

---

# Dokumentacja projektu doc-digit

Rafał Aspras, Mateusz Koczan

## 1. Technologie

### 1. Aplikacja - C#

- WPF
- Twain

### 2. Serwer - Java

- Tesseract

### 3. API - Python

- FastAPI
- SQLAlchemy

### 4. Magazyn danych

- MinIO

### 5. Baza danych

- PostgreSQL

### 6. Dokumentacja

- AsciiDoctor
- Swagger

### 7. Testowanie

- JUnit
- CircleCi

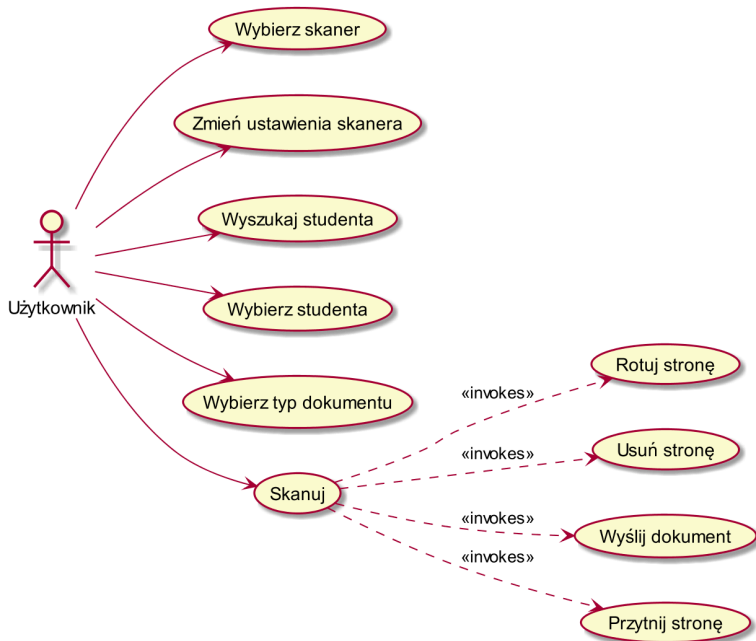
## **2. Model uprawnień do zasobów**

- Admin: all
- Użytkownik: Wysyła pliki, dostęp do wszystkich plików
- Student: Dostęp do swoich oraz publicznych plików

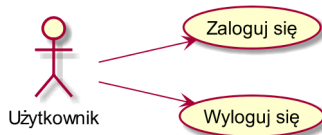
### 3. Model dziedziny użytkownika

#### 3.1. Diagram przypadku użycia

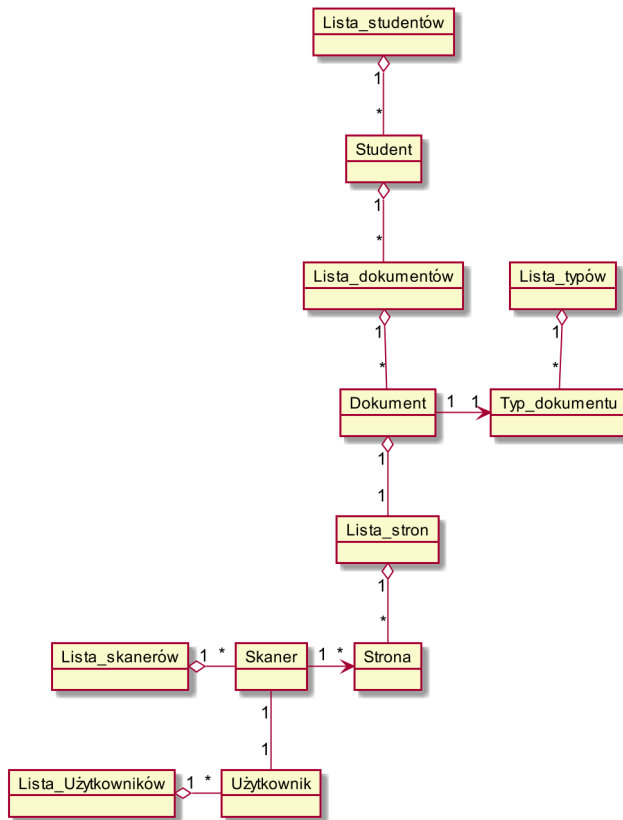
- Zarządzanie skanowaniem dokumentu



- Autoryzacja



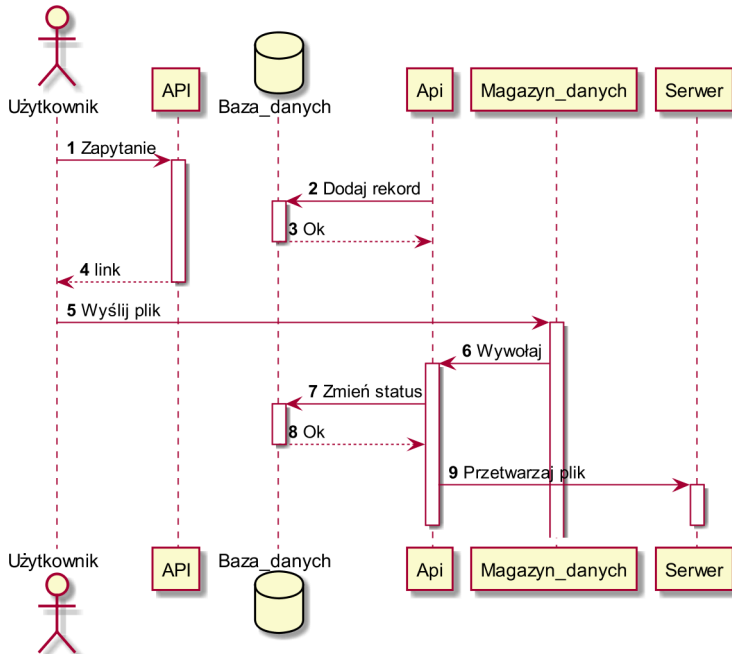
### 3.2. Diagram obiektów



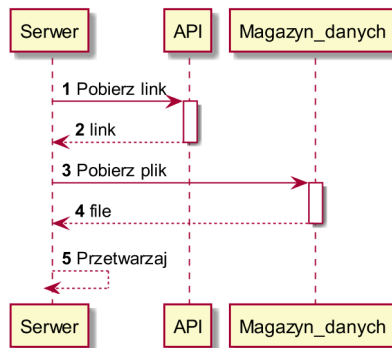
## 4. Model dziedziny interfejsu

### 4.1. Diagram sekwencji

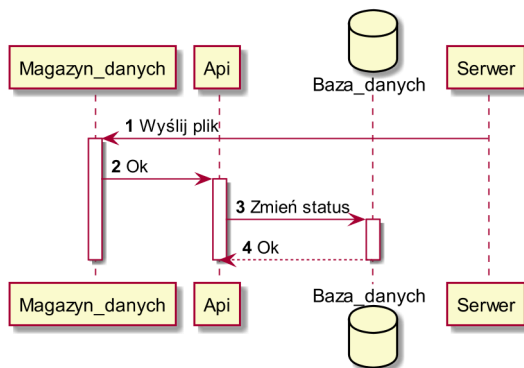
- Wysłanie pliku do przetworzenia



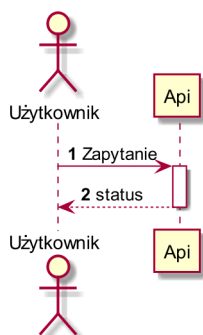
- Przetwarzanie pliku



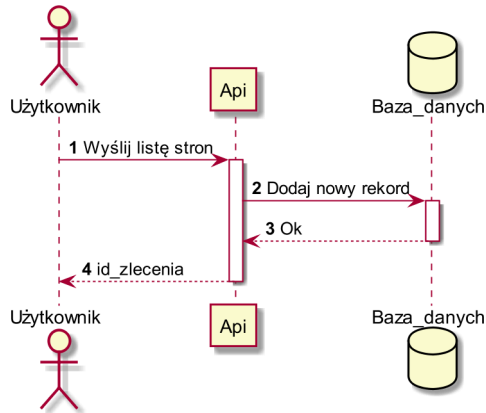
- Wgranie pliku po przetworzeniu



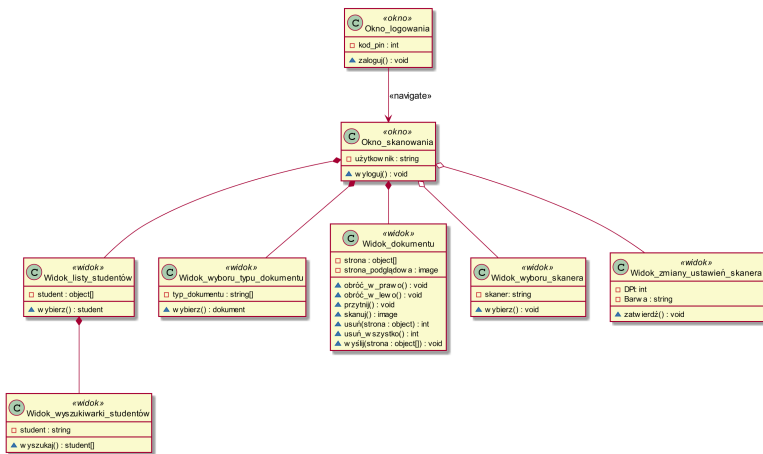
- Klient sprawdza status przetworzonego pliku



- Zapisanie sesji skanowania



## 4.2. Diagram klas



## 5. Prototyp

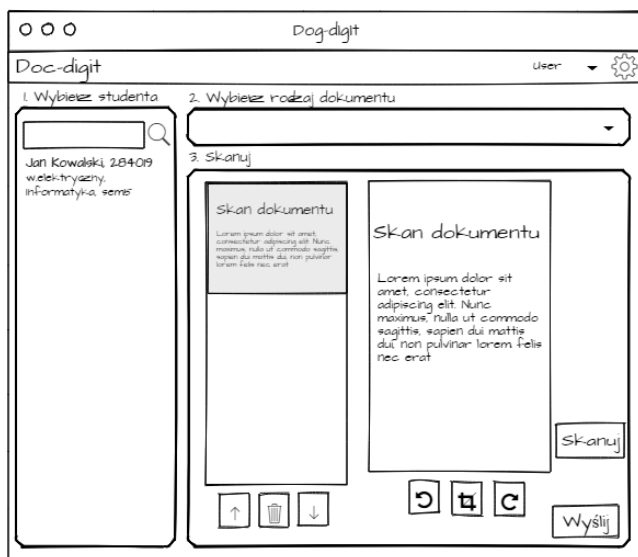
### 5.1. Ekran logowania

Uruchomienie aplikacji powoduje pojawienie się na początku okienka logowania. W celu zalogowania się, należy wpisać kod pin oraz kliknąć w przycisk „zaloguj”



## 5.2. Ekran skanowania

Poniższy obrazek przedstawia okno skanowania. Wyświetlane jest lista studentów, widok skanowania oraz widok typu dokumentu.



## 6. Architektura

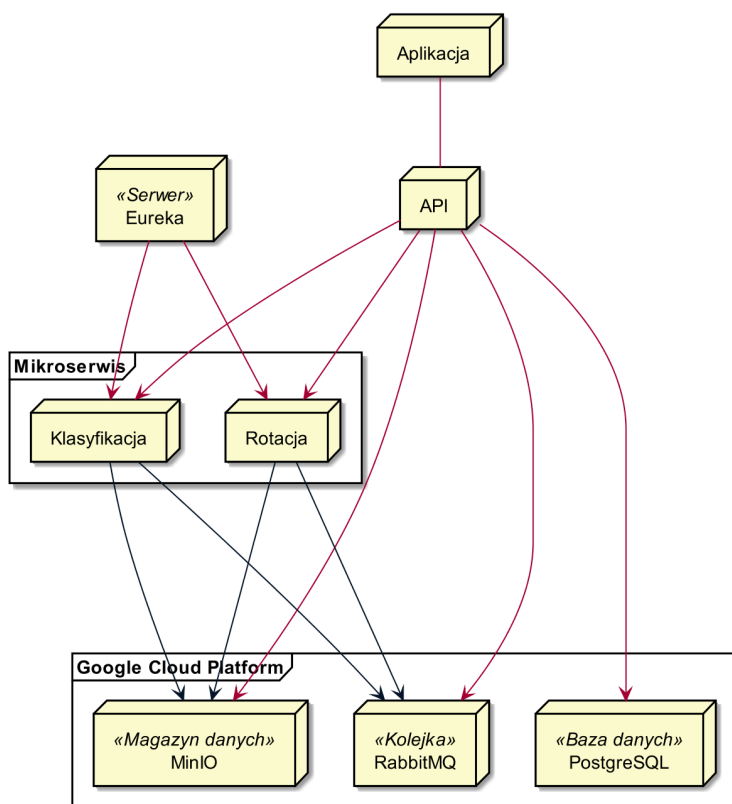
System składa się z następujących komponentów:

- Aplikacja - zajmuje się skanowaniem dokumentów, które odbywa się za pomocą sterownika Twain. Umożliwia komunikację z API w celu rotacji oraz klasyfikacji dokumentu.

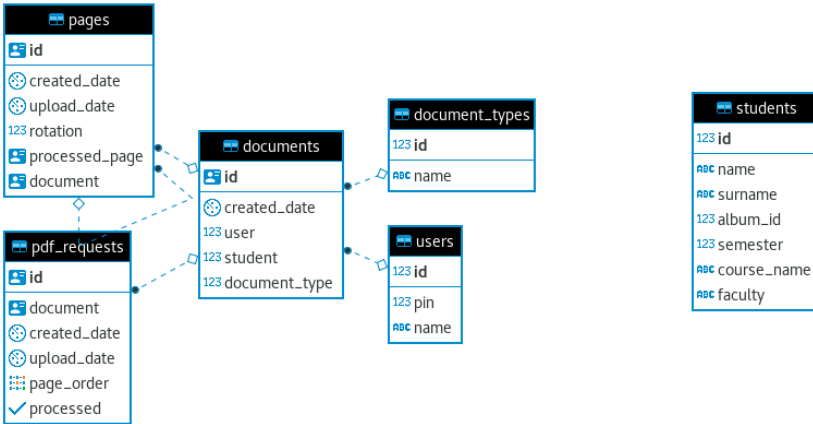


- Serwer - jest złożony z dwóch mikroservisów, które odpowiadają kolejno za rotację i klasyfikację dokumentów. Całość jest zarządzana przez serwer Eureka. Komunikacja z bazą danych odbywa się przy pomocy API.
- API - jest głównym komponentem odpowiedzialnym za komunikację z aplikacją oraz serwerem. Dodatkowo zapisuje w bazie danych status przetwarzania dokumentu, udostępnia linki do pobierania i zapisywania plików zawartych w magazynie danych. Dołącza ono także zadania do kolejki komunikatów.

### 6.1. Diagram architektury:



## 6.2. Baza danych



## 6.3. Kolejka

Służy do kolejowania żądań, takich jak przetwarzanie strony lub wygenerowanie pdfa. Zadanie te są zlecane przez API, a następnie przekazywane do serwera w celu ich wykonania.

## 6.4. Magazyn danych

Przechowuje pliki w dwóch "bucketach". Pierwszy z nich służy do trzymania obrazów skanów stron, natomiast w drugim znajdują się przetworzone i złączone w całość dokumenty w formacie pdf.

## 6.5. Klasyfikacja

Jest to mikroservis, który służy do klasyfikowania oraz generowania dokumentów w postaci pdfa. Kontaktuje się z kolejką, która zleca mu wykonanie danego zadania oraz z API w celu obsłużenia bazy danych i pobrania linków, aby można było wysłać przekonwertowany dokument do magazynu danych.

## 6.6. Rotacja

Jest to mikroservis, który obsługuje zadania rotacji znajdujące się w kolejce. W celu jego wykonania, najpierw pobiera skan strony z magazynu danych.

Następnie pobiera link od API do wysłania obróconego obrazka. I na koniec, po wykonaniu żądania, wysyła obrócony skan do magazynu danych.

## 7. Wdrażanie

Budowanie, testowanie i wdrażanie API zostało zautomatyzowane przy pomocy platformy CircleCi, która umożliwia pracę według praktyki CI/CD, czyli ciągłej integracji i ciągłego wdrażania, co zapewnia aktualną wersję API na produkcji. Środowisko testowe jest budowane i uruchamiane w kontenerach Docker, dzięki czemu można łatwo sprawdzić integrację API z innymi modułami (magazyn danych, kolejka komunikatów, baza danych). Po poprawnym przejściu wszystkich testów, budowany jest obraz API, który jest umieszczany na Docker Hub. Następnie przy pomocy SSH platforma CircleCi łączy się ze zdalnym serwerem i pobiera najnowszą wersję API. Wszystkie części systemu są uruchamiane na instancjach maszyn Google Cloud Platform, które mają włączoną autonaprawę, co zapobiega problemom z niedostępnością serwisów.

## 8. Swagger API

<https://35.193.72.232/docs>

