Dokumentacja projektowa
Aplikacja do zarządzania zadaniami

Artur Ziemba
Krzysztof Kołodziejczak

1. Specyfikacja problemu

Potrzebny jest nowoczesny system do zarządzania tablicami zadań, który będzie posiadał autoryzowany dostęp dla użytkowników systemu. Został zauważony problem braku prostej w obsłudze, intuicyjnej aplikacji pozwalającej we współpracy z innymi użytkownikami na zarządzanie zarówno podziałem jak i procesem realizacji wyszczególnionych zadań. W celu reakcji na zapotrzebowanie zostanie zaprojektowany system informatyczny, pracujący w technologii Node.js, ze względu na możliwą do osiągnięcia niezawodność oraz dostępność nawet przy jednoczesnej obsłudze wielu korzystających z aplikacji klientów. Nie powinien on wymagać żadnych procesów wdrożeniowych w celu zrozumienia obsługi narzędzia, ponieważ celem projektu jest stworzenie narzędzia, które wspiera, a nie dodatkowo komplikuje określone zadania. System powinien w przejrzysty sposób prezentować proces realizacji poszczególnych zadań zorganizowanych w tablicach. Właściciel określonej tablicy powinien mieć możliwość zarządzania dostępem do tablicy poprzez udostępnianie lub ograniczanie jej treści pozostałym użytkownikom. Aplikacja zapewni subskrybentom nieprzerwaną możliwość nadzoru, wglądu oraz określania aktualnych statusów dowolnych procesów.

2. Wymagania funkcjonalne

Analiza wymagań funkcjonalnych umożliwia zidentyfikowanie i opisanie pożądanego zachowania systemu, określenie usług oferowanych przez system, reakcji na dane wejściowe oraz zachowania w określonych sytuacjach. Zdefiniowane zostały następujące wymagania funkcjonalne:

- Wymiana komunikatów w modelu klient-serwer
- Możliwość utworzenia do 1.000.000 kont użytkowników
- Możliwość zakładania konta i logowania
- Możliwość zmiany danych użytkownika
- Weryfikacja adresu email przed aktywacją
- Możliwość resetowania hasła za pomocą adresu email
- Utrzymanie do 100 tablic jednocześnie dla użytkownika
- Obsługa do 1000 równoległych członków jednej tablicy
- Możliwość zarządzania członkami tablicy
- Obsługa do 1000 równoległych zadań dla tablicy
- Możliwość zarządzania zadaniami
- Utrzymanie do 1000 statusów do jednego zadania
- Zarządzanie statusami zadań
- Walidacja poprawności wprowadzanych przez użytkownika danych zarówno po stronie klienta jak i serwera
- W razie wystąpienia nieprawidłowych danych system powiadamia o błędach
- Zapewnienie okna pomocy opisującego korzystanie z aplikacji
- Możliwość otrzymywania powiadomień w postaci wiadomości email
- System współpracuje z bazą danych

3. Wymagania pozafunkcjonalne

Wymagania niefunkcjonalne opisują kryteria umożliwiające dokonanie oceny działania systemu i elementów mających wpływ na satysfakcję użytkownika. Zdefiniowane zostały następujące wymagania niefunkcjonalne:

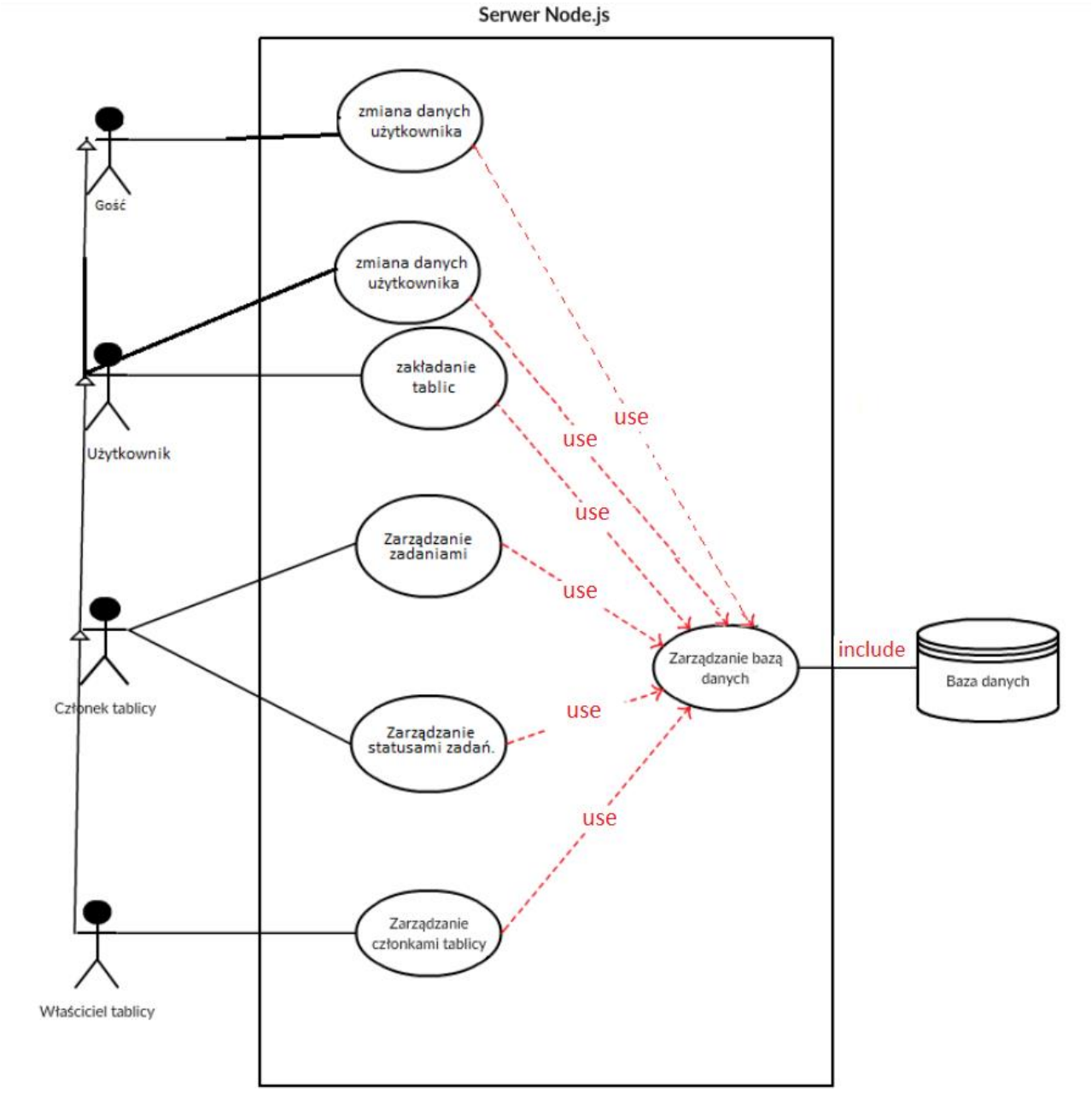
- Budowa interfejsu użytkownika musi być możliwie intuicyjna i prosta w obsłudze
- Dostęp do określonych zasobów musi być chroniony poprzez login i hasło użytkownika
- Proste analizowanie i diagnozowanie błędów oraz sytuacji problemowych
- Budowa zapewnia łatwe wprowadzanie koniecznych zmian
- Jest w stanie obsłużyć jednocześnie do 100.000 użytkowników
- Czas uruchomienia jest nie dłuższy niż 10 sekund
- Umożliwia efektywne testowanie wprowadzonych zmian
- Posiada możliwość obsługi poprzez różne przeglądarki internetowe i środowiska
- Zrozumiały dla wszystkich. Czas przyswojenia nie przekracza 15 minut
- Ma możliwość współistnienia z innymi modułami
- Spełnia wszystkie normy prawne w naszym kraju

4. Określenie aktorów

Aktor określa zbiór ról odgrywanych przez użytkowników systemu. Zostali wyróżnieni następujący aktorzy:
Gość - nie zalogowany gośc serwisu.
Funkcje: Zakładanie konta, logowanie, odzyskiwanie hasła, weryfikacja email.
Użytkownik - użytkownik zalogowany w serwisie, rozszerzenie funkcjonalności gościa
Funkcje: zmiana danych użytkownika, zakładanie tablic
Członek tablicy - użytkownik należący do tablicy, rozszerzenie funkcjonalności użytkownika
Funkcje: Zarządzanie zadaniami, zarządzanie statusami zadań.
Właściciel tablicy - właściciel tablicy, rozszerzenie funkcjonalności członka tablicy
Funkcje: Zarządzania członkami tablicy

5. Analiza statyczna:

a) **Diagram przypadków użycia**
Graficzna prezentacja aktorów, związków między nimi oraz funkcjonalności serwisu)



b) Struktura bazy danych

Baza danych została stworzona za pomocą technologii MongoDB, zorientowanej na dokumenty. Wyróżniono dwa dokumenty główne:

Users –dokument zawierający dane użytkowników serwisu.
Budowa:
_id: ObjectId - pole identyfikacyjne rekordu,
Login: string – login używany do autoryzacji,
Password: password - zaszyfrowane hasło używane do autoryzacji,
Email: ObjectEmail - adres email używany do wysyłania powiadomień,
emailConfirmed: boolean - informacja czy użytkownik potwierdził adres email,
boards: Collection<ObjectId> - lista kluczy wskazujących tablice w dokumencie Boards, do których użytkownik należy,
invitations: Collection<ObjectId> - lista kluczy wskazujących tablice w dokumencie Boards, do których użytkownik jest zaproszony.

Boards – dokument zawierający tablice zadań serwisu.
Budowa:
_id: ObjectId - pole identyfikacyjne rekordu,
Name: string - nazwa tablicy,
Owner: ObjectId - klucz wskazujący właściciela w dokumencie Users,
Tasks: Collection<Objects> - struktura danych opisująca zadania

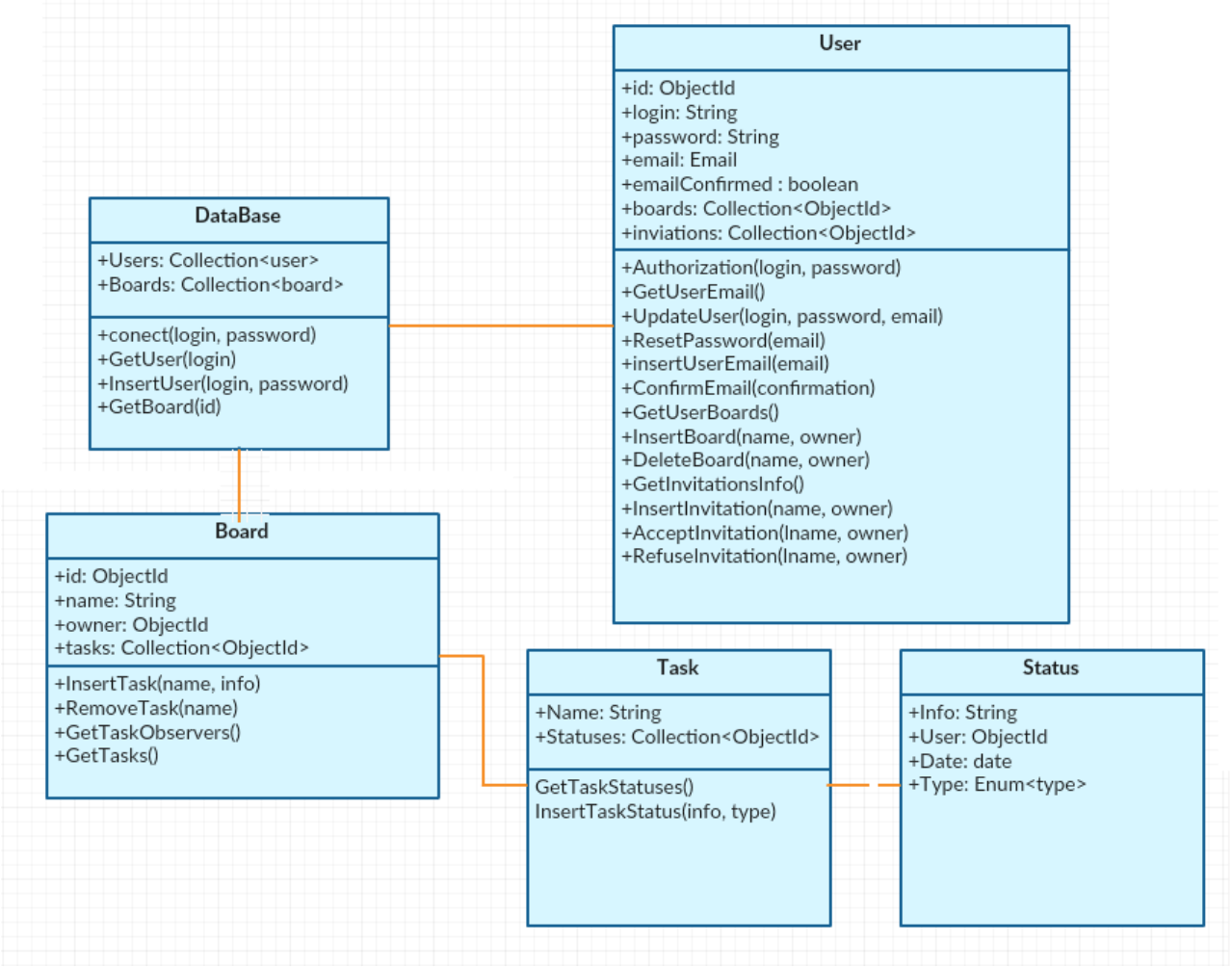
Task – struktura danych określająca zadanie w tablicy.
Budowa:
Name: string - nazwa zadania,
Statuses: Collection<Objects> - struktura danych opisująca statusy

Status – struktura danych określająca status zadania.
Budowa:
Type: enum:type - wartość opisująca typ status,
User: ObjectId - klucz wskazujący użytkownika w dokumencie Users, który dodał status,
Info: string - dodatkowy komentarz tekstowy,
Date: date - timestamb tworzony podczas dodania komentarza

Type – typ wyliczeniowy, określający rodzaj status
Możliwe wartości: New, Blocked, InProgress, Finished, Resumed.

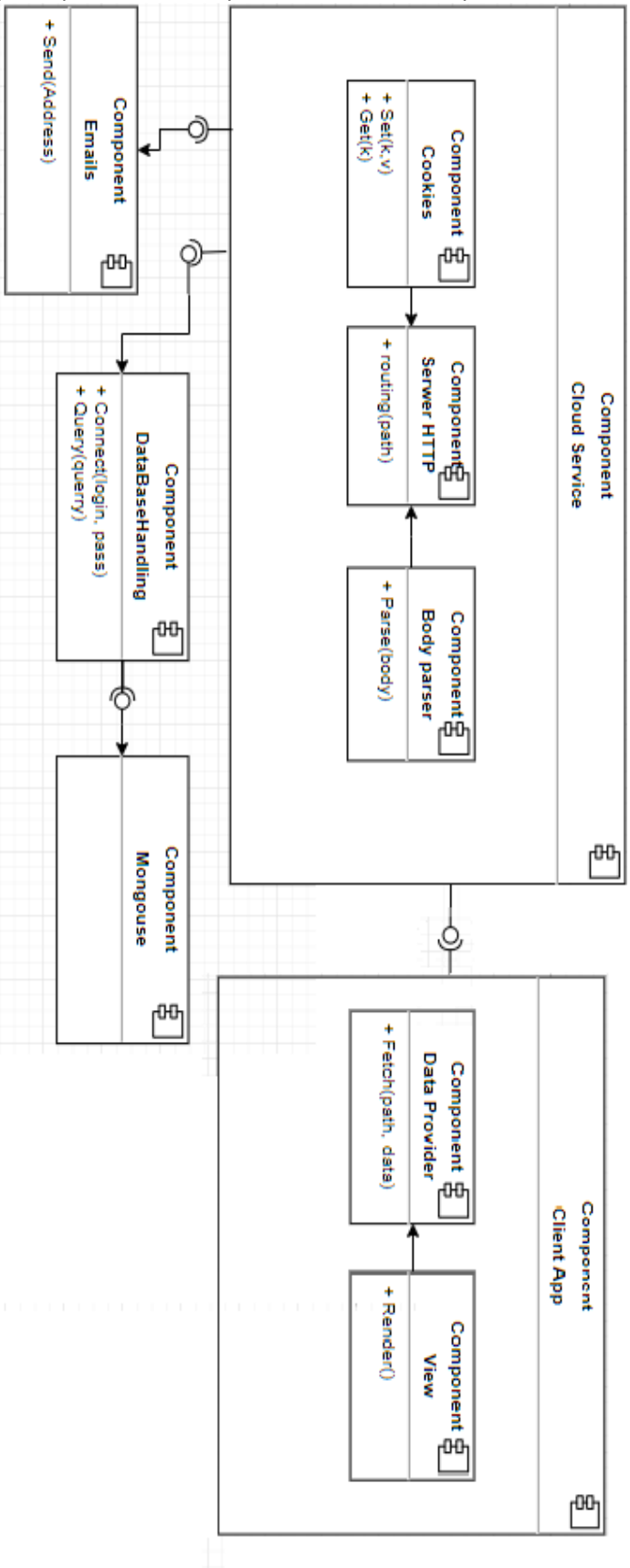
c) Diagram klas

Informacje określające statyczne związki między elementami serwisu. Został stworzony na podstawie bazy danych, ponieważ najwierniej odzwierciedla ona relacje między encjami.



d) Diagram komponentów

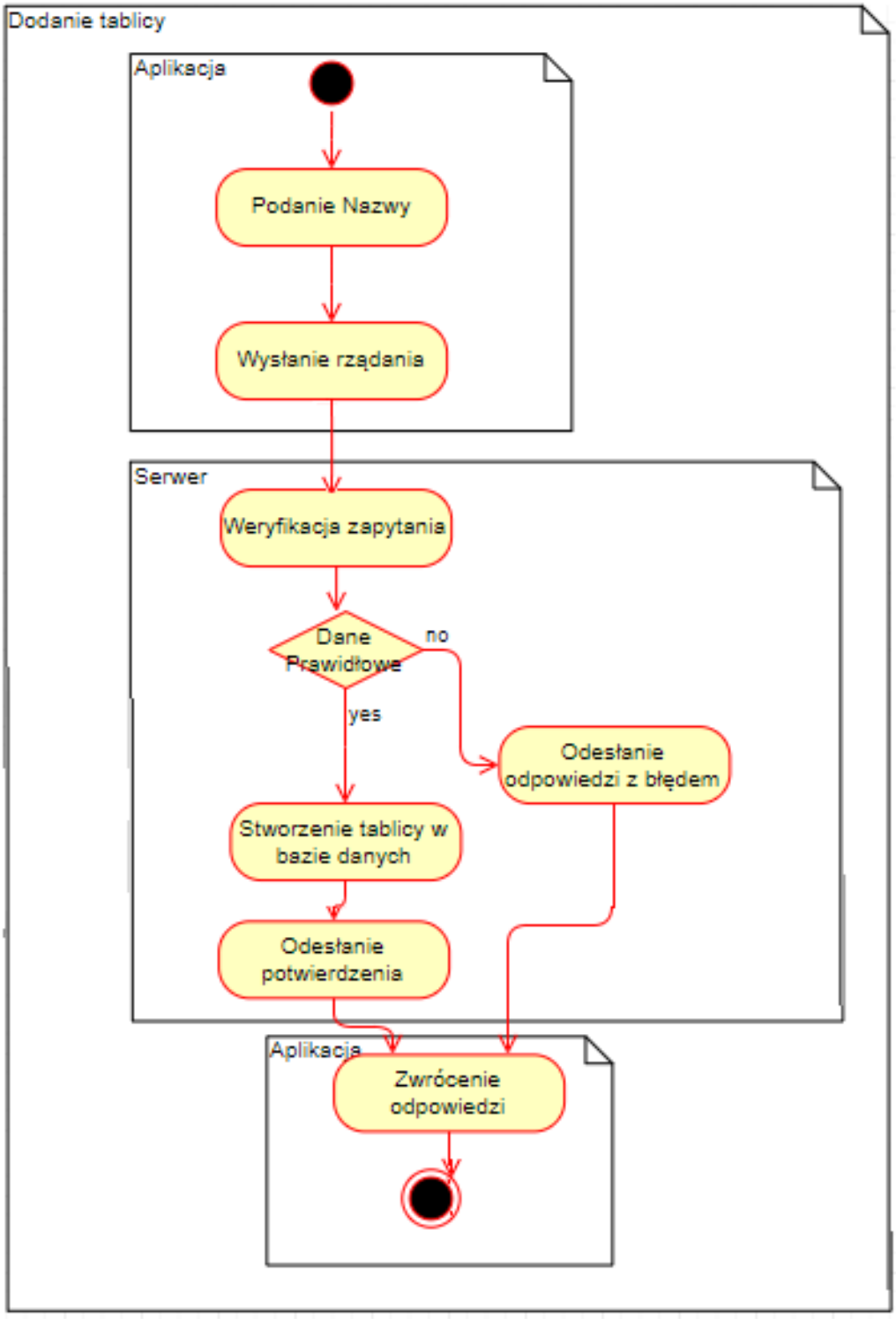
Prezentacja systemu z podziałem na mniejsze, jednostki programowe wykonujące określone zadania. Każdy komponent może być dowolnie zmieniany.



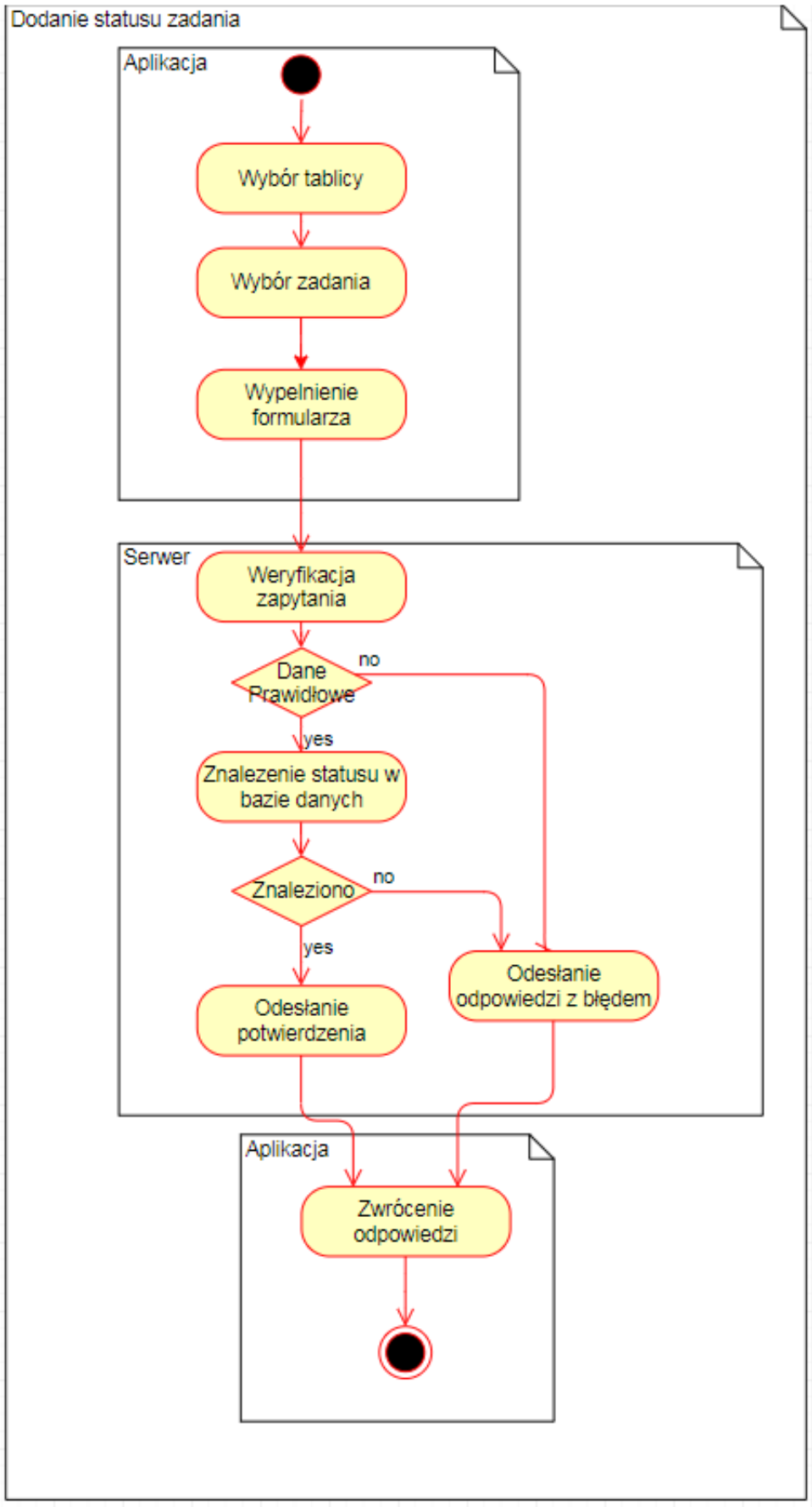
Cloud service – komponent reprezentujący aplikacje serwerową,
Cookies – komponent Cloud service'u, odpowiada za obsługe technologii ciastek http
Serwer HTTP – komponent Cloud service'u , działający w architekturze restapi dopasowujący adres i metody http do odpowiednich funkcji serwisu
Body parser - komponent Cloud service'u, odpowiedzia za obługe odebranych zasobów
Client app – komponent reprezentujący interfejs użytkownika
Data provider – komponent Client App'a odpowiadający za komunikacje z częścią serwerową
View – komponent Client App'a odpowiadający za prezentacje danych
Emails – komponent odpowiadający za wysyłanie powiadomień w postaci wiadomości email
DataBaseHandling – komponent kontroler do DBMS
Mongouse – komponent DBMS obsługujący komunikacje z bazą danych MongoDB

6. Analiza Dynamiczna

a) Diagram aktywnosci
Model kolejnych czynności oraz odpowiedzialności wybranych akcji
Dodanie nowej tablicy przez zalogowanego użytkownika:



Dodawanie nowego statusu zadania przez zalogowanego użytkownika



b) **Diagram kooperacji**

Prezentacja jak poszczególne elementy serwisu współpracują ze sobą w celu osiągnięcia danej funkcjonalności.

Diagram prezentujący współprace pomiędzy użytkownikami serwisu

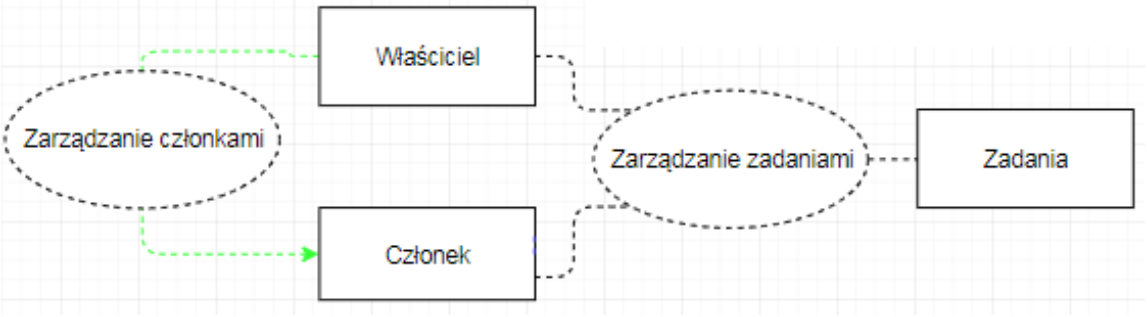
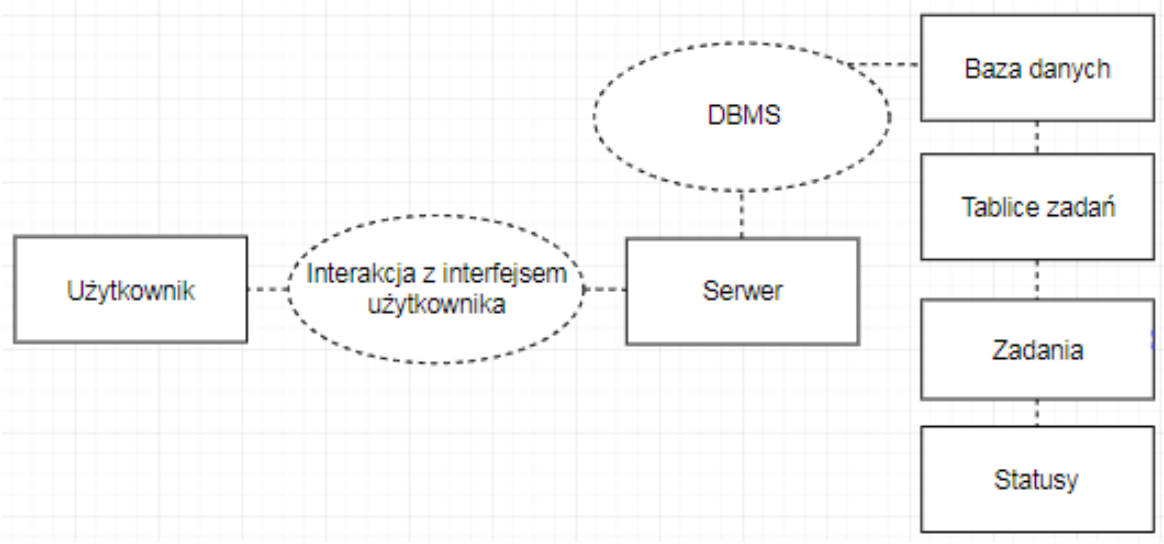


Diagram prezentujący współprace zasobów systemu z użytkownikiem



c) **Diagram maszyny stanow**

Reprezentacja wszystkich możliwych stanów na elementach serwisu.

Diagram tablicy z zadaniami

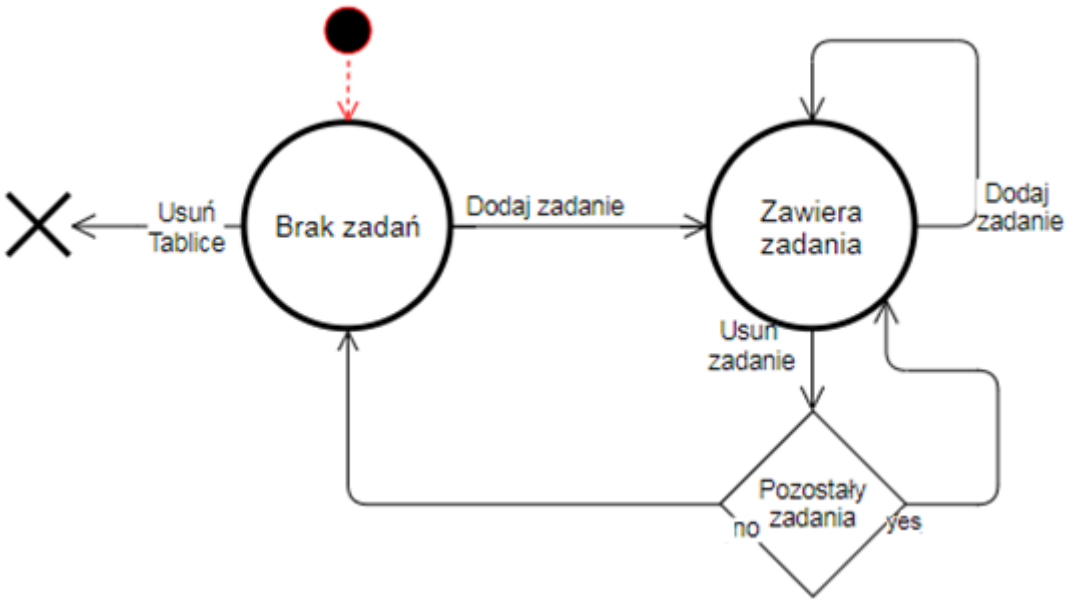
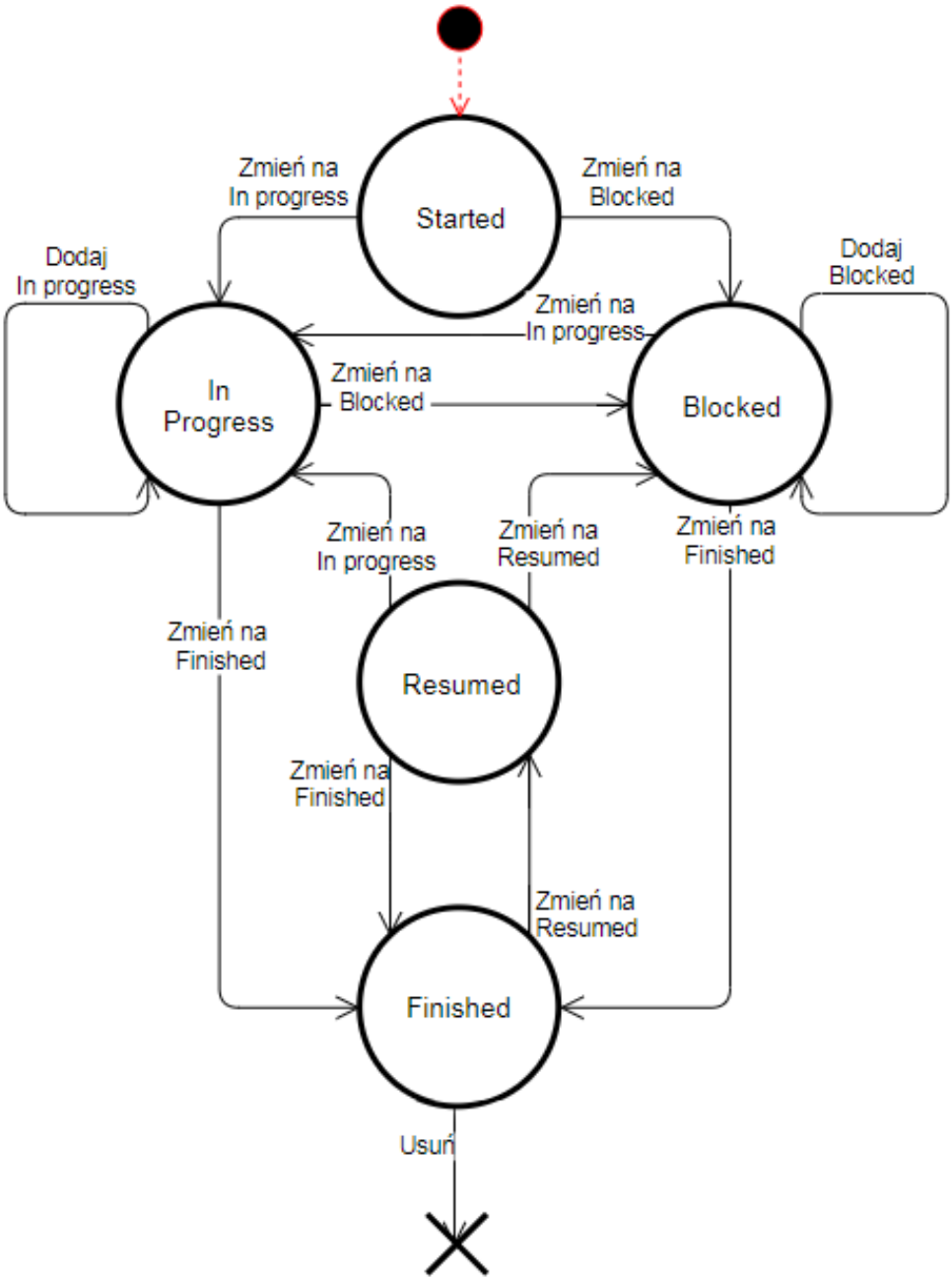


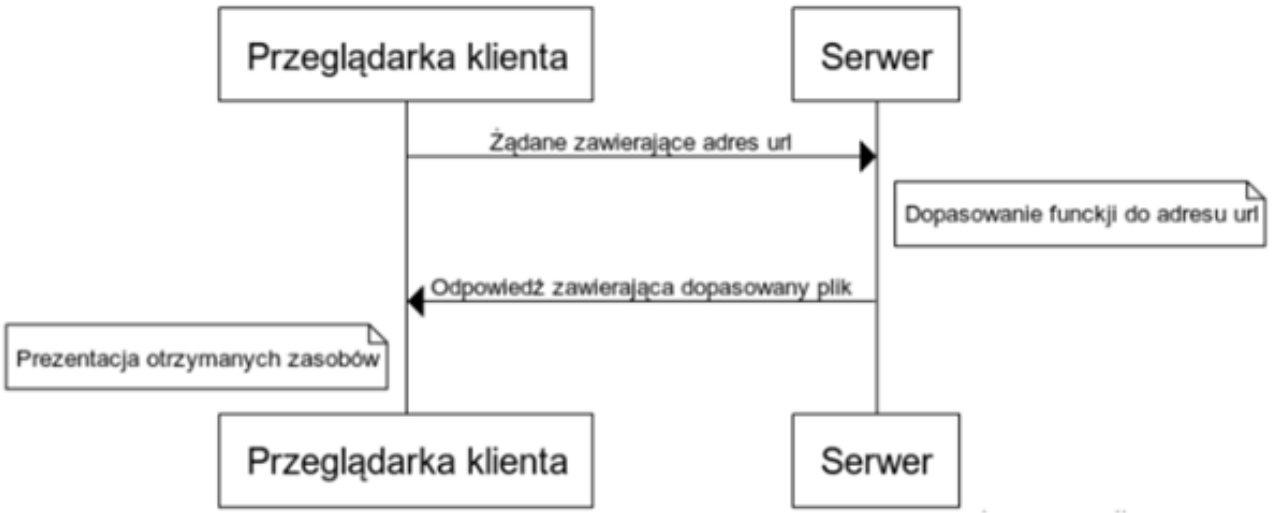
Diagram statusu zadania



d) Diagram sekwencji

Opis interakcji części systemu poprzez wymieniane komunikaty.

Serwowanie statycznych zasobów



Przepływ danych wejściowych od użytkownika