Università degli Studi di Milano - Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di Laurea in Informatica

Integrazione di un sistema di gestione meeting per una piattaforma di recruiting

Relatore: Prof.ssa Daniela Micucci

Tesi di Laurea di:
Simone Lesinigo
Matricola 899540

Anno Accademico 2023-2024

Indice

Elenco delle figure

Introduzione

Il campo del recruiting è in continua evoluzione, con una crescente necessità di strumenti che facilitino in modo semplice ed efficiente l'incontro tra domanda e offerta di lavoro. In questo contesto, la gestione dei colloqui di lavoro rappresenta una fase cruciale e complessa del processo di selezione.

L'obiettivo di questa esperienza di stage, svoltasi presso l'azienda Nesecom SRLS, è stato quello di integrare un sistema per la gestione dei colloqui di lavoro tramite meeting all'interno di un sito web in sviluppo, RisUma.

RisUma, acronimo di RISorse UMAne, è pensato per essere una piattaforma di recruiting intuitiva, che consente agli utenti e alle aziende di trovare efficacemente le opportunità di lavoro e le figure professionali richieste, gestendo l'intero processo di selezione su un'unica piattaforma.

Questo progetto è ancora nelle fasi iniziali del suo sviluppo, ma punta a eguagliare e superare le più famose piattaforme di recruiting grazie alle sue comode funzionalità.

Capitolo 1

Tecnologie utilizzate

Nel corso dello sviluppo del progetto sono state impiegate diverse tecnologie moderne, per garantire un'applicazione robusta, scalabile e facilmente manutenibile. La parte del progetto a me assegnata ha richiesto uno sviluppo full-stack, coinvolgendo sia il frontend che il backend.

1.1 Frontend

Il frontend rappresenta la parte dell'applicazione con cui l'utente interagisce direttamente. È stato sviluppato utilizzando **React** insieme a **TypeScript**, con l'obiettivo di garantire un'esperienza utente fluida e intuitiva.
Per gestire i pacchetti e le dipendenze, è stato utilizzato **npm** (Node Package Manager).
Inoltre, il design dell'interfaccia utente è stato standardizzato utilizzando il **MUI Theme** di Material-UI.

- **React**: React è una libreria JavaScript open source, sviluppata da Facebook, per la costruzione di interfacce utente. Utilizzando un approccio basato sui componenti, React consente di creare interfacce utente modulari e riutilizzabili. La sua capacità di aggiornare e mostrare efficientemente solo i componenti necessari in risposta ai cambiamenti rende React particolarmente adatto per lo sviluppo di applicazioni interattive e ad alte prestazioni. Inoltre, React permette di gestire lo stato delle applicazioni in modo prevedibile e scalabile, facilitando lo sviluppo di applicazioni complesse. [1]

- **TypeScript**: TypeScript è un linguaggio di programmazione open source sviluppato da Microsoft, che estende JavaScript aggiungendo tipi statici. L'adozione di TypeScript permette di ridurre significativamente gli errori durante lo sviluppo, grazie al controllo statico dei tipi che individua potenziali problemi prima ancora dell'esecuzione del codice. Inoltre, TypeScript facilita la manutenzione del codice in progetti di grandi dimensioni, migliorando la leggibilità e la documentazione attraverso la tipizzazione esplicita. [2]

- **MUI Theme (Material-UI)**: Material-UI è una libreria di componenti React che implementa le linee guida del **Material Design** di Google.

Utilizzare Material-UI permette di sviluppare interfacce utente coerenti e professionali senza dover creare e stilizzare i componenti da zero. La libreria offre una vasta gamma di componenti pre-stilizzati e altamente personalizzabili, che facilitano la creazione di un design uniforme e accattivante. Inoltre, Material-UI supporta nativamente la creazione di temi, consentendo di personalizzare l'aspetto dell'applicazione in modo centralizzato.

- **npm**: Node Package Manager è il gestore di pacchetti predefinito per l'ecosistema JavaScript, utilizzato per installare e gestire le dipendenze necessarie per lo sviluppo delle applicazioni. Npm facilita l'integrazione di librerie e strumenti di terze parti, permettendo agli sviluppatori di accedere rapidamente a un vasto repository di pacchetti open source. Questo strumento è fondamentale per mantenere aggiornate le dipendenze del progetto e per gestire le versioni delle librerie utilizzate.

1.2 Backend

Il backend rappresenta la parte dell'applicazione che gestisce la logica di business, l'elaborazione dei dati e la comunicazione con il database. È responsabile del funzionamento lato server dell'applicazione, elaborando le richieste degli utenti e restituendo le risposte appropriate al frontend.
Il backend dell'applicazione è stato sviluppato utilizzando **Java** con il framework **Spring Boot**. Inoltre, viene utilizzato **Maven** come strumento di gestione delle dipendenze.
Per la gestione della persistenza dei dati è stato utilizzato **Java Persistence API (JPA)**, che permette di interfacciarsi con il database in modo efficiente e strutturato. Il database utilizzato si basa su **PostgreSQL 14**.

- **Java**: Java è un linguaggio di programmazione ad alto livello, orientato agli oggetti e a tipizzazione statica, sviluppato da Sun Microsystems (ora di proprietà di Oracle). È noto per la sua robustezza, sicurezza e portabilità, grazie alla Java Virtual Machine (JVM) che permette di eseguire il codice Java su qualsiasi piattaforma. Java è ampiamente utilizzato nello sviluppo di applicazioni enterprise, sistemi embedded, applicazioni mobili e web. La sua vasta ecosistema di librerie e strumenti lo rende una scelta ideale per lo sviluppo backend. [3]

- **Spring Boot**: Spring Boot è un framework open source per lo sviluppo di applicazioni Java basato su Spring, che semplifica il processo di creazione di app web e microservizi. Spring Boot offre una configurazione automatica dei componenti, riducendo significativamente il tempo necessario per avviare un progetto. Fornisce anche una serie di funzionalità integrate, come la gestione della sicurezza e il logging che facilitano lo sviluppo e la manutenzione delle applicazioni. [4]

- **Maven**: Maven è uno strumento di build automation e gestione delle dipendenze per progetti Java sviluppato dalla Apache Software Foundation. Utilizzando un modello basato su pom.xml (Project Object Model), Maven facilita la gestione del ciclo di vita del progetto, dall'inizializzazione alla distribuzione. Consente di integrare facilmente le librerie di terze

parti e di gestire le versioni delle dipendenze, migliorando la coerenza e la riproducibilità del build. [5]

- **Java Persistence API (JPA)**: La Java Persistence API (JPA) è una specifica standard per la gestione della persistenza dei dati nelle applicazioni Java. JPA offre un modo standardizzato per mappare le classi Java agli oggetti del database, permettendo di eseguire operazioni CRUD (Create, Read, Update, Delete) in modo semplice e intuitivo.
  Inoltre fornisce un linguaggio per effettuare query SQL, chiamato **JPQL** (Java Persistence Query Language), che è indipendente dal DBMS utilizzato.
  Utilizzando JPA, gli sviluppatori possono concentrarsi sulla logica di business senza doversi preoccupare dei dettagli specifici dell'interazione con il database, migliorando la produttività e la manutenibilità del codice. [6]

- **PostgreSQL**: PostgreSQL è un sistema open source di database relazionale a oggetti sviluppato da Michael Stonebraker. È famoso per la sua affidabilità e per l'integrità dei dati, oltre alla community che continua a sostenerlo continuando a sviluppare nuove soluzioni. [7]

Capitolo 2

Approccio Proposto

2.1 Problema e Obiettivo

Il problema principale affrontato riguarda:

- Per le **aziende**: ottimizzazione dell'efficienza nella gestione dei meeting per i colloqui di lavoro, fornendo uno strumento adeguato e completo di tutte le funzionalità necessarie. L'obiettivo principale è migliorare l'esperienza dei recruiter, che potrebbero trovarsi a gestire un elevato numero di meeting.
- Per gli **utenti**: eliminare la necessità per i candidati di segnare manualmente gli appuntamenti, offrendo loro uno strumento che ricordi in modo comodo e automatico tutti i colloqui programmati, facilitandone l'accesso.

2.1.1 Caratteristiche dei Meeting

Un ulteriore obiettivo riguarda le proprietà che devono caratterizzare un meeting:

- Non è richiesto avere un account per partecipare al meeting, ma sarà sufficiente inserire un nome a propria scelta al momento dell'accesso.
- Nessuno può accedere al meeting prima di un quarto d'ora rispetto all'orario di inizio prestabilito o dopo che sia trascorso l'orario di fine.

2.2 Metodologia

Di seguito sono riportati i passi seguiti per la realizzazione di questo progetto. Si vuole far notare che RisUma è un'idea del direttore aziendale, e pertanto non vi è un cliente con cui confrontarsi. Inoltre, poiché non sono state disponibili direttive scritte ma solo alcune indicazioni orali, tutte le scelte sono state prese a mia completa discrezione.

1. **Servizio**: il primo passo è stato scegliere quale servizio di terze parti utilizzare per le chiamate online. Si cercava un servizio che:

   (a) avesse le caratteristiche richieste

   (b) si integrasse bene con il progetto (API disponibili, compatibilità con le tecnologie utilizzate...)

   (c) fosse facilmente usabile anche dagli utenti meno esperti

   Per tali motivi sono state prese in considerazione le piattaforme più note, quali Cisco Webex, Google Meet e Zoom.

2. **Interfaccia utente**: è stato scelto il design della pagina frontend. Basandosi anche sulle 10 euristiche di Jakob Nielsen, si sono perseguiti i seguenti obiettivi:

   (a) Adattare il sistema al mondo reale. (Euristica 2)

   (b) La pagina doveva essere facile da navigare, riducendo al minimo il carico cognitivo per l'utente. (Euristica 6)

   (c) Optare per uno stile minimalista e intuitivo. (Euristica 7)

   (d) Implementare funzionalità di feedback per guidare l'utente in caso di errori o problemi durante l'utilizzo della pagina. (Euristiche 8 e 9)

3. **Creazione della tabella a database**: è stata progettata e creata la tabella nel database aziendale, includendo tutte le colonne necessarie, al fine di garantire il corretto funzionamento della pagina web.

4. **Integrazione delle API**: sono state integrate le API per consentire le operazioni CRUD sui meeting, sviluppando contemporaneamente le parti di frontend e backend secondo il seguente ordine:

   (a) **Creazione**: implementazione della funzionalità per creare nuovi meeting, gestendo i parametri necessari come data, ora e invitati.

   (b) **Recupero**: realizzazione della logica per recuperare e visualizzare i meeting di un utente.

   (c) **Modifica**: implementazione della possibilità dei modificare i parametri dei meeting già esistenti.

   (d) **Eliminazione**: implementazione della funzionalità per annullare meeting programmati.

   Questa parte è stata cruciale nel corso della mia esperienza di stage in quanto ha costituito il nucleo essenziale della pagina che ho progettato e integrato.

5. **Test**: Il sistema è stato sottoposto da parte mia e di alcuni colleghi a dei test per garantire che tutte le funzionalità operassero correttamente.

6. **Deploy**: È stato eseguito un deploy su un server per testare le prestazioni in un ambiente più realistico rispetto a quello locale.

2.3 Possibili sfide

Nel corso dello sviluppo del progetto sono emerse due principali sfide:

1. **Consistenza tra i database**: assicurare la consistenza tra il database aziendale e quello del servizio di terze parti durante le operazioni di creazione, modifica o eliminazione di un meeting. Una eventuale inconsistenza porterebbe a gravi problemi legati allo svolgimento dei meeting, compromettendo l'affidabilità e le funzionalità del sistema.

2. **Creazione dell'interfaccia utente**: la realizzazione di un'interfaccia utente piacevole e funzionale ha rappresentato un'altra grande sfida. Nel corso dello sviluppo è stata più volte modificata sulla base dei feedback dei colleghi.

Capitolo 3

Sviluppo

In questa sezione sono riportati in dettaglio tutti gli step che si sono seguiti per la realizzazione del progetto.

3.1 L'idea

L'idea alla base del progetto è che l'azienda presso cui si è svolto lo stage assuma il ruolo di organizzatrice di tutti i meeting, garantendone il pieno controllo. Gli utenti di RisUma dovranno semplicemente partecipare come ospiti.

Un altro punto cruciale è impedire alle aziende di mettersi in contatto con i candidati tramite mezzi alternativi a RisUma almeno fino al momento del meeting. Questo obbliga le aziende e gli utenti a utilizzare esclusivamente il sito, garantendo che tutte le comunicazioni avvengano tramite la piattaforma.

3.2 Scelta del servizio

Come riportato precedentemente, il primo passo è stato selezionare il servizio di terze parti da utilizzare per effettuare le chiamate online.

Le possibili opzioni considerate sono state:

- Cisco Webex
- Google Meet
- Zoom

Dopo un'attenta lettura della documentazione e delle funzionalità offerte dai vari servizi, la scelta è ricaduta su **Cisco Webex** per i seguenti motivi:

- **Facilità di integrazione**: Cisco Webex consente di integrare facilmente la propria applicazione nel loro sito, permettendo di abilitare tutte le funzionalità necessarie in modo comodo e rapido.
- **Gestione tramite API RESTful**: la gestione dei meeting avviene interamente attraverso API RESTful, le quali possono essere chiamate da qualsiasi ambiente, garantendo una notevole flessibilità.

- **Completezza delle API**: Cisco Webex offre una vasta gamma di API per soddisfare qualsiasi esigenza, ognuna delle quali è altamente personalizzabile in base alle specifiche necessità del progetto.
- **Documentazione chiara ed efficace**: la documentazione fornita da Cisco è ben strutturata e di facile comprensione, facilitando il processo di sviluppo.
- **Ambiente di prova delle API**: Cisco Webex offre la possibilità di testare le API in un ambiente dedicato senza la necessità di effettuare il login, utilizzando esempi predefiniti o personalizzati.
- **Supporto rapido ed efficiente**: in caso di necessità, il supporto fornito da Cisco è tempestivo e competente.

3.3 Configurazione di Webex

Per utilizzare i servizi offerti da Cisco Webex, è necessario compiere una serie di passaggi preliminari per ottenere un **access token**, che verrà utilizzato per autenticare tutte le chiamate alle API.

1. **Creazione di un account**: il primo passo consiste nel creare un account sul sito Webex for Developers, che permette l'accesso agli strumenti necessari per lo sviluppo.
2. **Creazione dell'integrazione**: successivamente si deve creare l'integrazione per l'applicazione sul proprio account. Durante questo processo sono stati specificati:
   - Il **nome dell'applicazione**
   - Una **descrizione dell'applicazione**
   - Un'**icona rappresentativa**
   - Gli **scopes**: di fondamentale importanza sono gli scopes. Questi definiscono tutte le operazioni che il nostro account potrà andare ad eseguire quando si effettuano chiamate alle API. È necessario selezionare tutti gli scopes necessari alle finalità del progetto, con la possibilità di abilitarli o disabilitarli in qualsiasi momento.
3. **Richiesta di un account sandbox**: per effettuare test senza limitazioni durante lo sviluppo, è necessario richiedere un account Sandbox tramite il proprio account creato in precedenza. Una volta ottenute le credenziali, si può usufruire di questo account speciale per accedere come amministratore di un'organizzazione e gestire tutti i meeting e le relative impostazioni attraverso una comoda interfaccia, Webex Control Hub. [8]
4. **Recupero dell'access token**: l'ultimo passaggio è stato ottenere l'access token necessario per autenticare le chiamate alle API. Bisogna effettuare una richiesta POST all'API `https://webexapis.com/v1/access_token` con il seguente body:

```
{
    "grant_type": "authorization_code",
    "client_id": "1234567890abcdef123456",
    "client_secret": "abcdef1234567890abcdef1234567890",
    "code":"12345678abcdef12345678abcdef"
}
```

I valori di `client_id`, `client_secret`, e `code` sono stati reperiti dalla pagina di integrazione dell'applicazione sul proprio account Webex for Developers. Se tutti i dati passati sono corretti, l'access token viene restituito come risposta.

3.4 Interfaccia Utente

L'obiettivo dell'interfaccia utente era fornire una visualizzazione chiara e intuitiva dei meeting programmati. A tal fine, è stato scelto un calendario come elemento principale dell'interfaccia.

In particolare, si è deciso di utilizzare *FullCalendar* [9], una famosa libreria Javascript open source che si integra perfettamente con React.

Grazie al **MUI Theme**, l'utente ha la possibilità di personalizzare i colori principali del sito a proprio piacimento tramite una pratica interfaccia laterale:

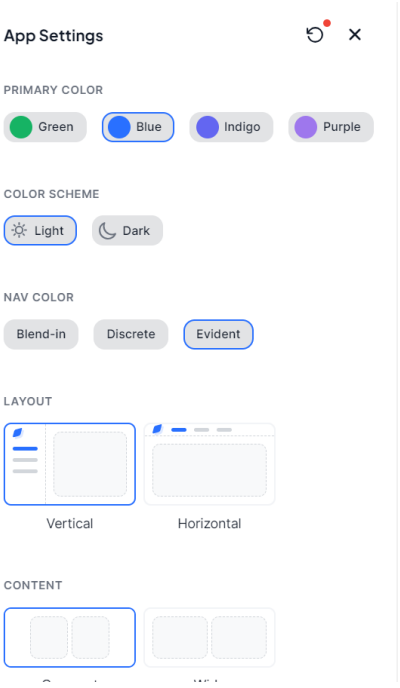

Figura 3.1: Tema chiaro

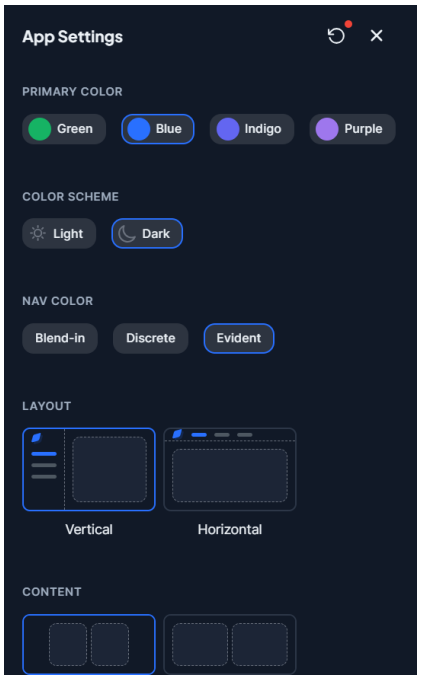

Figura 3.2: Tema scuro

Un esempio di visualizzazione della pagina utilizzando entrambi i temi:

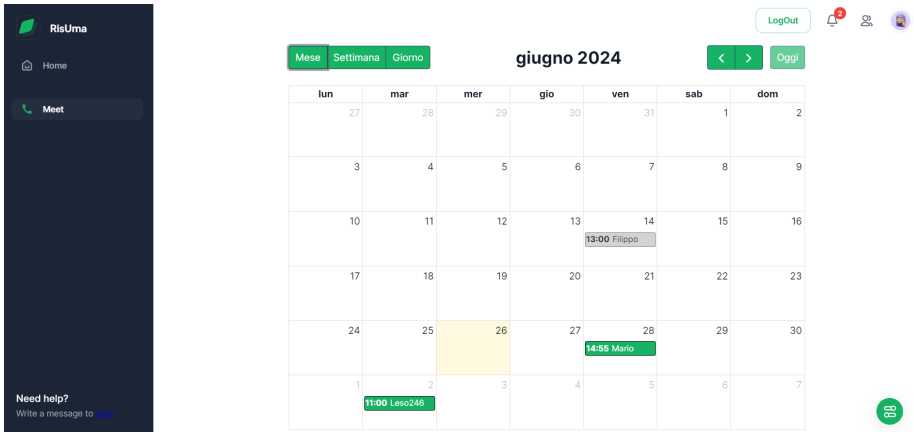

Figura 3.3: Visualizzazione Calendario - tema chiaro

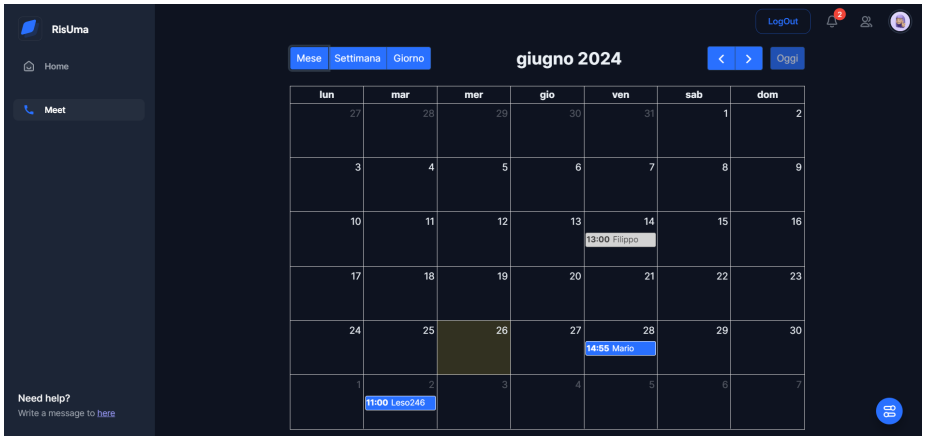

Figura 3.4: Visualizzazione Calendario - tema scuro

Nel caso siano presenti più eventi in un giorno nella visualizzazione mensile del calendario, è possibile utilizzare un pratico popup per avere una visione migliore:

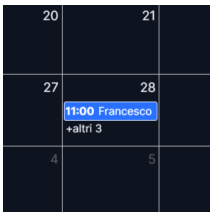

Figura 3.5: + altri meet

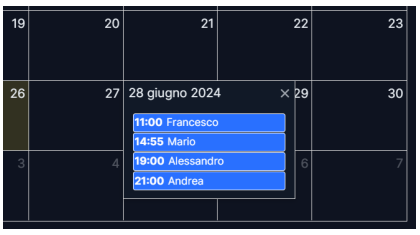

Figura 3.6: popup + meet

È anche disponibile una visualizzazione settimanale e giornaliera:

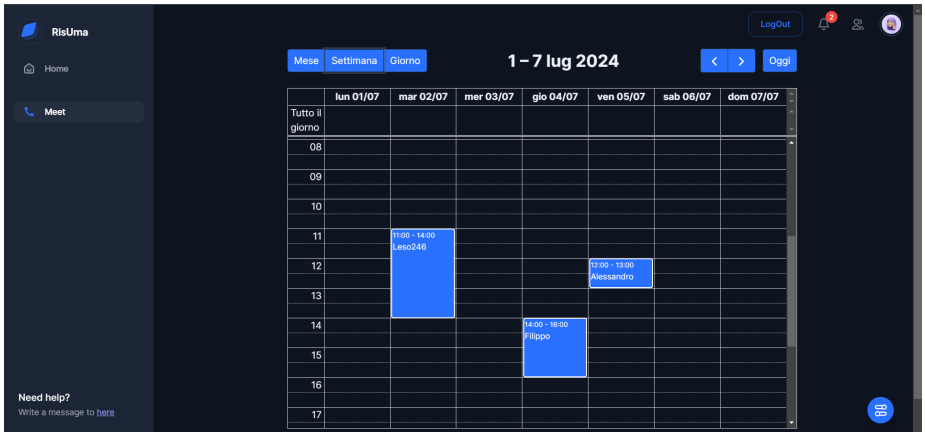

Figura 3.7: Visualizzazione Settimanale - tema scuro

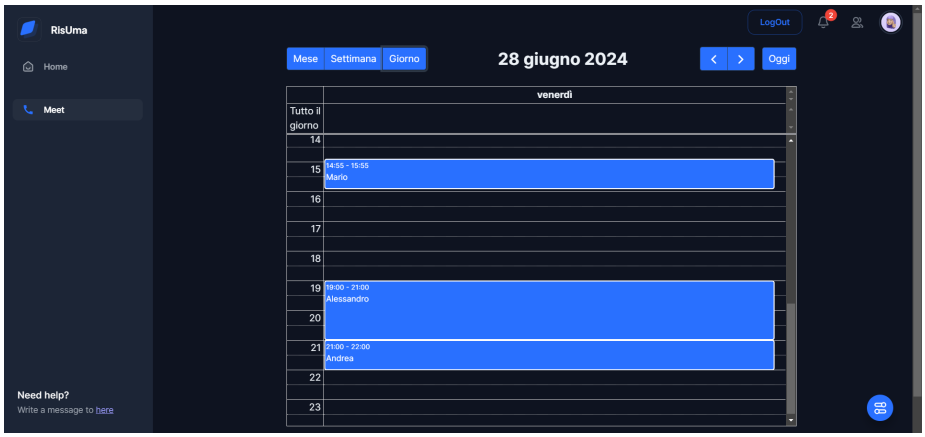

Figura 3.8: Visualizzazione Giornaliera - tema scuro

In seguito si entrerà nel dettaglio dei restanti elementi dell'interfaccia utente.

3.5 Creazione della tabella a database

Per garantire un'efficace gestione dei meeting, è stato necessario creare una tabella dedicata nel database aziendale. Questa tabella è stata progettata per memorizzare tutte le informazioni rilevanti relative ai meeting, le quali saranno utilizzate da *FullCalendar* [9] per permettere la corretta visualizzazione e gestione da parte degli utenti.

Per collegare un evento nel database aziendale con il corrispondente meeting su Webex, viene memorizzato l'ID univoco assegnato da Webex al momento della creazione del meeting. Inoltre, per ragioni di efficienza e praticità, viene salvato anche il link del meeting, generato anch'esso al momento della creazione.

Viene riportata di seguito la query con cui è stata generata la tabella:

```
CREATE TABLE meet (
    id BIGSERIAL PRIMARY KEY,
    data_inizio TIMESTAMP NOT NULL,
    data_fine TIMESTAMP NOT NULL,
    id_utente BIGINT NOT NULL,
    id_azienda BIGINT NOT NULL,
    link VARCHAR(200) NOT NULL,
    webex_id VARCHAR(200) NOT NULL,
    FOREIGN KEY (id_utente) REFERENCES utente(id_utente),
    FOREIGN KEY (id_azienda) REFERENCES aziende(id_azienda)
);
```

id	data_inizio	data_fine	id_utente	id_azienda	link	webex_id
50	2024-06-28 15:00:00	2024-06-28 16:00:00	73	51	webex.com/meet/ex1	93d7d864bd9b4
51	2024-07-13 14:30:00	2024-07-13 16:30:00	34	62	webex.com/meet/ex2	34421188b307b9

Tabella 3.1: Tabella a database di esempio

Nota: i timestamp a database vengono automaticamente visualizzati in un formato leggibile da un umano.

Segue una descrizione dettagliata delle colonne:

1. **id**:
    - **Tipo**: BIGSERIAL
    - **Descrizione**: chiave primaria della tabella. È un identificatore univoco generato automaticamente per ogni record.
2. **data_inizio**:
    - **Tipo**: TIMESTAMP
    - **Descrizione**: indica la data e l'ora di inizio del meeting. Questo campo non può essere nullo.
3. **data_fine**:
    - **Tipo**: TIMESTAMP
    - **Descrizione**: indica la data e l'ora di fine del meeting. Questo campo non può essere nullo.
4. **id_utente**:
    - **Tipo**: BIGINT
    - **Descrizione**: identificatore dell'utente associato al meeting. Questo campo è una chiave esterna che fa riferimento alla colonna `id_utente` della tabella `utente`. Non può essere nullo.

5. **id_azienda**:
    - **Tipo**: BIGINT
    - **Descrizione**: identificatore dell'azienda associato al meeting. Questo campo è una chiave esterna che fa riferimento alla colonna `id_azienda` della tabella `azienda`. Non può essere nullo.
6. **link**:
    - **Tipo**: VARCHAR(200)
    - **Descrizione**: contiene il link univoco del meeting generato da Webex.
7. **webex_id**:
    - **Tipo**: VARCHAR(200)
    - **Descrizione**: id univoco del meeting generato da Webex. Viene utilizzato per gestire tutte le operazioni che si effettuano sul meeting tramite le API di Webex.

Dato l'utilizzo di **JPA**, si è resa necessaria la mappatura della tabella a backend:

```
@Entity
@Table(name = "meet")
@Data
public class Meet {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  @Column(name = "id", nullable = false)
  private long id;

  @Column(name = "data_inizio", nullable = false)
  private Timestamp dataInizio;

  @Column(name = "data_fine",nullable = false)
  private Timestamp dataFine;

  @ManyToOne
  @JoinColumn(name = "id_utente", referencedColumnName = "id_utente", nullable = false)
  private Utente utente;

  @ManyToOne
  @JoinColumn(name = "id_azienda", referencedColumnName = "id_azienda", nullable = false)
  private Azienda azienda;

  @Column(name = "link", nullable = false)
  private String link;

  @Column(name = "webex_id", nullable = false)
  private String webexId;
}
```

3.6 Creazione dei meeting

3.6.1 Frontend

3.6.2 Backend

3.7 Recupero dei meeting

3.7.1 Frontend

3.7.2 Backend

3.8 Modifica dei meeting

3.8.1 Frontend

3.8.2 Backend

3.9 Eliminazione dei meeting

3.9.1 Frontend

3.9.2 Backend

Capitolo 4

Idee future

Capitolo 5

Ulteriori sviluppi secondari

Bibliografia

[1] Wikipedia. «React (web framework).» (), indirizzo: https://it.wikipedia.org/wiki/React_(web_framework). (Data di accesso: 25.06.2024).

[2] Wikipedia. «TypeScript.» (), indirizzo: https://it.wikipedia.org/wiki/TypeScript. (Data di accesso: 25.06.2024).

[3] Wikipedia. «Java (linguaggio di programmazione).» (), indirizzo: https://it.wikipedia.org/wiki/Java_(linguaggio_di_programmazione). (Data di accesso: 24.06.2024).

[4] Microsoft. «Che cos'è Spring Boot di Java?» (), indirizzo: https://azure.microsoft.com/it-it/resources/cloud-computing-dictionary/what-is-java-spring-boot. (Data di accesso: 25.06.2024).

[5] A. Gatu. «Cos'è Maven, a cosa serve e come si usa.» (2024), indirizzo: https://www.nextre.it/cose-maven-si-usa/. (Data di accesso: 25.06.2024).

[6] V. Racca. «Introduzione a JPA e mapping delle relazioni.» (2020), indirizzo: https://www.vincenzoracca.com/blog/framework/jpa/jpa-reletions/. (Data di accesso: 25.06.2024).

[7] Geekandjob. «Cos'è PostgreSQL.» (), indirizzo: https://www.geekandjob.com/wiki/postgresql. (Data di accesso: 25.06.2024).

[8] Webex. «Developer Sandbox.» (), indirizzo: https://developer.webex.com/docs/developer-sandbox-guide.

[9] A. Arshaw. «FullCalendar.» (), indirizzo: https://fullcalendar.io/.

[10] S. Lesinigo, *Appunti del corso di Interazione Uomo-Macchina*, Università degli Studi di Milano-Bicocca, Docente: Cabitza Federico Antonio Niccolò Amedeo, Anno Accademico 2023-2024.

Appendice A

Euristiche di Nielsen

Delineate da Jakob Nielsen in collaborazione con Rolf Morich nel 1994, sono un insieme di principi generali ampaimente utilizzati ancora oggi che supportano la progettazione di sistemi interattivi usabili [10]:

1. Visibilità dello stato del sistema
2. Corrispondenza tra sistema e mondo reale
3. Controllo e libertà dell'utente
4. Consistenza e standard
5. Riconoscimento piuttosto che ricordo
6. Flessibilità ed efficienza d'uso
7. Estetica e minimalismo
8. Prevenzione degli errori
9. Aiuto nel rilevare e correggere gli errori
10. Aiuto e documentazione