



**UNIFI**, *Santa Marta*,  
*Giugno 2020*



UNIVERSITÀ  
DEGLI STUDI  
FIRENZE

# Applicativo SIM per la gestione e la consultazione dei servizi di infomobilità

Giulio Calamai    Marco Loschiavo

✉ [giulio.calamai1@stud.unifi.it](mailto:giulio.calamai1@stud.unifi.it)

✉ [marco.loschiavo1@stud.unifi.it](mailto:marco.loschiavo1@stud.unifi.it)

🏛 Laboratorio STLAb, UNIFI Santa Marta, FI

# Sommario

- 1 Introduzione
- 2 Tecnologie Adottate
- 3 Documentazione
- 4 Panoramica dell'applicazione
- 5 Conclusioni



# Processo di sviluppo

Abbiamo adottato un approccio simil-aziendale,  
ovvero con le figure di:

- **Project Manager** (STLAB): fornitura di documentazione iniziale di progetto e tutoraggio in corso d'opera.
- **Developer**: sviluppo del software secondo la documentazione fornita con modifiche e aggiunte.



# STINGRAY Project

Il progetto **STINGRAY (SmarT station INtelliGent RAilwaY)** è focalizzato sullo studio, la progettazione e lo sviluppo di un'infrastruttura di comunicazione di stazione, integrando la linea elettrica e le tecnologie wireless, che:

- realizza una rete LAN sugli impianti della stazione;
- consente il controllo e il monitoraggio delle apparecchiature della stazione;
- crea servizi a valore-aggiunto aggiunti quali connettività, monitoraggio, video-sorveglianza, rilevamento ambientale, infomobilità, per clienti e personale.



# Panoramica

## Smart Station

Stazione intelligente che aggrega più servizi di mobilità.

## Infomobility Service

Servizio di mobilità esposto in una stazione intelligente.



# Il progetto

## Obiettivo

Realizzazione di una soluzione pratica per l'accesso ai servizi di mobilità attraverso delle stazioni intelligenti, dislocate nella città. Un viaggiatore può accedere a queste stazioni per consultare o prenotare un servizio.



# Il progetto

- Fornire tutti i servizi di mobilità ad un utente viaggiatore in modo da facilitargli gli spostamenti.
- Consentire agli amministratori semplici la gestione di stazioni intelligenti e servizi di infomobilità. Essi hanno la possibilità di creare, modificare e disabilitare le smart stations presenti nelle città alle quali sono autorizzati. Inoltre possono aggiungere o rimuovere i servizi di infomobilità ai quali sono autorizzati. Quest'ultimi possono a loro volta essere abilitati, disabilitati e modificati.
- Consentire agli amministratori di sistema (alto livello) di autorizzare gli amministratori semplici ad operare su determinate città e servizi. Possono inoltre aggiungere nuovi infomobility service provider oltre alle operazioni già citate.



# Realizzazione

- **Frontend:** applicativo web per accedere sia come amministratore per la gestione delle Smart Station, sia come utente per la fruizione del servizio.
- **Backend:** implementazione dei domini e le relazioni tra di essi. In particolare, si vuole modellare la gestione di tutti i fornitori di servizi di infomobilità presenti nella città di varia tipologia (e. g. bike sharing, car sharing, etc.) e come questi possono essere utilizzati da un cittadino.



# Tecnologie Adottate

- **Java Enterprise Edition**, con le specifiche:
  - ▶ **CDI** (Context and Dependency Injection)
  - ▶ **JPA** (Java Persistence API)
  - ▶ **JAX-RS** (Java API for RESTful Web Services)
- **MySQL Server** come DBMS relazionale
- **JBoss Wildfly** come Application Server per il deployment
- **Maven** come dependecies manager
- **Repository** di sviluppo fornito da STLAB
- **Postman** per il test dei contratti dei dati di ogni servizio esposto da SIM Backend
- **Java JWT** per la generazione del token durante la fase di login
- Framework **Angular 2+** con linguaggio **TypeScript** per il frontend



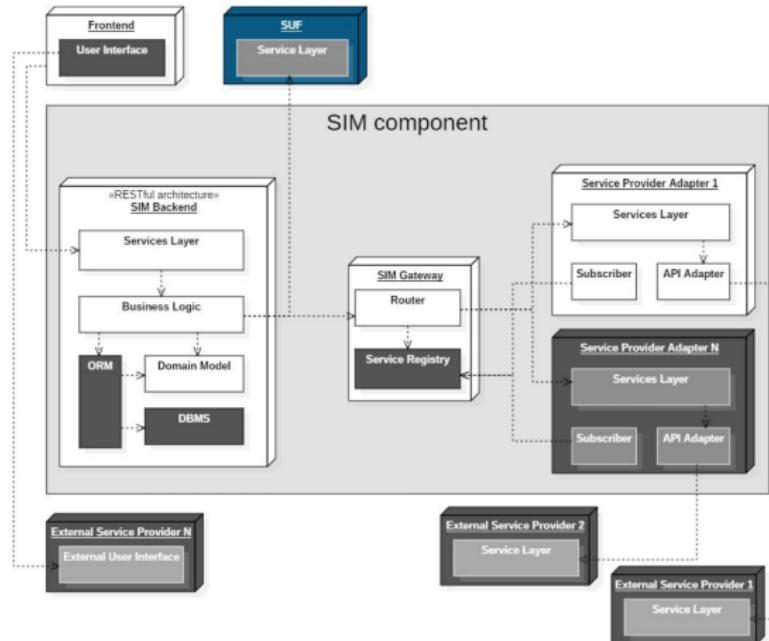
# Architettura SIM

Caratteristiche:

- **RESTful** (Representational State Transfer)
- **Service-Oriented**
- Mantiene **separati** i moduli frontend e backend.
- Espone delle **REST API** nel backend, fruibili da diversi consumatori.



# Deployment UML



- **Frontend:** parte client dell'applicazione
- **SIM Backend:** parte server dell'applicazione
- **SUF:** Fornitore dei dati dei treni e ambientali (Stringray che sono stati adattati in SIM Backend)

# SIM Backend

- Espone servizi che sono consumabili direttamente dall'applicazione client di frontend, differenziandosi in base a condizioni di autenticazione.
  - ▶ **Autenticato:** utente con ruolo “SUPER ADMIN” o “ADMIN” dotato di maggiori diritti rispetto agli utenti semplici non autenticati
  - ▶ **Non Autenticato:** utente senza alcun ruolo, ossia un viaggiatore che ha diritti di fruizione ridotti rispetto ad un amministratore.

## Endpoint

raccoglitore di servizi REST dedicati alla risorsa di uno specifico tipo



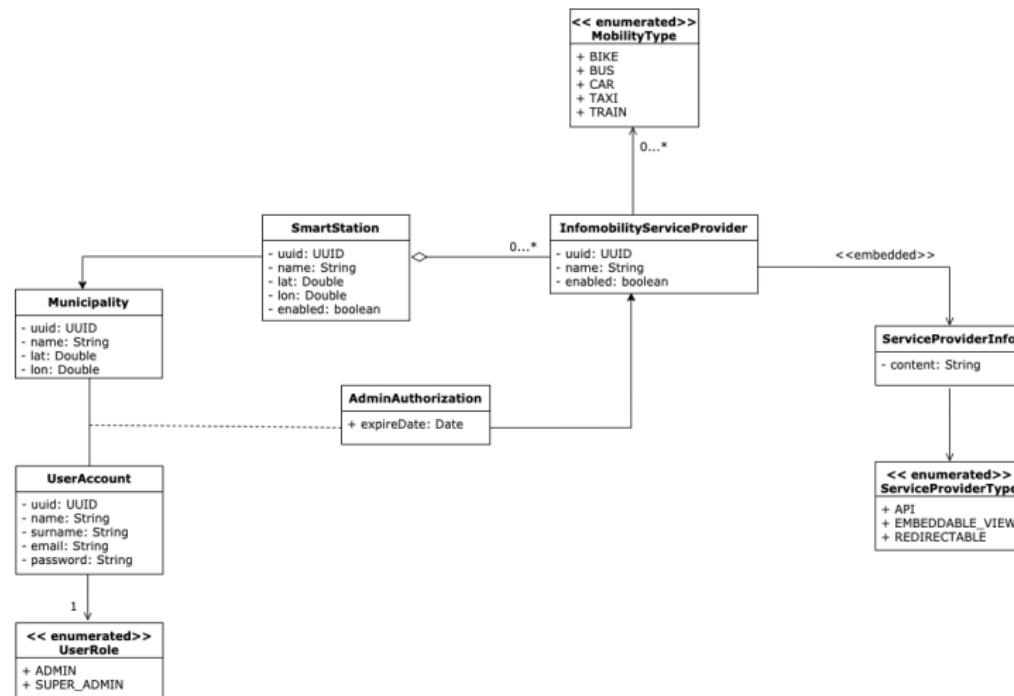
# Endpoint SIM Backend

Sono previsti:

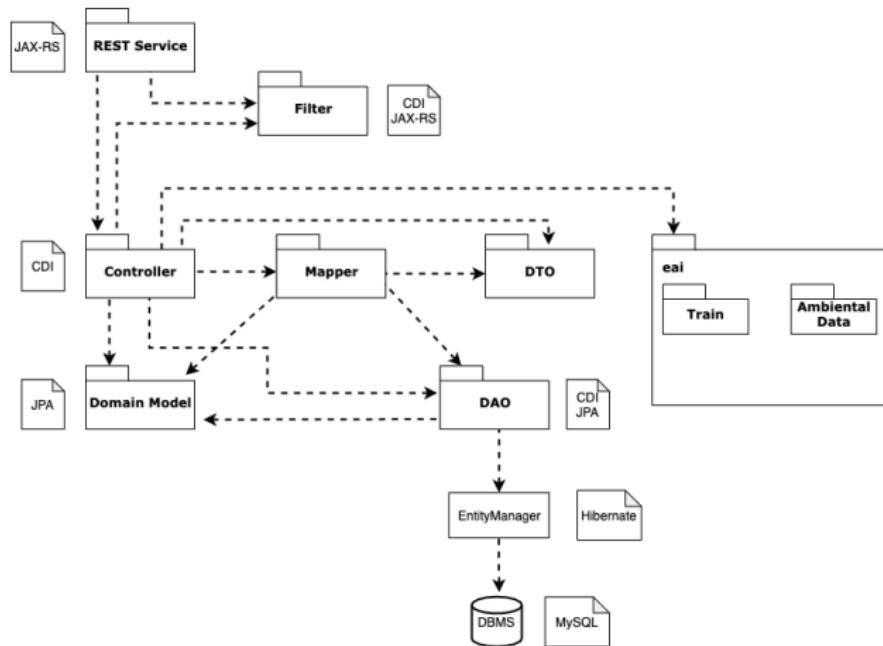
- **User Account Endpoint:** dedicato alla risorsa di tipo utente. Le funzionalità “in scrittura” di gestione degli account saranno permesse solo a SUPER ADMIN;
- **Smart Station Endpoint:** dedicato alla risorsa di tipo stazione intelligente. Permette anche agli utenti con ruolo ADMIN di associare l’elenco di servizi di infomobilità attivi per essa;
- **Infomobility Service Endpoint:** dedicato alla risorsa di tipo servizio di infomobilità. Le specifiche funzionalità per la gestione delle configurazioni sono permesse a utenti con ruolo ADMIN
- **Municipality Endpoint:** dedicato alla risorsa di tipo municipalità. Le specifiche funzionalità per la gestione delle configurazioni sono permesse solo ad utenti SUPER ADMIN
- **Authentication Endpoint:** dedicato al meccanismo di autenticazione di un utente, generando un token JWT
- **Admin Authorization Endpoint:** dedicato alla risorsa di tipo autorizzazioni fornite agli ADMIN. Le specifiche funzionalità per la gestione sono permesse solo a SUPER ADMIN



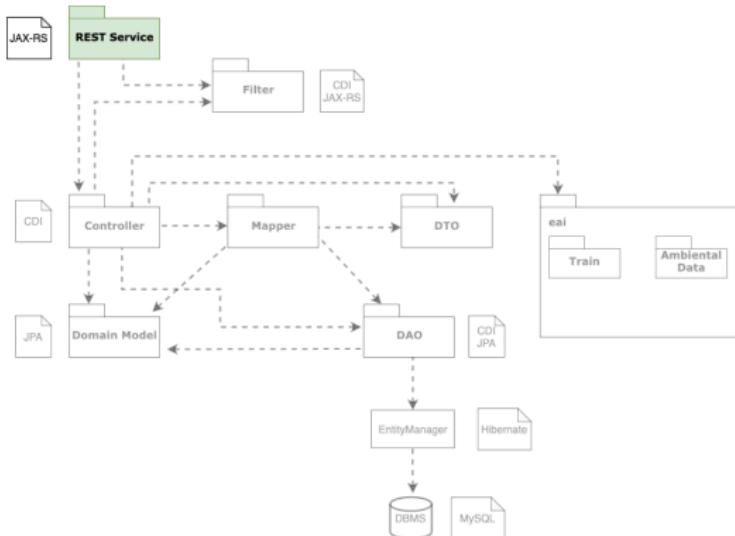
# Modello di Dominio



# Implementazione



# Implementazione

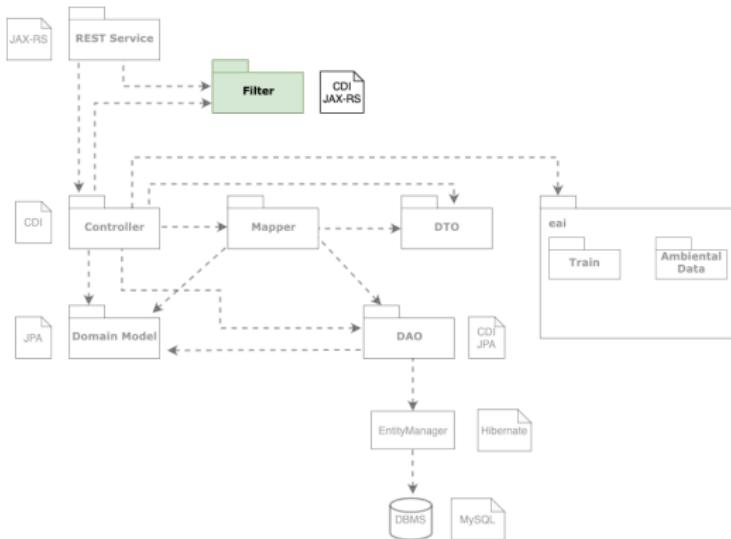


## Rest Service

- Tutti gli endpoint messi a disposizione dal BE
- JAX-RS
- Alcuni servizi degli endpoint sono protetti e possono essere usufruiti solamente da personale autorizzato(SUPER ADMIN, ADMIN)



# Implementazione

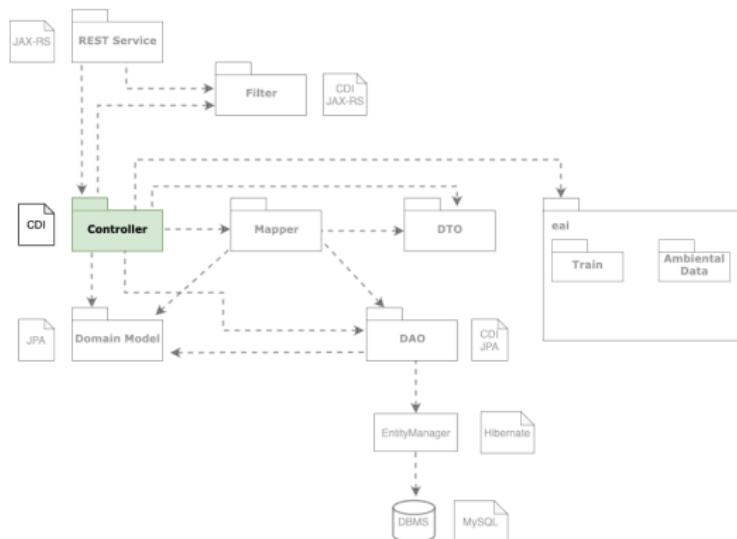


## Filter

- Impostazioni riguardanti le Request e le Response
- si configurano le impostazioni per l'ACAO (access-control-origin-allow)
- Protezione degli endpoints
  - ▶ JWT per l'autenticazione
  - ▶ JAX-RS per l'autorizzazione ROLE and CONDITION-base



# Implementazione

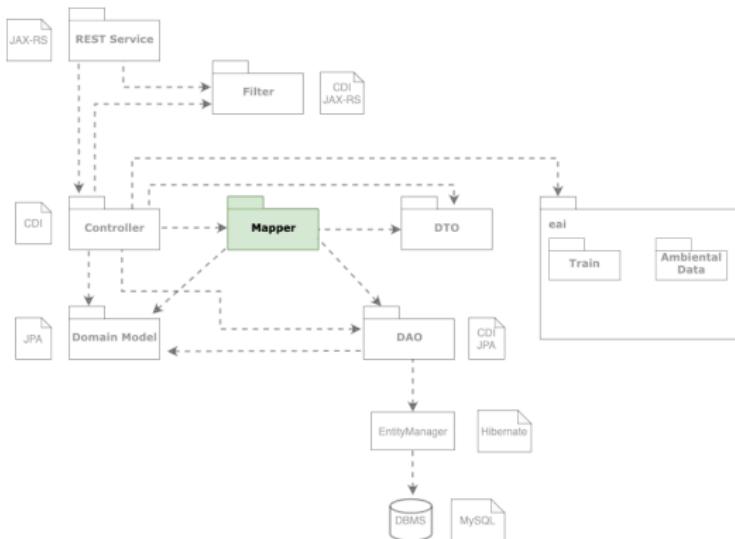


## Controller

- Assolvono i casi d'uso invocando od implementando direttamente i metodi caratterizzanti
- Manipolano entità del Domain Model per intermediazione del DAO o direttamente
- Elaborano ed interpretano istanze di DTO, solitamente per intermediazione dei Mapper



# Implementazione

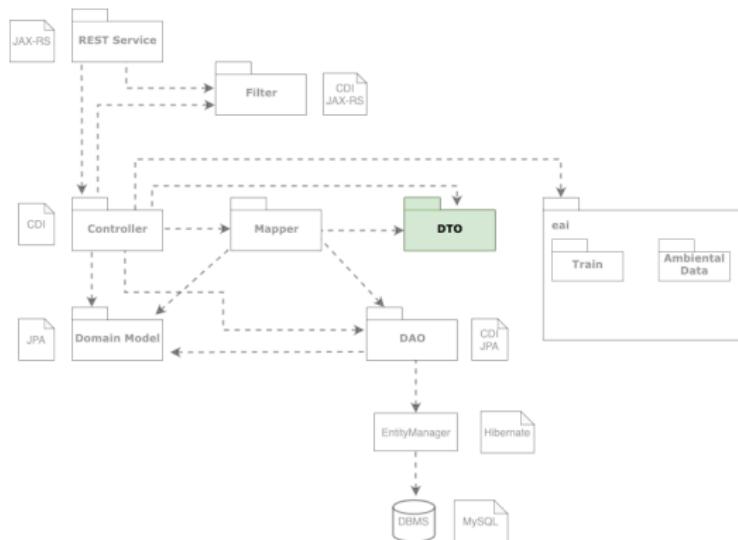


## Mapper

- si occupano di fare la conversione bidirezionale tra entità proprie del modello di dominio e oggetti di trasferimento(DTO)



# Implementazione

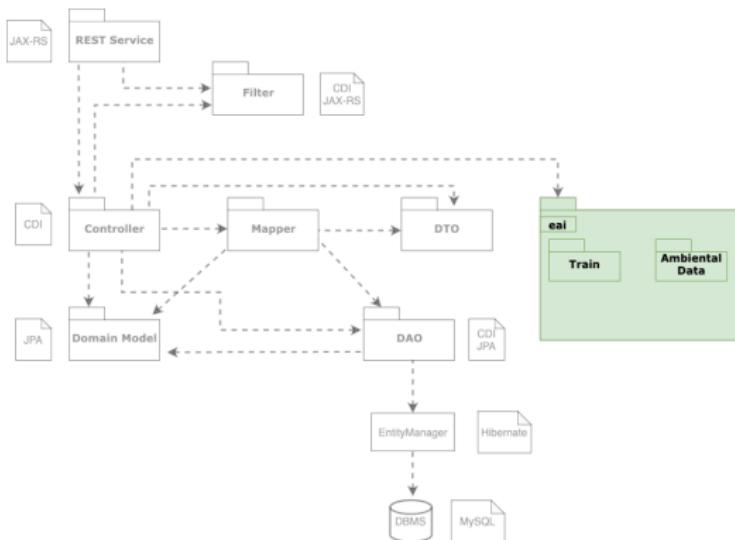


## DTO

- Rappresentano una versione "semplificata" o "adeguata" di oggetti riferiti al modello di dominio
- Svolgono il ruolo di "trasportatore di dati" tra processi comunicanti (ie Front-end e Back-end)



# Implementazione

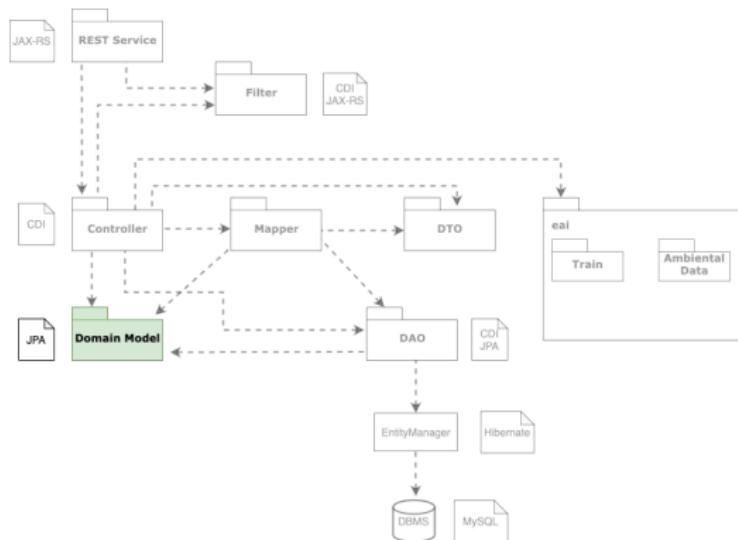


## eai

- Servizi esterni (Treni e dati ambientali) forniti da stringray
- Conversione da XML a JSON tramite libreria Jackson



# Implementazione

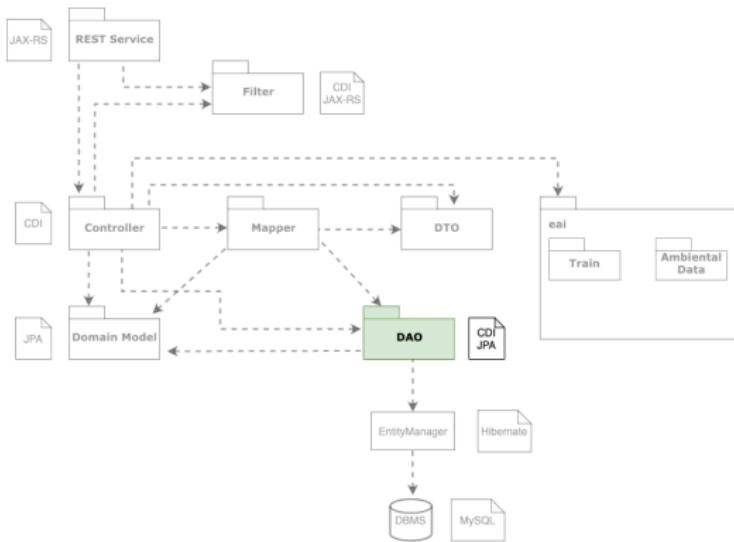


## Domain Model

- Classi del modello di dominio
- JPA con le annotazioni offerte per fare il mapping sul DB delle entità



# Implementazione

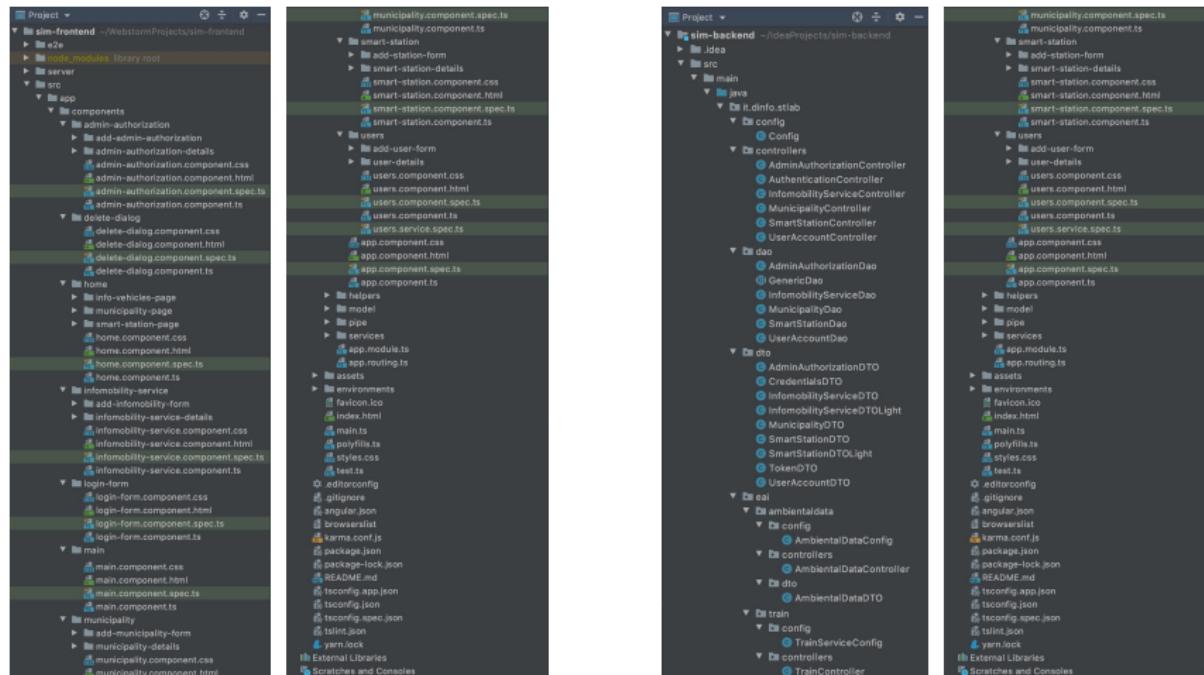


## DAO

- Interfaccia astratta verso il livello di persistenza
- Intermediario tra le entità del modello di dominio e le “tabelle reali” del database
- Si avvalgono del supporto di un EntityManager che consente di effettuare delle operazioni sulle entità come creare, rimuovere e trovare istanze di una entità e interrogare entità mediante specifiche query.



# Struttura SIM Frontend & Backend



# Code Snippet-1

```
private urlReal = 'http://79.35.20.111/sim-backend-1.0/infomobility-services';

// send a PUT request to the API to update a data object
public updateInfomobilityService(infomobilityService): Observable<InfomobilityService> {
  const body = JSON.stringify(infomobilityService);
  return this.http.put<InfomobilityService>(`url: ${this.urlReal} + '/' + infomobilityService.id, body, httpOptions);
}
```

(a) **Frontend** - consume update isp service

```
/*
 * @URI: SIM_BACKEND_URI/infomobility-services/{uuid}
 * @PathParams: uuid: stringa in formato UUIDv4 riferita all'infomobility di interesse
 * @RequestBody: JSON contenente una rappresentazione dell'infomobility service da modificare
 * @ResponseBody: JSON contenente l'infomobility service modificato */
@PUT
@Path("/{uuid}")
@Consumes({MediaType.APPLICATION_JSON})
@Produces({MediaType.APPLICATION_JSON})
@RolesAllowed({"SUPER_ADMIN", "ADMIN"})
public Response update(@PathParam("uuid") String uuid, InfomobilityServiceDTO infomobilityDTOReceived) {
    infomobilityServiceController.update(uuid, infomobilityDTOReceived);
    InfomobilityServiceDTO dto = infomobilityServiceController.getById(uuid);
    return Response.status(Response.Status.OK).entity(dto).build();
}
```

(b) **Backend** - update isp service



# Code Snippet-2

```

public TrainsDeparturesResponseDTO getDepartureInfo(String placeId) throws IOException {
    String path = TrainServiceConfig.TRAIN_SERVICE_ENDPOINT;

    //Si utilizza la specifica
    Client client = ClientBuilder.newBuilder()
        .path("GetDepartures")
        .queryParam("placeId", placeId)
        .request(MediaType.APPLICATION_XML)
        .get(String.class);

    //JACKSON
    XmlMapper xmlMapper = new XmlMapper();
    xmlMapper.disable(DeserializationFeature.FAIL_ON_UNKNOWN_PROPERTIES);
    TrainsDeparturesResponseDTO trainsDeparturesResponseDTO = xmlMapper.readValue(xml, TrainsDeparturesResponseDTO.class);
    return trainsDeparturesResponseDTO;
}

```

(a) Client request to STINGRAY service

```

@JacksonXmlRootElement(localName = "DeparturesResponse") //Define root element name in XML. localName is the name of the XML root element
public class TrainsDeparturesResponseDTO {

    @JacksonXmlElementWrapper(localName = "Departures") //Define wrapper to use for collection types. localName is the name of the XML wrapper element
    @JacksonXmlProperty(localName = "Departure") //Define XML property, can be attribute or element. localName is the name of the XML element
    @JsonProperty("departures") //Nome di uscita in json. Altrimenti il nome sarebbe quello della variabile (departures)
    private List<Departure> departures;
    public List<Departure> getDepartures() { return departures; }

    public void setDepartures(List<Departure> departures) { this.departures = departures; }

    static class Departure {

        @JacksonXmlProperty(localName = "Cancelled")
        @JsonProperty("cancelled")
        private Boolean cancelled;
    }
}

```

(b) Adapter XML response to JSON



# Frontend Web App

Parte visibile all'utente di un programma, tipicamente un'interfaccia (UI). Responsabile dell'acquisizione dei dati di ingresso e della loro elaborazione.

Vedremo in seguito alcune parti dell'UI:

- Viaggiatori
- Login
- ADMIN
- SUPER ADMIN



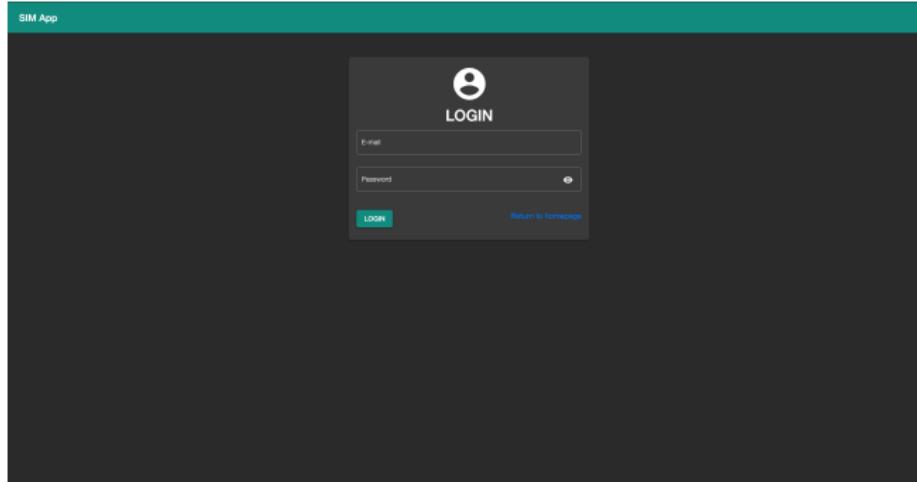
# Viaggiatori

- Un utente **viaggiatore** ha accesso unicamente alla home page nella quale può scegliere la municipalità, la stazione, il servizio ed infine i veicoli disponibili, sfruttando anche un filtro di ricerca.



# Login

- Un utente che possiede le **credenziali di accesso** può accedere all'app per la gestione dei servizi che offre, nel rispetto del suo ruolo.



# ADMIN

- Ha accesso alle pagine di smart stations e infomobility services. In entrambi i casi saranno mostrate solo quelle per le quali ha un'autorizzazione fornita da un SUPER ADMIN, così come le operazioni CRUD su di esse.



# SUPER ADMIN

- Può visualizzare e modificare le stazioni, i servizi di infomobilità, le municipalità e gli utenti senza restrizioni. Per quest'ultimi può fornire delle autorizzazioni per limitarne il "raggio d'azione".



# Conclusioni

- L'applicazione risponde bene agli obiettivi prefissati;
- Esperienza molto costruttiva e preziosa;
- In futuro, opportuna integrazione di:
  - ▶ Altri servizi di infomobilità
  - ▶ Servizi meteorologici
- Ringraziamo l'STLAB per il costante supporto.

