

## **Załącznik 6. Diagramy architektury oraz przepływu wiadomości w systemie p2p-chat**

### **1. Architektura systemu**

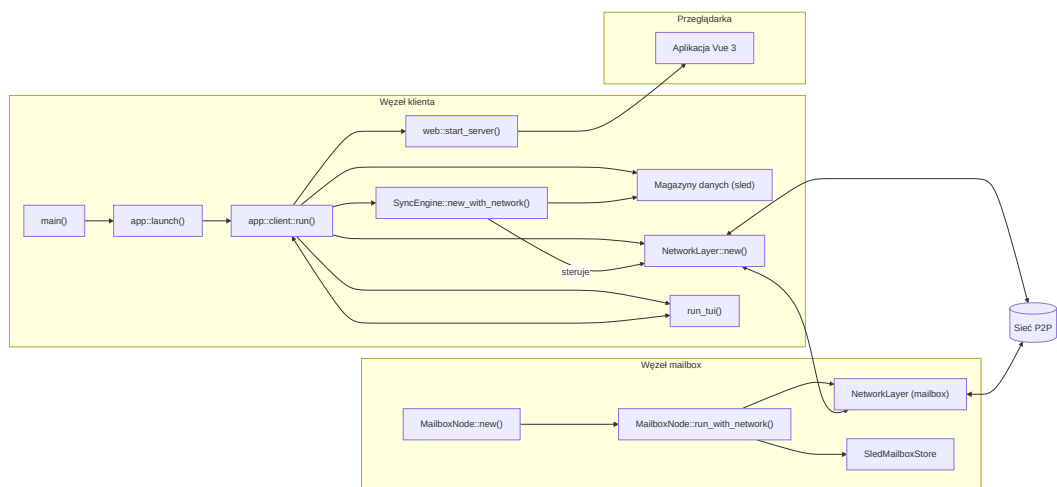
Na rysunku 1 przedstawiono ogólną architekturę systemu p2p-chat. Pokazano powiązania pomiędzy warstwą aplikacyjną (`app::client::run()`), warstwą sieciową (`NetworkLayer`), silnikiem synchronizacji (`SyncEngine`), magazynami danych opartymi na bazie `sled`, interfejsem tekstowym (`run_tui()`) oraz wbudowanym serwerem webowym (`web::start_server()`). Dodatkowo uwzględniono odrębny węzeł `MailboxNode`, odpowiedzialny za przechowywanie wiadomości dla nieobecnych użytkowników.

### **2. Przepływ wiadomości między klientami**

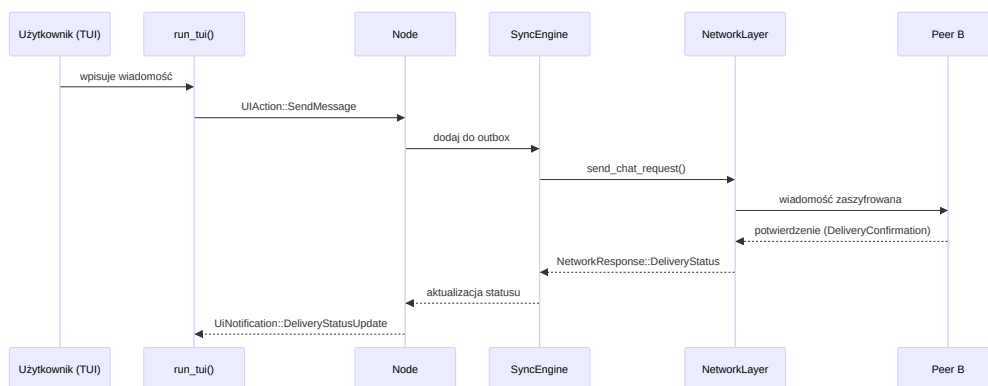
Rysunek 2 przedstawia uproszczony przebieg wysyłania pojedynczej wiadomości pomiędzy dwoma klientami, gdy odbiorca jest dostępny online. Użytkownik w interfejsie tekstowym (`run_tui()`) inicjuje akcję wysłania wiadomości, która jest mapowana na `UIAction::SendMessage` i przekazywana do struktury `Node`. Następnie wiadomość trafia do silnika synchronizacji (`SyncEngine`), który zapisuje ją w *outboxie* i zleca warstwie sieciowej (`NetworkLayer`) wywołanie metody `send_chat_request()` skierowanej do docelowego *PeerId*. Odbiorca przesyła zwrotne potwierdzenie dostarczenia (`DeliveryConfirmation`), które jest propagowane poprzez `NetworkLayer` i `SyncEngine` z powrotem do `Node`. Końcowym efektem jest powiadomienie interfejsu użytkownika o zmianie statusu dostarczenia w postaci zdarzenia `DeliveryStatusUpdate`.

### **3. Przepływ wiadomości z wykorzystaniem węzła MailboxNode**

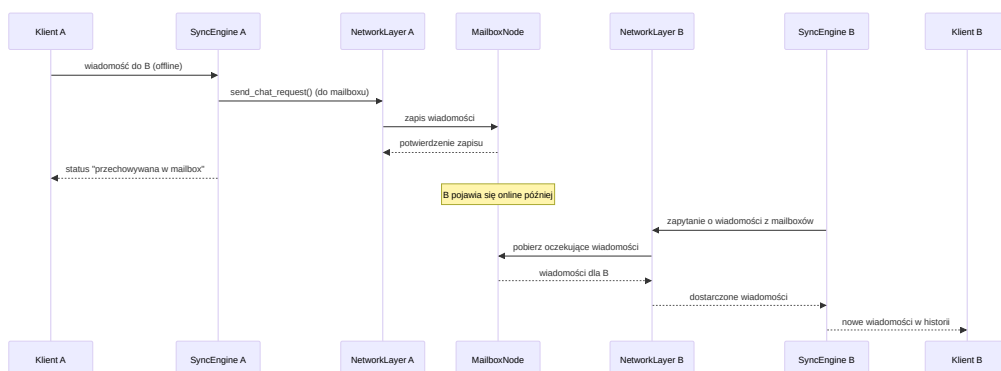
Na rysunku 3 pokazano scenariusz, w którym odbiorca jest chwilowo niedostępny, a wiadomość jest przekazywana do węzła MailboxNode. Silnik synchronizacji po stronie nadawcy identyfikuje odpowiedni węzeł mailbox (na podstawie DHT) i wysyła żądanie poprzez NetworkLayer do MailboxNode, który zapisuje wiadomość w lokalnym SledMailboxStore. W momencie ponownego pojawienia się odbiorcy jego SyncEngine inicjuje pobranie oczekujących wiadomości z mailboxów, a następnie zapisuje je w lokalnej historii, aktualizując interfejs użytkownika.



Rysunek 1: Architektura systemu p2p-chat obejmująca węzeł klienta, węzeł MailboxNode, warstwę sieciową NetworkLayer oraz interfejs webowy osadzony w przeglądarce.



Rysunek 2: Przepływ wiadomości pomiędzy dwoma klientami online z wykorzystaniem SyncEngine, NetworkLayer oraz powiadomień UiNotification.



Rysunek 3: Przepływ wiadomości z wykorzystaniem węzła MailboxNode w sytuacji, gdy odbiorca jest czasowo nieobecny w sieci.