

Projekt zespołowy systemu informatycznego

Sprint 1, grupa B , raporty ze spotkań *Scrum Meetings*

Sprint Backlog

- task1:** Inicjalizacja środowiska – konfiguracja plików startowych projektu
- task2:** Utworzenie struktury frontendu (Vite + React + Tailwind + routing, App.tsx, main.tsx, konfiguracja routera)
- task3:** Przygotowanie struktury backendu (Express + routery, konfiguracja błędów)
- task4:** Implementacja bazy danych SQLite – konfiguracja typów i schematów (ORM)
- task5:** Konfiguracja GitHub – dodanie repozytorium i pliku README.md
- task6:** Opracowanie przepływu logowania i rejestracji – formularze i logika sesji
- task7:** Widok profilu użytkownika – dane + działające wylogowanie
- task14:** Formularz rezerwacji i logika widoku klienta
- task16:** Konfiguracja systemu do wysyłki wiadomości e-mail – potwierdzenia rezerwacji i pliki ICS
- task17:** Middleware Express Rate-Limit – ograniczenie (1 rezerwacja / 30 s na IP + e-mail)
- task18:** Widok „Moje rezerwacje” / Dashboard użytkownika
- task19:** Middleware autoryzacji API wg ról OWNER/WORKER + kontekstu companyId
- task20:** Panel ustawień firmy – interfejs właściciela
- task32:** Przygotowanie panelu administratora – układ i komponenty

Raport z pierwszego tygodnia realizacji sprintu 1

06.11.2025

Scrum Meeting 1 (27.10.2024)

Na pierwszym spotkaniu rozmawialiśmy o doprecyzowaniu pomysłu i tym, jak poukładać narzędzia pracy zespołowej. Ustaliliśmy, że idziemy w monorepo i omówiliśmy zasady pracy z Jira i GitHubem: jak ma wyglądać projekt i board w Jira (kolumny, workflow, etykiety), jak nazywamy zadania i gałęzie oraz jaki format commitów chcemy przyjąć. Ustaliliśmy też, że repozytorium na GitHubie będzie naszym centralnym miejscem pracy. W części technicznej uzgodniliśmy, że przygotujemy jedynie szkielet: frontend na Vite + React + Tailwind oraz backend na Expressie z ts-node, a w głównym package.json dodamy skrypty do uruchamiania obu serwerów. Uzgodniliśmy minimalne połączenie front–back (CORS i prosty health-check), żeby można było lokalnie włączyć środowisko i płynnie ruszyć z pracami w kolejnych tygodniach.

Scrum Meeting 2 (06.11.2025)

Na drugim spotkaniu skupiliśmy się na tym, jak rozwijać funkcjonalności na bazie przygotowanego szkieletu. Ustaliliśmy, że po stronie backendu potrzebujemy kluczowych middleware'ów (helmet, cors, sesje, parser JSON) oraz logowania żądań, a autoryzację opieramy na sesji: po zalogowaniu przechowujemy w sesji identyfikator, e-mail i rolę (USER/WORKER/OWNER/PLATFORM_ADMIN), żeby móc sprawnie weryfikować uprawnienia. Po stronie frontendu omówiliśmy pełny przepływ logowania i rejestracji: LoginPage z walidacjami, podglądem hasła, komunikatami i przekierowaniem zależnym od roli oraz RegisterPage z walidacją e-maila, wskaźnikiem siły hasła, potwierdzeniem hasła i akceptacją

Projekt zespołowy systemu informatycznego

regulaminu. Uzgodniliśmy też dodanie AuthContext do obsługi sesji i ProtectedRoute do kontroli dostępu oraz automatycznych przekierowań.

Zadania realizowane przez poszczególnych deweloperów:

Rafał Gola:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Inicjalizacja środowiska – konfiguracja plików startowych projektu	4	ukończone
Konfiguracja GitHub, dodanie repozytorium i pliku	2	ukończone

Przemysław Habdas:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Przygotowanie struktury backendu	3	Ukończone
Middleware autoryzacji API wg ról OWNER/WORKER + kontekstu companyId.	4	W realizacji
Middleware Express Rate-Limit (1 rezerwacja/30 s per IP + e-mail).	2	W realizacji

Kamil:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Utworzenie struktury frontendu (Vite + React + Tailwind, `App.tsx`, `main.tsx`, konfiguracja routera)	5	Ukończone

Jakub:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Opracowanie przepływu logowania i rejestracji – koncepcja widoków i logiki sesji`	2	Ukończony
Widok profilu użytkownika z danymi i działającym wylogowaniem.	4	W realizacji

Daniel:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Wstępna struktura plików pod przyszłą bazę i ORM	2	Ukończony

Projekt zespołowy systemu informatycznego

Cele osiągnięte w bieżącym tygodniu

- Ustaliliśmy strukturę projektu w architekturze monorepo (frontend + backend).
- Skonfigurowaliśmy środowisko deweloperskie i repozytorium GitHub.
- Uruchomiliśmy podstawowy szkielet aplikacji (Vite + React + Tailwind + Express).
- Ustaliliśmy zasady pracy zespołowej w Jira oraz format commitów.
- Połączylismy frontend z backendem testowym endpointem (health-check).
- Przygotowaliśmy wspólne pliki konfiguracyjne TypeScript.

Stan tablicy Kanban na dzień 27.10.2024

The Kanban board displays the following tasks:

- OGRÓDNE**: 3 items
 - Formularz rezerwacji (GRUPAB-14, PH)
 - Auto-przydział pracownika (GRUPAB-15, PH)
 - E-mail potwierdzenia rezerwacji (GRUPAB-16, NR)
- TODO**: 7 items
 - Panel Admina (approve/reject) (GRUPAB-32, R)
 - Godzinny pracy (per dzień) (GRUPAB-11, DC)
 - Algorytm wolnych slotów (GRUPAB-13, PH)
 - Rate limit POST / reservations (GRUPAB-17, PH)
 - Wyszukiwarka firm (GRUPAB-9, R)
 - Moje konto + wylogowanie (GRUPAB-7, R)
 - RBAC firmy (Owner/Worker) (GRUPAB-19, PH)
- IN PROGRESS**: 3 items
 - Backend scaffold (GRUPAB-3, ✓, PH)
 - Rejestracja i logowanie (sesje) (GRUPAB-6, ✓, R)
 - Baza i ORM (Prisma + SQLite) (GRUPAB-4, ✓, DC)
- REVIEW**: 3 items
 - Inicjalizacja środowiska (GRUPAB-1, NR)
 - GitHub — konfiguracja repozytorium (GRUPAB-5, NR)
 - Frontend scaffold (GRUPAB-2, NR)
- II SPRINT DONE**: 0 items

Stan tablicy Kanban na dzień 06.11.2025

The Kanban board displays the following tasks:

- OGRÓDNE**: 3 items
 - Formularz rezerwacji (GRUPAB-14, KJ)
 - Auto-przydział pracownika (GRUPAB-15, PH)
 - E-mail potwierdzenia rezerwacji (GRUPAB-16, NR)
- TODO**: 4 items
 - Panel Admina (approve/reject) (GRUPAB-32, R)
 - Godzinny pracy (per dzień) (GRUPAB-11, DC)
 - Algorytm wolnych slotów (GRUPAB-13, PH)
 - Wyszukiwarka firm (GRUPAB-9, R)
- IN PROGRESS**: 3 items
 - RBAC firmy (Owner/Worker) (GRUPAB-19, ✓, PH)
 - Rate limit POST / reservations (GRUPAB-17, ✓, PH)
 - Moje konto + wylogowanie (GRUPAB-7, ✓, R)
- REVIEW**: 6 items
 - Inicjalizacja środowiska (GRUPAB-1, NR)
 - Frontend scaffold (GRUPAB-2, KJ)
 - Backend scaffold (GRUPAB-3, ✓, PH)
 - Rejestracja i logowanie (sesje) (GRUPAB-6, B)
 - Baza i ORM (Prisma + SQLite) (GRUPAB-4, DC)
 - GitHub — konfiguracja repozytorium (GRUPAB-5, NR)
- II SPRINT DONE**: 0 items

Projekt zespołowy systemu informatycznego

Commity z Githuba – stan na dzień 06.11.2025

Fix missing newline at end of AdminDashboard.tsx	...
Fix missing newline at end of RegisterPage.tsx	...
Fix missing newline at end of LoginPage.tsx	...
Refactor HomePage component layout and content	...
Implement ProtectedRoute component for route protection	...
Add Logo component with customizable properties	...
Implement routing with React Router	...
Refactor App component with navigation and footer	...
Add main.tsx to initialize React application	...
Add initial HTML structure for frontend	...
Add companies route with CRUD operations	...
Refactor auth routes to use local db methods	...

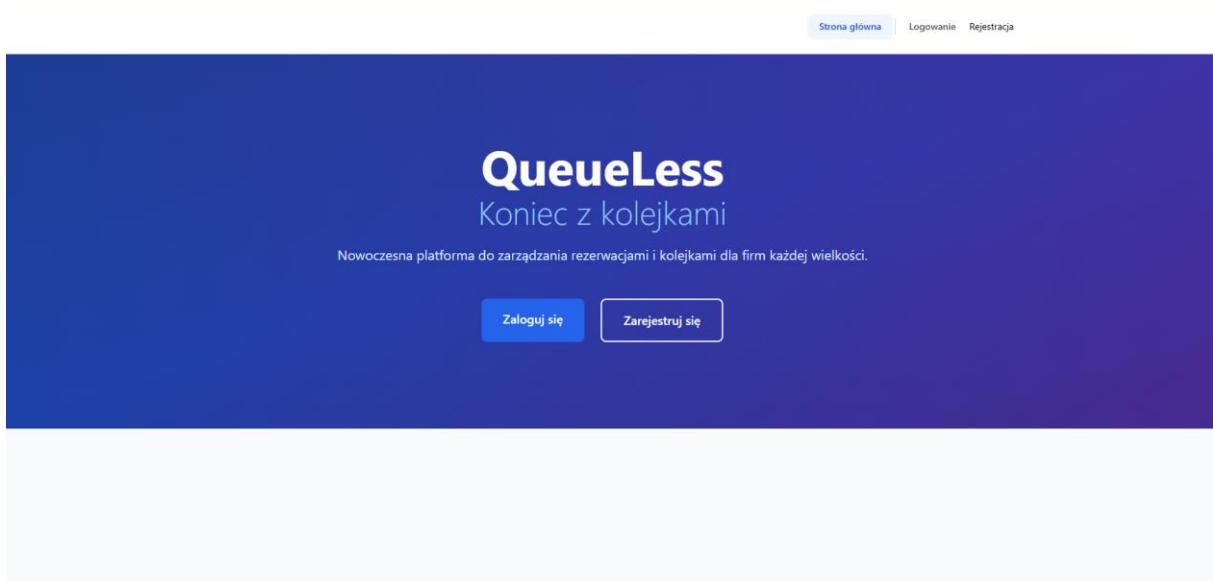
Refactor LoginPage component for improved structure	...
Update README.md	...
Add PostCSS configuration with Tailwind CSS	...
Modify Tailwind config for content and plugins	...
Add Tailwind CSS configuration file	...
Add Vite configuration for React app with proxy	...
Add TypeScript configuration for frontend package	...
Create tsconfig.json for backend package	...
Add package.json for frontend with dependencies	...
Initialize package.json for backend project	...
Add entries to .gitignore file	...
Add Prettier configuration file	...
Add ESLint configuration for TypeScript project	...
Add tsconfig.base.json with compiler options	...

Projekt zespołowy systemu informatycznego

Add ESLint configuration for TypeScript project	...
Refaue pushed 1 commit • 330d594...1e49845 • 4 days ago	...
Add tsconfig.base.json with compiler options	...
Refaue pushed 1 commit • 54fc579...330d594 • 4 days ago	...
Create package-lock.json	...
Refaue pushed 1 commit • 955c123...54fc579 • 4 days ago	...
Refactor package.json for monorepo structure	...
Refaue pushed 1 commit • 306e032...955c123 • 4 days ago	...
GRUPAB-3: Backend scaffold	...
x-PrMo pushed 1 commit • c53dd4...006e032 • 9 days ago	...
Create reservationService.ts	...
Dan69Dan pushed 1 commit • c628cd4...c53dd4 • 9 days ago	...
Create shifts.ts	...
Dan69Dan pushed 1 commit • 6b3ef5e...c628cd4 • 9 days ago	...
Add ReservationsPage component for managing reservations	...
kkkojus pushed 1 commit • 7e14bd6...6b3ef5e • 9 days ago	...
Add HomePage component with features and CTA sections	...
kkkojus pushed 1 commit • 3de99f1...7e14bd6 • 9 days ago	...
Initialize App component with routing setup	...
kkkojus pushed 1 commit • aada48f...3de99f1 • 9 days ago	...
Create seed.ts	...
Dan69Dan pushed 1 commit • a1087e3...aade48f • 9 days ago	...
Add initial package.json configuration	...
Refaue pushed 1 commit • d742dd7...a1087e3 • 9 days ago	...
Create schema.prisma	...
Dan69Dan pushed 1 commit • ecf2e53...d742dd7 • 9 days ago	...
Add CI workflow configuration for QueueLess project	...
Refaue pushed 1 commit • c88122b...ecf2e53 • 10 days ago	...

Implementowane widoki aplikacji

Ekran startowy:



Projekt zespołowy systemu informatycznego

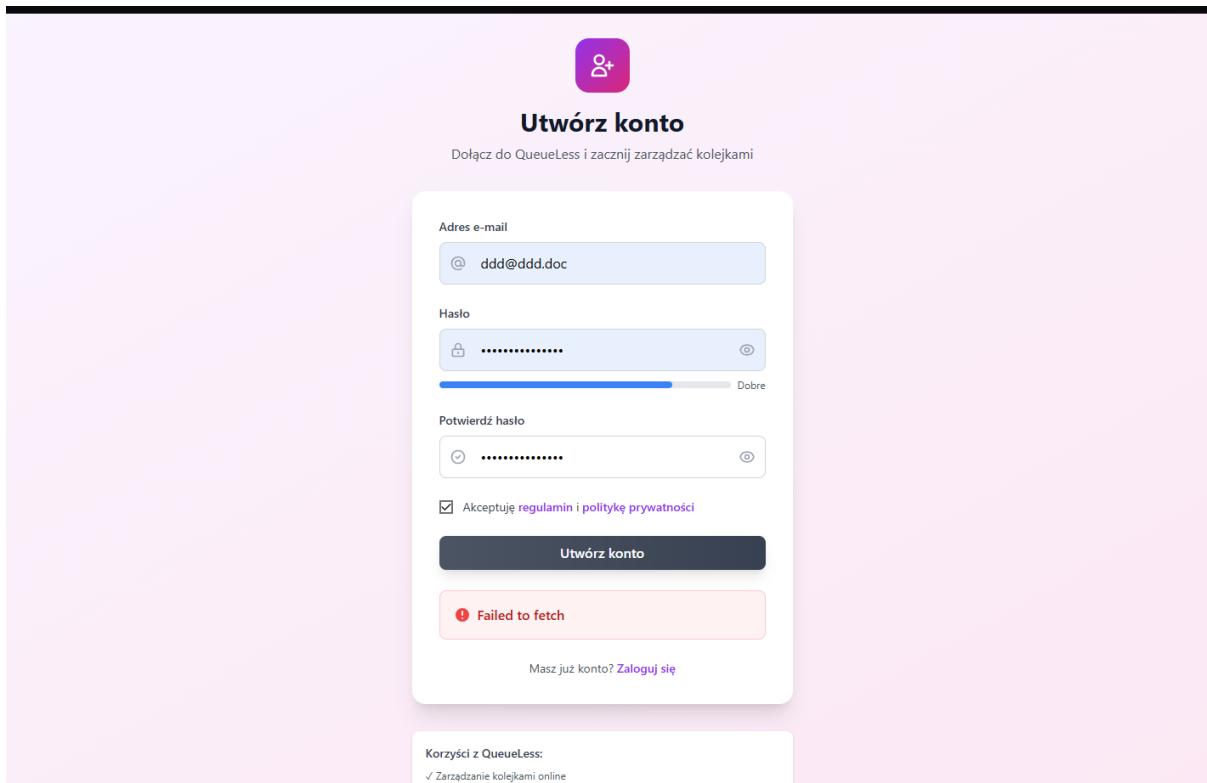
Ekran logowania:

Strona główna | **Logowanie** | Rejestracja

The screenshot shows a light blue-themed login form. At the top center is a purple lock icon. Below it, the text "Zaloguj się" is displayed in bold black font. A sub-instruction "Witaj ponownie! Zaloguj się do swojego konta" follows. The main input fields are labeled "Adres e-mail" and "Hasło". The "Adres e-mail" field contains "np. jan.kowalski@example.com". The "Hasło" field has placeholder text "Wprowadź swoje hasło". Below these fields are two small checkboxes: "Zapamiętaj mnie" and "Zapomniałeś hasła?". A large blue button at the bottom left of the form area contains the text "Zaloguj się". At the very bottom of the page, there is a small note about demo accounts: "Konta demo: Admin: admin@admin.com / admin123".

Projekt zespołowy systemu informatycznego

Ekran Rejestracji:



Kod źródłowy – istotne fragmenty opracowane w bieżącym tygodniu

```
QueueLess > packages > backend > src > routes > TS auth.ts > router.post('/register') callback
12
13   router.post('/register', async (req, res) => {
14     const parsed = registerSchema.safeParse(req.body);
15     if (!parsed.success) return res.status(400).json({ error: 'Invalid input' });
16
17     const { email, password } = parsed.data;
18
19     // Sprawdź czy użytkownik już istnieje
20     const existingUser = db.getUserByEmail(email);
21     if (existingUser) return res.status(409).json({ error: 'Email already registered' });
22
23     // Hashuj hasło i utwórz użytkownika
24     const hash = await bcrypt.hash(password, 10);
25     const user = db.createUser({
26       email,
27       password: hash,
28       role: 'USER',
29       isActive: true
30     });
31
```

Ten fragment kodu obsługuje rejestrację użytkownika w aplikacji. Sprawdza poprawność danych wejściowych, weryfikuje, czy podany e-mail nie jest już zarejestrowany, a następnie

Projekt zespołowy systemu informatycznego

hashuje hasło za pomocą bcrypts i tworzy nowego użytkownika z rolą „USER” i statusem aktywnym.

```
QueueLess > packages > frontend > src > components > ProtectedRoute.tsx > ProtectedRoute

10
11  export default function ProtectedRoute({
12    children,
13    requiredRoles = [],
14    requireAuth = true
15  }: ProtectedRouteProps) {
16    const { user, isLoading } = useAuth();
17
18    if (isLoading) {
19      return (
20        <div className="min-h-screen flex items-center justify-center">
21          <div className="text-center">
22            <div className="brand-spinner mx-auto mb-4"></div>
23            <p className="text-gray-600">Ładowanie...</p>
24          </div>
25        </div>
26      );
27    }
28
29    // Jeśli wymaga uwierzytelnienia, ale użytkownik nie jest zalogowany
30    if (requireAuth && !user) {
31      return <Navigate to="/login" replace />;
32    }
33
34    // Jeśli nie wymaga uwierzytelnienia i użytkownik jest zalogowany, przekieruj do dashboardu
35    if (!requireAuth && user) {
36      return <Navigate to={getDashboardPath(user.role)} replace />;
37    }
38
39    // Jeśli są wymagane konkretne role
40    if (requiredRoles.length > 0 && user && !requiredRoles.includes(user.role)) {
41      return <Navigate to="/unauthorized" replace />;
42    }
43
44    return <>{children}</>;
45  }
46
```

Ten fragment kodu odpowiada za ochronę tras w aplikacji — komponent ProtectedRoute sprawdza, czy użytkownik jest zalogowany i czy ma odpowiednią rolę dostępu. W zależności od wyniku przekierowuje go na stronę logowania, dashboard lub „unauthorized”, a jeśli wszystko jest poprawne, wyświetla żądzany widok.

Projekt zespołowy systemu informatycznego

Raport z drugiego tygodnia realizacji sprintu 1

10.11.2025

Na meetingu 3 skoncentrowaliśmy się na tym, żeby logowanie działało po prostu bez stresu. Zaczęliśmy od przejrzenia ekranu i konsoli, bo część osób widziała komunikat „Not found”. Potem sprawdziliśmy „ręcznie” połaczenie z serwerem i szybko okazało się, że aplikacja woła pod inny adres niż ten, którego oczekuje backend. Ustaliliśmy prostą, konkretną poprawkę: przepisać ścieżkę w ustawieniach frontu tak, żeby trafiała we właściwe miejsce, zrestartować oba serwery i sprawdzić jeszcze raz cały przepływ. Zrobiliśmy krótki test po zmianie, żeby mieć pewność, że sesja się poprawnie zakłada, a użytkownik dostaje czytelną informację. Przy okazji uporządkowaliśmy detale na ekranie logowania: poprawne dane demo, prostsze komunikaty w błędnych przypadkach i sensowne podpowiedzi co wpisać. Ustaliliśmy też, że w najbliższych dniach ruszymy z pierwszymi „widocznymi” elementami — listą firm i szkicem panelu admina — tak, żeby każdy mógł zobaczyć postęp bez zaglądania w technikalia.

14.11.2025

Na meetingu 4 usiedliśmy do podziału ról i zaplanowaliśmy, jak ma wyglądać pierwsza, lekka wersja panelu administratora. Chcieliśmy, żeby była prosta w obsłudze: lista firm, widoczny status firmy (aktywna/wyłączona) i kilka przełączników, które od razu dają poczucie kontroli. Ustaliliśmy też „bezp pieczniki” — żeby nikt nie mógł mączyć logowania w nieskończoność i żeby dane, które wpisujemy, były od razu sensownie sprawdzane. Zdecydowaliśmy, że przygotujemy podstawowy zestaw danych startowych (admin, jedna przykładowa firma, pracownik), dzięki czemu demo będzie od pierwszego dnia wyglądać „jak działa”. Dopecozywaliśmy, co dla nas znaczy „zrobione” w tej części (czyli jasna definicja gotowości), a na koniec ułożyliśmy prosty plan kontroli jakości: czy da się bez problemu wejść, czy sesja trzyma, czy komunikaty są zrozumiałe i czy aplikacja zachowuje się grzecznie w typowych błędach. Na bazie tego zrobiliśmy krótki plan następnego kroku: widok listy firm na froncie, prosty profil firmy z przyciskiem „Zarezerwuj”, oraz pierwszy ekran „kolejki dnia” dla pracownika, gdzie można w prosty sposób zacząć i zakończyć wizytę. Dla użytkownika końcowego chcemy dorzucić przyjazne wiadomości (np. „sprawdź e-mail lub hasło”), kilka drobnych udogodnień na telefonie i szybkie potwierdzenie rezerwacji e-mailem z linkiem do kalendarza, żeby całość czuła się „gotowa do użycia”.

Projekt zespołowy systemu informatycznego

Zadania realizowane przez poszczególnych deweloperów:

Rafał Gola:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Konfiguracja GitHub oraz utworzenie repozytorium i pipeline CI (T5)	2	Ukończone
Konfiguracja systemu do wysyłki wiadomości e-mail – potwierdzenia rezerwacji i ICS (T16)	3	W realizacji

Przemysław Habdas:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Przygotowanie struktury backendu (Express + routery, konfiguracja błędów) (T3)	2	Ukończone
Implementacja middleware autoryzacji w rolach OWNER/WORKER i kontekstu companyId (T19)	3	W realizacji
Middleware Express Rate-Limit (autoryzacja/rezerwacje) (T17)	2	W realizacji

Kamil:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Utworzenie struktury frontendu (Vite + React + Tailwind + routing) (T2)	5	Ukończone
Formularz rezerwacji i logika widoku klienta (T14)	4	W realizacji
Widok „Moje rezerwacje” / Dashboard użytkownika (T18)	3	W realizacji
Panel ustawień firmy – interfejs właściciela (T20)	4	W realizacji

Jakub:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Opracowanie przepływu logowania i rejestracji – formularze i logika sesji (T6)	2	Ukończone
Widok profilu użytkownika – dane + wylogowanie (T7)	2	W realizacji
Przygotowanie panelu administratora – układ i komponenty (T32, backlog Sprintu 2)	4	Zaplanowane

Daniel:

Realizowane zadanie	Tygodniowy czas pracy [h]	Status realizowanego zadania
Implementacja bazy danych SQLite– konfiguracja typów i schematów (T4)	4	W realizacji
Migracje i pliki inicjalizujące strukturę tabel oraz dane przykładowe (T4)	2	Ukończone

Projekt zespołowy systemu informatycznego

Cele osiągnięte w bieżącym tygodniu

- Zainicjalizowaliśmy projekt w architekturze monorepo (frontend + backend).
- Skonfigurowaliśmy środowisko programistyczne oraz repozytorium GitHub z działającym CI.
- Utworzyliśmy i połączylismy bazę danych SQLite.
- Zaimplementowaliśmy system logowania i rejestracji użytkowników (formularze + sesje).
- Przygotowaliśmy wspólne pliki konfiguracyjne TypeScript, ESLint, Prettier.
- Ustaliliśmy zasady pracy zespołowej i strukturę commitów w repozytorium.

Stan tablicy Kanban na dzień 10.11.2025

The Kanban board displays the following tasks:

- OGRÓDNE**: 3 tasks (Formularz rezerwacji, Auto-przydział pracownika, E-mail potwierdzenia rezerwacji)
- TODO**: 4 tasks (Panel Admina (approve/reject), Godziny pracy (per dzień), Algorytm wolnych slotów, Wyszukiwarka firm)
- INPROGRESS**: 3 tasks (RBAC firmy (Owner/Worker), Rate limit POST / reservations, Moje konto + wylogowanie)
- REVIEW**: 6 tasks (Inicjalizacja środowiska, Frontend scaffold, Backend scaffold, Rejestracja i logowanie (sesje), Baza i ORM (Prisma + SQLite), GitHub — konfiguracja repozytorium)
- II SPRINT DONE**: 0 tasks

Stan tablicy Kanban na dzień 14.11.2025

The Kanban board displays the following tasks:

- OGRÓDNE**: 1 task (Auto-przydział pracownika)
- TODO**: 3 tasks (Godziny pracy (per dzień), Algorytm wolnych slotów, Wyszukiwarka firm)
- INPROGRESS**: 6 tasks (Moje konto + wylogowanie, Panel Admina (approve/reject), E-mail potwierdzenia rezerwacji, Rate limit POST / reservations, RBAC firmy (Owner/Worker), Formularz rezerwacji)
- REVIEW**: 0 tasks
- ODDANE**: 6 tasks (Frontend scaffold, Inicjalizacja środowiska, Backend scaffold, Rejestracja i logowanie (sesje), Baza i ORM (SQLite), GitHub — konfiguracja repozytorium)
- II SPRINT DONE**: 0 tasks

Commity z Githuba – stan na dzień 14.11.2025

Projekt zespołowy systemu informatycznego

<p>Fix missing newline at end of UnauthorizedPage.tsx</p> <p> Kubaaa04 pushed 1 commit to main · d69093c...5e82d70 · yesterday</p>	...
<p>Delete packages/frontend/src/pages/UnauthorizedPage</p> <p> Kubaaa04 pushed 1 commit to main · 6d47b56...d69093c · yesterday</p>	...
<p>Create UnauthorizedPage component for access control</p> <p> Kubaaa04 pushed 1 commit to main · a902c4e...6d47b56 · yesterday</p>	...
<p>Update RegisterPage.tsx</p> <p> Kubaaa04 pushed 1 commit to main · f35ece0...a902c4e · yesterday</p>	...
<p>Update demo account credentials on LoginPage</p> <p> Kubaaa04 pushed 1 commit to main · 88f106b...f35ece0 · yesterday</p>	...
<p>Refactor mailer to use Ethereal test account</p> <p> Rafaue pushed 1 commit to main · ff04ce9...88f106b · yesterday</p>	...
<p>Update TypeScript config to include db directory</p> <p> Rafaue pushed 1 commit to main · f47a695...ff04ce9 · yesterday</p>	...
<p>Update package.json for backend configuration</p> <p> Rafaue pushed 1 commit to main · 9abffd1...47a695 · yesterday</p>	...
<p>Add imports for migration and seeding scripts</p> <p> Rafaue pushed 1 commit to main · 4e60522...9abffd1 · yesterday</p>	...
<p>Rename path</p> <p> Dan69Dan pushed 1 commit to main · 28f0105...dc76cd6 · 5 hours ago</p>	...
<p>Create seed.ts</p> <p> Dan69Dan pushed 1 commit to main · be13edc...28f0105 · 5 hours ago</p>	...
<p>Create 001_init.sql</p> <p> Dan69Dan pushed 1 commit to main · 5ca5fd0...be13edc · 5 hours ago</p>	...
<p>Update db.ts</p> <p> Dan69Dan pushed 1 commit to main · 75125bd...5ca5fd0 · 5 hours ago</p>	...
<p>Create migrate.ts</p> <p> Dan69Dan pushed 1 commit to main · b2c3532...75125bd · 5 hours ago</p>	...
<p>Create .gitkeep</p> <p> kkkejus pushed 1 commit to main · 88f33a...b2c3532 · 6 hours ago</p>	...
<p>Implement Worker Dashboard component</p> <p> kkkejus pushed 1 commit to main · 4e78625...88f33a · 6 hours ago</p>	...
<p>Add OwnerDashboard component for managing companies</p> <p> kkkejus pushed 1 commit to main · 860f82d...4e78625 · 6 hours ago</p>	...
<p>Add UserDashboard component for managing reservations</p> <p> kkkejus pushed 1 commit to main · f649639...860f82d · 6 hours ago</p>	...
<p>Add QueuePage component to display queue status</p> <p> kkkejus pushed 1 commit to main · 437a408...f649639 · 6 hours ago</p>	...
<p>Create CompanySettingsPage.tsx</p> <p> kkkejus pushed 1 commit to main · df9f27a...437a408 · 6 hours ago</p>	...
<p>Add ReservationPage component for booking process</p> <p> kkkejus pushed 1 commit to main · f9e4858...df9f27a · 6 hours ago</p>	...
<p>Ensure root elements have full height</p> <p> kkkejus pushed 1 commit to main · 76f9305...f9e4858 · 6 hours ago</p>	...
<p>Fix CSS syntax for body-responsive class</p> <p> kkkejus pushed 1 commit to main · c480537...76f9305 · 6 hours ago</p>	...
<p>Revise homepage text and links for clarity</p> <p> kkkejus pushed 1 commit to main · f81f5b1...c480537 · 6 hours ago</p>	...
<p>Fix formatting in ProtectedRoute component</p> <p> kkkejus pushed 1 commit to main · fc19fc3...f81f5b1 · 6 hours ago</p>	...
<p>Fix formatting in Logo.tsx component</p> <p> kkkejus pushed 1 commit to main · cb73695...fc19fc3 · 6 hours ago</p>	...
<p>Add protected routes for reservation and dashboards</p> <p> kkkejus pushed 1 commit to main · 671d571...cb73695 · 6 hours ago</p>	...

Projekt zespołowy systemu informatycznego

Ensure root elements have full height	...
Fix CSS syntax for body-responsive class	...
Revise homepage text and links for clarity	...
Fix formatting in ProtectedRoute component	...
Fix formatting in Logo.tsx component	...
Add protected routes for reservation and dashboards	...
Update user role handling in dashboard path	...
Fix formatting in main.tsx	...
Fix missing newline at end of index.html	...
Refactor reservations route with Zod and direct DB access	...
Implement rate limiting middleware	...
Refactor authentication routes with database queries	...
Enhance company routes with timezone and logo upload	...
Add reservations routes to the backend	...
Add user management and loading functionality	...
Handle error response in API calls	...
Fix missing newline at end of AuthContext.tsx	...
Fix missing newline at end of AuthContext.tsx	...
Fix missing newline at end of AuthContext.tsx	...

Projekt zespołowy systemu informatycznego

Implementowane widoki aplikacji

Strona główna:

The screenshot shows the QueueLess homepage. At the top, there's a dark blue header with the QueueLess logo, navigation links for "Strona główna", "Logowanie", and "Rejestracja", and a link to "Platforma nowej generacji". Below the header is a large dark blue banner with the text "QueueLess" and "Koniec z kolejkami". It claims to be a modern platform for managing reservations and queues for businesses of all sizes, increasing efficiency by reducing client wait times through intelligent solutions. Key statistics are displayed: "10,000+" satisfied clients, "500+" companies using it, and a "95% reduction in waiting time". At the bottom of the banner, there's a section titled "Jak to działa?" (How does it work?) with a small explanatory text and a link to "Dochód z naszych kredytów".

Zakładanie konta użytkownika:

The screenshot shows the "Utwórz konto" (Create account) page. It features a purple header with a user icon and the text "Utwórz konto". Below this, a sub-header reads "Dobierz do QueueLess i zacznij zarządzać kolejkami". The main form includes fields for "Adres e-mail" (Email address) with placeholder "np. jan.kowalski@example.com", "Hasło" (Password) with placeholder "Wprowadź bezpieczne hasło", "Potwierdź hasło" (Confirm password) with placeholder "Powtórz hasło", and a checkbox for "Akceptuję regulamin i politykę prywatności" (I accept the Terms and Privacy Policy). A "Utwórz konto" (Create account) button is at the bottom. A note below the form says "Masz już konto? Zaloguj się" (Already have an account? Log in).

Strona rezerwacji wizyty:

The screenshot shows the "Zarezerwuj wizytę" (Book an appointment) page. At the top, there's a sub-header "Wybierz firmę" (Select company) with a note "Znajdź firmę, w której chcesz zarezerwować wizytę". Below this, a step-by-step navigation bar shows "1 Wybierz firmę", "2 Szczegóły", and "3 Potwierdzenie". The "Wybierz firmę" step displays two company options: "Demo Salon" (Europe/Warsaw) and "Test Company" (Europe/Warsaw). The "Szczegóły" and "Potwierdzenie" steps are currently hidden or not visible.

Projekt zespołowy systemu informatycznego

Panel Administratora (Dla klienta prowadzącego firmę):

The screenshot shows the Queueless Admin Panel interface. At the top, there are navigation links: Strona główna, Rezerwacja, Panel (highlighted in blue), Witaj_admin@queueless.local, Admin, and Wyloguj. Below the header, a banner reads "Panel administratora" and "Witaj, admin@queueless.local Zarządzaj firmami i użytkownikami platformy Queueless.". There are four summary boxes: "Firmy" (2), "Użytkownicy" (0), "Aktywne Firmy" (0), and "Aktywni użytkownicy" (0). A section titled "Dodaj nową firmę" contains fields for "Nazwa firmy" (e.g., "np. Salon fryzjerski") and "Kategoria (opcjonalnie)" (e.g., "np. Salon fryzjerski"). A "Dodaj firmę" button is present. Below this is a table titled "Zarządzanie firmami" with columns: NAZWA, KATEGORIA, STATUS, and DATA UTWORZENIA. It lists two entries: "Demo Salon" (Status: Nieaktywne, Data: Invalid Date) and "Test Company" (Status: Nieaktywne, Data: Invalid Date). A note below the table states: "Brak danych o użytkownikach (endpoint może wymagać implementacji)". The bottom section is titled "Zarządzanie użytkownikami".

Panel logowania:

The screenshot shows the Queueless Login page. At the top, there are navigation links: Strona główna, Logowanie (highlighted in blue), and Rejestracja. The main area features a "Zaloguj się" button with a lock icon. Below it, a message says "Witaj ponownie! Zaloguj się do swojego konta". The login form has fields for "Adres e-mail" (e.g., np. jan.kowalski@example.com) and "Hasło" (with placeholder "Wprowadź swoje hasło"). There are checkboxes for "Zapamiętaj mnie" and "Zapomniane hasła?". A "Zaloguj się" button is at the bottom. A note below the form says "Nie masz jeszcze konta? Zarejestruj się". At the bottom, there is a "Konta demo:" section with the text: "Admin: admin@queueless.local / admin123" and "Widok: owner@queueless.local / owner123".

Projekt zespołowy systemu informatycznego

Kod źródłowy – istotne fragmenty opracowane w bieżącym tygodniu

```
packages > backend > src > middleware > ts rateLimit.ts > ...
1 import { Request, Response, NextFunction } from 'express';
2
3 type Keyer = (req: Request) => string;
4
5 interface Rule {
6   windowMs: number;
7   max: number;
8   keyer: Keyer;
9 }
10
11 const store = new Map<string, { count: number; resetAt: number }>();
12
13 export function rateLimit(rule: Rule) {
14   return (req: Request, res: Response, next: NextFunction) => {
15     const key = rule.keyer(req);
16     const now = Date.now();
17     const entry = store.get(key);
18     if (!entry || now > entry.resetAt) {
19       store.set(key, { count: 1, resetAt: now + rule.windowMs });
20       return next();
21     }
22     if (entry.count >= rule.max) {
23       return res.status(429).json({ error: 'Too many requests' });
24     }
25     entry.count += 1;
26     next();
27   };
}
```

Kod dostarcza funkcję getTransport() tworzącą transporter SMTP przez nodemailer .
W trybie dev automatycznie zakłada konto Ethereal i zwraca gotowy transporter (host smtp.ethereal.email , port 587 , secure: false).

W bloku catch zwracany jest „no-op” transport z metodą sendMail , który nic nie robi, dzięki czemu wywołania mailera nie wywalają aplikacji bez sieci/SMTP.

Taki fallback ułatwia lokalny rozwój i testy/CI — kod wysyłki można wołać bez warunków na środowisko.

Projekt zespołowy systemu informatycznego

```
packages > backend > src > lib > TS mailer.ts > ⚡ getTransport

1 import nodemailer from 'nodemailer';
2
3 export async function getTransport() {
4     try {
5         // W dev korzystamy z konta testowego Ethereal
6         const testAccount = await nodemailer.createTestAccount();
7         const transporter = nodemailer.createTransport({
8             host: 'smtp.ethereal.email',
9             port: 587,
10            secure: false,
11            auth: {
12                user: testAccount.user,
13                pass: testAccount.pass,
14            },
15        });
16        return transporter;
17    } catch (e) {
18        // Brak sieci → zwróć obiekt transportu, który nic nie robi
19        return {
20            sendMail: async () => ({})
21        } as any;
22    }
}
```

Kod implementuje prosty, generyczny rate-limiter: funkcja rateLimit(rule) przyjmuje okno czasowe (windowMs), limit (max) i funkcję klucza (keyer). Stan trzymany jest w Map , resetowany po wygaśnięciu okna; przy przekroczeniu zwracany jest HTTP 429.

Gotowe reguły:

loginRateLimit : 5 prób logowania na minutę per IP.

reservationRateLimit : 1 rezerwacja na 30 s per IP+email (lowercase).