LiveLuxe

Losi Lorenzo, Apolone Lorenzo.

Corso di laurea: LITINF

Progetto Tecnologie Internet.

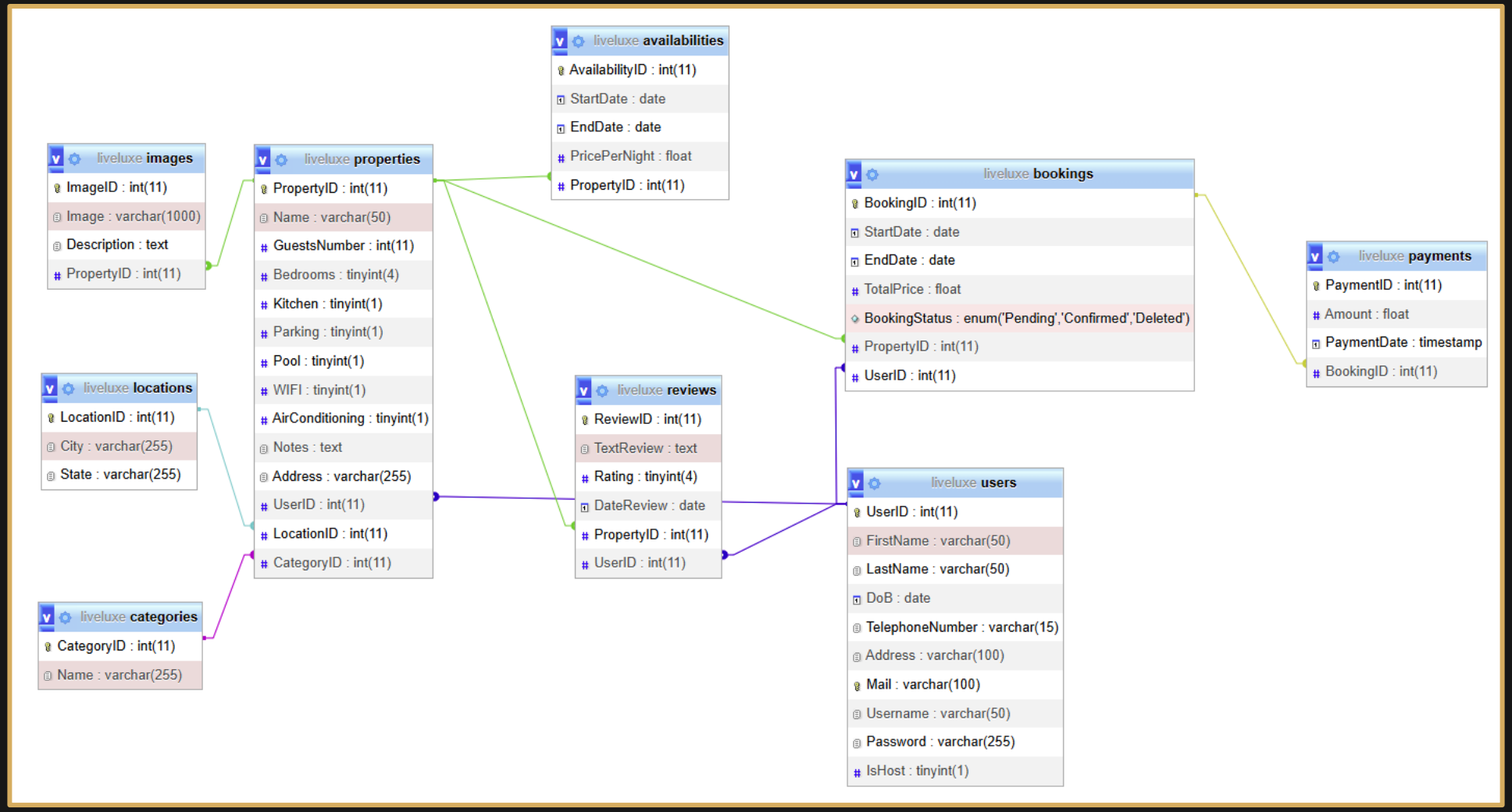
FUNZIONALITÀ PRINCIPALI

LiveLuxe mette a disposizione immobili di tutte le categorie, offre la possibilità di ricercare appartamenti in modo dettagliato, fornisce informazioni su di essi e la possibilità di affittarli per un breve periodo.

Inoltre offre la possibilità di mettere in affitto i propri immobili e

pubblicarli su sito.

STRUTTURA DATABASE



STRUTTURA SITO

1) HOME-PAGE: Pagina iniziale del sito a cui tutti gli utenti possono accedere e visualizzare immobili disponibili.

2) PARTE PUBBLICA: Parte comprendente login, registrazione, risultati di ricerca, informazioni e date di disponibilità di un immobile.

3) PARTE PRIVATA GUEST: Parte comprendente pagina di prenotazione immobile e conseguente pagamento.

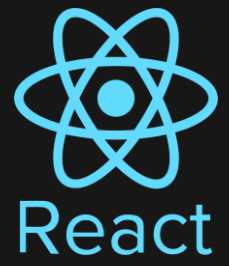
4) PARTE PRIVATA HOST: Parte in cui un utente può mettere in affitto e pubblicare un immobile proprio, visualizzare propri immobili e prenotazioni ricevute.

TECNOLOGIE UTILIZZATE 1/2

REACT

Permette di suddividere l'interfaccia utente in componenti riutilizzabili, il che rende il codice più modulare, mantenibile e facilmente testabile. Inoltre offre un sistema di gestione dello stato che aiuta a tenere traccia dei cambiamenti nell'interfaccia utente in modo efficiente, senza bisogno di manipolare direttamente il DOM (rappresenta la struttura della pagina web).

Invece di aggiornare direttamente il DOM ogni volta che cambia lo stato dell'applicazione, React prima aggiorna il Virtual DOM, lo confronta con il DOM reale e applica solo le modifiche necessarie. Questo rende React più veloce ed efficiente rispetto alla manipolazione diretta del DOM tradizionale.



TECNOLOGIE UTILIZZATE 2/2

NODE JS

Consente di utilizzare JavaScript sia sul lato client che sul lato server, il che semplifica lo sviluppo di applicazioni full-stack.

Utilizza un ciclo di eventi (event loop), che consente di gestire le richieste in modo asincrono. Questo significa che non blocca il programma in attesa di una risposta da una richiesta di rete o da un'operazione di I/O, permettendo di elaborare più richieste contemporaneamente.

Inoltre è ben adatto per lo sviluppo di applicazioni basate su microservizi, dove diverse componenti dell'applicazione sono suddivise in piccoli servizi indipendenti.

CYBERSECURITY 1/3

In fase di registrazione e login la password viene criptata con SHA-256 e viene inviata al server.

A lato server viene salvata la stringa HASH corrispondente.

Quando viene effettuato un accesso si fornisce al server il risultato della funzione HASH della password.

In questo modo anche se il server venisse compromesso o i dati inviati dal client fossero intercettati, il malintenzionato avrebbe accesso alla password già criptata.

```
// Creating SHA-256 hash password to send to server.
const hashPassword = async (password) => {
    const encoder = new TextEncoder();
    const data = encoder.encode(password);
    const hash = await crypto.subtle.digest('SHA-256', data);
    return Array.from(new Uint8Array(hash))
        .map((byte) => byte.toString(16).padStart(2, '0'))
        .join('');
};
```

CYBERSECURITY 2/3

Il processo di Login è resistente alla SQL Injection, poiché quando un utente effettua il login il server utilizza una query parametrizzata e non concatenata.

In questo modo se un malintenzionato provasse ad attaccare il database con una stringa del tipo OR 1=1 questa verrebbe trattata come dato e non come parte della query.

```
function getUserByUsernameAndPassword(username, password, callback) {
    const connection = createConnection();
    const query = `SELECT UserID, Username, IsHost, FirstName, LastName FROM users WHERE Username = ? AND Password = ?`;

    connection.query(query, [username, password], (err, results) => {
        if (err) {
            console.error("Error fetching user:", err.message);
            callback(err, null);
        } else {

            if (results.length === 0) {
                callback(null, []);
            } else {
                callback(null, results);
            }
        }
        connection.end();
    });
}
```

CYBERSECURITY 3/3

Una Protected Route serve a limitare l'accesso a determinate pagine solo agli utenti autenticati.

Controlla se l'utente è autenticato, se sì reindirizza il componente richiesto altrimenti reindirizza l'utente alla pagina di Login.

```
<Route path="/house/:id" element={ <HouseDescription /> } />
<Route path="/login" element={<Login />} />
<Route path="/registration" element={<Registration />} />
<Route path="/booking-page" element={<ProtectedRoute><BookingPage /></ProtectedRoute>} />
<Route path="/host-properties" element={<ProtectedRoute><HostProperties /></ProtectedRoute>} />
```

IMPLEMENTAZIONI FUTURE

1) RECENSIONI: Gestire meccanismo di recensioni e possibilità di inserimento da parte dei soli utenti che hanno soggiornato nell'immobile.

2) MAPPA: Visualizzare mappa per indirizzo dell'immobile.

3) CANCELLAZIONE PRENOTAZIONE: Possibilità di cancellare una prenotazione.

4) PAGAMENTI: Migliorare gestione pagamenti.