BAZY DANYCH - PROJEKT

DOKUMENTACJA DO ETAPU 2

Kowalczyk Emilia 272539

Julia Rojek 272529

# Wybrane technologie

## 1.2 Backend

W backendzie zastosowano szereg technologii, które wspierają efektywne przetwarzanie danych, bezpieczeństwo oraz łatwą integrację z frontendem. Główne technologie użyte w tej części projektu to:

1. **Node.js** - środowisko uruchomieniowe JavaScript, które pozwala na realizację aplikacji serwerowych przy użyciu języka JavaScript. Dzięki swojej asynchroniczności i architekturze opartej na zdarzeniach, Node.js jest wydajny i umożliwia obsługę wielu jednoczesnych połączeń, co jest kluczowe dla skalowalności aplikacji.
2. **Express.js** - framework aplikacji webowych oparty na Node.js, który upraszcza tworzenie interfejsów API i zarządzanie trasami w aplikacji. Express.js pozwala na szybkie i elastyczne konfigurowanie endpointów, obsługę żądań HTTP oraz middleware, co ułatwia implementację.
3. **Multer** - middleware do obsługi przesyłania plików w aplikacji Express. Używany jest głównie do przesyłania obrazów i ich walidacji w backendzie (np. weryfikacji typów plików oraz obsługi błędów podczas przesyłania). W projekcie multer jest wykorzystywany do obsługi zdjęć użytkowników i domków, dzięki czemu pozwala na łatwe i bezpieczne przetwarzanie danych multimedialnych.
4. **Body-parser** - biblioteka umożliwiająca parsowanie treści żądań HTTP do formatu JSON, co jest niezbędne do poprawnej interpretacji danych wysyłanych przez klienta (frontend) w żądaniach POST i PUT.
5. **CORS** (Cross-Origin Resource Sharing) - middleware pozwalający na dostęp do API z różnych domen i portów, co jest przydatne w przypadku, gdy frontend i backend aplikacji są hostowane oddzielnie (np. na różnych portach w lokalnym środowisku). CORS w projekcie pozwala frontendowi na wykonywanie żądań do backendu, zachowując odpowiednie poziomy bezpieczeństwa.
6. **MySQL** - relacyjna baza danych użyta do przechowywania informacji o użytkownikach, rezerwacjach, domkach oraz innych istotnych danych. Zastosowanie relacyjnej bazy danych pozwala na wydajne zarządzanie i przetwarzanie danych z użyciem języka SQL.
7. **MySQL2** - biblioteka pozwalająca na połączenie Node.js z bazą danych MySQL. W projekcie biblioteka ta odpowiada za nawiązywanie połączenia z bazą, wykonywanie zapytań SQL oraz zwracanie wyników do aplikacji. Wersja MySQL2 jest zoptymalizowana pod kątem wydajności oraz lepszego wsparcia dla operacji asynchronicznych.
8. **Dotenv** - biblioteka do zarządzania zmiennymi środowiskowymi. Umożliwia przechowywanie poufnych danych (takich jak dane dostępowe do bazy danych) w pliku .env, co pozwala na bezpieczniejsze zarządzanie konfiguracją bez ujawniania danych w kodzie źródłowym.
9. **Path** - moduł wbudowany w Node.js, używany do zarządzania ścieżkami dostępu do plików i katalogów. Path jest wykorzystywany do serwowania statycznych zasobów, takich jak obrazy czy pliki, w sposób niezależny od systemu operacyjnego.
10. **Bcrypt** – biblioteka przeznaczona do bezpiecznego haszowania haseł. Bcrypt umożliwia szyfrowanie haseł użytkowników przed zapisaniem ich do bazy danych, co znacząco zwiększa bezpieczeństwo projektu i chroni dane użytkowników przed potencjalnym przechwyceniem.

## 1.3 Frontend

W projekcie frontend został zaprojektowany z wykorzystaniem nowoczesnych technologii i narzędzi, które wspierają dynamiczne działanie aplikacji oraz zapewniają czytelny i intuicyjny interfejs użytkownika. Poniżej znajdują się technologie wykorzystane w warstwie frontendowej:

1. **React** - jest to biblioteka JavaScript, która umożliwia tworzenie komponentowych i dynamicznych aplikacji internetowych. Wykorzystuje **Virtual DOM** do efektywnego renderowania interfejsu użytkownika, co znacząco przyspiesza proces aktualizacji elementów strony bez konieczności przeładowywania całej aplikacji.
2. **React Router** - pozwala na nawigację pomiędzy różnymi widokami aplikacji bez przeładowania strony, co jest istotne w aplikacjach typu SPA (Single Page Application). W projekcie zastosowano na przykład komponenty takie jak Private\_client\_route oraz Private\_owner\_route, aby kontrolować dostęp do stron zarezerwowanych tylko dla klientów lub właścicieli, bazując na sesji użytkownika.
3. **Axios** - służy do wykonywania zapytań HTTP do serwera. Jest ona używana, na przykład, w komponentach Waiting oraz Confirmed, do komunikacji z backendem i pobierania danych o rezerwacjach. axios umożliwia łatwe przetwarzanie odpowiedzi serwera oraz obsługę błędów, co zapewnia spójność komunikacji między frontendem a backendem.
4. **CSS i Sass** - stylowanie zostało zrealizowane z wykorzystaniem CSS, wspomaganego przez preprocesor Sass. Sass umożliwia bardziej modularne podejście do stylowania, dzięki wsparciu dla zmiennych, zagnieżdżania selektorów oraz dziedziczenia stylów.
5. **React-datpicker** - pozwala na wygodne wybieranie zakresu dat, co jest widoczne w komponencie np. Search\_form. Komponent umożliwia użytkownikom wybór dat w wygodny sposób oraz obsługę kalendarza, co podnosi użyteczność aplikacji.

# encje itp

# transakcje itp.

# 4 progznoza blabla