**Medibot**

**Medibot** è un’applicazione web moderna e sicura, progettata per facilitare la gestione delle prenotazioni di visite mediche all’interno di uno studio. La piattaforma consente ai **pazienti** di registrarsi, accedere, consultare le disponibilità dei **medici** e prenotare o cancellare appuntamenti in maniera semplice e intuitiva. I **medici**, a loro volta, possono gestire in autonomia la propria agenda, definendo i giorni e le fasce orarie in cui sono disponibili a ricevere i pazienti.

L’applicazione è stata sviluppata secondo un’architettura modulare che ne facilita l’estensione e la manutenzione. Il **frontend** è stato realizzato con HTML, CSS e JavaScript, mentre il **backend** si basa su Node.js con Express.js. L’**autenticazione** è gestita tramite **JSON Web Token (JWT)**, garantendo un accesso sicuro alle funzionalità riservate. Le informazioni su utenti, disponibilità e prenotazioni sono memorizzate in un database relazionale **MySQL**, strutturato in modo efficiente e normalizzato.

**Struttura del database**

Il database si compone di tre tabelle principali:

* **users**: memorizza i dati degli utenti (sia pazienti che medici), tra cui ID, nome, email, password e ruolo.
* **doctor\_availability**: raccoglie le disponibilità fornite dai medici, con riferimento al medico, data, orario di inizio e orario di fine.
* **appointments**: rappresenta le prenotazioni effettuate, collegate a un paziente, a un medico e a una specifica fascia oraria. Le prenotazioni possono avere stato “confermata” o “cancellata”.

Sono inoltre implementate **relazioni a cascata**, che garantiscono l’eliminazione automatica di disponibilità e appuntamenti nel caso in cui un utente venga rimosso.

**Funzionalità e API REST**

Medibot implementa una gamma completa di **API RESTful**, che permettono l’interazione con il sistema:

* **Autenticazione e autorizzazione**:
  + POST /auth/register: registrazione nuovi utenti
  + POST /auth/login: login e generazione del token JWT
  + GET /auth/me: recupero informazioni dell’utente autenticato
* **Gestione disponibilità (solo medici)**:
  + POST /doctor/availability: inserimento nuove fasce orarie
  + GET /doctor/availability: visualizzazione delle disponibilità esistenti
  + DELETE /doctor/availability/:id: eliminazione di una disponibilità
* **Prenotazioni (solo pazienti)**:
  + POST /appointments/book: prenotazione appuntamenti
  + GET /appointments: visualizzazione degli appuntamenti propri
  + DELETE /appointments/:id: cancellazione prenotazioni

Il sistema controlla i ruoli utente tramite middleware di autorizzazione, assicurando che solo i **pazienti** possano prenotare o cancellare appuntamenti, e solo i **medici** possano gestire le disponibilità.

**Testing con cURL**

È possibile verificare il corretto funzionamento delle API tramite strumenti come **cURL**, seguendo questi passaggi:

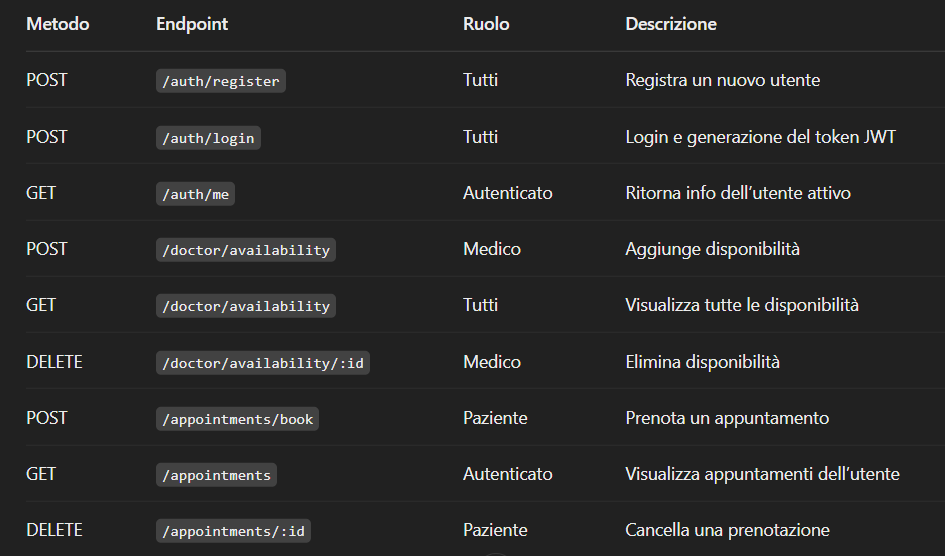
1. **Registrazione** di un utente medico o paziente.
2. **Login** e ottenimento del token JWT.
3. Uso del **token** nelle intestazioni delle richieste per autenticare l’utente.
4. I medici possono aggiungere disponibilità, mentre i pazienti possono consultarle, prenotare, visualizzare e cancellare le visite.

**1. Panoramica dell’Applicazione**

Tecnologie utilizzate:  
- Frontend: HTML, CSS, JavaScript  
- Backend: Node.js + Express.js  
- Autenticazione: JWT  
- Database: MySQL

3. Funzionamento delle Richieste lato Backend

Il backend segue l’architettura MVC (Model-View-Controller):  
1. Il client invia una richiesta HTTP (GET, POST, DELETE...).  
2. La route riceve la richiesta e la inoltra al relativo controller.  
3. Il controller gestisce la logica e comunica con il modello.  
4. Il modello accede al database MySQL, esegue query e restituisce i dati.  
5. I dati vengono inviati come risposta JSON al client.

**2. Metodi HTTP e Funzionalità Implementate****3. Casi di Test delle API (via CURL)**

**Registrazione**:  
curl -X POST http://localhost:3000/auth/register \  
-H "Content-Type: application/json" \  
-d '{"name":"Laura Bianchi","email":"laura@example.com","password":"test123","role":"patient"}'  
  
**Login**:  
curl -X POST http://localhost:3000/auth/login \  
-H "Content-Type: application/json" \  
-d '{"email":"laura@example.com","password":"test123"}'  
  
**Aggiunta disponibilità (medico):**  
curl -X POST http://localhost:3000/doctor/availability \  
-H "Authorization: Bearer <TOKEN>" \  
-H "Content-Type: application/json" \  
-d '{"available\_date":"2025-04-08","start\_time":"09:00","end\_time":"12:00"}'  
  
**Prenotazione visita (paziente):**  
curl -X POST http://localhost:3000/appointments/book \  
-H "Authorization: Bearer <TOKEN>" \  
-H "Content-Type: application/json" \  
-d '{"doctor\_id":1,"availability\_id":4}'  
  
**Cancellazione prenotazione:**  
curl -X DELETE http://localhost:3000/appointments/3 \  
-H "Authorization: Bearer <TOKEN>"