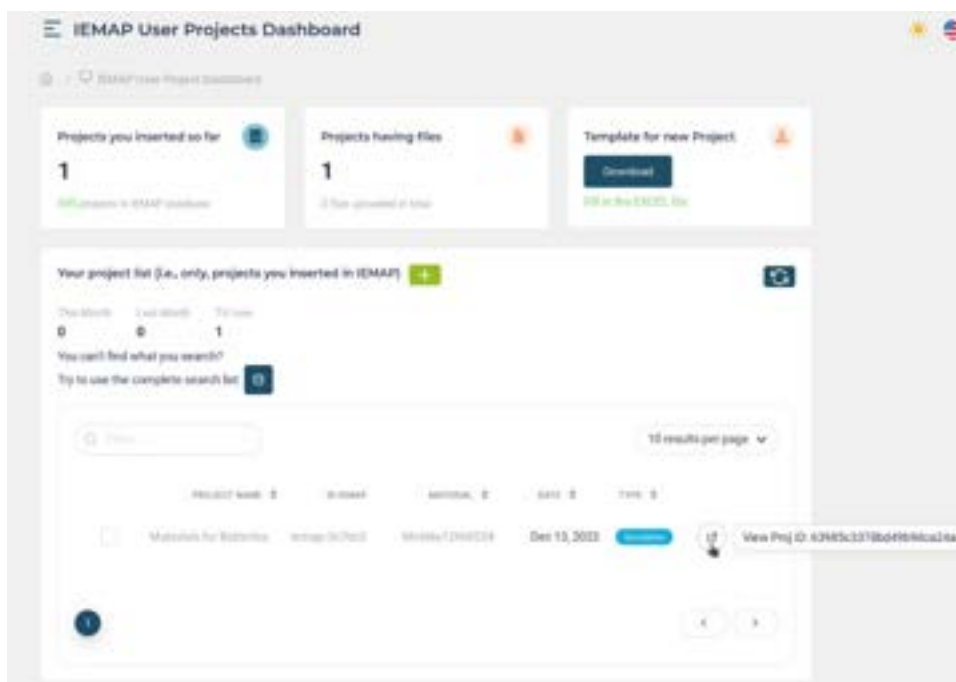


Fig. 20 Pagina elenco progetti utente corrente

La pagina di dettagli di progetto appare come segue

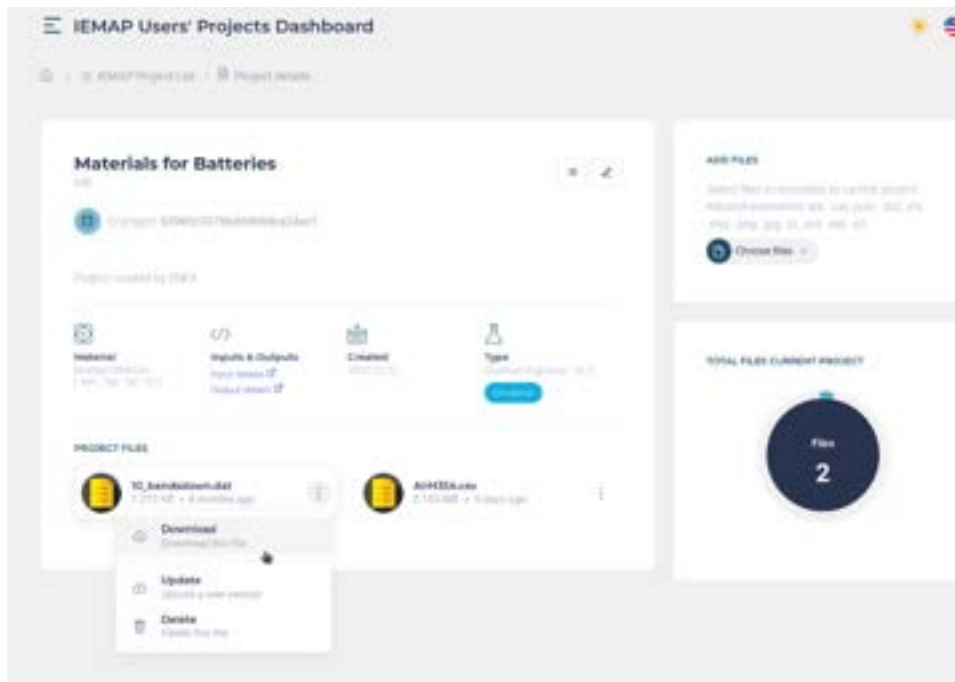


Fig. 21 Pagina dettagli singolo progetto per utente corrente

Qui è possibile scaricare file e/o farne l'upload di nuovi. (Alcune funzionalità quali update/delete file, sono ancora da implementare).

L'inserimento di un nuovo progetto avviene dalla pagina principale dei progetti dell'utente e consta essenzialmente di due fasi:

- 1 Inserimento metadati
- 2 Upload file

Prima vanno inseriti i metadati di progetto tramite file JSON oppure compilando il template .xlsx scaricabile tramite il pulsante "Download" nel riquadro "Template for new Project".

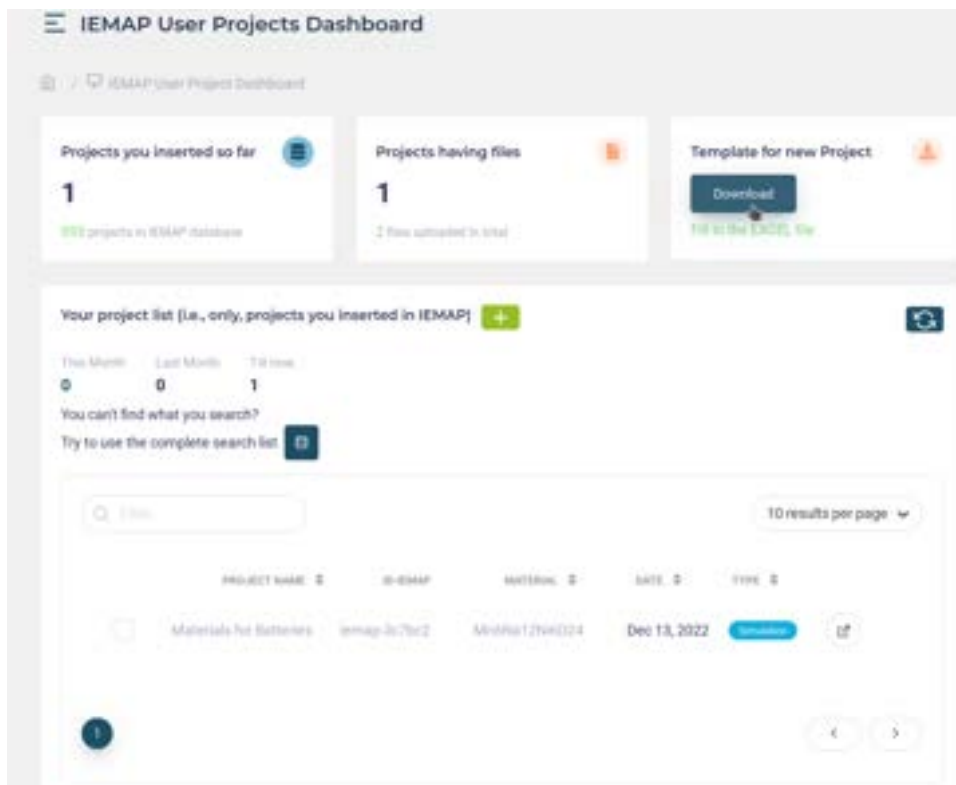


Fig. 22 Dettaglio del pulsante per il “download” del template .xlsx da compilare con i metadati di progetto

L’upload del file di metadati, sia esso un JSON oppure il file Excel del template che va opportunamente compilato, avviene con il pulsante “browse metadata file”

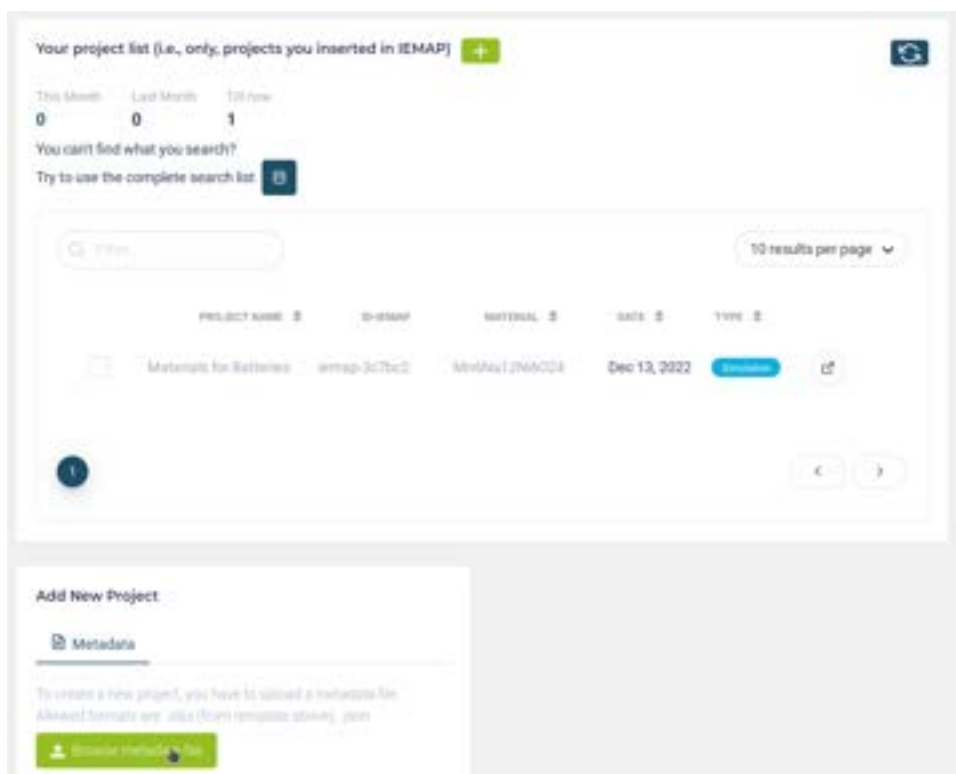


Fig. 23 Per inserire il file (JSON/Excel) di metadati utilizzare “browse metadata file”

Selezionato il file di metadati fin qui non si è salvato nulla nell'archivio IEMAP) il sistema mostra eventuali errori oppure una visualizzazione del JSON che andrà caricato nell'archivio IEMAP (estratto dal template Excel o caricato direttamente in tale formato) consentendo l'effettivo upload tramite il pulsante "Create Project". Solamente a progetto creato, ovvero una volta salvati i metadati di progetto, si potrà caricare uno o più file (dalla pagina di dettagli del progetto, inf Fig.21).

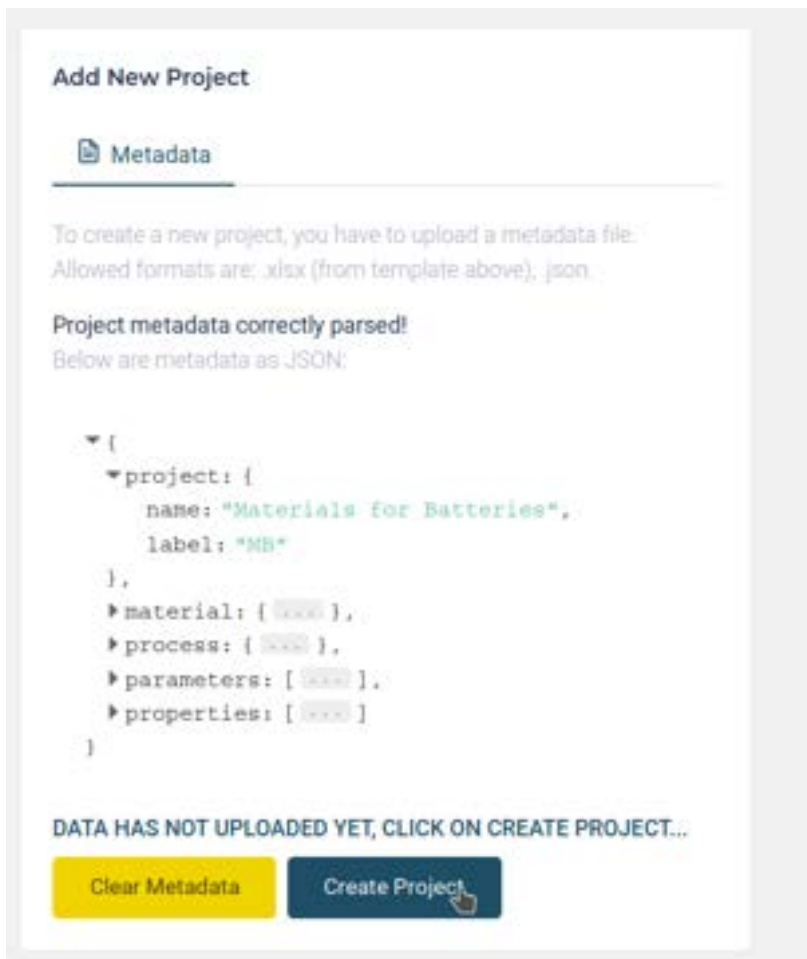


Fig. 24 Risultato elaborazione file metadati prima della creazione del progetto (tramite pulsante "create project")

7 Modulo Python iemap

Per poter facilmente integrare nel proprio flusso di lavoro le funzionalità della piattaforma IEMAP si è sviluppato un modulo Python disponibile al link

<https://pypi.org/project/iemap/>

Tale modulo è facilmente installabile tramite "pip install iemap". Inoltre, al seguente link, è disponibile tramite servizio di cloud Google Colab, un esempio su come utilizzare tale modulo Python.

<https://colab.research.google.com/drive/14T2ReGfx-xdWj9mblOtswO5RCGbFPHUO?usp=sharing>

Si tratta di un notebook Jupyter che illustra le funzionalità base del modulo Python e non richiede la configurazione dei moduli necessari Python da parte dell'utente, il tutto è eseguito sul cloud tramite interfaccia web. Tale modalità di fruizione dei servizi IEMAP è sicuramente quella da preferire per chi ha un minimo di conoscenze Python soprattutto in previsione delle prossime funzionalità che verranno messa a disposizione per l'integrazione di modelli di Machine Learning.

8 Sviluppi futuri

Nel prosieguo del progetto si conta di sviluppare e/o completare le seguenti funzionalità:

- Internazionalizzazione sito web
- Passaggio, ove applicabile, ad una architettura Kubernetes ai fini di una maggiore scalabilità
- Ampliamento funzionalità di ricerca dati e/o file
- Integrazione di un servizio Jupyter Hub per la semplice fruizione di Jupyter Notebooks con modelli di Machine Learning d'interesse per le attività di progetto
- Manutenzione e aggiornamento codici lato backend e frontend

9 Conclusioni

L'infrastruttura dati IEMAP sin qui sviluppata e già disponibile nella sua forma prototipale sul sito iemap.enea.it, consente l'inserimento dei metadati e dei dati di progetto con la possibilità di caricare uno o più file di progetto. La parte di consultazione e ricerca dati o download file di progetto è liberamente accessibile mentre per la creazione di progetti occorre registrare e validare un account tramite la propria mail istituzionale. L'architettura sviluppata utilizza le più moderne tecnologie software disponibili in ognuno degli ambiti di pertinenza e si compone di differenti parti di cui se ne è dettagliato il funzionamento.

10 Bibliografia

Ceph: l'archiviazione pratica per tutte le aziende - IONOS. (n.d.). Retrieved June 9, 2023, from <https://www.ionos.it/digitalguide/server/know-how/cose-ceph/>

Introduction to Ceph · Better Tomorrow with Computer Science. (n.d.). Retrieved June 8, 2023, from <https://insujang.github.io/2020-08-30/introduction-to-ceph/>

Puccini, M., Prestazioni, A., Legale, S., Mariano, R. A., Cannataro, G., & Caretto, G. (n.d.). *PRESTAZIONI DEL FILESYSTEM BASATO SU CEPH NELL'INFRASTRUTTURA ENEAGRID*. Retrieved June 9, 2023, from www.enea.it

Draxl, C. a. (2018). NOMAD: The FAIR concept for big data-driven materials science. *Mrs Bulletin*, 23(9), 676-682.

Jain, A. a. (2013). Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL materials*, 1(1), 011002.

11 Abbreviazioni ed acronimi

API: An application programming interface (API) is a way for two or more computer programs to communicate with each other.

Back-end : development comprises a site's structure, system, data, and logic.

CEPH: The name "Ceph" is an abbreviation of "cephalopod", it is an open source storage platform

Docker : open platform for developing, shipping, and running applications

Endpoint : are the locations of the resources, and the API uses endpoint URLs to retrieve the requested resources

Frontend : focuses on the visual aspects of a website — the part that users see and interact with

JSON: Javascript Object Notation

MongoDB: NoSQL database

NGINX : Reverse Proxy Server & Load Balancer

REST API: Representational State Transfer Application Programming Interface

SPA – Single Page Application

URI: Uniform Resource Identifier

URL: Uniform Resource Locator

VUEJS: Javascript framework for web applications

VMware : virtualization and cloud computing software provider