Chat app

Приложение, в което трябва да се регистрираш, за да имаш право да си говориш с останалите участници. След регистрацията можеш да пишеш на който и да е от участниците.

Използвани технологии

Frontend

**React.js** – за изграждане на интерфейса

**React Router** – за навигация между страниците

**Axios** – за подаване на заявки към backend-а

Backend

**Node.js / Express.js** – за приемане и изпълнение на заявките

**JWT (JSON Web Tokens)** – за заверка и контрол на сесиите

**bcrypt** – за сигурно криптиране на паролите

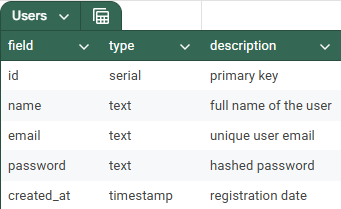
**CORS** – за осигуряването на безопасни заявки от frontend-а

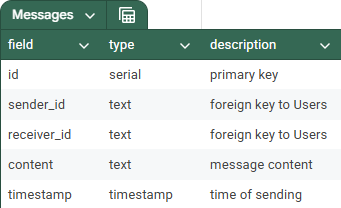
Database

**PostgreSQL** – за записване на всичко в базата данни

**pg** (Node.js PostgreSQL client) – за управление на базата данни

Database design





Deployment

Проектът е деплойнат с помощта на Terraform.

## CI/CD & Code Quality

## **ESLint** за подобряване на кода.

## **Prettier** за по-добро форматиране.

## **Unit тестове с** Jest за контролерите.

User stories

Sign Up / Log in

Като потребител искам да мога да се регистрирам и да си пиша с всички други регистрирани потребители.

Chat

Като потребител искам да мога да си пиша с всички и чатовете ми да бъдат запазени.

Use cases

### Use Case 1: Регистрация на потребител

**Title**: Регистрация на потребител

**Primary Actor**: Новият потребител

**Goal**: Създаване на профил

**Preconditions**: Потребителят не е регистриран

**Trigger**: Потребителят използва функционалността "Sign Up"

**Main Success Scenario**:

Потребителят въвежда данните.

Системата валидира данните.

Системата ги запазва и хешира паролата.

Системата връща потвърждение и token.

**Postconditions**: Потребителят има създаден профил, в който влиза с регистрацията.

**Alternative Paths**:

Показва грешка, ако вече съществува потребител с такъв имейл.

Ако някое от полетата е празно или грешно, се показва грешка да се попълни или поправи.

### Use Case 2: Влизане в профил

**Title**: Влизане в профил

**Primary Actor**: Регистриран потребител

**Goal**: Потребителят да си влезе в профила

**Preconditions**: Потребителят е регистриран

**Trigger**: Потребителят използва функционалността "Log In"

**Main Success Scenario**:

Потребителят въвежда името и паролата.

Системата проверява данните.

Системата връща валиден JWT token.

Потребителят е отведен на страницата с чатовете си.

**Postconditions**: Потребителят е в профила си.

**Alternative Paths**:

Грешни данни → съобщение за грешка се извежда на екрана

### Use Case 3: Изпращане на съобщение

**Title**: Изпращане на съобщение

**Primary Actor**: Регистриран потребител, влязъл в профила си

**Goal**: Изпращане на съобщение до друг потребител

**Preconditions**: Потребителят е в профила си

**Trigger**: Потребителят изпраща съобщение

**Main Success Scenario**:

Потребителят написва съобщение и натиска „изпращане“

Системата запазва съобщението в базата данни

Системата обновява сайта и вече се вижда новото съобщение

**Postconditions**: Съобщението е видимо в чата

**Alternative Paths**:

Съобщението е празно → извежда се съобщение за грешка

Грешка в базата данни → извежда се съобщение за грешка

### Use Case 4: Преглед на чата

**Title**: Преглед на чата

**Primary Actor**: Регистриран потребител, влязъл в профила си

**Goal**: Да мога да преглеждам стари чатове

**Preconditions**: Има съобщения с даден потребител в базата данни

**Trigger**: Потребителят отваря чата

**Main Success Scenario**:

Потребителят избира друг потребител

Системата показва всички съобщения между двамата потребители в хронологичен ред

* **Postconditions**: Потребителят вижда всички съобщения
* **Alternative Paths**:

Няма съобщения → страницата е празна

Проблем с базата данни → извежда се съобщение за грешка

### Use Case 5: Излизане от профила

**Title**: Излизане от профила

**Primary Actor**: Регистриран потребител, влязъл в профила си

**Goal**: Излизане от профила в приложението

**Preconditions**: Потребителят е в профила си

**Trigger**: Потребителят използва функционалността "Log Out"

**Main Success Scenario**:

JWT token-а бива изтрит

Потребителят е отведен на страницата за влизане в профила

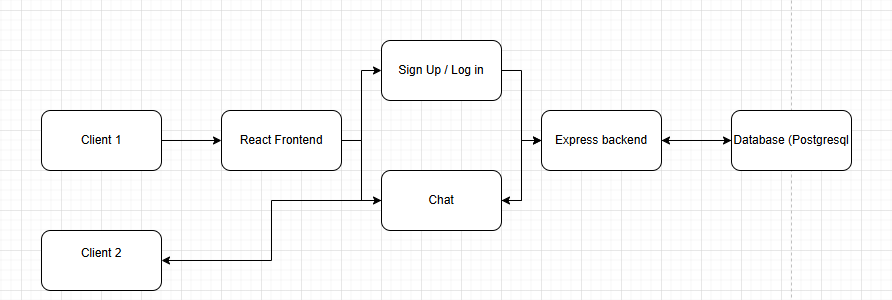
**Postconditions**: Край на сесията на потребителя

**Alternative Paths**:

грешка

Project architecture

Реших да използвам архитектурата Client-Server, защото има добро scalability и изисква намалена поддръжка.

Design Patterns

Използвам Facade във client частта и Singleton в частта с базата данни.

User stories

#### Scenario 1: Регистрация на потребител

**Title**: Регистрация

**Role**: Нов потребител

**Goal**: Регистрация с имейл, име и парола

**Reason**: Да мога да използвам приложението

**Acceptance Criteria**:

Потребителят попълва всички полета

Получава token

#### Scenario 2: Влизане в профила

**Title**: Влизане в профила

**Role**: Регистриран потребител

**Goal**: Влизане с име и парола

**Reason**: Да мога да влизам в чатовете си

**Acceptance Criteria**:

Правилни име и парола

#### Scenario 3: Изпращане на съобщение

**Title**: Изпращане на съобщение

**Role**: Регистриран потребител

**Goal**: Изпращане на съобщение

**Reason**: Да пиша съобщения

**Acceptance Criteria**:

Съобщението се запазва в базата данни

#### Scenario 4: Преглед на съобщенията

**Title**: Преглед на съобщенията

**Role**: Регистриран потребител

**Goal**: Да видя стари съобщения

**Reason**: Да продължа разговора

**Acceptance Criteria**:

Съобщенията се извеждат по реда на изпращане

Показват се само съобщенията между двамата потребители в чата

#### Scenario 5: Излизане от профила

**Title**: Излизане от профила

**Role**: Регистриран потребител

**Goal**: Излизане от профила

**Reason**: Да не може да се ползва от други потребители на същото устройство

**Acceptance Criteria**:

JWT token-а се премахва

Потребителят бива отведен на началния екран за влизане в профила