**Progetto Gestionale Dipendenti**

Fatto da Moreno Frigo Turco, Filippo Santoro e Neha Bhatti

**INTRODUZIONE**

Nella seguente realtà, il gruppo ha creato una web app per la gestione degli orari di entrata e di uscita in un’azienda. La parte di logica del back-end viene gestita tramite l’utilizzo del framework di Java Spring Boot, con il collegamento ad un database basato su MySQL. Il front-end invece viene gestito tramite il framework Angular, con l’aggiunta di Bootstrap per migliorare lo stile delle pagine e per renderle l’applicazione responsive.

L’applicazione prevede un login iniziale effettuabile sia da un admin che da un dipendente. Da cui si accede ad una homepage in cui è possibile effettuare timbrare l’entrata e l’uscita.



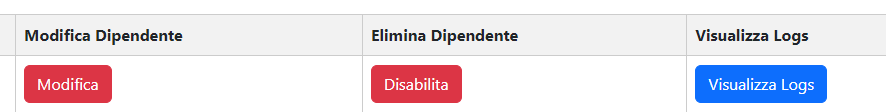
La timbratura al completamento della sessione, quindi una volta timbrata l’entrata e anche l’uscita viene registrata nel database e mostrata nella sezione dei logs, accessibile dalla navbar (“Storico Logs”).

Nella homepage è inoltre possibile visualizzare le ore totali lavorate.

Accedendo allo “Storico Logs” con un account Admin è possibile cancellare la propria sessione di timbratura, mentre per il dipendente questa funzionalità non è disponibile.

L’utente Admin dalla navbar ha accesso all “Admin Panel” in cui può trovare le seguenti funzionalità:

* inserire un nuovo dipendente
* storico logs di tutti i dipendenti e degli admin
* una tabella con i record dei singoli dipendenti e le relative opzioni.

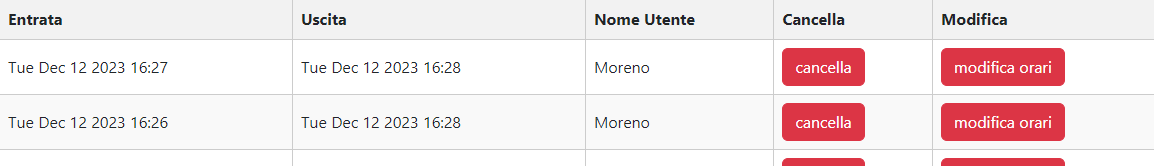


Le opzioni attuabili sul singolo dipendente sono la modifica, disabilita e Visualizza logs.

Con la modifica possiamo accedere ad un form per modificare i dati presenti nel database.

Per disabilitare un dipendente in modo che non possa accedere e alla piattaforma è sufficiente premere il pulsante. Funzionalità inversa per abilitarlo.

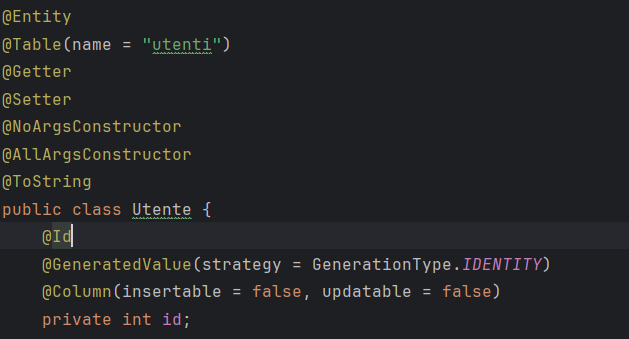
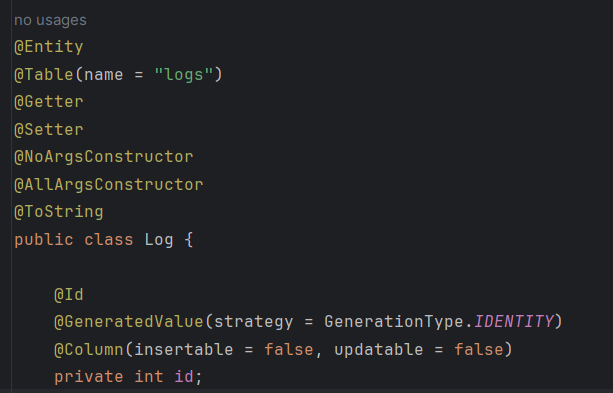
Con l’ultimo si può visualizzare la lista delle sessioni di timbratura del singolo dipendente.



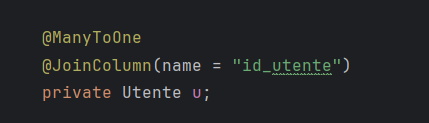
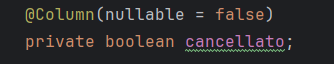
In questa sezione di può inoltre cancellare le singole sessioni oppure modificarne gli orari.

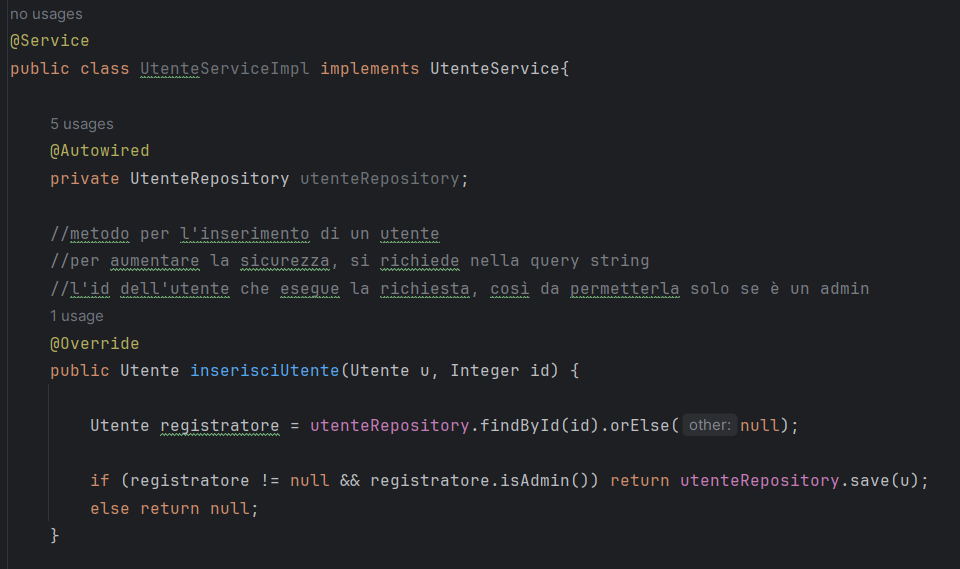
In alto a sinistra è presente logout.

**BACKEND**



La parte di back-end inizia con la creazione delle due entità principali, utente e log. Queste vengono create automaticamente nel database tramite l’utilizzo delle properties di Hibernate. Viene utilizzata la libreria di Lombok per rendere più pulito il codice e automatizzare la creazione dei getters, setters e costruttori. L’utente può essere admin oppure no, così da avere permessi diversi.

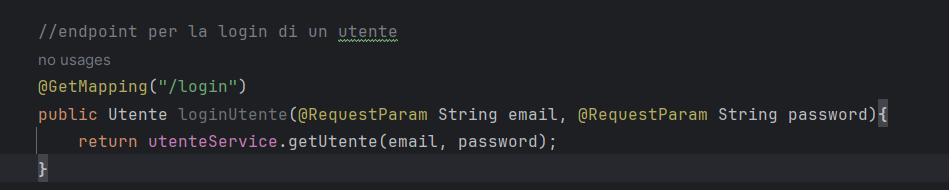
Inoltre, contiene un flag di tipo booleano, che ti dice se è stato cancellato o meno, così da avere uno storico dei logs salvato. Infatti, i logs sono collegati con l’utente tramite un’annotazione ManyToOne, così da avere una relazione nel database.



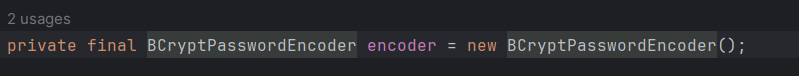
Abbiamo poi un service per gli utenti, che richiama la sua repository per eseguire le operazioni verso il database. Un esempio si trova nell’inserisci utente, dove riceverà dal front-end un utente e un id, quest’ultimo appartenente all’utente che va ad eseguire la richiesta dell’operazione, eseguibile solo da un admin.



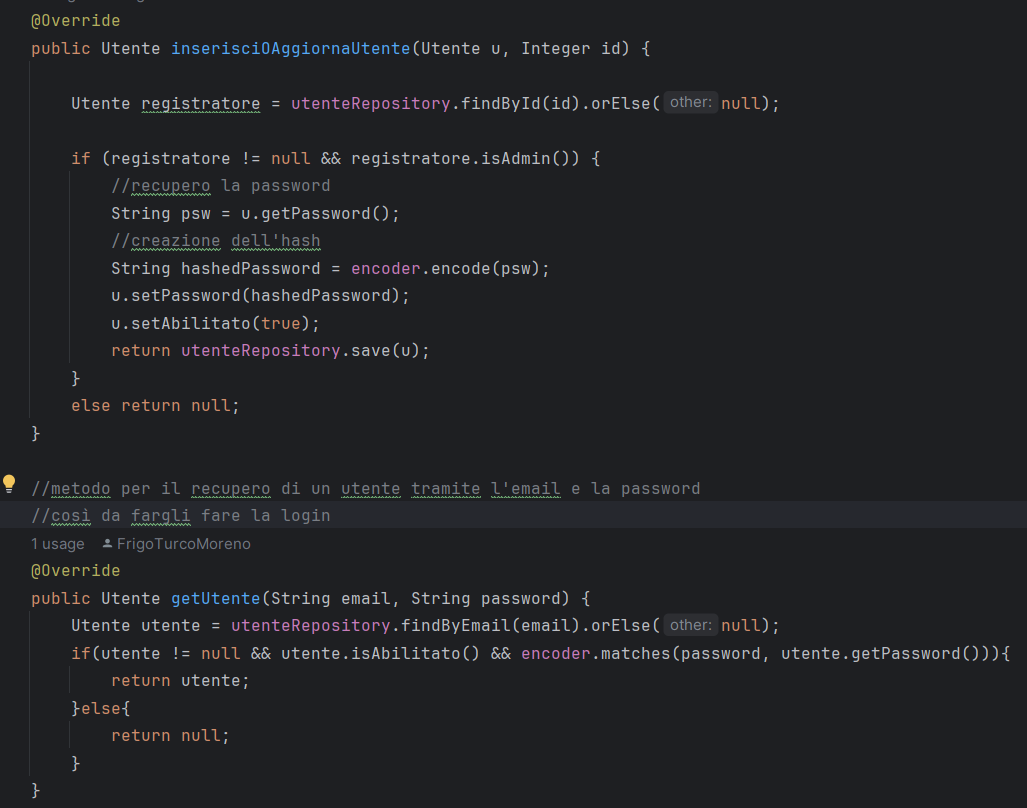
Per fare le richieste per operazioni riguardanti gli utenti, viene creato un rest controller, con l’aggiunta dell’annotazione cross origin per non essere bloccati dal problema del Cors. Nel controller si inseriscono gli endpoint che vengono raggiunti dal front-end per ricevere dei dati, e come possiamo vedere da esempio, nella registrazione, che utilizza il Post perchè sta inviando dei dati al database, si va a passare l’id dell’utente che sta eseguendo l’operazione tramite l’url, per controllare che sia un admin, mentre l’utente da inserire lo si passa direttamente tramite il body.



Per la login la situazione è un attimo diversa, visto che si va ad utilizzare la query string per passargli i dati dell’utente, essendo che si utilizza una Get.



Abbiamo aggiunto un grado di sicurezza, decidendo di importare spring security per utilizzare il suo password encoder, così da inserire nel database la password hashata.



Come si può vedere in questa foto, si va ad inserire la password hashata nella registrazione, per poi fare un match tramite l’encoder, mettendo a confronto la password passata dal form e la password salata nel db.

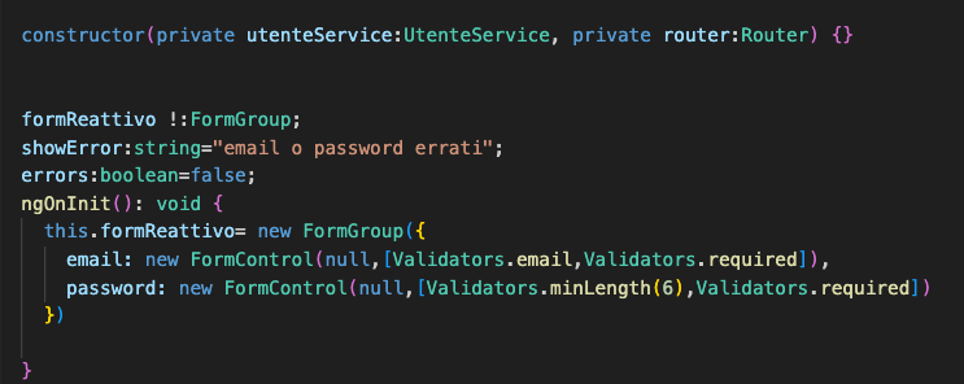
**FRONTEND**

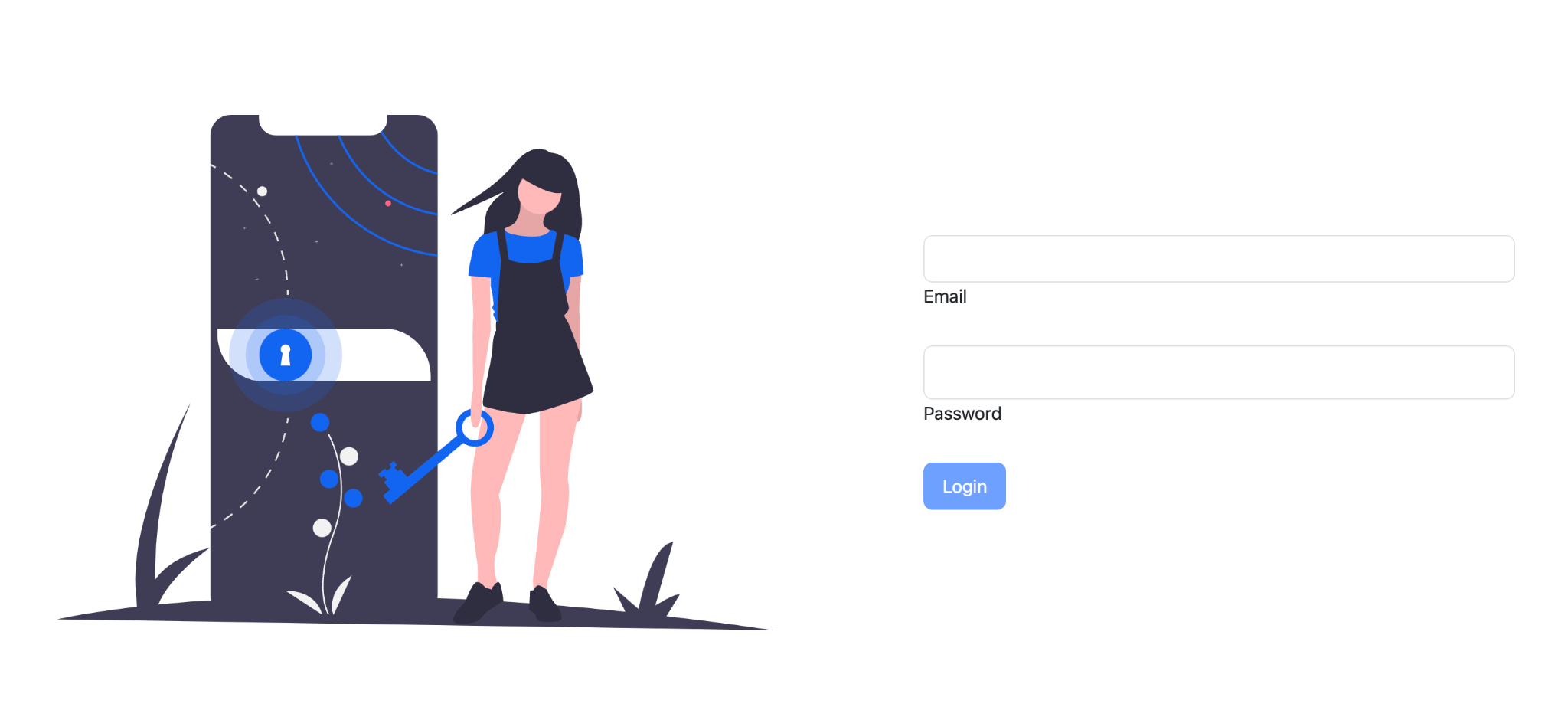
In angular il progetto è stato suddiviso nelle cartelle componenti, interfacce, guard e servizi.

I componenti sviluppati sono:

* admin-panel
* home
* login
* logs
* nav-bar
* register
* update

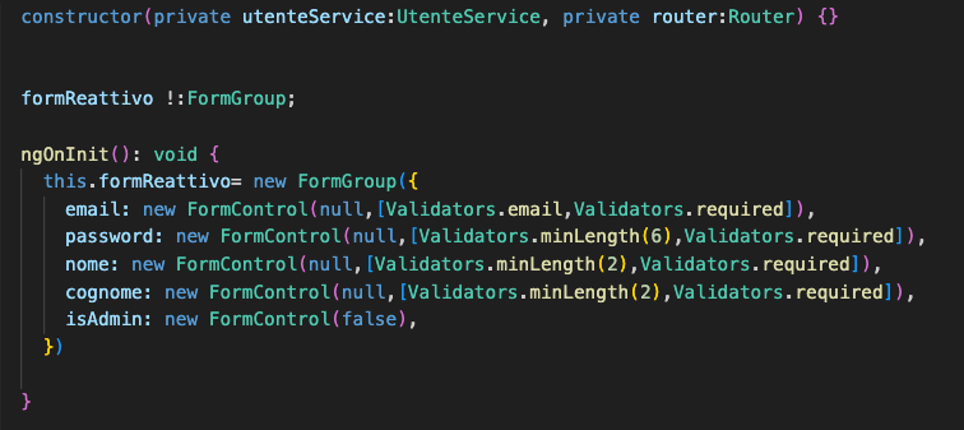
Il componente login presenta un FormGroup per la validazione dei dati dove viene fatto un controllo sulla lunghezza minima della password e un controllo che sia effettivamente un’e-mail nel formato corretto.

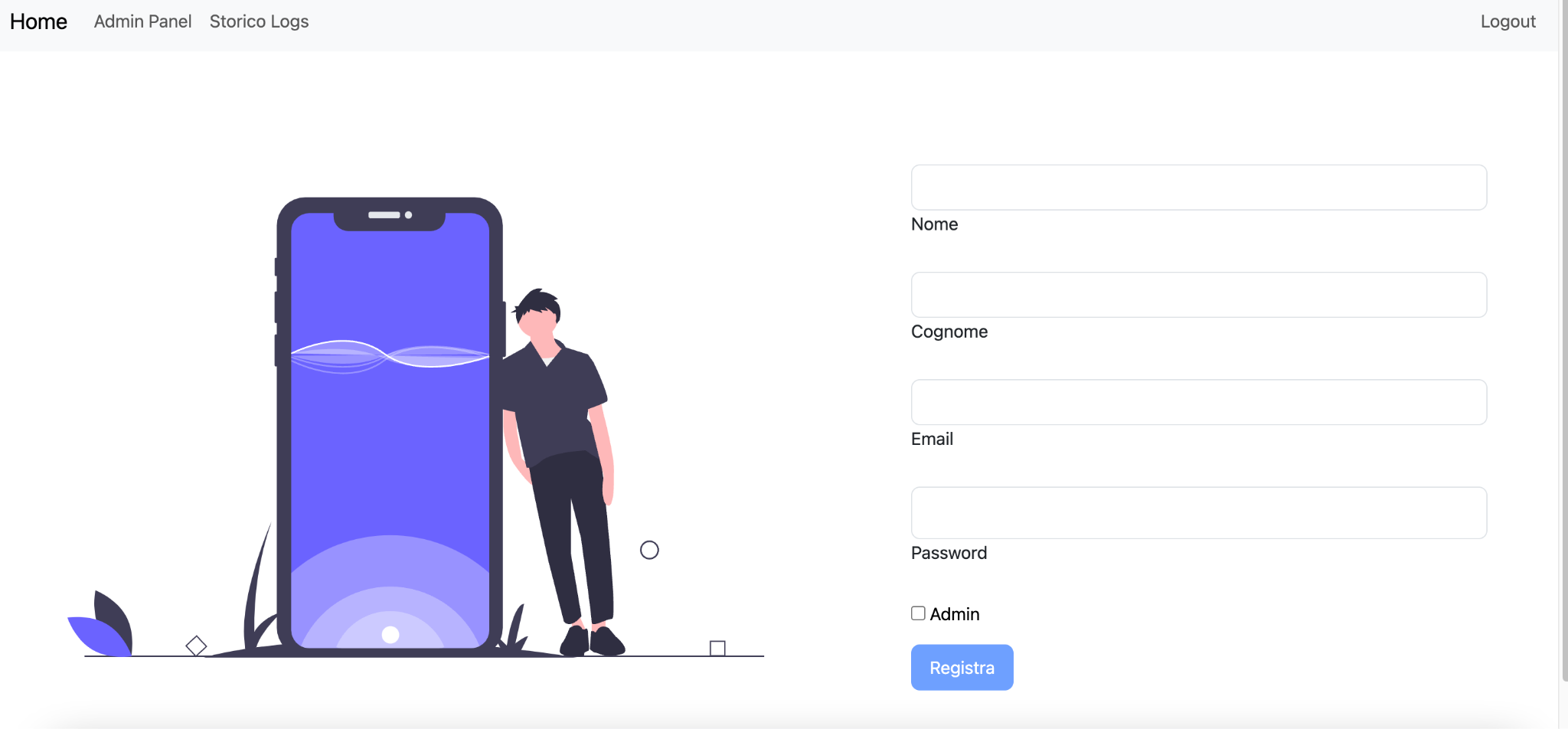




Non appena i dati inseriti risultano nel formato corretto verrà abilitato il pulsante di login e se risultano corretti i dati inseriti verrai ridirezionato alla home mentre se non sono corretti verrà visualizzato un messaggio di errore.

Mentre per la registrazione di un nuovo utente è necessario essere un utente Admin e sarà presente il pulsante per la creazione di quest’ultimo nell’admin panel. La pagina di registrazione presenta un FormGroup per la validazione dei dati dove vengono validati email,password,nome,cognome.





.

Non appena i dati inseriti risultano nel formato corretto verrà abilitato il pulsante di registrazione e se risultano corretti i dati inseriti verrai ridirezionato alla home.

Guardie

Sono presenti due guardie: una per verificare se l’utente ha effettuato il login ed un’altra per verificare se l’utente loggato ha il ruolo di admin.

**Admin panel**

è stata creata una tabella nell'admin panel per gestire tutti i dipendenti dell'azienda. Sono stati aggiunti tre bottoni: uno di modifica, per permettere di modificare i dati dei dipendenti, uno per disattivarli (quest'ultimo non elimina i dati del dipendente dal database, ma permetti di disattivarne l'accesso alla piattaforma, l'utente che ha effettuato l'accesso viene escluso a priori e non insierito nella tabella).

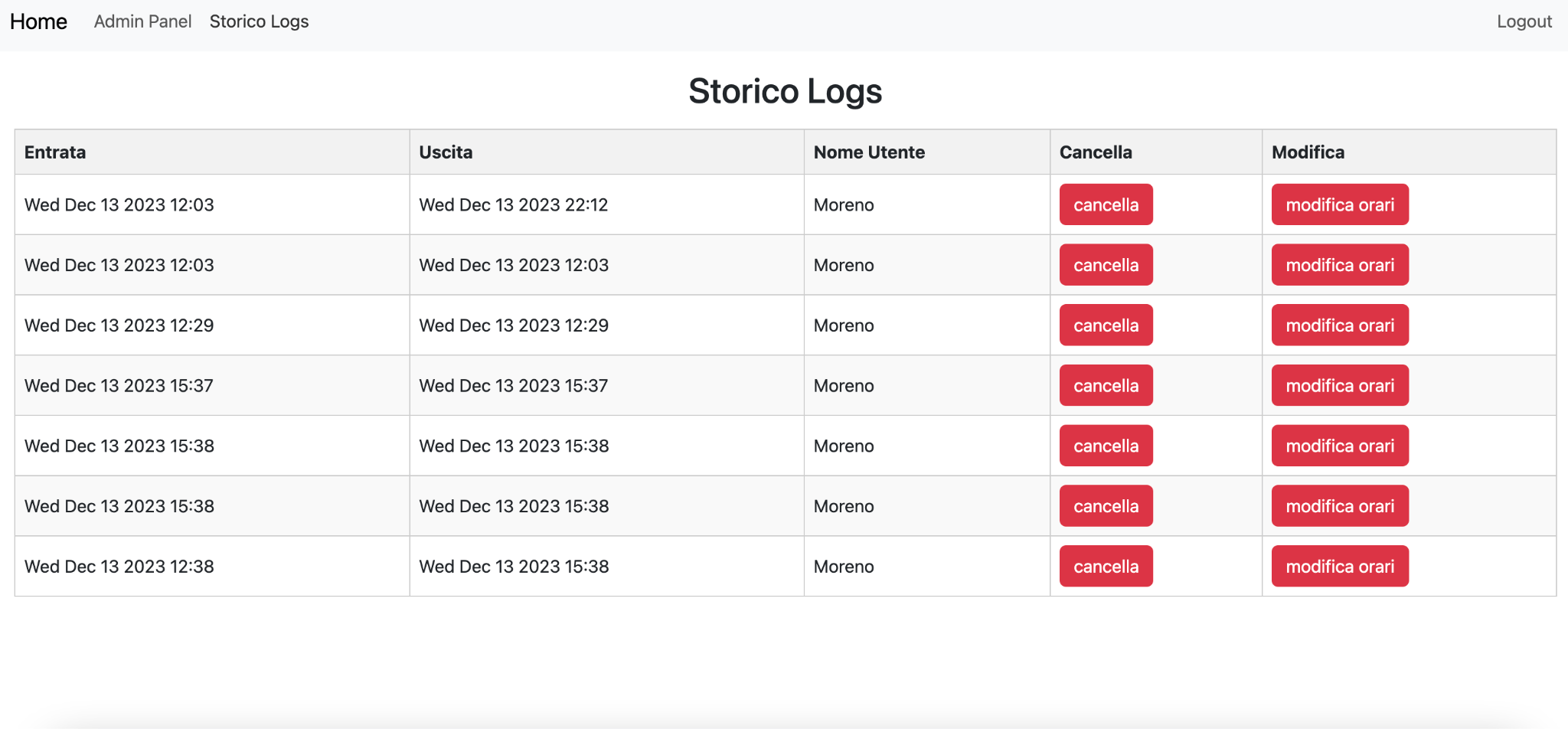
L'ultimo pulsante permette di visualizzare le singole sessioni di timbratura dei dipendenti.

**Logs**

Per mostrare i logs è stato creato un componente logs dove una volta inizializzato va a prendere da LogsService l’array contenente i logs. In questo modo, si può riutilizzare questo componente per recuperare sia i logs personali, sia i logs di tutti gli utenti, sia i logs di uno specifico utente su richiesta di un admin.







Inoltre un utente con ruolo admin ha la possibilità di cancellare o modificare gli orari di uscita o entrata di un determinato log.

**COME INIZIALIZZARE LA WEB APP**

* Recuperare dal backend il file db.sql ed eseguire la creazione del db e iniziare ad utilizzarlo in MySQL Workbench;
* Avviare il backend tramite IntelliJ per permettere la creazione automatica delle entità e delle relazioni nel db (cambiare l’accesso al db nelle properties in base al proprio ambiente);
* Inserire l’utente presente nel file db.sql in MySQL Workbench (la password è 123456);
* Scaricare nel frontend tutti i node modules necessari (angular 16.2.10, bootstrap, popper.js, jquery);
* Avviare il frontend, scrivendo nel terminale ng serve;
* Loggarsi con l’account inserito precedentemente e iniziare ad utilizzare la web app.

**Github repository:** <https://github.com/FrigoTurcoMoreno/Gestionale-Dipendenti>