# Messenger - dokumentacja

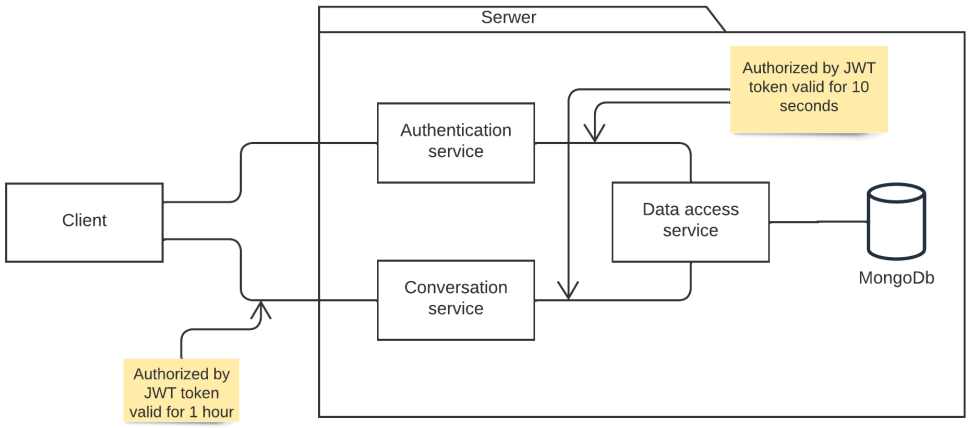


## 1. Opis systemu

Aplikacja ma funkcjonalność prostego komunikatora, w którym można wymieniać się wiadomościami z innymi zarejestrowanymi użytkownikami. Możliwe do wykonania operacje to: logowanie, rejestracja, wyświetlenie utworzonych grup, wyświetlenie użytkowników, stworzenie grup oraz wysyłanie wiadomości. Wszystkie te operacje z wyjątkiem logowania i rejestracji są odpowiednio zabezpieczone i niedostępne z dla zwykłego niezarejestrowanego użytkownika.

## 2. Architektura

Przyjęto architekturę mikro serwisową z podziałem na 3 współpracujące mikroserwisy. Są nimi: mikroserwis dostępu do danych, mikroserwis autoryzacyjny oraz mikroserwis służący do obsługi komunikacji z innymi użytkownikami. Poniższy diagram przedstawia połączenia oraz zależności pomiędzy serwisami.




Jako silnik bazodanowy zostało wykorzystane MongoDB. Aplikacja została skonkretyzowana i można ją uruchomić za pomocą jednego polecenia docker.

3. API

W aplikacji dostępnych jest wiele endpointów. Poniżej znajdują się ich proste opisy.

- Serwis autoryzacyjny
    - **/api/v1/auth/authenticate** - endpoint służący do autoryzacji. Wymaga w body JSON'a z loginem oraz hasłem
    - **/api/v1/auth/valid** - endpoint służący do sprawdzenia ważności tokenu. Wymaga w sekcji body JSON'a z tokenem
    - **/api/v1/auth/register** - endpoint służący do rejestracji. Wymaga w sekcji body JSON'a z loginem, hasłem oraz emailem.

- Serwis obsługujący komunikację
    - **/api/v1/conv/users/all** - endpoint zwraca listę loginów wszystkich użytkowników. Wymaga bearer token do autoryzacji.
    - **/api/v1/conv/group/all** - endpoint zwraca listę nazw wszystkich grup. Wymaga bearer token do autoryzacji.
    - **/api/v1/conv/group/create** - endpoint tworzy nową grupę. Wymaga sekcji body w której będzie nazwa grupy oraz lista loginów osób do niej należących. Wymaga bearer token do autoryzacji.
    - **/api/v1/conv/group/id** - endpoint zwraca id grupy której nazwę przesyłamy jako parametr 'name'. Wymaga bearer token do autoryzacji.
    - **/api/v1/conv/group/messages** - endpoint zwraca listę wiadomości grupy której id przesłano jako parametr 'id'. Pojedyncza wiadomość składa się z jej treści, loginu autora oraz czasu wysłania. Wymaga bearer token do autoryzacji.
    - **/api/v1/conv/message** - endpoint wysyła wiadomość w grupie. Wymaga body w postaci JSON'a który posiada treść wiadomości oraz id grupy. Wymaga bearer token do autoryzacji.

- Serwis dostępu do bazy danych
    - **/api/v1/data/auth/login** - endpoint służący do logowania serwisu. Zwraca token JWT ważny przez 10 sekund. Wymaga sekcji body w której będzie login i hasło serwisu.
    - **/api/v1/data/auth/prolong** - endpoint służący do przedłużenia czasu ważności tokenu. Wymaga sekcji body w której będzie ważny token JWT. Zwraca nowy token JWT ważny przez 10 sekund
    - **/api/v1/data/users/all** - endpoint służący do pobrania listy wszystkich użytkowników dostępnych w bazie. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/users/all-logins** - endpoint służący do pobrania listy loginów wszystkich użytkowników dostępnych w bazie. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/user/get** - endpoint służący do pobrania użytkownika. Wymaga sekcji body w postaci JSON'a z loginem użytkownika. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/user/add** - endpoint służący do dodania użytkownika. Wymaga sekcji body w postaci JSON'a z loginem, hasłem oraz emailem użytkownika. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/user/delete** - endpoint służący do usuwania użytkownika. Wymaga sekcji body w postaci JSON'a z loginem użytkownika. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/group/all** - endpoint służący do pobrania listy wszystkich grup dostępnych w bazie. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/group/all-names** - endpoint służący do pobrania listy nazw wszystkich grup dostępnych w bazie. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/group/add** - endpoint służący do dodawania grupy do bazy danych. Wymaga JSON'a z utworzoną grupą. Wymaga bearer token do autoryzacji.
    - **/api/v1/data/group/update** - endpoint służący do aktualizacji grupy w bazie danych. Wymaga JSON'a z zaktualizowaną grupą. Wymaga bearer token do autoryzacji.

## 4. Bezpieczeństwo

- Komunikacja klient - serwis
  Bezpieczeństwo aplikacji gwarantuje zabezpieczenie komunikacji z serwerem przez token JWT. Jest to skuteczny sposób zabezpieczania komunikacji. Każde zapytanie HTTP, które wymaga autoryzacji musi otrzymać ważny token JWT w przeciwnym razie zwrócony zostanie odpowiedni status HTTP informujący o braku autoryzacji.

- Komunikacja serwis-serwis
  Każde zapytanie pomiędzy serwisami jest również autoryzowane przez token JWT. Jednak w celu zwiększenia bezpieczeństwa token służący do autoryzacji komunikacji pomiędzy serwisami ważny jest tylko przez 10 sekund. Po tym czasie musi być on wymieniony na nowy.

## 5. Uruchamianie

Konfiguracja wstępna.

1. W głównym katalogu projektu (Messenger) należy wykonać polecenie “mvn install”
2. Następnie należy utworzyć obrazy dla docker
   a. W katalogu AuthenticationService należy uruchomić następujące polecenie “docker build -t auth:latest .”
   b. W katalogu ConversationService należy uruchomić następujące polecenie “docker build -t conversation:latest .”
   c. W katalogu DataAccess należy uruchomić następujące polecenie “docker build -t data-access:latest .”
3. W głównym katalogu (Messenger) należy wykonać następujące polecenie “docker compose up”

Przejście wstępnej konfiguracji utworzy odpowiednie obrazy oraz kontenery w docker. Po jej zakończeniu wszystkie usługi będą dostępne oraz skonfigurowane. Skonfigurowane kontenery można podejrzeć w aplikacji desktopowej docker.

Uruchomienie klienta. Należy przejść do katalogu Client/target i wykonać w nim polecenie “java -jar client.jar”. Utworzy to klienta, który połączy się do dostępnych serwisów.