Java dokumentacja

> Oddanie
>
> czas developmentu: 5 tygodni
> oddanie koło 20 stycznia ~20 min na osobę
> repozytorium publiczne github: https://github.com/julnac/aplikacje-przemyslowe

1. Opis projektu

Celem projektu jest stworzenie **kompleksowej aplikacji webowej w technologii Java Spring Boot** do zarządzania salonem usługowym (fryzjer / kosmetyczka). System obsługuje trzy główne strefy funkcjonalne:

- **Strefa publiczna** – dostępna bez logowania, prezentująca ofertę usług, cennik oraz opinie klientów.
- **Panel klienta** – umożliwiający rejestrację, logowanie, rezerwację wizyt w inteligentnym kalendarzu oraz zarządzanie rezerwacjami.
- **Panel administratora** – pozwalający na zarządzanie wizytami, ofertą usług, klientami oraz analizę statystyk.

Kluczowym elementem systemu jest **algorytm rezerwacji**, który:

- sumuje czas trwania wybranych usług,
- wyszukuje odpowiednio długie wolne przedziały czasowe,
- zarządza cyklem życia rezerwacji (UTWORZONA → POTWIERDZONA PRZEZ KLIENTA → ZATWIERDZONA PRZEZ ADMINA → ZABLOKOWANA).

Projekt realizuje pełne spektrum zagadnień Spring: JPA, JdbcTemplate, REST API, Security, Thymeleaf, testy oraz dokumentację API.

---

2. Architektura systemu

2.1 Styl architektoniczny

- Architektura **warstwowa (Layered Architecture)**
- Backend: **Spring Boot (REST + MVC)**
- Frontend: Angular (na początku sam swagger wystarczy)
- Baza danych: **PostgreSQL / MySQL**

Warstwy:

- Controller (REST + MVC)

- Service (logika biznesowa)
- Repository / DAO
- Model (Encje JPA)
- DTO + Mapper

---

3. Model domenowy (Encje JPA)

3.1 Lista encji

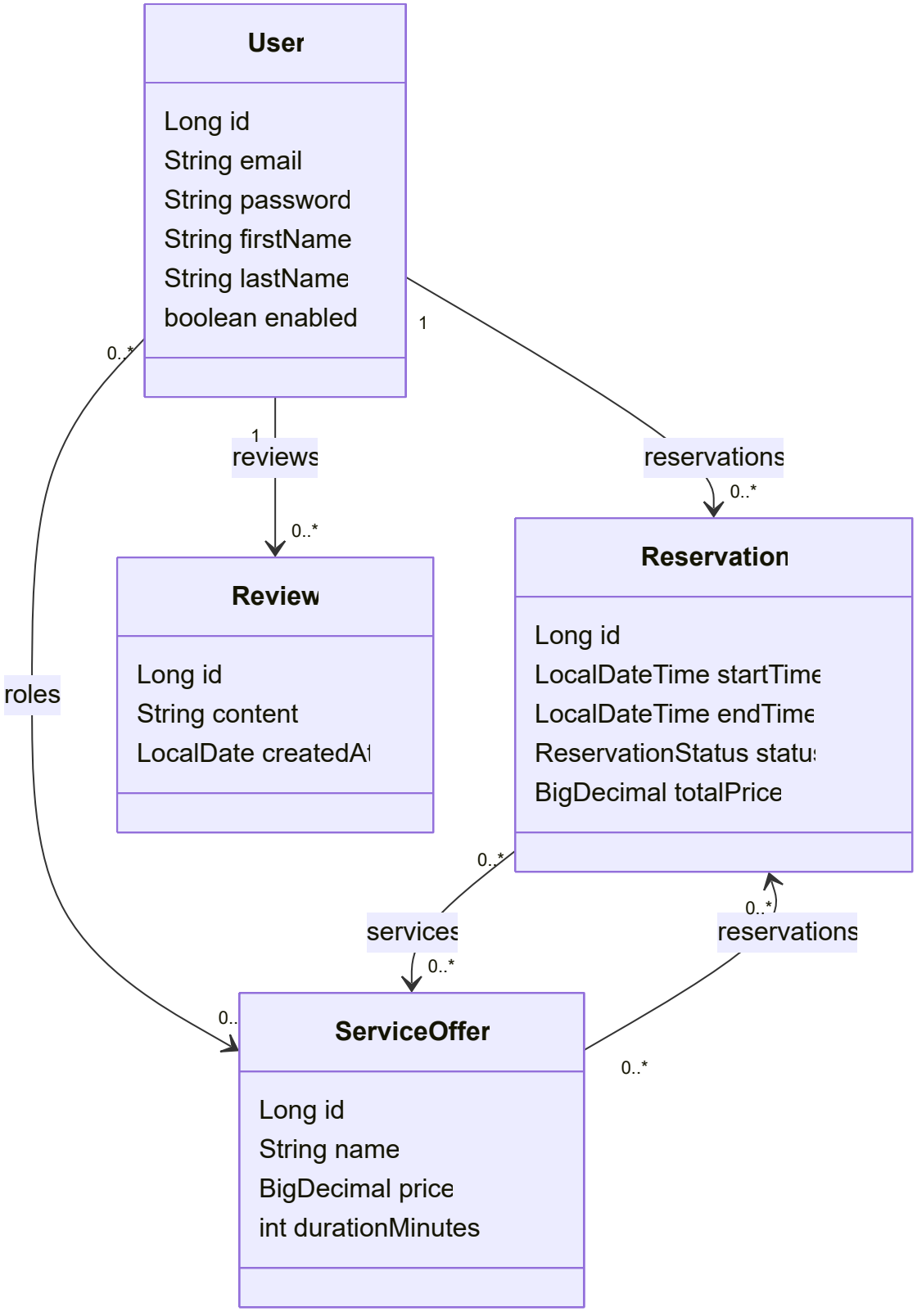
- User (role)
- ServiceOffer (Usługa)
- Employee (Pracownik)
- Reservation (Wizyta)
- EmployeeService (specjalizacja pracownika)
- Review (Opinia)
- ReservationService (encja łącząca)

3.2 Relacje

- Employee .. ServiceOffer (specjalizacje)
- Employee 1..* Reservation
- User 1..* Reservation
- Reservation .. ServiceOffer
- User 1..* Review
- User .. Role

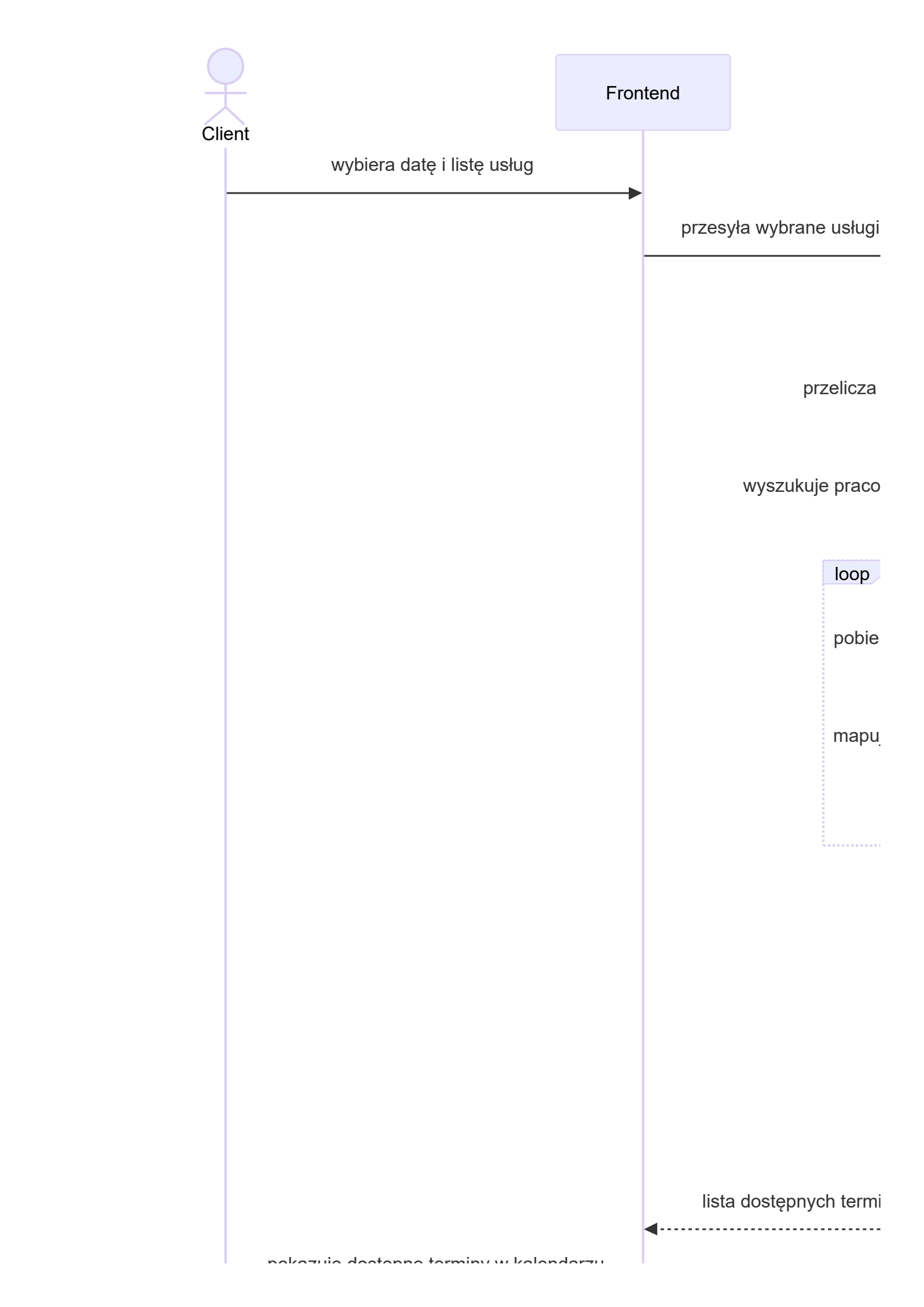
---

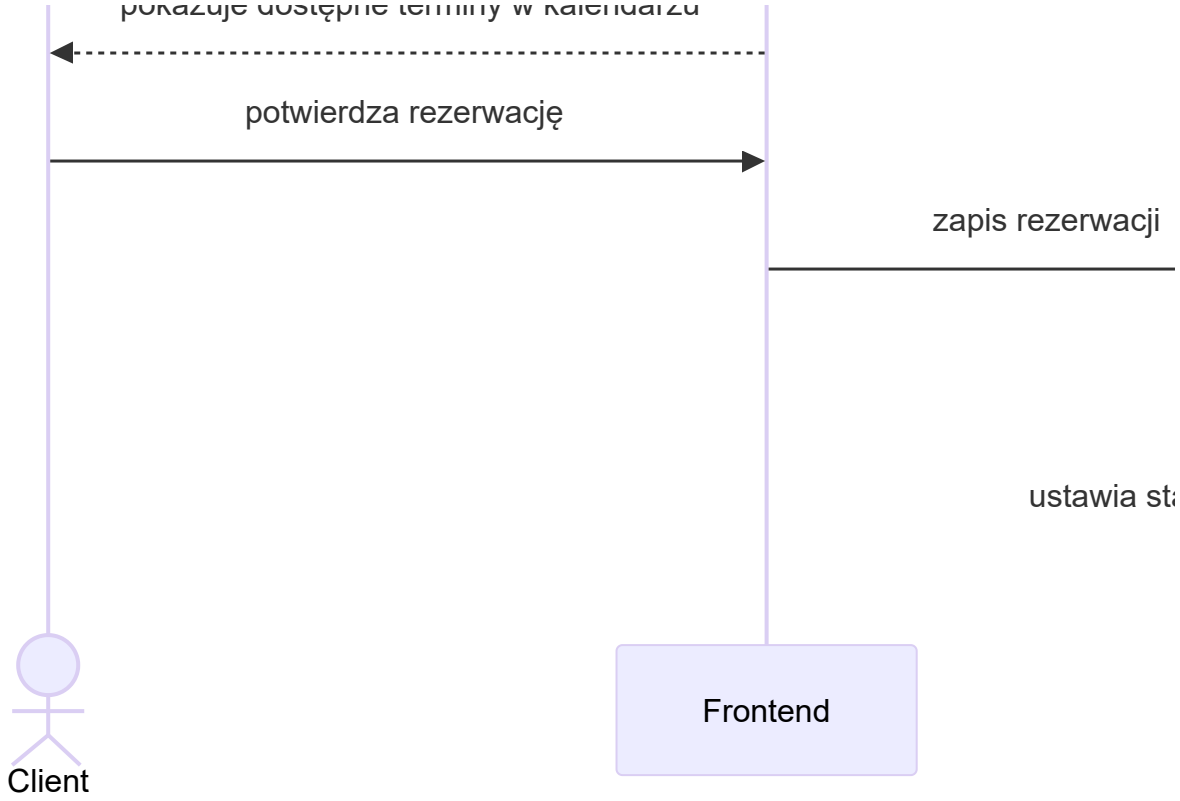
4. Diagram klas



5. Algorytm rezerwacji

5.1 Kroki algorytmu


Client
Frontend
wybiera datę i listę usług
przesyła wybrane usługi
przelicza
wyszukuje praco
loop
pobie
mapu
lista dostępnych termi



1. Klient wybiera:
   - datę
   - listę usług
2. System:
   - sumuje czas usług
   - przelicza na liczbę slotów 30-minutowych
3. System wyszukuje:
   - pracowników, którzy **obsługują wszystkie wybrane usługi**
4. Dla każdego pracownika:
   - pobiera rezerwacje z danego dnia
   - mapuje je na zajęte sloty
5. Algorytm:
   - generuje wszystkie możliwe sloty w godzinach otwarcia
   - wyszukuje **ciąg wolnych slotów ≥ wymagany czas**
6. Frontend:
   - pokazuje dostępne terminy w kalendarzu
7. Zapis rezerwacji:
   - blokuje sloty
   - ustawia status `CREATED`

5.2 Statusy rezerwacji

- CREATED – utworzona
- CONFIRMED_BY_CLIENT – potwierdzona przez klienta
- APPROVED – zatwierdzona przez admina

- CANCELLED – anulowana

Po statusie APPROVED edycja po stronie klienta jest blokowana, wtedy wizyta może być już tylko anulowana CANCELLED.

---

6. REST API – struktura

6.1 Przykładowe endpointy

Usługi

- GET /api/v1/services
- POST /api/v1/services (ADMIN)
- PUT /api/v1/services/{id}
- DELETE /api/v1/services/{id}

Rezerwacje

- GET /api/v1/reservations/my
- POST /api/v1/reservations
- PUT /api/v1/reservations/{id}/confirm
- PUT /api/v1/reservations/{id}/approve (ADMIN)

---

7. JdbcTemplate – Statystyki

Przykładowe funkcjonalności:

- Średnia kwota wizyty klienta
- Średni czas trwania wizyty

Realizacja:

- Dedykowane DAO
- Zapytania SQL z GROUP BY
- RowMapper

---

8. Spring Security

- Role: ROLE_CLIENT, ROLE_ADMIN
- BCryptPasswordEncoder
- Formularz logowania
- Ochrona endpointów REST
- @WithMockUser w testach

9. Testowanie

9.1 Rodzaje testów

- @DataJpaTest – repozytoria
- Mockito – serwisy
- @WebMvcTest – kontrolery
- @SpringBootTest – integracyjne

9.2 Wymagania

- Min. 70% coverage (JaCoCo)
- Happy Path + Error Cases

---

10. Harmonogram prac (propozycja)

---

11. Możliwe rozszerzenia (opcjonalne)

- Powiadomienia email
- Rezerwacje cykliczne
- Wielu pracowników
- Front SPA (React/Angular)