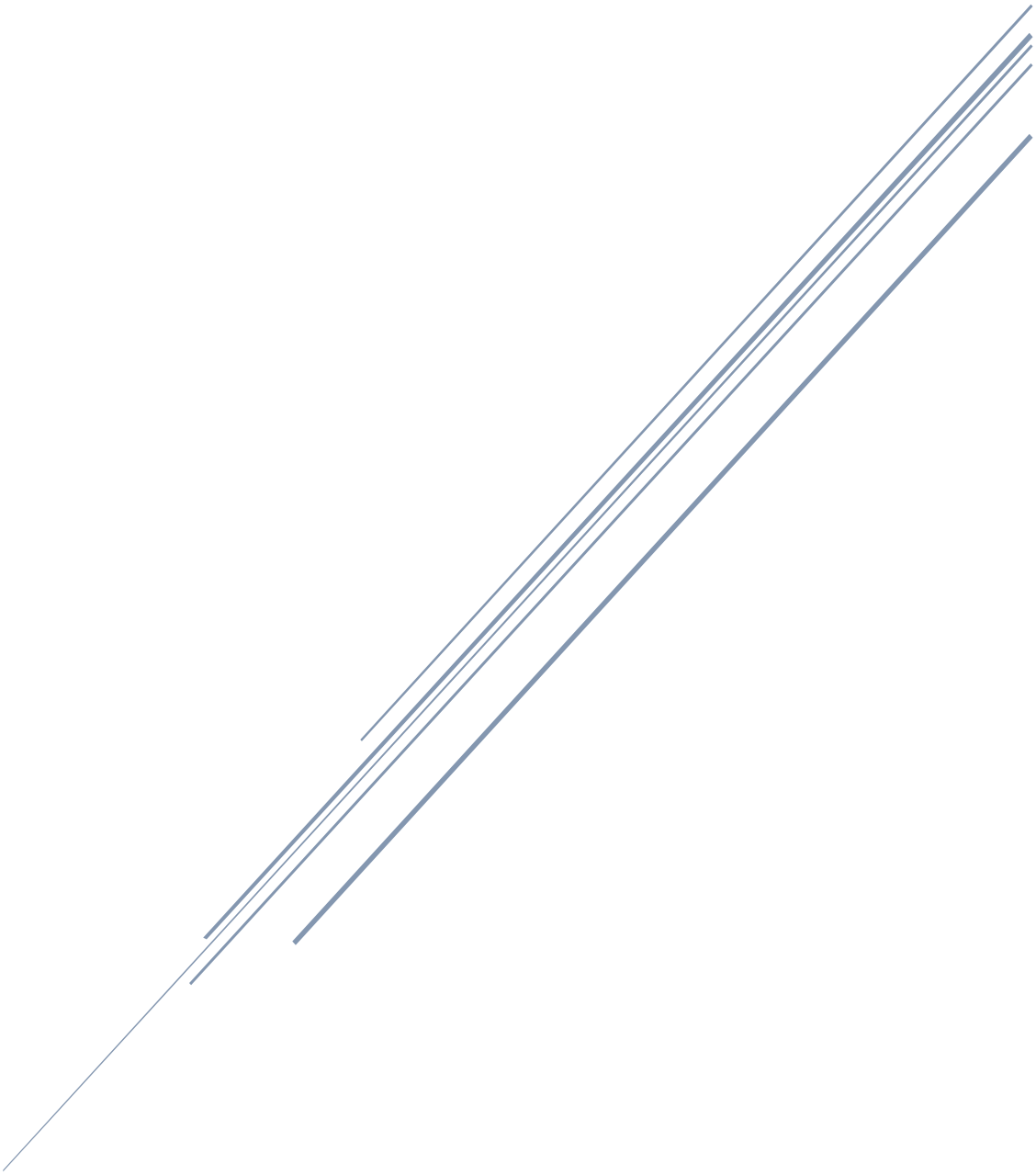
MAP VIEW

Progetto del modulo “Applicazione web e mobile”



Università di Camerino - UNICAM
Corso di Informatica per la Comunicazione Digitale

Indice

Introduzione

Definizione del progetto

Il progetto ha come scopo la realizzazione di una piattaforma di valorizzazione e digitalizzazione di un territorio comunale attraverso il caricamento di informazioni culturali, turistiche, sportive e di qualsiasi altra natura che possano essere di interesse per chi vive e frequenta un territorio. Il caricamento di qualsiasi contenuto sarà da riferirsi a un punto geolocalizzato presente all'interno del territorio comunale, pertanto non potranno essere aggiunte informazioni su altri comuni. Come per i contenuti, anche nuovi punti geolocalizzati potranno essere definiti dall'utenza. È inoltre possibile creare itinerari o esperienze che rappresentano insiemi ordinati, o non, di punti di interesse. Tali itinerari saranno definiti secondo le stesse strategie dei punti di interesse, ovvero caricati da responsabili o in maniera collaborativa e successivamente confermati. Si immagina una piattaforma collaborativa in cui la cittadinanza stessa possa contribuire ad arricchire la piattaforma caricando contenuti. In tal caso quest'ultimi verranno pubblicati in piattaforma soltanto dopo la verifica di conformità del contenuto, salvo utenti certificati dalla piattaforma stessa.

Committenti e team di sviluppo

Il progetto "MapView" è stato commissionato da:

- Prof. Andrea Polini e Andrea Morichetta: sviluppare il backend in Java e successivamente portarlo su Spring Boot. Il progetto risultante dovrà fornire delle API REST con cui poter realizzare l'interazione, dunque performare le funzionalità specificate in precedenza;

- Prof. Diego Bonura: sviluppare il frontend dell'applicativo in modo da essere fruibile in modalità web anche da dispositivi mobile oppure sia una app mobile (ionic, xamarin, maui, nativescript, react native, flutter…), implementi un pattern MVC o Single Page Application e che abbia un backend ed un DBMS per il popolamento dei dati;

La prima parte del progetto è da riferirsi al modulo di "Ingegneria del Software" dell'anno accademico 2023/2024, mentre la seconda parte al modulo di "Applicazione web e mobile" del medesimo anno accademico. Il team di sviluppo responsabile di questo progetto, è composto dai seguenti membri:

1. Matteo Giaccaglia (N. matricola);
2. Michele Polenta (N. matricola);
3. Samuele Pirani (N. matricola 118535);
4. Aris Vaccarini (N. matricola 118536);

Tutti i membri del gruppo provengo del corso di Informatica per la Comunicazione Digitale dell'anno accademico 2023/2024, presso l'università di Camerino.

Descrizione del sistema

Backend

Il software è stato progettato e sviluppato in Java, traducendolo poi in Spring-Boot, framework open-source ideato per semplificare il processo di sviluppo, configurazione e distribuzione delle applicazioni Java, in particolare sfruttando il suo modulo web per lo sviluppo delle REST API. Infatti, grazie all'inclusione nel framework di server integrati, come TomCat, Jetty o Undertow, questi consentiranno la verificabilità funzionale delle API, potendole testare direttamente da browser oppure utilizzando software di supporto come Postman. Tra le facilitazioni introdotte dal framework, spicca quella riguardante la persistenza dei dati, implementata nel progetto tramite JPA (Java Persistence API), specifiche Java incaricate nella comunicazione con il database, e la manipolazione delle informazioni contenute in quest'ultimo. La base di dati è stata creata tramite PostgreSQL, successivamente caricata su ElephantSql, in maniera di renderla accessibile ovunque al di fuori del proprio dispositivo. Di seguito sono riportate tutte le dipendenze sulle quali il backend fa affidamento:

- spring-boot-starter-web: Questa dipendenza fornisce tutte le funzionalità necessarie per sviluppare applicazioni web usando Spring Boot;
- spring-boot-starter-data-jpa: Abilita l'utilizzo delle JPA per quanto riguarda la persistenza delle informazioni;
- spring-boot-starter-tomcat: Abilita l'utilizzo del server TomCat per quanto riguarda l'esecuzione e la distribuzione dell'applicativo web;
- overpass-api: è un servizio web che fornisce un'interfaccia per interrogare e recuperare dati da OpenStreetMap (OSM) una mappa del mondo libera e modellata dalla comunità che fornisce dati geografici dettagliati, come strade, punti di interesse, confini amministrativi e molto altro ancora;
- spring-boot-starter-security: è una dipendenza di Spring Boot che fornisce un supporto integrato per la sicurezza delle applicazioni;
- jjwt: è una libreria Java per la gestione di JSON Web Token (JWT);
- postgresql: driver JDBC per abilitare la comunicazione tra applicativo e database PostgreSQL;

Frontend

Il frontend è stato realizzato utilizzando il framework open-source Angular, implementando il pattern Single Page Application.

Aspetti di sicurezza

Durante lo sviluppo del progetto sono stati tenuti in conto una serie di aspetti legati alla sicurezza da implementare necessariamente all'interno del software:

- SQL Injection: SQL injection è una vulnerabilità di sicurezza che si verifica quando i dati forniti dall'utente vengono inseriti direttamente nelle query SQL senza una corretta sanitizzazione o validazione. Con JPA (Java Persistence API), SQL injection è mitigata in modo significativo perché JPA offre un'astrazione del livello di persistenza senza la necessità di scrivere ed eseguire manualmente le query SQL;

- Autenticazione, autorizzazione e gestione dei ruoli: grazie al modulo Spring Security, viene gestito correttamente l'accesso a determinate funzionalità legate al ruolo dell'utente, previa autenticazione. L'autenticazione e autorizzazione fanno affidamento sullo standard JWT che definisce un modo compatto e autonomo per rappresentare informazioni tra due parti come oggetto JSON. Un JWT è costituito da tre sezioni separate da un punto ('.'): header, payload e signature. Ogni sezione è codificata in Base64 e concatenate per formare il token completo. Il token risultato conterrà tutte le informazioni necessarie al sistema per autenticare l'utente e autorizzarlo ad usare specifiche funzionalità;

- Attacchi CORS: per evitare problemi di CORS durante lo sviluppo, il client Angular è stato configurato in modo da inoltrare le richieste che vengono effettuate al backend tramite proxy. Questa configurazione provvisoria è stata adottata per facilitare l'interazione tra frontend e backend, mentre in un reale ambiente di produzione è necessario configurare correttamente il server per gestire le autorizzazioni CORS;

- Protezione delle password: Ogni volta che viene registrato un nuovo account, le password vengono criptate grazie all'algoritmo bcrypt, una funzione di hashing di password basata sulla cifratura Blowfish. Oltre a incorporare un salt per proteggere la password contro attacchi rainbow table, bcrypt è una funzione adattiva: col tempo, il conteggio dell'iterazione può essere aumentato per renderla più lenta, in modo da essere resistente ad attacchi brute force anche con capacità computazionale crescente;