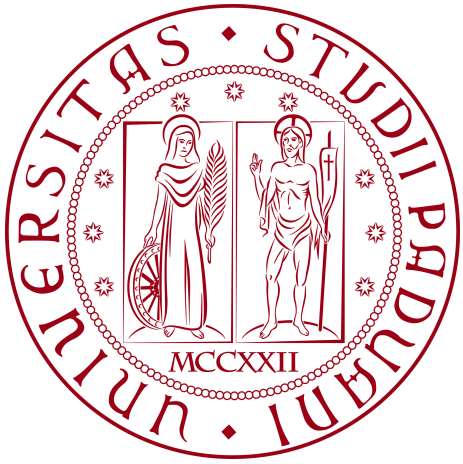
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



Sviluppo del Frontend di TripHippie: una applicazione web per il Car Pooling.

Tesi di Laurea Triennale

Relatore

Prof. Bresolin Davide

Laureando

Marco Gobbo

Matricola 2042362

Anno Accademico 2023-2024

Ringraziamenti

Desidero esprimere la mia gratitudine al professor Bresolin Davide, mio relatore, per l'aiuto e il sostegno che mi ha dato durante la stesura dell'elaborato.

Vorrei anche ringraziare, con affetto, i miei genitori, mio fratello e i miei nonni per il loro sostegno, il grande aiuto e la loro presenza in ogni momento durante gli anni di studio.

Desidero poi ringraziare tutti i miei amici, in modo particolare Tommaso e Giovanni Mauro, per i bellissimi anni trascorsi insieme e le mille avventure vissute.

Padova, settembre 2024 *Marco Gobbo*

Sommario

Questo documento descrive in modo dettagliato il tirocinio da me svolto presso l'azienda Sync Lab S.r.l. nella sede di Padova nel periodo che va dal 03/06/2024 al 26/07/2024.

La durata totale del tirocinio è stata di 8 settimane da 40 ore di lavoro ciascuna, per un totale di 320 ore complessive.

Lo studente si è visto impegnato durante l'attività di stage nella creazione di un'app di car pooling, in particolare nella realizzazione front end di alcune delle maschere principali dell'applicazione come la schermata home, la sezione per l'autenticazione e la sezione inerente ai viaggi. Per fare questo sono state usate varie tecnologie, tra cui le principali sono state TypeScript e il framework Angular.

Il percorso di tirocinio si è composto prevalentemente da due parti ben distinte tra di loro, una prima parte di studio e ripasso sia di nuove tecnologie sia di concetti già studiati durante il corso di studi e una seconda parte in cui le cose studiate nel primo periodo sono state effettivamente applicate al progetto.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

1.1 L'azienda

Sync Lab è una Innovative Company attenta ai paradigmi della trasformazione digitale che realizza prodotti e soluzioni per diversi mercati quali: Sanità, Industria, Energia, Telco, Finanza, Trasporti e Logistica.
L'azienda propone un'offerta di consulenza specialistica, sintesi di circa 20 anni nel settore IT in diversi domini tecnologici quali: GDPR, Big Data, Cloud Computing, IoT, Mobile e Cyber Security.

Figura 1.1: Logo e sito della Sync Lab

In questo scenario Sync Lab si posiziona come system integrator nel panorama italiano delle aziende ICT.

L'azienda al momento è costituita da:

- Più di 300 dipendenti
- Più di 6 sedi sparse per tutta Italia rispettivamente a: Napoli, Roma, Milano, Padova, Verona, Como
- Più di 150 clienti diretti e finali tra cui spiccano: Eni, Enel, Vodafone, Trenitalia, Tim, Fastweb, Poste Italiane, Unicredit

Figura 1.2: Statistiche Sync Lab

1.2 L'idea

L'obiettivo di questo tirocinio è sviluppare un'app di car pooling, un sistema di condivisione dell'auto tra più persone che devono percorrere lo stesso tragitto, con l'intento di ridurre il numero di veicoli sulle strade e diminuire la congestione del traffico. Nonostante l'esistenza di diverse app di car pooling, il problema della congestione e del traffico nei weekend, specialmente per chi si sposta verso mete di svago come il mare o la montagna, rimane ancora rilevante.
L'idea alla base di questa nuova app è di provare a creare un prodotto web che combini la funzionalità del car pooling con la possibilità di favorire l'incontro tra persone, ispirandosi all'Hippie Trail, un fenomeno degli anni '60 e '70 in cui gruppi di viaggiatori percorrevano rotte alternative verso l'Asia, condividendo esperienze e creando legami sociali lungo il cammino.

Oltre alle classiche funzionalità di un carpooling l'app può fornire:

- Profilazione utenti;
- Disponibilità ad approfondire conoscenza tra passeggeri;
- Autostop digitale;
- Possibilità di pranzo/cena tra il gruppo in viaggio (pizza, ristorante, sushi);
- Possibilità di inviare inviti di viaggio privati tra utenti;
- La scelta (consapevole) di accettare passaggi durante il percorso.

TripHippie vuole, per chi desidera, essere un modo alternativo per conoscere altre persone attraverso un percorso in auto.
Il percorso di tirocinio ha previsto una parte di implementazione delle maschere di front end principali dell'applicazione come, ad esempio, il login/registrazione di un utente oppure la visualizzazione/modifica/cancellazione di una proposta di viaggio.

1.3 Quello che ho fatto

Durante il mio tirocinio, ho svolto diverse attività che si possono raggruppare in due fasi principali.

Nella prima fase, che ha coperto le prime quattro settimane, mi sono focalizzato sull'acquisizione delle competenze tecniche necessarie per sviluppare il progetto, in pratica questo primo periodo è servito come periodo di studio in preparazione al secondo periodo dove avrei dovuto effettivamente implementare nel concreto le nozioni acquisite. Come prima cosa ho iniziato con un ripasso della metodologia Agile/Scrum e del linguaggio Java SE, per poi finire con il ripasso dei concetti Web come Servlet, servizi REST, JSON ecc. Inoltre, ho studiato in modo più approfondito Spring Core e Spring Boot, fondamentali

per il back-end. Ho anche dedicato tempo a migliorare la mia conoscenza in materia di linguaggi e framework utilizzabili per il front-end, come JavaScript, TypeScript, NodeJS e Angular.

La seconda fase, che ha occupato le ultime quattro settimane, è stata dedicata invece all'implementazione delle principali funzionalità dell'applicazione. Ho iniziato progettando e sviluppando le schermate per la gestione del profilo utente, inclusi i processi di login e registrazione. Successivamente, mi sono concentrato sulla creazione delle schermate per l'inserimento e la modifica dei viaggi, scrivendo anche i servizi necessari per collegare il front end al back end, inoltre ho anche implementato delle mappe interattive con il framework leaflet. Alla fine, ho completato l'implementazione delle diverse interfacce previste dal progetto, integrandole e testandole per assicurare che l'applicazione funzionasse correttamente

1.4 Struttura del documento

Il documento qui presente è separato in 6 capitoli differenti, ognuno dei quali copre un diverso aspetto di quanto fatto durante il periodo di tirocinio:

Il primo capitolo serve a dare un'introduzione al documento, descrive l'azienda e il progetto.

Il secondo capitolo descrive che cosa sia il car Pooling nello specifico, andando ad evidenziare il come mai sia necessaria un'altra app di car pooling.

Il terzo capitolo approfondisce l'attività di tirocinio in generale fornendo maggiori dettagli sul piano di lavoro e sugli obiettivi da completare.

Il quarto capitolo entra più nel dettaglio degli strumenti e le tecnologie principali utilizzate all'interno del progetto.

Il quinto capitolo approfondisce la progettazione e la codifica di "TripHippie" mostrando anche le maschere prodotte.

Nel sesto capitolo si riporta una conclusione delle attività svolte con alcune considerazioni personali sul percorso di tirocinio in generale.

Capitolo 2

Car Pooling

In questo capitolo viene descritto nel dettaglio lo stato dell'arte del dominio applicativo del progetto: il Car Pooling.

2.1 Descrizione

Il car pooling, o covetturaggio, è una modalità di trasporto in cui più persone condividono un'auto privata per ridurre i costi e contribuire alla mobilità sostenibile. È particolarmente diffuso in contesti lavorativi o universitari, dove persone che percorrono la stessa tratta si accordano per viaggiare insieme, condividendo le spese.

Il car pooling offre vari vantaggi, come la riduzione del traffico, il risparmio economico e la diminuzione dell'inquinamento. Tuttavia, presenta anche delle sfide, come la ridotta flessibilità negli orari e la sicurezza/fiducia nel conducente.

Negli ultimi anni, il car pooling ha visto un notevole incremento grazie a piattaforme digitali che mettono in contatto chi offre e chi cerca un passaggio. Queste piattaforme aiutano a superare la diffidenza iniziale verso l'idea di viaggiare con sconosciuti.

In Italia, il car pooling è più diffuso nel Nord e nelle aree metropolitane, con una particolare concentrazione in Lombardia, Lazio e Piemonte. È praticato soprattutto da giovani, con una crescente partecipazione femminile. Tuttavia, nel Sud Italia e in Sardegna, la pratica è ancora poco diffusa.

Esistono diverse tipologie di car pooling, ognuna con le sue caratteristiche che la contraddistinguono:

- **Carpooling**: Questo è il modo più semplice e comune di condividere un viaggio. Di solito, un gruppo di persone che devono andare nello stesso posto (come amici oppure colleghi di lavoro) si organizza per condividere un'auto privata. Spesso il carpooling nasce in maniera spontanea e informale, ma grazie alle piattaforme digitali è diventato più facile anche per chi non si conosce mettersi d'accordo e viaggiare insieme.

- **Ridesharing Company**: Qui parliamo di aziende che hanno cambiato il mondo della mobilità grazie alle loro app. Queste piattaforme mettono in contatto conducenti e passeggeri senza possedere veicoli propri. Le più famose, come Uber e Lyft, non si limitano al carpooling ma funzionano più come taxi, con un autista che porta i passeggeri a destinazione a pagamento.

- **Peer-to-Peer Carsharing**: Questa è un'idea più avanzata e flessibile di condivisione dell'auto. In pratica, chi possiede un'auto può affittarla a chi ne ha bisogno tramite una piattaforma online. Questo sistema, basato su una logica peer-to-peer, permette di utilizzare auto che altrimenti resterebbero ferme, riducendo la necessità di possederne una propria.

La principale differenza tra il *carpooling* tradizionale e il *peer-to-peer carsharing* sta nel fatto che, nel primo caso il conducente è al tempo stesso il proprietario e condivide la propria auto con le persone che l'hanno richiesta, guidando effettivamente lui il veicolo. Nel secondo caso invece la macchina viene semplicemente messa a disposizione del passeggero che può usufruirne anche senza il proprietario a bordo.

2.1.1 Vantaggi e Sfide del Car Pooling

Il car pooling ha numerosi vantaggi:

- **Riduzione dei Costi:** Il car pooling è sicuramente una delle soluzioni più economiche per viaggiare, sia per il passeggero che per il conducente.

Le spese che si possono dividere possono essere ad esempio carburante, pedaggi ecc.

- **Impatto Ambientale:** Il car pooling aiuta sicuramente a controllare e limitare le emissioni di gas serra dovute alla massiccia circolazione di automobili per le strade di tutto il mondo. Secondo alcune ricerche condotte dal jojob nel 2023 in Italia sono stati effettuati 373.767 viaggi condividendo l'auto privata tra colleghi risparmiando così 4.931.175 km, che hanno evitato l'emissione di 641 tonnellate di CO2 e generato un risparmio economico di quasi 1 milione di euro.

- **Benefici Sociali:** Il car pooling è un'occasione per creare interazioni sociali. Condividere il viaggio con altre persone, oltre a portarti a destinazione, potrebbe essere un'opportunità per conoscere nuove persone scambiandosi idee e creando connessioni sociali.

Tuttavia, il car pooling non è esente da sfide:

- **Coordinamento e Flessibilità:** Uno dei principali ostacoli alla diffusione del car pooling ha sicuramente a che fare con il coordinamento e la flessibilità degli impegni tra le persone. Non utilizzando una propria macchina non è possibile, la maggior parte delle volte, scegliere un orario preciso in cui mettersi in viaggio, dovendo invece aspettare il conducente che molto spesso potrebbe non essere immediatamente disponibile.

- **Sicurezza e Fiducia:** Andare in viaggio con una persona che non si conosce sicuramente per alcuni potrebbe risultare un ostacolo. Le piattaforme digitali come, ad esempio, BlaBlaCar mettono a disposizione degli utenti un sistema di recensioni in modo tale l'utente prima di scegliere il suo autista possa farsi un'idea di che persona il conducente possa essere rimanendo più tranquilla durante il viaggio.

- **Accettazione Culturale:** In alcune culture, l'idea di condividere il proprio spazio privato, come un'auto, con sconosciuti non è ancora del tutto accettata. Questo limita l'adozione del car pooling in determinati contesti geografici e sociali.

2.2 Piattaforme esistenti

Negli ultimi anni, l'avvento del car pooling ha preso sempre più piede anche e soprattutto grazie alle varie piattaforme digitali esistenti:

- **BlaBlaCar:** BlaBlaCar è l'app leader mondiale per i viaggi comunitari, con 27 milioni di membri attivi all'anno in 21 paesi. La sua tecnologia abbina conducenti con posti vuoti a passeggeri che si dirigono nella stessa direzione, permettendo loro di condividere i costi del viaggio. Con la missione di diventare il punto di riferimento per i viaggi condivisi, BlaBlaCar combina il carpooling con i viaggi in autobus di oltre 5.000 operatori, offrendo un'ampia scelta di soluzioni di viaggio economiche e sostenibili, tutte in un'unica app.
  Nel 2023, la comunità di BlaBlaCar ha connesso 2,4 milioni di punti d'incontro in tutto il mondo, permettendo 104 milioni di incontri tra persone. I conducenti che hanno usato il carpooling hanno risparmiato 513 milioni di euro, e tutti i servizi di mobilità di BlaBlaCar hanno contribuito a evitare 2 milioni di tonnellate di emissioni di CO2.

- **Waze Carpool:** Nel 2009, Waze è stata lanciata come un'app di navigazione basata sul crowdsourcing, progettata per migliorare l'esperienza di guida attraverso la collaborazione degli utenti. Con oltre 140 milioni di utenti che contribuiscono aggiornando le mappe e segnalando problemi, l'app fornisce indicazioni precise e tempestive, rendendo il traffico più gestibile. La community di "Waze Carpool" e le collaborazioni con città e autorità aiutano a ottimizzare i percorsi, risparmiando tempo e aumentando la sicurezza sulla strada.

- **LiftShare:** LiftShare è una piattaforma britannica che facilita il carpooling, permettendo agli utenti di condividere viaggi in auto con altre persone che percorrono la stessa rotta. L'obiettivo principale dell'applicazione è quello di connettere tra di loro persone che viaggiano nella stessa direzione, in questo modo posso allo stesso tempo: spartirsi i costi, ridurre l'inquinamento atmosferico e ridurre il traffico sulle strade.

2.3 Perché un'altra app di Car Pooling

Come visto nel paragrafo sopra le applicazioni di car pooling esistono e sembrerebbero essere anche molto funzionali, ma allora perché c'è bisogno di un'altra app di car pooling?
Le lacune lasciate dalle app sopra citate purtroppo esistono esse, infatti, hanno come target principale i pendolari o comunque utenti che vanno su tratte ben coperte, non pensando a chi viaggia magari durante i weekend per puro piacere personale. In situazioni come queste il traffico e la congestione stradale sono ancora problemi attuali e a cui sembra non esserci soluzione. Inoltre, le app esistenti tendono a ridurre il car pooling a un semplice servizio di trasporto, senza valorizzare l'aspetto sociale del viaggio. Mancano strumenti che incentivino realmente l'interazione e la creazione di legami tra i viaggiatori.

TripHippie nasce proprio per colmare queste carenze. L'app mette a disposizione una lunga serie di opzioni per gli utenti come la disponibilità ad approfondire una conoscenza tra passeggeri, la possibilità di fare Autostop digitale, la possibilità di profilazione degli utenti, l'occasione di pranzare/cenare tra il gruppo in viaggio ecc. Tutto questo rende l'app dinamica e coinvolgente, diversa dalle vecchie app di car pooling. Lo scopo di TripHippie, dunque, non è solo quello di portare la persona da un certo punto A ad un altro punto B ma è anche e soprattutto quello di creare connessioni umane tra gli utenti, instaurare veri e propri rapporti utilizzando come forma di viaggio quella dell'Hippie Trail.

Capitolo 3

Il progetto di tirocinio

In questo capitolo viene presentato più nel dettaglio il tirocinio, più nello specifico gli obiettivi attesi con relativo piano di lavoro e cambiamenti allo stesso.

3.1 Piano di Lavoro

Durante la fase preliminare dell'attività di tirocinio è stato definito un piano di lavoro nel quale sono presenti le attività svolte durante tutto il periodo di stage, tutto questo è stato stabilito insieme all'azienda Sync Lab S.r.l. .
Il piano di lavoro è suddiviso settimanalmente e comprende un monte ore di 320 ore totali suddiviso in 8 settimane.
Le attività stabilite inizialmente nel piano di lavoro qui presentato hanno subito delle leggere variazioni durante il periodo di tirocinio, questi cambiamenti sono definiti all'interno del paragrafo Variazioni/Aggiunte alle attività del piano di lavoro.

3.1.1 Prima Settimana (40 ore)

Svolta dal 3 giugno al 7 giugno.
Le attività proposte sono state:

- Incontro con le persone coinvolte nel progetto per discutere dei requisiti e delle richieste relativamente al sistema da sviluppare;

- Presentazione degli strumenti di lavoro per la condivisione del materiale di studio e per la gestione dell'avanzamento;
- Condivisione della scaletta degli argomenti;
- Ripasso dei concetti di Metodologia Agile/Scrum;
- Ripasso del linguaggio Java SE;
- Ripasso dei concetti Web (Servlet, servizi REST, JSON, ecc.);
- Studio dei principi generali di Spring Core (IoC, Dependency Injection).

3.1.2 Seconda Settimana (40 ore)

Svolta dal 10 giugno al 14 giugno.
Le attività proposte sono state:

- Studio SpringBoot;
- Studio Spring Data/DataRest.

3.1.3 Terza Settimana (40 ore)

Svolta dal 17 giugno al 21 giugno.
Le attività proposte sono state:

- Ripasso linguaggio Javascript;
- Studio del linguaggio TypeScript.

3.1.4 Quarta Settimana (40 ore)

Svolta dal 24 giugno al 28 giugno.
Le attività proposte sono state:

- Studio piattaforma NodeJS e AngularCLI;
- Studio framework Angular.

- Implementazione di un mini-prototipo di una maschera in Angular con 5 campi generici di input con chiamata al back end tramite un service dedicato.

3.1.5 Quinta Settimana (40 ore)

Svolta dal 1 luglio al 5 luglio.
Le attività proposte sono state:

- Analisi e studio del progetto TripHippie;
- Progettazione ed implementazione della nuova maschera di gestione profilo/login/registrazione utente.

3.1.6 Sesta Settimana (40 ore)

Svolta dal 8 luglio al 12 luglio.
Le attività proposte sono state:

- Progettazione ed implementazione nuova maschera "Inserimento/Modifica Viaggi";
- Scrittura dei service (su front-end) di chiamata al back-end.

3.1.7 Settima Settimana (40 ore)

Svolta dal 15 luglio al 19 luglio.
Le attività proposte sono state:

- Termine implementazioni maschere;
- Scrittura dei service (su front-end) di chiamata al back-end.

3.1.8 Ottava Settimana (40 ore)

Svolta dal 22 luglio al 26 luglio.
Le attività proposte sono state:

- Termine integrazioni e collaudo finale.

Dal piano di lavoro qui presentato possiamo notare principalmente due periodi:

- Un primo periodo che comprende le prime 4 settimane e che si focalizza principalmente sullo studio e sulla comprensione delle tecnologie da utilizzare e sul come utilizzarle al meglio
- Un secondo periodo che comprende le ultime 4 settimane e che si focalizza sull'implementazione e creazione delle maschere dell'applicazione tramite le competenze acquisite durante la prima parte di tirocinio

3.2 Obiettivi

Si farà riferimento ai requisiti secondo le seguenti notazioni:

- **OB** per i requisiti obbligatori, vincolanti in quanto obiettivo primario richiesto dal committente;
- **DE** per i requisiti desiderabili, non vincolanti o strettamente necessari, ma dal riconoscibile valore aggiunto;
- **OP** per i requisiti opzionali, rappresentanti valore aggiunto non strettamente competitivo

Le sigle precedentemente indicate saranno seguite da un numero, identificativo del requisito. La tabella 3.1 indica gli obiettivi che sono stati identificati.

Obbligatorio	
OB 1	Acquisizione competenze sulle tematiche sopra descritte.
OB 2	Capacità di raggiungere gli obiettivi richiesti in autonomia seguendo il crono programma.
OB 3	Portare a termine le implementazioni previste con una percentuale di superamento pari al 80% (equivalente alla maschera di Gestione profilo Utente e loro service con le chiamate al backend).
Desiderabile	
DE 1	Portare a termine le implementazioni previste con una percentuale di superamento pari al 100% (equivalente alla maschera di Gestione Profilo Utente, Gestione Viaggio e loro service con le chiamate al backend).
Opzionale	
OP 1	Apportare un valore aggiunto al gruppo di lavoro durante le fasi di progettazione delle interfacce.

Tabella 3.1: Obiettivi del tirocinio

3.2.1 Requisiti non formalizzati

Durante il periodo di tirocinio non è stata fatta una vera e propria analisi dei requisiti; tuttavia, per poter procedere in modo ordinato con il lavoro e con il resto del progetto ho ritenuto opportuno stilare comunque una lista di requisiti anche se non formalizzati, la quale mi è stata di aiuto durante lo svolgimento del progetto.

Qui di seguito i requisiti individuati per il progetto:

- L'utente può accedere all'area personale con username e password
- L'utente visualizza un messaggio di errore se inserisce delle credenziali sbagliate
- L'utente può registrarsi creando un proprio account
- L'utente può navigare nella sezione "Cerca un Viaggio!" una volta loggato
- L'utente può visualizzare tutti i viaggi disponibili nella sezione "Cerca un Viaggio!"
- L'utente può filtrare i viaggi per "Data Inizio" e/o "Data Fine"
- L'utente può visualizzare i dettagli di un determinato viaggio, come le tappe ad esso associate
- L'utente può navigare nella "Area Personale" una volta loggato
- L'utente può eseguire l'operazione di logout
- L'utente può modificare il proprio account nella sezione "Il mio profilo"
- L'utente può eliminare il proprio account nella sezione "Il mio profilo"
- L'utente può aggiungere una foto profilo al suo account nella sezione "Il mio profilo"
- L'utente può modificare la foto profilo del suo account nella sezione "Il mio profilo"
- L'utente può eliminare la foto profilo del suo account nella sezione "Il mio profilo"
- L'utente può iniziare la procedura di creazione di un viaggio nella sezione "Inserisci un viaggio!"
- L'utente può creare un viaggio senza tappe intermedie nella sezione "Inserisci un viaggio!"

- L'utente può inserire una o più tappe intermedie ad un viaggio nella sezione "Inserisci un viaggio!"
- L'utente può creare un viaggio con tappe intermedie nella sezione "Inserisci un viaggio!"
- L'utente può visualizzare i viaggi da lui creati nella sezione "I miei viaggi!"
- L'utente può modificare i viaggi da lui creati nella sezione "I miei viaggi!"
- L'utente può eliminare i viaggi da lui creati nella sezione "I miei viaggi!"
- L'utente può visualizzare le tappe associate ad un singolo viaggio da lui creato nella sezione "I miei viaggi!"
- L'utente può modificare le tappe associate ad un singolo viaggio da lui creato nella sezione "I miei viaggi!"
- L'utente può eliminare le tappe associate ad un singolo viaggio da lui creato nella sezione "I miei viaggi!"

3.3 Variazioni/Aggiunte alle attività del piano di lavoro

Durante il percorso di tirocinio, alcune attività previste dal piano di lavoro hanno subito delle variazioni.
Nella seconda settimana, oltre allo studio di Spring Boot e di Spring Data/Data Rest, ho implementato un prototipo di backend utilizzando queste tecnologie. Il tema del prototipo era libero, permettendomi di scegliere le entità da sviluppare. Grazie a questa flessibilità, sono riuscito a soddisfare rapidamente la richiesta del tutor aziendale. Di conseguenza, la seconda settimana è così diventata:

3.3.1 Seconda Settimana (40 ore)

Svolta dal 10 giugno al 14 giugno.
Le attività proposte sono state:

- Studio SpringBoot;
- Studio Spring Data/DataRest.
- Implementazione di un prototipo di backend utilizzando le competenze acquisite da SpringBoot e Spring Data/DataRest.

All'inizio della Settima settimana tutti gli obiettivi e le maschere da implementare erano state completate, così di comune accordo con il tutor aziendale abbiamo deciso di aggiungere un'ulteriore attività al di fuori del piano di lavoro originale ovvero l'implementazione di mappe interattive tramite la libreria LeafLet di Javascript. Questa implementazione è stata, come detto precedentemente, una cosa in più per quanto riguarda il piano di lavoro andando a modificare quindi la settima settimana:

3.3.2 Settima Settimana (40 ore)

Svolta dal 15 luglio al 19 luglio.
Le attività proposte sono state:

- Studio libreria Leaflet
- Implementazione di mappe interattive tramite la libreria Leaflet

Capitolo 4

Strumenti e tecnologie

In questo capitolo viene data un'analisi dettagliata di tutti i Linguaggi, Framework, librerie e strumenti principali utilizzati durante il periodo di tirocinio.

4.1 Linguaggi

4.1.1 HTML

Figura 4.1: Logo HTML

HTML (HyperText Markup Language) è il linguaggio di markup più elementare del Web. Definisce la composizione e la struttura dei contenuti web. Altre tecnologie, oltre a HTML vengono generalmente utilizzate per descrivere l'aspetto/presentazione di una pagina web (CSS) o la sua funzionalità/comportamento (JavaScript).
HTML utilizza "markup" per annotare testo, immagini e altri contenuti da visualizzare in un browser web. Il markup HTML include elementi speciali co-

me <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , , <img>, <aside>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <search>, <output>, <progress>, <video>, <ul>, <ol>, <li> e molti altri.
Un elemento HTML è distinto dagli altri testi in un documento tramite "tag", che consistono nel nome dell'elemento racchiuso tra "<" e ">".
All'interno del progetto è stato utilizzato per creare le pagine web dei componenti Angular.

4.1.2 CSS

Figura 4.2: Logo CSS

CSS (Cascading Style Sheets) è un linguaggio di markup di stile utilizzato per la presentazione di un documento scritto in HTML. CSS descrive come gli elementi devono essere visualizzati sullo schermo. CSS è uno dei linguaggi fondamentali del web ed è standardizzato nei browser web secondo le specifiche del W3C.
All'interno del progetto è stato utilizzato per dare uno stile generale al contenuto delle pagine web e per cambiare lo stile di alcuni componenti predefiniti tramite PrimeNG.

4.1.3 Typescript

Figura 4.3: Logo Typescript

TypeScript è un linguaggio di programmazione open-source sviluppato da Microsoft che estende JavaScript aggiungendo il supporto per la tipizzazione statica. Questo permette ai programmatori di rilevare errori direttamente nell'editor prima di eseguire il codice, migliorando la sicurezza e la manutenibilità del codice stesso. TypeScript è un "superset" di JavaScript, il che significa che tutto il codice JavaScript valido è automaticamente valido anche in TypeScript. Il codice TypeScript viene poi compilato in JavaScript, rendendolo eseguibile ovunque sia supportato JavaScript.
All'interno del progetto è stato utilizzato come linguaggio di programmazione principale di Angular per la creazione di interfacce web.

4.1.4 Java

Figura 4.4: Logo Java

Java è un linguaggio di programmazione e una piattaforma di elaborazione essenziale per lo sviluppo di software. È utilizzato in milioni di dispositivi, dai

computer ai telefoni cellulari, passando per le console videogiochi.
All'interno del progetto è stato usato nella creazione del prototipo di backend.

4.2 Framework e Librerie

4.2.1 Spring Boot

Figura 4.5: Logo Spring Boot

Spring Boot è un framework open-source che facilita lo sviluppo di applicazioni basate su Java, integrando il potente Spring Framework. Esso semplifica la configurazione delle applicazioni, permettendo di partire rapidamente grazie a un sistema di "auto-configurazione" che riduce la necessità di configurazioni manuali. Spring Boot è progettato per creare applicazioni standalone, complete di server web integrato come Tomcat, e supporta la creazione di microservizi, rendendolo una scelta popolare per l'architettura moderna delle applicazioni.
All'interno del progetto è stato utilizzato per creare la parte di backend, nel mio caso in rapporto ai miei compiti è stato utilizzato per poter comprendere il backend e poterne creare un prototipo.

4.2.2 Angular

Figura 4.6: Logo Angular

Angular è un framework di sviluppo web open-source, creato da Google, che utilizza TypeScript. Viene impiegato per costruire applicazioni web dinamiche e complesse grazie alla sua capacità di gestire sia la parte frontend che la logica dell'applicazione. Il framework Angular è utilizzato per creare delle Single Page Applications (anche dette SPA). Il concetto fondamentale è che non ci sia più una pagina HTML per ogni pagina web del sito ma al contrario esisterà un'unica pagina HTML nella quale a seconda di che cosa bisogna far vedere all'utente farà vedere o meno certi componenti che insieme formeranno la pagina.
All'interno dell'applicazione è stato utilizzato come framework principale per il front end ed è stato il framework che ho utilizzato maggiormente durante il tirocinio.

4.2.3 PrimeNG

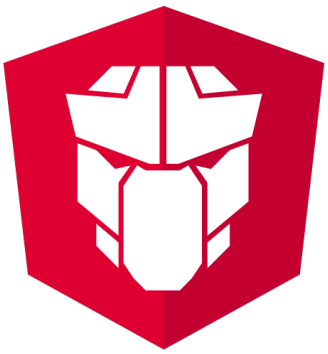


Figura 4.7: Logo PrimeNG

PrimeNG è una libreria completa di componenti UI per Angular, sviluppata da PrimeTek. Offre un'ampia gamma di componenti ricchi di funzionalità, altamente personalizzabili, per migliorare le applicazioni web. Inoltre, PrimeNG fornisce blocchi UI pronti all'uso per costruire rapidamente applicazioni spettacolari, oltre a template professionali per avviare progetti con stile.
All'interno del progetto è stata utilizzata per utilizzare elementi come ad esempio bottoni, contenitori, pop-up già predefiniti pronti per essere utilizzati. In molti casi è successo che utilizzassi il css per modificare degli stili già impostati di alcuni di questi elementi.

4.2.4 Leaflet

Figura 4.8: Logo Leaflet

Leaflet è una libreria JavaScript open-source leggera, ideale per creare mappe interattive ottimizzate. Con un peso di circa 42 KB, offre tutte le funzionalità essenziali per la creazione di mappe, come livelli di piastrelle, marker, polilinee, popup. È progettata per essere semplice da usare, altamente performante, e facilmente estensibile tramite numerosi plugin. Leaflet è compatibile con tutte

le principali piattaforme desktop e mobili.
All'interno del progetto è stata utilizzata per inserire le coordinate geografiche di determinati luoghi scelti dall'utente che rappresentano: il punto di partenza, il punto di arrivo e le possibili tappe intermedie.

4.3 Strumenti

4.3.1 IntelliJ

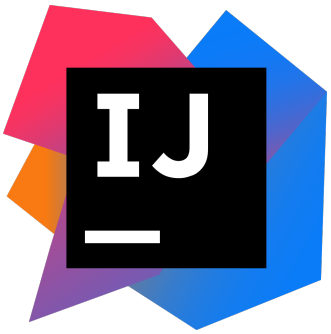


Figura 4.9: Logo IntelliJ

IntelliJ IDEA è un ambiente di sviluppo integrato (IDE) per Java, sviluppato da JetBrains. Rilasciato per la prima volta nel 2001, è stato pionieristico per funzionalità come la navigazione del codice e il refactoring. Disponibile sia in versione open-source che commerciale, IntelliJ IDEA è stato anche la base per Android Studio e altri IDE JetBrains. È riconosciuto per la sua efficienza nello sviluppo Java, ottenendo punteggi elevati in vari confronti tra IDE.
All'interno del progetto, per la parte di backend è stato utilizzato questo specifico ambiente di sviluppo. Anche se, avendo lavorato principalmente come frontendista, questo non è stato il mio ambiente di sviluppo principale, la sua integrazione è risultata fondamentale per il successo complessivo del progetto.

4.3.2 Visual Studio Code

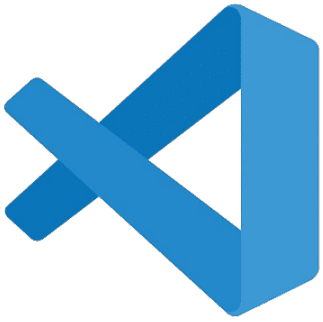


Figura 4.10: Logo Visual Studio Code

Visual Studio Code (VS Code) è un editor di codice sorgente sviluppato da Microsoft per Windows, Linux e macOS. È gratuito e open-source con funzionalità come debugging, controllo Git integrato, IntelliSense e supporto per vari linguaggi di programmazione. Visual Studio Code è anche molto utile per la sua propensione ad essere integrato con estensioni oppure temi personalizzati che rendono l'esperienza di codifica più gradevole e semplice.
All'interno del progetto è stato utilizzato come ambiente di sviluppo principale per la creazione frontend delle maschere dell'applicazione.

4.3.3 Postman

Figura 4.11: Logo Postman

Postman è un ambiente di sviluppo API che permette agli sviluppatori di inviare e gestire richieste API, effettuare debug e testare endpoint. Supporta vari tipi di richieste (REST, SOAP, GraphQL) e formati di risposta (JSON, XML).

Offre funzionalità avanzate come l'organizzazione e la condivisione di richieste, la personalizzazione delle richieste con intestazioni e parametri, e la simulazione di risposte API senza un server reale. Postman è disponibile come applicazione standalone o estensione per Chrome, rendendolo uno strumento versatile per lo sviluppo collaborativo.
All'interno del progetto è stato utilizzato per testare le chiamate al backend tramite i relativi endpoint. È stata fondamentale per verificare, quando le cose nel frontend non funzionavano, che effettivamente nel backend invece funzionassero.

4.3.4 Stoplight

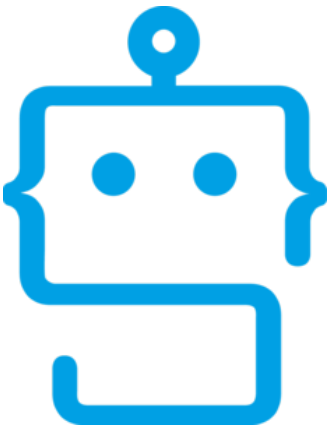


Figura 4.12: Logo Stoplight

Stoplight è una piattaforma progettata per facilitare l'intero ciclo di vita delle API, dalla progettazione alla documentazione, fino alla costruzione. Stoplight mette a disposizione un'interfaccia chiara ed intuitiva per la visualizzazione delle API andando a specificare: parametri, struttura, nomi precisi per i campi ecc. Inoltre, fornisce funzionalità per gestire e riutilizzare componenti API.
All'interno del progetto è stato utilizzato per visionare i documenti relativi agli endpoint delle API forniti dal backend, specificatamente per vedere quali fossero i requisiti degli endpoint e come fossero strutturati.

Capitolo 5

Progettazione e sviluppo

In questo capitolo viene data una visione più dettagliata del progetto in sé, più nello specifico vengono mostrate le maschere e il processo per realizzarle

5.1 Premessa Progettuale

Dopo le prime quattro settimane di studio delle tecnologie necessarie, nelle ultime quattro mi sono dedicato interamente allo sviluppo del front end, utilizzando Angular come framework principale.
Quando ho iniziato a lavorare, il back end era già in corso di sviluppo grazie ad un altro stagista, che aveva già messo in piedi lo "UserService" ovvero il servizio che gestisce tutte le operazioni degli utenti, come registrazione, login e gestione del profilo. Man mano che il progetto andava avanti è stato aggiunto anche il "TravelService", che si occupa di tutto ciò che riguarda i viaggi, come l'inserimento, la modifica e la visualizzazione.
Il back end veniva aggiornato settimanalmente, e ogni settimana ricevevo le nuove funzionalità integrandole nel front end. Il back end era costituito da un'architettura a microservizi, con un API Gateway per gestire le richieste e un Service Registry per tenere traccia dei servizi attivi, tutto questo facilitava il collegamento al front end che risultava fluido ed efficiente.
La mia responsabilità principale è stata quella di sviluppare l'intera interfaccia utente: ho progettato e implementato le maschere per la gestione degli utenti e dei viaggi, collegandole ai servizi del back end per garantire un funzionamento

ottimale dell'applicazione.
Ho lavorato a stretto contatto con lo stagista che si occupava del back end, e questo ci ha permesso di risolvere rapidamente eventuali problemi relativi alle corrispettive parti di progetto e di mantenere lo stesso in linea con le aspettative. È stato un lavoro di squadra che ha funzionato davvero bene.

5.2 Design Pattern utilizzati

Come detto nel paragrafo precedente, Angular è stato il framework principale con cui ho passato più tempo durante il tirocinio. Angular utilizza un pattern architetturale un po' diverso dal classico MVC, questo pattern si chiama MVVM.
Il Model–View–ViewModel (MVVM) è un pattern architetturale usato per separare la logica di presentazione dalla logica di business, in particolare nelle interfacce utente.

Il MVVM si compone di quattro componenti principali:

- **Model:** Rappresenta il dominio dell'applicazione e include i dati, la logica di business e di validazione.
- **View:** Definisce l'aspetto e il layout dell'interfaccia utente.
- **ViewModel:** Funziona come intermediario tra la View e il Model, gestendo la logica della vista e formattando i dati per la visualizzazione.
- **Binder:** Mantiene sincronizzati il ViewModel e la View, assicurando che le modifiche in uno si riflettano automaticamente nell'altro, semplificando il lavoro dello sviluppatore.

Il pattern è particolarmente utile per applicazioni con interfacce utente complesse, dove la separazione delle responsabilità tra dati, logica e presentazione è cruciale.

5.3 Architettura di un'applicazione Angular

Prima di parlare delle maschere prodotte durante il periodo di tirocinio è opportuno spiegare come sia strutturata un'applicazione in Angular. Essa come concetto principale ha quello di mostrare un'unica pagina HTML dove al suo interno cambi dinamicamente il contenuto a seconda dell'url digitato; quindi, a differenza di un normale sito web dove ad ogni pagina del sito è associata una pagina html, in un'applicazione Angular se ne utilizza solamente una.
A cambiare quindi non è la pagina HTML principale ma il contenuto della pagina ovvero i componenti. Ogni componente è formato da 3 file che sono:

- **nome.component.html:** Questo file contiene gli elementi che compongono il componente; un componente al suo interno può avere altri componenti che avranno a loro volta i 3 file.
- **nome.component.css:** Esso contiene il codice CSS che andrà ad influenzare lo stile del componente.
- **nome.component.ts:** La parte più importante del componente, contiene il codice in TypeScript del componente e si può dire che funzioni come il cervello del componente.

I componenti quindi a seconda dell'url vengono caricati all'interno della pagina HTML principale.
Degli esempi di componenti nella mia applicazione possono essere il login oppure la registrazione tutti e due con i loro 3 file.
Un'altra parte importante di un'applicazione Angular è sicuramente quella relativa ai servizi, essi sono composti solo da un file e permettono all'utente di poter connettere il back end con il front end. Degli esempi di servizi nell'applicazione sono "auth.service.ts" che gestisce tutto ciò che riguarda l'autenticazione oppure "profile.service.ts" che gestisce tutto ciò che è riguardante al profilo.

5.4 Maschere Realizzate

In questa sezione verranno mostrate più nel dettaglio le maschere prodotte durante il periodo di tirocinio, in particolare ci sarà una parte descrittiva della pagina e, dove più opportuno, sarà anche presente una parte di spiegazione del codice prodotto.

5.4.1 Home

Questa è la pagina di benvenuto dell'applicazione web. Essa mostra un messaggio di benvenuto e una descrizione dell'applicazione soffermandosi soprattutto al perché un utente dovrebbe scegliere TripHippie piuttosto che un'altra app di car pooling.
In alto viene mostrata il menù sottoforma di barra di navigazione, esso cambia a seconda che l'utente sia loggato o meno.
Se l'utente non è loggato il menù mostrato sarà composto solamente da "Home" e "Area Riservata"(Figura 5.1). Se al contrario, l'utente risulterà loggato, verrà mostrata un'altra voce nel menù ovvero "Cerca un Viaggio!"(Figura 5.2)

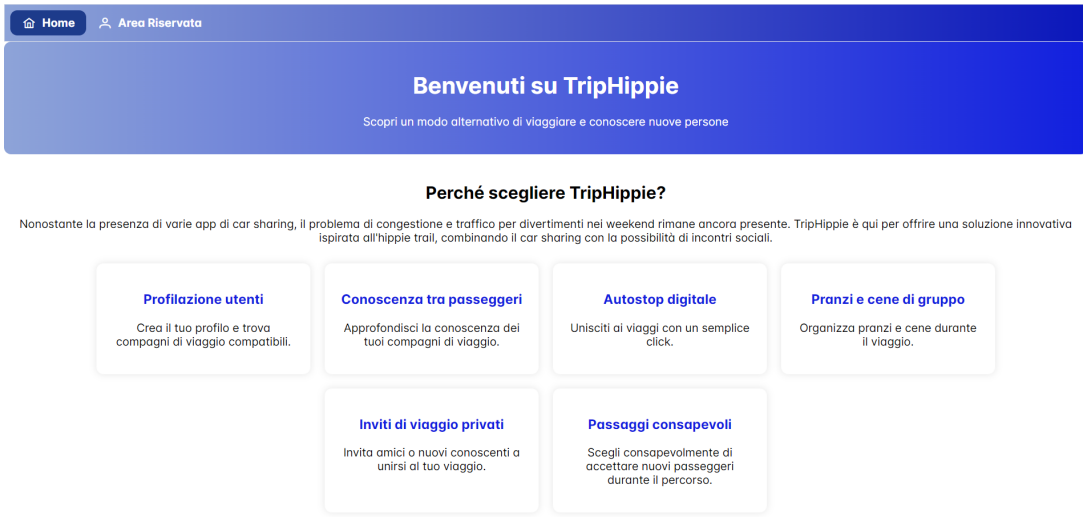


Figura 5.1: Pagina Home

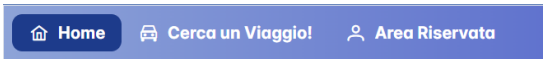


Figura 5.2: Menù con utente loggato

5.4.2 Login

Quando si clicca sulla voce del menù "Area Riservata" e non si è loggati si verrà reindirizzati alla pagina di login. Questa pagina è dedicata all'accesso degli utenti già in possesso di un account e quindi già registrati ma non ancora loggati nell'applicazione. All'interno della pagina sarà presente un form con "username" e "password", i campi non potranno essere vuoti, altrimenti il bottone per il login rimarrà disabilitato, inoltre le credenziali dovranno essere corrette. Se l'utente dovesse inserire delle credenziali errate verrà mostrato un messaggio di errore. Una volta terminato il login si verrà reindirizzati alla pagina home; tuttavia, essendo loggati comparirà una nuova voce nel menù come mostrato in figura 5.2 e il riferimento ad "Area riservata" sarà cambiato.

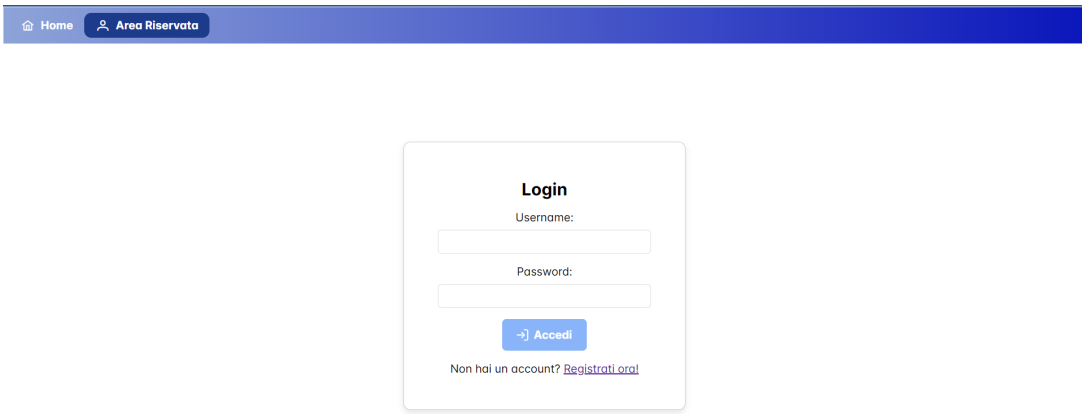


Figura 5.3: Pagina Login

Codifica

"LoginComponent" è uno dei componenti più importanti del progetto, esso è composto da 3 parti:

- HTML: All'interno dell'html è presente il form composto solamente da due campi ed un bottone. Il bottone è collegato alla funzione *onLogin(form: NgForm)* che passa i dati del form inseriti e chiama una funzione di "authService" ovvero *Login(user)*. AuthService non è altro che il servizio dedicato all'autenticazione che collega il front end con il back end ed esegue

la chiamata http://localhost:8765/api/users/login con il metodo POST, progettata per fare il login.

- TS: Per poter dichiarare un form in Angular e poter usarne gli elementi bisogna importare all'interno del "login.component.ts" i form facendo *import { FormsModule, NgForm } from '@angular/forms'*, all'interno di questo file sono inoltre descritte tutte le funzioni sopra citate tranne quelle del service.

Una volta fatto il login viene creato un token dal back end all'interno del localStorage, esso servirà per capire se l'utente è loggato o meno a seconda della presenza o meno del token.

5.4.3 Registrazione

Per tutti gli utenti non ancora registrati e quindi non in possesso di un account è possibile, compilando il form della pagina di registrazione, creare il proprio profilo. È necessario inserire tutti i campi obbligatori, che sono tutti eccetto la descrizione, così da sbloccare il tasto per registrarsi che altrimenti sarebbe disabilitato.
Una volta terminata la registrazione si verrà reindirizzati alla pagina di login dove si potranno inserire le credenziali appena create per entrare nell'account.

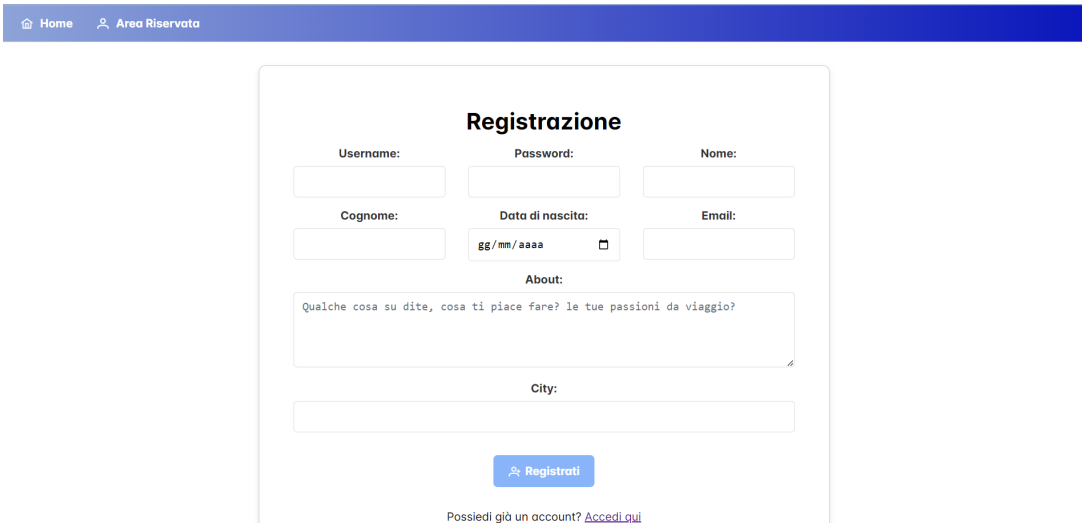


Figura 5.4: Pagina Registrazione

Codifica

La codifica di "RegistrazioneComponent" è molto simile strutturalmente a quella del login, sono presenti molti più campi all'interno del form e viene utilizzato di nuovo come servizio dedicato "authService". La chiamata viene fatta con http://localhost:8765/api/users con metodo POST e se va a buon fine l'utente viene creato. Per indirizzare alla pagina di login alla fine della procedura di registrazione si utilizza il "Router" o, meglio, una funzione di esso ovvero *this.router.navigateByUrl('/login')*.

5.4.4 Area Riservata

Se si clicca sulla voce del menù "Area Riservata" e non si è loggati, come descritto prima si verrà indirizzati nella pagina di Login (figura 5.3). Tuttavia, se si è loggati si verrà reindirizzati ad una pagina differente ovvero la pagina di gestione del profilo personale, essa è costituita da un componente principale costituito da un menù fisso sulla sinistra e uno spazio sulla destra che cambierà a seconda del contenuto da mostrare dalla quale si potranno eseguire determinate azioni differenti:

5.4.4.1 Il mio profilo

Selezionando la voce "Il mio profilo" del sottomenù situato nel lato sinistro della pagina "Area Riservata" sarà possibile visionare le proprie informazioni personali inerenti al profilo (figura 5.5), inoltre sarà possibile compiere due azioni principali:

5.4.4.1.1 Inserimento/Modifica/Cancellazione Immagine profilo

Tramite il bottone "Aggiungi Immagine" sarà possibile aggiungere un'immagine dal proprio computer come propria foto profilo, essa dovrà necessariamente essere in formato png.
Selezionando l'immagine essa verrà caricata all'interno del database effettuando una chiamata POST di "http://localhost:8765/api/users/id/profileImage" do-

ve id è l'id dell'utente ricavato dal token presente nel localStorage tramite la funzione getIdFromToken(token: string) di "profiloService". ProfiloService è un nuovo servizio dedicato utilizzato per tutte le chiamate che coinvolgono il profilo dell'utente. Una volta inserita l'immagine essa verrà visualizzata al posto del pulsante all'interno di un cerchio (figura 5.6) e sarà possibile tramite i relativi pulsanti posti sotto di essa modificarla inserendo un'altra immagine oppure eliminarla. La modifica e l'eliminazione sono eseguite dalle chiamate in DELETE e PUT allo stesso url di prima. Ad ogni ricaricamento della pagina viene fatta, tramite il metodo OnInit() ovvero un metodo che ti permette di eseguire del codice appena la pagina viene caricata, una chiamata GET sempre a http://localhost:8765/api/users/id/profileImage, se l'immagine è presenta nel database allora verrà caricata altrimenti non succederà nulla.

5.4.4.1.2 Modifica/Cancellazione del profilo

In fondo alla pagina del profilo sono posti due pulsanti, rispettivamente "Modifica Profilo" ed "Elimina Profilo".
Cliccando su "Elimina Profilo" l'account dell'utente verrà cancellato, ed insieme a questo tutte le proposte di viaggio create dall'utente. Verrà infatti eseguita una chiamata tramite "profiloService" di tipo DELETE a "http://localhost:8765/api/users/id" inoltre una volta cancellato l'utente si eseguirà il logout che non è altro che lo svuotamento del localStorage.
Cliccando invece sul pulsante di "Modifica Profilo" si verrà reindirizzati verso la pagina di modifica del profilo (figura 5.7) nella quale ci sarà un form, con i dati dell'utente già caricati nelle caselle di input, dove sarà possibile salvare le modifiche al profilo. Una volta premuto il bottone per salvare le modifiche al profilo verrà fatta una chiamata a "http://localhost:8765/api/users/id" con metodo PUT e le informazioni dell'utente cambieranno.

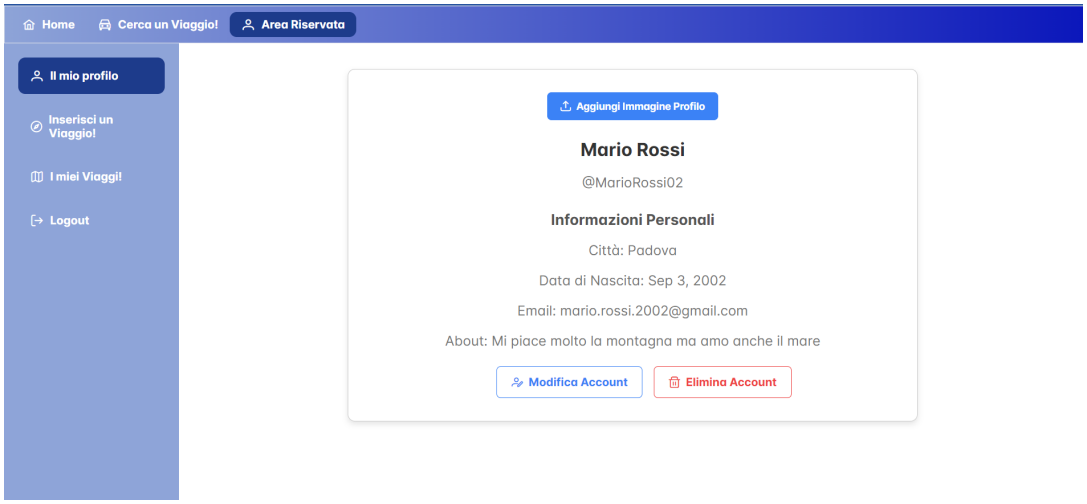


Figura 5.5: Pagina Profilo

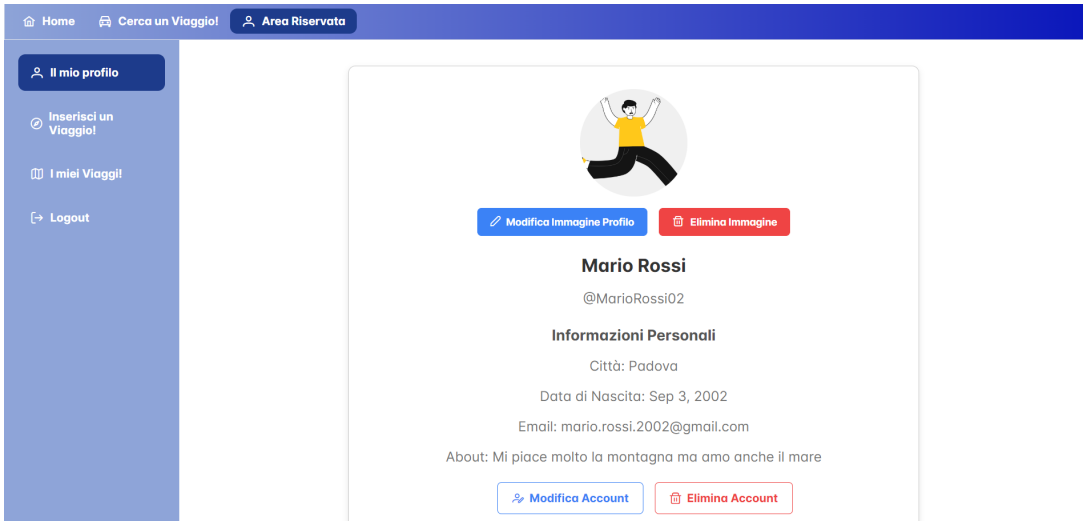


Figura 5.6: Pagina Profilo con immagine

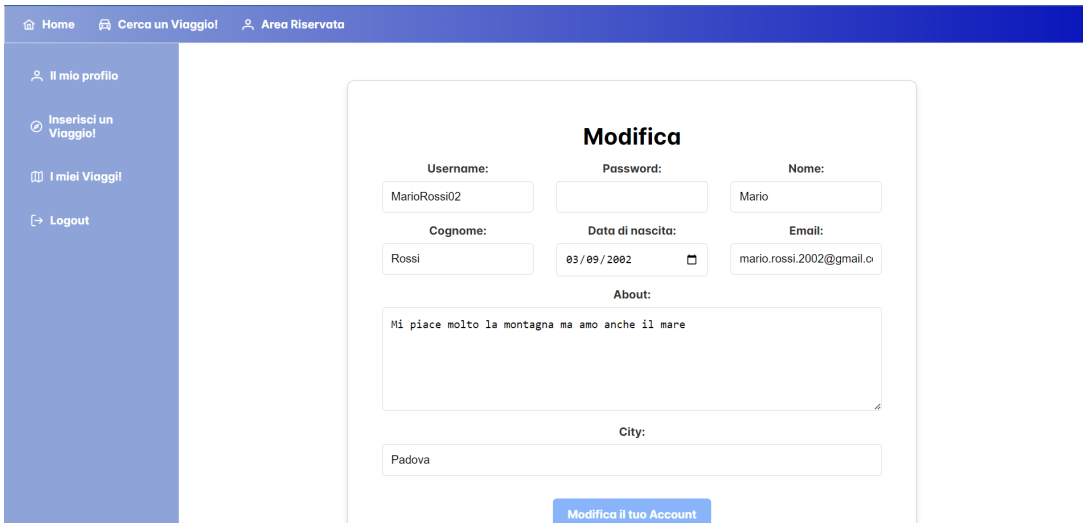


Figura 5.7: Pagina Modifica Profilo

5.4.4.2 Inserisci un Viaggio

Selezionando la voce nel menù "Inserisci un viaggio!" l'utente avrà l'occasione di poter creare una proposta di viaggio, visibile poi a tutti gli utenti. È importante notare come l'utente si troverà ancora nel componente "Area Riservata", l'unico componente che cambierà infatti sarà quello di destra dove verrà caricato il componente "CreaViaggioComponent" invece del precedente.
Per poter creare un viaggio ci sono due step fondamentali da seguire; Una prima parte dove si inseriscono i dati fondamentali del viaggio e una parte dove, se si vuole, si inseriscono le tappe intermedie.

5.4.4.2.1 Inserimento Dati fondamentali del Viaggio

Durante questa parte di inserimento del viaggio, verranno inseriti i dati fondamentali del viaggio (figura 5.8), in particolare verranno inseriti il punto di partenza e la destinazione, esse verranno inserite tramite l'ausilio di una mappa popup creata tramite il framework LeafLet (figura 5.9).
Una volta inseriti i dati del viaggio si preme il bottone di invio del form per continuare con la creazione delle tappe.

Codifica

"CreaViaggioComponent" ha al suo interno, nella parte HTML, un form dove è possibile inserire i dati del viaggio. Una particolarità di questo form è la mappa creata tramite il framework LeafLet.
Tramite la funzione toggleMap() la mappa viene caricata e mostrata come popup sullo schermo e tramite la funzione *addMarker(markerData: MarkerData)* viene inserito un marker di tipo markerData definito dall'omonima interfaccia *interface MarkerData {lat: number;lng: number;name: string;}*. Questo marker rappresenta un punto geografico sulla mappa scelto dall'utente tramite un click, i dati come la latitudine la longitudine e il nome vengono poi ricavati automaticamente tramite il punto inserito nella mappa.
Una volta che tutti i dati sono inseriti, tramite *this.router.navigateByUrl("/area-*

riservata/crea-tappa"); si accede alla schermata di inserimento delle tappe; i dati del viaggio vengono momentaneamente salvati all'interno del sessionStorage.

5.4.4.2.2 Inserimento Tappe Intermedie

Durante questa seconda parte di inserimento del viaggio l'utente può scegliere se aggiungere o meno delle tappe intermedie al suo viaggio. (figura 5.10)
Nel caso decidesse di non inserirle allora potrà cliccare sul tasto "Crea Viaggio senza Tappe!", in questo modo il viaggio verrà creato senza tappe intermedie.
Nel caso in cui invece l'utente volesse inserire delle tappe intermedie gli basterà aprire la mappa e selezionare i punti di interesse. Una volta inserite le tappe potrà cliccare sul tasto "Crea Viaggio con Tappe". Se questo pulsante viene schiacciato senza aver inserito delle tappe tramite la mappa allora verrà visualizzato un messaggio d'errore.

Codifica

All'interno del componente "CreaTappaComponent" ci sono principalmente due metodi:

- **createTrip():** Prende i dati del viaggio dal sessionStorage e li utilizza per fare una chiamata POST a http://localhost:8765/api/trips creando così il viaggio
- **createJourney():** Innanzitutto viene creato il Trip tramite il metodo createTrip() richiamato una sola volta all'inizio della funzione, poi in base al numero di punti inseriti nella mappa come tappe esse vengono create ciclicamente facendo una chiamata POST a http://localhost:8765/api/journeys.

Tutte le chiamate vengono eseguite tramite "TripService" ovvero il servizio dedicato che collega il back end al front end per tutte le operazioni riguardanti i viaggi.

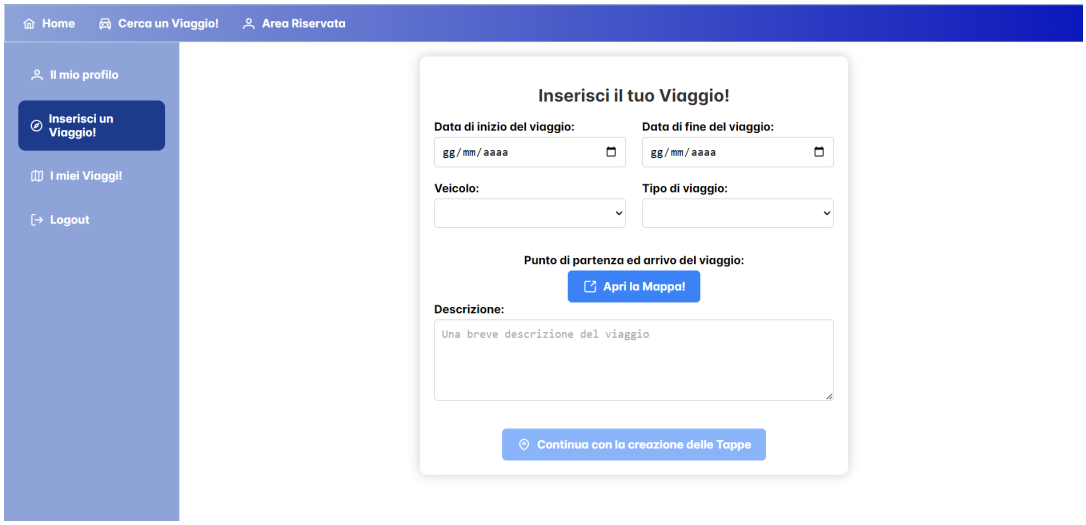


Figura 5.8: Pagina Inserimento Viaggio

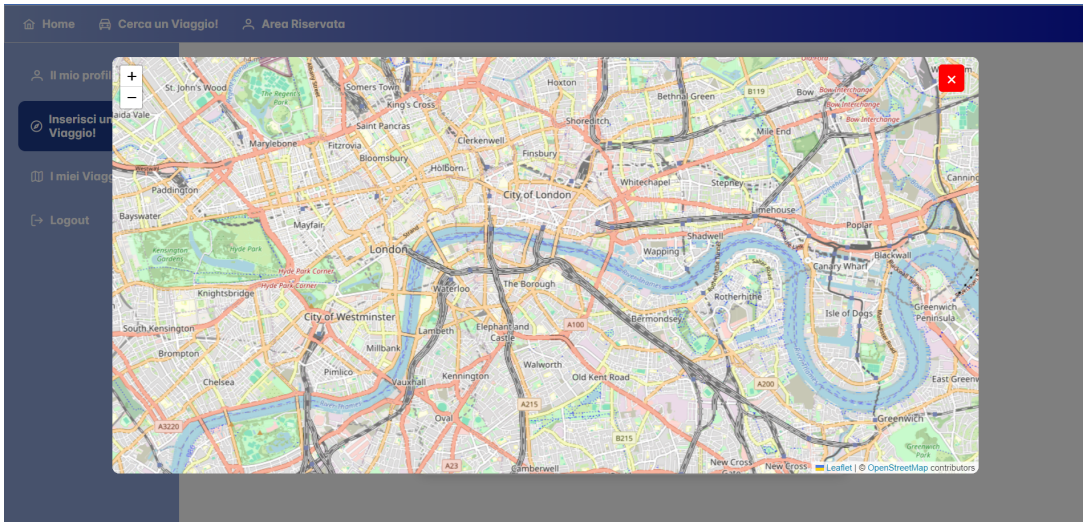


Figura 5.9: Mappa

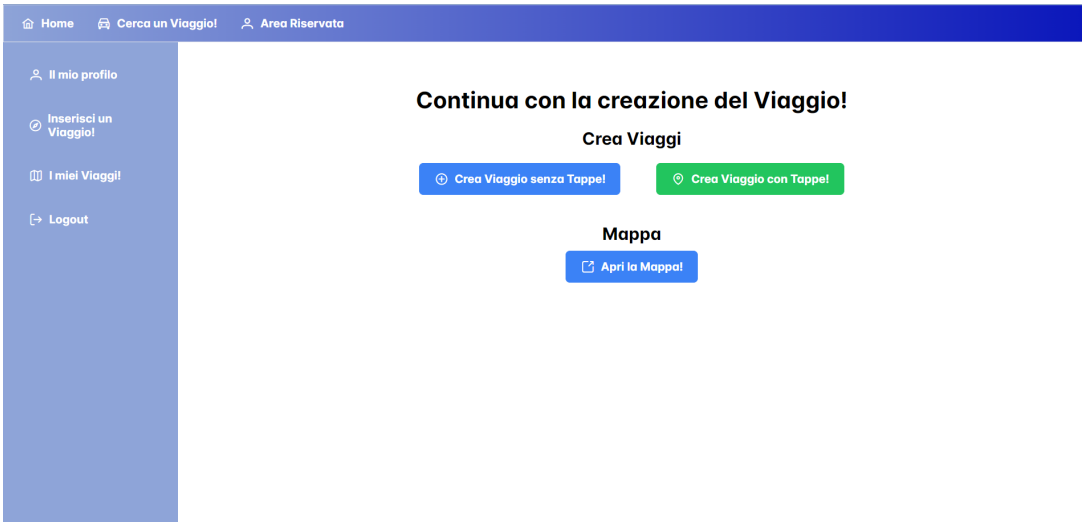


Figura 5.10: Pagina Inserimento Tappe

5.4.4.3 I miei Viaggi

Selezionando la voce del menù "I miei Viaggi!" l'utente potrà visionare i propri viaggi creati, avendo anche la possibilità di modificarli ed eliminarli (figura 5.11).

Premendo sul tasto "Elimina" verrà eliminato il viaggio, questo significa che ci sarà prima un'eliminazione delle tappe associate al viaggio, e poi ci sarà l'eliminazione del viaggio stesso.

Codifica

Questo viene fatto tramite la funzione *deleteTrip(trip: any)* presente in "IMieiViaggiComponent", essa itera tutti le tappe intermedie che hanno il campo di riferimento del Trip associato uguale all'id del Trip e le elimina una per volta, una volta eliminate tutte le tappe intermedie verrà eliminato il viaggio per intero. L'eliminazione avviene tramite una chiamata di tipo DELETE ad http://localhost:8765/api/journeys per ogni tappa presente, ed a http://localhost:8765/api/trips per l'eliminazione del viaggio principale.

5.4.4.3.1 Modifica di un viaggio

Premendo invece sul tasto "Modifica" si verrà reindirizzati alla pagina di modifica del viaggio. All'interno di questa pagina sarà presente un form con i campi già precompilati in base al viaggio scelto da modificare (figura 5.12).

Una volta modificati campi si può scegliere se modificare solo il viaggio oppure se continuare con le modifiche anche delle tappe associate.

5.4.4.3.2 Modifica di una tappa

Una volta che l'utente clicca su "Modifica le tappe associate!" si verrà nuovamente reindirizzati ad una nuova pagina di modifica delle tappe (figura 5.13). Qui saranno presenti principalmente 3 pulsanti:

- **Modifica:** Se l'utente clicca sul pulsante "Modifica" si aprirà la mappa dalla quale sarà possibile reimpostare la posizione geografica e la descrizione della Tappa.
- **Elimina:** Se l'utente clicca sul pulsante "Elimina" la tappa verrà eliminata e non sarà più visualizzabile nella pagina delle Tappe associate ad un determinato viaggio.
- **Aggiungi:** L'utente potrebbe anche decidere di aggiungere un'ulteriore tappa al di fuori del momento della creazione del viaggio. Premendo quindi il pulsante "Aggiungi Tappa" la mappa si aprirà permettendo all'utente di poter aggiungere un'ulteriore tappa a quelle già esistenti.

Codifica

Tramite la funzione OnInit() ad ogni caricamento della pagina verrà fatta una chiamata tramite il metodo GET a http://localhost:8765/api/journeys per caricare le tappe del viaggio. Sempre tramite lo stesso url è possibile eseguire le 3 azioni citate precedentemente solamente con metodi diversi, rispettivamente PUT, DELETE e POST.

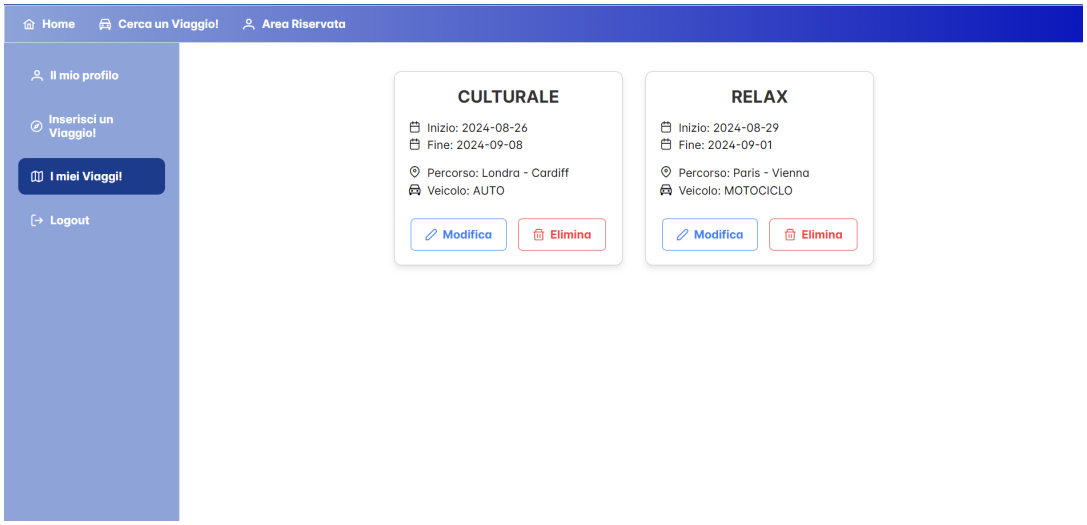


Figura 5.11: Pagina i miei viaggi

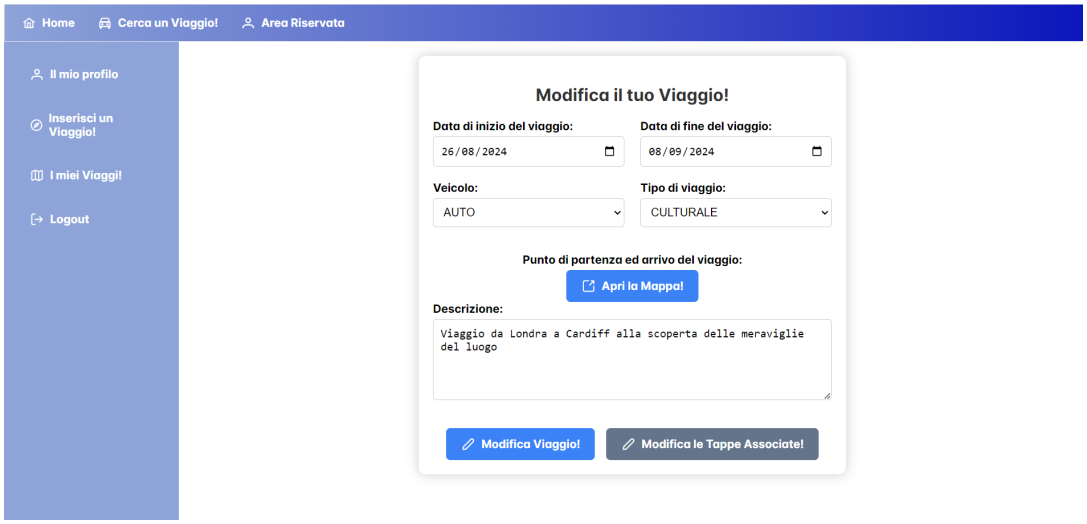


Figura 5.12: Pagina Modifica Viaggio

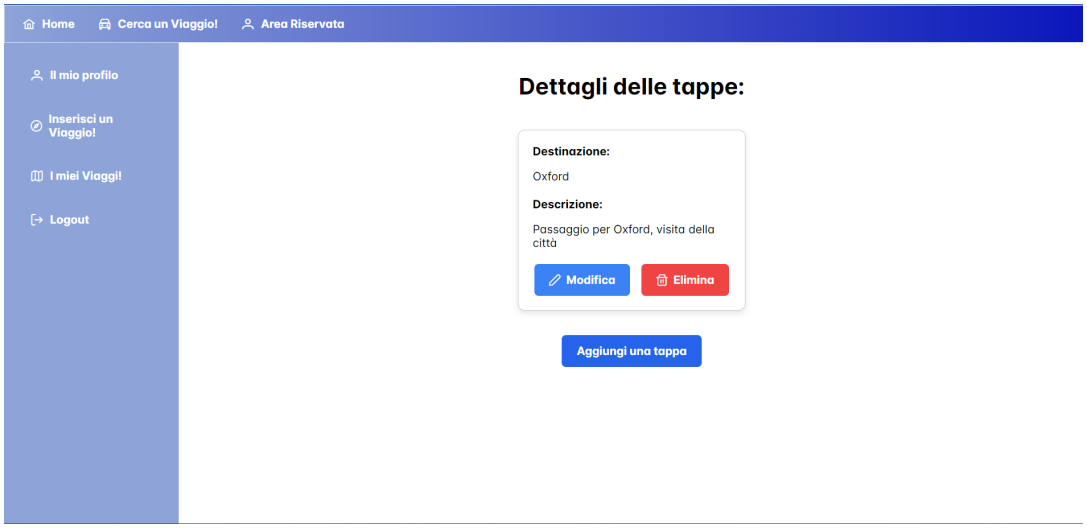


Figura 5.13: Pagina Modifica Tappe

5.4.5 Cerca un Viaggio

Selezionando la voce del menù "Cerca un viaggio!"(disponibile solo per gli utenti loggati) si verrà reindirizzati alla pagina delle proposte di viaggio disponibili. In questa pagina verranno mostrati tutte le proposte di viaggio a cui è possibile partecipare all'interno dell'applicazione.
Con i viaggi sarà possibile principalmente fare due cose:

- Cercare i viaggi disponibili tramite un filtro, esso restituirà i viaggi che partono da una certa data e finiscono entro un'altra data.
- Guardare i dettagli di un determinato viaggio (figura 5.15)

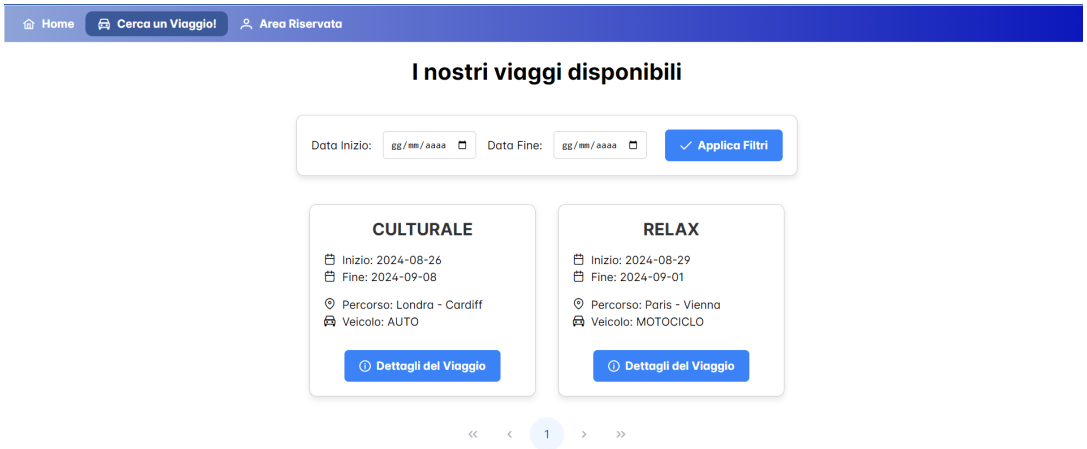


Figura 5.14: Pagina Viaggi

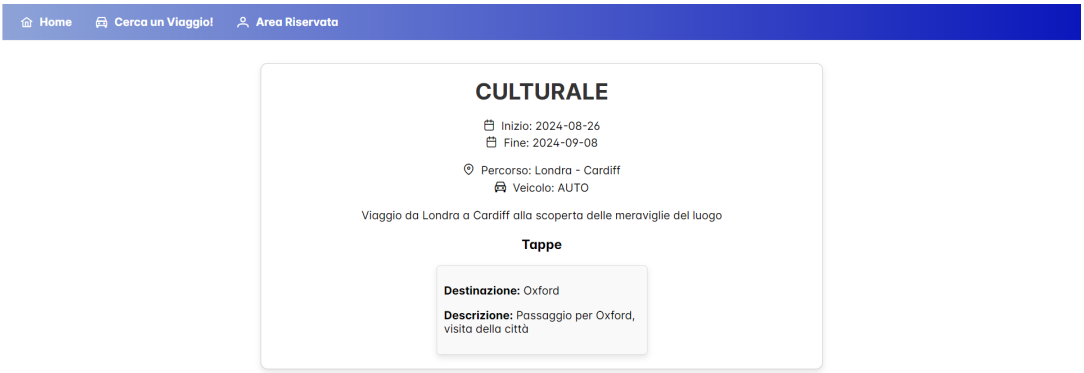


Figura 5.15: Pagina dettaglio di un viaggio

Codifica

Tramite un controllo all'interno del componente "DashboardComponent", ovvero il componente della pagina in figura 5.14, si va vedere il contenuto del localStorage per verificare che l'utente sia loggato. Se l'utente è effettivamente loggato allora mostrerà il menù presente in figura 5.2 altrimenti quello base. In questa pagina tramite la funzione OnInit verranno mostrati tutti i viaggi di tutti gli utenti, inoltre a piè di pagina sarà presente anche un paginatore per i viaggi impostato in modo tale da mostrare massimo 10 viaggi per pagina. I Viaggi vengono caricati tramite un GET della chiamata "http://localhost:8765/api/trips/params" dove params sono una serie di parametri molto importanti.
Questi parametri sono 4 di cui 2 sono già decisi ovvero tripsSize=10 e page=0 (per scelta progettuale si vedranno 10 viaggi per pagina e si comincia dalla pagina 0), mentre gli altri 2 ovvero startDate e endDate si decidono aggiungendo i filtri situati appena sotto il menù e cliccando il pulsante che farà partire un'altra chiamata GET con anche questi 2 parametri.

Capitolo 6

Conclusioni

In questo capitolo vengono illustrate le conclusioni relative al percorso di stage ovvero: il consuntivo finale sulle ore di lavoro, il raggiungimento degli obiettivi e alcune riflessioni personali.

6.1 Consuntivo finale

In confronto con il Piano di Lavoro iniziale non ci sono state variazioni in termini di ore previste. (Vedi Tabella 6.1)
Le ore previste erano 320 e le ore effettive fatte sono state 320.

Settimana	Ore Previste	Ore Effettive
Settimana 1 - 03/06/2024-07/06/2024	40	40
Settimana 2 - 10/06/2024-14/06/2024	40	40
Settimana 3 - 17/06/2024-21/06/2024	40	40
Settimana 4 - 24/06/2024-28/06/2024	40	40
Settimana 5 - 01/07/2024-05/07/2024	40	40
Settimana 6 - 08/07/2024-12/07/2024	40	40
Settimana 7 - 15/07/2024-19/07/2024	40	40
Settimana 8 - 22/07/2024-26/07/2024	40	40

Tabella 6.1: Tabella Consuntivo

6.2 Raggiungimento degli obiettivi

Alla fine del periodo di tirocinio ho avuto un colloquio con il mio tutor aziendale Fabio Pallaro per discutere lo stato di raggiungimento degli obiettivi concordati ad inizio stage.

Durante il colloquio abbiamo analizzato il prodotto effettivo rispetto a quello atteso giungendo alla conclusione che tutti gli obiettivi, sia obbligatori che desiderabili che opzionali siano stati raggiunti, come mostrato in tabella 6.2

Obbligatorio		
OB 1	Acquisizione competenze sulle tematiche sopra descritte.	Sì
OB 2	Capacità di raggiungere gli obiettivi richiesti in autonomia seguendo il crono programma.	Sì
OB 3	Portare a termine le implementazioni previste con una percentuale di superamento pari al 80% (equivalente alla maschera di Gestione profilo Utente e loro service con le chiamate al backend).	Sì
Desiderabile		
DE 1	Portare a termine le implementazioni previste con una percentuale di superamento pari al 100% (equivalente alla maschera di Gestione Profilo Utente, Gestione Viaggio e loro service con le chiamate al backend).	Sì
Opzionale		
OP 1	Apportare un valore aggiunto al gruppo di lavoro durante le fasi di progettazione delle interfacce.	Sì

Tabella 6.2: Tabella stato di raggiungimento degli obiettivi

6.3 Conoscenze acquisite

Durante il periodo di stage, soprattutto durante la prima parte, ho dovuto studiare ed apprendere un numero significativo di linguaggi, framework e strumenti lavorativi, questo ha arricchito enormemente il mio bagaglio culturale portan-

domi ad una conoscenza superiore in ambito di sviluppo web.
Le conoscenze acquisite principali sicuramente sono state:

- **Angular e TypeScript:** Ho acquisito una solida padronanza di Angular, uno dei principali framework per lo sviluppo front-end, e di TypeScript, un linguaggio che estende JavaScript con una tipizzazione forte. Questi strumenti mi hanno permesso di sviluppare interfacce utente complesse, dinamiche e facilmente mantenibili, come le schermate di login/registrazione e gestione del profilo utente.

- **Integrazione front-end e back-end:** Anche se il focus principale del mio lavoro è stato sul front-end, ho sviluppato un prototipo di back-end utilizzando Spring Boot per creare servizi REST. Questo mi ha permesso di gestire le chiamate API dal front-end, collegando in modo efficace l'interfaccia utente con i dati provenienti dal server. Questa esperienza mi ha dato una visione più ampia ed approfondita di come le due parti si integrino ed interagiscano tra di loro sinergicamente.

6.4 Valutazione personale

Durante il mio periodo di tirocinio ho avuto l'opportunità di crescere sia come lavoratore che come persona andando a migliorare aspetti professionali riguardanti le mie competenze e aspetti personali. Fin dall'inizio mi sono immerso in un ambiente che mi ha permesso di sviluppare ulteriormente le mie competenze come sviluppatore web. In particolare, ho potuto approfondire l'uso del framework Angular, con cui non avevo molta familiarità prima. Ho dedicato tempo a studiare autonomamente molte tecnologie (linguaggi, framework ecc.), alcune nuove altre invece che avevano solo bisogno di una rispolverata nella mia mente, per poi applicare quanto appreso al progetto reale, migliorando così le mie capacità tecniche.
Oltre all'aspetto tecnico però, questo tirocinio mi ha insegnato molto anche su altre competenze, come il lavoro di squadra e la comunicazione. Collaborare con altri studenti durante il tirocinio per, ad esempio, coordinarsi sul passaggio

delle proprie parti del progetto oppure sulla risoluzione di problemi mi ha fatto capire quanto sia importante confrontarsi e lavorare insieme per raggiungere gli obiettivi comuni. Inoltre, ho imparato il valore della comunicazione soprattutto di come essa debba essere efficiente per riuscire a lavorare al meglio.
Per concludere la mia valutazione e quindi anche la tesi in sé, questo tirocinio mi ha fatto capire la differenza tra il mondo universitario e il mondo lavorativo; all'università sicuramente ho imparato molte cose e mi sono costruito delle ottime fondamenta teoriche tuttavia il mondo lavorativo è tutt'altra cosa. Ho capito quanto sia fondamentale applicare le conoscenze in contesti concreti e risolvere così problemi reali.

Bibliografia

Articoli

Econopoly. «Carpooling e mobilità sostenibile: il ruolo cruciale delle aziende». In: *IlSole24ore* (2024). DOI: https://www.econopoly.ilsole24ore.com/2024/07/17/carpooling-aziendale-mobilita-sostenibile-aziende/?refresh_ce=1.

Siti

BlaBlaCar. URL: https://blog.blablacar.com/about-us.

Carpooling. en: https://en.wikipedia.org/wiki/Carpool. URL: https://it.wikipedia.org/wiki/Car_pooling.

Documentazione Angular. Altri link:https://angular.dev/overview. URL: https://formazione.jdk.it/angular-a-cosa-serve/#:~:text=Angular%20%C3%A8%20un%20framework%20di,JavaScript%20e%20la%20libreria%20TypeScript..

Documentazione CSS. URL: https://developer.mozilla.org/en-US/docs/Web/CSS.

Documentazione HTML. URL: https://developer.mozilla.org/en-US/docs/Web/HTML.

Documentazione Intellij. Altri link: https://it.wikipedia.org/wiki/IntelliJ_IDEA. URL: https://www.jetbrains.com/idea/.

Documentazione Java. URL: https://www.java.com/en/download/help/whatis_java.html.

Documentazione LeafLet. URL: https://leafletjs.com/index.html.

Documentazione PrimeNG. URL: https://primeng.org/.

Documentazione Visual Studio Code. Altri link: https://it.wikipedia.org/wiki/Visual_Studio_Code. URL: https://visualstudio.microsoft.com/it/.

Documentazione-Typescript. URL: https://www.typescriptlang.org/.

LiftShare: Altri link: https://hub.liftshare.com/en/knowledge/what-is-car-sharing, https://blog.liftshare.com/liftshare/product-release-scoping-smart-mobility. URL: https://liftshare.com/uk/frequently-asked-questions.

MVVM. URL: https://it.wikipedia.org/wiki/Model-view-viewmodel.

Peer-to-peer carsharing. URL: https://en.wikipedia.org/wiki/Peer-to-peer_carsharing.

Postman. Altri Link: https://www.geekandjob.com/wiki/postman. URL: https://www.postman.com/.

Postman. URL: https://stoplight.io/.

Ridesharing company. URL: https://en.wikipedia.org/wiki/Ridesharing_company.

SpringBoot. URL: https://www.ibm.com/it-it/topics/java-spring-boot.

Sync Lab s.rl. URL: https://www.synclab.it.

Waze CarPooling. URL: https://support.google.com/waze/answer/6071177?hl=it#zippy=%2Cper-chi-%C3%A8-progettata-waze%2Ccome-funziona-waze%2Cin-che-modo-waze-impara-i-percorsi.