# Dokumentacja Projektu: Automatyzacja Oceny Sprawozdań Studenckich przy Użyciu Dużych Modeli Jezykowych (StudentReportLLMs)

Magdalena Pakuła Jakub Pawlak Piotr Hynasiński
Artur Pietrzak Rafał Górniak

June 16, 2024

Przedmiot: Zarzadzanie procesem wytwarzania oprogramowania
WFTIMS, Politechnika Łódzka

# Contents

# 1 Wstep

## 1.1 Cel dokumentu

Celem niniejszego dokumentu jest przedstawienie szczegółowej dokumentacji projektowej dla systemu Automatyzacja Oceny Sprawozdań Studenckich przy Użyciu Dużych Modeli Jezykowych (StudentReportLLMs).

Dokument ma na celu opisanie zakresu projektu, jego celów, oraz definicji zwiazanych z projektem. Jest on przeznaczony dla wszystkich interesariuszy, w tym członków zespołu projektowego, kadry dydaktycznej oraz potencjalnych użytkowników systemu.

## 1.2 Zakres projektu

Projekt StudentReportLLMs obejmuje zaprojektowanie i implementacje systemu wykorzystujacego zaawansowane modele jezykowe do automatycznej oceny sprawozdań studenckich.

Ocena ma dotyczyć dwóch kluczowych aspektów: jakości treści oraz zgodności z wytycznymi projektowymi (konkretnego zadania, którego dotyczy sprawozdanie). System ma również oceniać oryginalność tekstu w celu zapobiegania plagiatom. Zakres prac obejmuje:

- Analize i projektowanie systemu, w tym zdefiniowanie głównych funkcji oraz wymagań technicznych.
- Wybór odpowiednich technologii i modeli jezykowych.
- Implementacje systemu, w tym rozwój interfejsu użytkownika oraz integracje modeli jezykowych.
- Testowanie i ewaluacje systemu.

## 1.3 Terminy i definicje

| **Termin** | **Definicja** |
|---|---|
| LLMs (Large Language Models) | Zaawansowane modele jezykowe, takie jak GPT-4, BERT, T5, wykorzystywane do analizy i generowania tekstu. |
| Ocena jakości treści | Proces analizy sprawozdań pod katem poprawności jezykowej, spójności oraz klarowności przekazywanych informacji. |
| Ocena zgodności z wytycznymi | Proces sprawdzania, czy sprawozdanie spełnia określone wymagania projektowe. |
| Plagiat | Nieuprawnione wykorzystanie cudzej pracy bez odpowiedniego uznania źródła, co projekt ma na celu wykrywać i zapobiegać. |
| Interfejs użytkownika (UI) | Platforma webowa umożliwiajaca interakcje z systemem przez studentów i nauczycieli. |
| Information Retrieval | Proces wyszukiwania informacji w dużych zbiorach danych, który w projekcie bedzie wspierany przez bazy danych i technologie zarzadzania wektorami. |
| Duże modele jezykowe (LLMs) | Duże modele jezykowe to zaawansowane systemy sztucznej inteligencji, które zostały wytrenowane na ogromnych zbiorach danych tekstowych. LLMs sa w stanie rozumieć i generować ludzka mowe w sposób, który imituje ludzkie pisanie. |

Kontynuacja na nastepnej stronie

| Termin | Definicja |
|---|---|
| Automatyzacja | Automatyzacja odnosi sie do procesu zastepowania lub uzupełniania zadań wykonywanych przez ludzi zadaniami wykonywanymi przez maszyny lub programy komputerowe. Celem automatyzacji jest zwiekszenie wydajności, obniżenie kosztów i minimalizacja błedów poprzez eliminacje ludzkiego udziału lub zautomatyzowanie powtarzalnych zadań. |
| Tokenizacja | Proces podziału tekstu na mniejsze jednostki, zwane tokenami, co jest czesto używane w przetwarzaniu jezyka naturalnego, aby lepiej zrozumieć i analizować zawartość tekstu. |
| OCR (Optical Character Recognition) | Technologia pozwalajaca na konwersje obrazów lub zeskanowanych dokumentów na edytowalny tekst, co jest istotne dla odczytu i analizy zawartości plików PDF. |
| Skalowalność systemu | Skalowalność systemu odnosi sie do zdolności systemu do efektywnego dostosowywania sie do wzrostu obciażenia lub zasobów. System jest uznawany za skalowalny, jeśli może rosnać lub kurczyć sie w zależności od potrzeb, zachowujac przy tym swoja wydajność, niezawodność i funkcjonalność. |
| API (Application Programming Interface) | Interfejs programowania aplikacji, który umożliwia komunikacje miedzy różnymi komponentami oprogramowania, na przykład miedzy systemem automatyzacji oceny a silnikami modeli jezykowych. |
| System odczytu i analizy plików | System zdolny do przetwarzania plików w formatach PDF, Word, LaTeX w celu analizy ich treści przez modele jezykowe. |

## 1.4 Opis potrzeby

Głównym założeniem, a zarazem potrzeba jest wiarygodne, dokładne oraz z jak najmniejszym błedem sprawdzanie prac naukowych napisanych przez usługobiorców akademii, czyli studentów przy pomocy określonego narzedzia. Przy danych z góry kryteriach, program ma zachowywać sie w należyty sposób. Potrzeba zaistniała ze wzgledu na możliwa stronniczość wykładowcy. Oznacza to, że osoby, które chodziły na wykłady, krewni badź znajomi, osoby faworyzowane przez różne wzgledy mogłyby być oceniane wyżej w porównaniu do reszty, za taka sama prace.

Nastepnym możliwym zagrożeniem jest bład ludzki. W danej chwili, przy braku stu-procentowej pewności osoba sprawdzajaca może niesłusznie odjać punkty, podczas gdy rozwiazania, fraza zaproponowana przez studenta jest zupełnie poprawna i vice versa. Nastepna coraz wieksza "plaga" jest plagiat oraz paragrafy generowane przez sztuczna inteligencje. To również powinno być rozróżnione oraz wyłapane.

Dlatego wiec, aby każdy uczeń był równy wobec systemu sprawdzania, potrzebna jest technika, program to zapewniajacy. Ponadto każde konto uczelniane powinno mieć dostep do tego rozwiazania, a zatem potrzebna jest autentykacja użytkownika dla studenta oraz nauczyciela, jako pośredniego administratora narzedzia.

## 1.5 Zidentyfikowani odbiorcy

| Nazwa | Opis |
|---|---|
| Wykładowca (Nauczyciel akademicki) | Osoba pracujaca w systemie oświaty w sektorze publicznym lub prywatnym w zależności od rodzaju miejsca pracy. Jednym z obowiazków zadawanie, a w konsekwencji sprawdzanie prac pisemnych, napisanych przez klientów uczelni lub kadry akademickiej w roli recenzenta. |
| Student (Uczeń) | Osoba realizujaca oferte uczelni, na której sie znajduje w charakterze usługobiorcy. Pisze zadany przez nauczyciela raport, a nastepnie według kryteriów oceniania dostaje należyta informacje zwrotna w postaci oceny. |

# 2 Opis systemu

Projekt *StudentRaportLLMs* polega na stworzeniu kompleksowego systemu opartego na zaawansowanych modelach jezykowych, który automatycznie oceni sprawozdania studenckie. System bedzie przetwarzał tekst sprawozdań, analizujac treść i sprawdzajac zgodność z wytycznymi projektowymi lub innymi kryteriami.

Na podstawie analizy, system bedzie przyznawał oceny, dostosowane do konkretnego zadania lub przedmiotu. Aby zapewnić wiarygodność ocen, wyniki automatycznej oceny beda porównywane z wynikami recznie ocenionych sprawozdań, co pozwoli na dostosowanie parametrów oceniania i udoskonalenie procesu.

Ponadto, system bedzie integrowany z istniejacymi platformami edukacyjnymi, ułatwiajac przesyłanie prac i wymiane informacji miedzy studentami a nauczycielami. Projekt bedzie elastyczny i przystosowany do zmian w technologiach oraz przepisach prawnych dotyczacych ochrony danych osobowych.

## 2.1 Zakres funkcjonalny

System ma wykorzystywać duże modele jezykowe do oceny sprawozdań studenckich. Ocena ma dotyczyć dwóch kluczowych aspektów: jakości treści i zgodności z wytycznymi projektowymi (konkretnego zadania, którego dotyczy sprawozdanie). Dzieki temu możliwe bedzie szybkie, obiektywne i dokładne ocenianie treści sprawozdań oraz ich zgodności z wytycznymi projektowymi. To rozwiazanie ma potencjał zrewolucjonizować proces oceniania, poprawiajac jakość edukacji poprzez wprowadzenie bardziej obiektywnych i zgodnych z wytycznymi ocen.

Wymagane bedzie zintegrowanie i wykorzystanie zaawansowanych modeli jezykowych, bez konieczności ich dodatkowego szkolenia (fine-tuning). Projekt powinien obejmować funkcjonalność odczytu i analizy plików w formatach PDF, Word, LaTeX. Dodatkowo, system musi umożliwiać definiowanie przez operatora specyficznych kryteriów oceny, adaptowalnych do różnorodnych wymagań projektowych. Projekt zakłada również opracowanie metody oceny oryginalności tekstu, aby zapobiegać plagiatom i promować unikalność prac studenckich.

## 2.2 Zakres niefunkcjonalny

Ważnym elementem bedzie stworzenie intuicyjnego interfejsu użytkownika, który pozwoli na łatwe formułowanie zapytań dotyczacych specyficznych kryteriów oceny dla ogółu dostepnych prac, takich jak czesto popełniane błedy czy powtarzajace sie braki w sprawozdaniach. Kluczowymi elementami beda efektywność, precyzja oraz użyteczność systemu. Ważne bedzie również stworzenie szczegółowej dokumentacji projektowej.

# 3 Wymagania

Poniżej znajduja sie wszystkie wymagania funkcjonalne i niefunkcjonalne projektu:

## 3.1 Wymagania funkcjonalne

1. Zarzadzanie kontami użytkowników
   (a) Tworzenie nowych kont
   (b) Modyfikacja przywilejów istniejacych kont
2. Formułowanie przez nauczyciela kryteriów oceny pracy
3. Analiza jakościowa treści sprawozdania
4. Analiza zgodności z wymaganiami treści sprawozdania
5. Porównanie pod katem plagiatu z historycznymi pracami zapisanymi w uczelnianym archiwum
6. Odczyt różnych formatów plików (.docx, .pdf, .tex)
7. Zabezpieczenie przed atakami typu *prompt injection*

## 3.2 Wymagania niefunkcjonalne

1. Intuicyjność i reaktywność interfejsu użytkownika
2. Wysokie wartości miar porównujacych wyniki automatycznego oraz manualnego sprawdzania:
   (a) F1-score
   (b) Precision
   (c) Recall
   (d) Accuracy
3. Pozytywna korelacja pomiedzy wynikami automatycznego i manualnego sprawdzania
4. Automatyczne testy poprawności działania systemu
5. Szczegółowa dokumentacja projektowa
6. Satysfakcjonujacy czas weryfikacji pracy przez system

## 3.3 Diagramy

### Diagram przepływu danych

DFD sa szczególnie przydatne do przedstawienia przepływu danych i procesów w ramach systemu. Poniżej na fig. 1 znajduje sie diagram przedstawiajacy, jak przetwarzane sa dane w systemie.

### Diagram Business Process Model and Notation

BPMN idealnie nadaje sie do modelowania procesów biznesowych. Poniżej na fig. 2 znajduje sie diagram, który wskaże kluczowe procesy biznesowe zwiazane z funkcjonalnościa systemu.

### Diagram SWOT

Diagram SWOT (Strengths, Weaknesses, Opportunities, Threats) pokazany niżej na fig. 3 pomoże zidentyfikować wewnetrzne mocne strony, słabości oraz zewnetrzne szanse i zagrożenia.

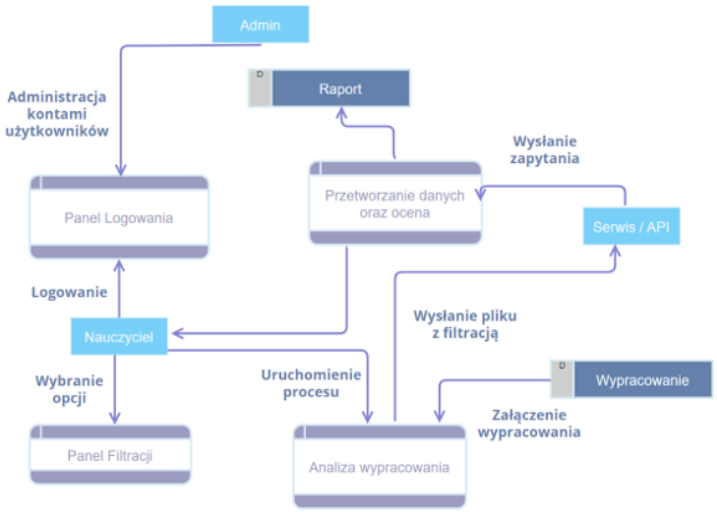


Figure 1: Diagram przepływu danych w systemie *StudentReportLLMs*

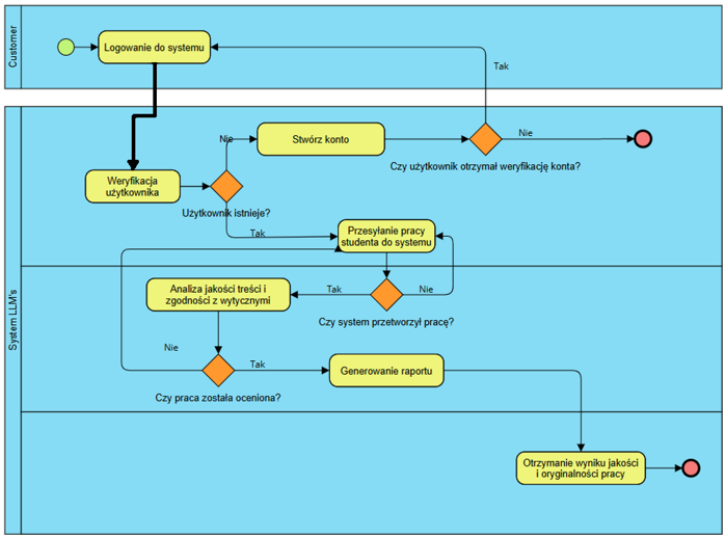


Figure 2: Diagram Business Process Model and Notation systemu *StudentReportLLMs*

## 3.4 Opis przypadków użycia

### Przypadek użycia 1: Przesyłanie raportu do systemu

Table 2: Przypadek użycia: Przesyłanie raportu do systemu

| | |
|---|---|
| **Typ** | Operacyjny |
| **Użytkownicy** | Nauczyciel akademicki |
| **Warunki wstepne** | • Nauczyciel musi być zalogowany do systemu.<br>• Raport studencki musi być w odpowiednim formacie (Word, PDF, LaTeX). |
| **Typowa kolejność kroków** | 1. Nauczyciel loguje sie do systemu StudentReportLLMs.<br>2. Nauczyciel wybiera opcje przesyłania nowego raportu.<br>3. System wyświetla formularz do przesyłania pliku.<br>4. Nauczyciel wypełnia formularz, dodajac odpowiedni plik raportu oraz wszelkie dodatkowe informacje wymagane przez system (np. kategoria, autor, przedmiot).<br>5. Nauczyciel przesyła formularz, a system zapisuje raport do bazy danych i wyświetla potwierdzenie poprawnego przesłania. |
| **Warunki końcowe** | • Raport jest zapisany w systemie i oczekuje na automatyczna ocene.<br>• Nauczyciel otrzymuje powiadomienie o poprawnym przesłaniu raportu. |

| | |
|---|---|
| **Alternatywna kolejność kroków** | 1. Nauczyciel loguje sie do systemu StudentReportLLMs.<br>2. Nauczyciel wybiera opcje przesyłania nowego raportu.<br>3. System wyświetla formularz do przesyłania pliku.<br>4. Nauczyciel próbuje przesłać plik w nieprawidłowym formacie.<br>5. System wyświetla komunikat o błedzie i prosi o poprawienie formatu pliku.<br>6. Nauczyciel poprawia format pliku i ponownie przesyła formularz.<br>7. System zapisuje raport do bazy danych i wyświetla potwierdzenie poprawnego przesłania. |
| **Komentarze i uwagi** | • System powinien obsługiwać różne formaty plików i weryfikować ich poprawność przed przesłaniem.<br>• System powinien być intuicyjny i łatwy w użyciu, aby nauczyciele mogli szybko i bezproblemowo przesyłać raporty. |

## Przypadek użycia 2: Automatyczna ocena treści raportów

Table 3: Przypadek użycia: Automatyczna ocena treści raportów

| | |
|---|---|
| **Typ** | Operacyjny |
| **Użytkownicy** | Nauczyciel akademicki |
| **Warunki wstepne** | • Nauczyciel musi być zalogowany do systemu.<br>• Raport studencki musi być w odpowiednim formacie (Word, PDF, LaTeX).<br>• System musi mieć zdefiniowane kryteria oceny. |

| | |
|---|---|
| **Typowa kolejność kroków** | 1. System wykrywa nowy przesłany raport oczekujacy na ocene.<br>2. System analizuje treść raportu przy użyciu zaawansowanych modeli jezykowych.<br>3. System sprawdza zgodność treści raportu z wytycznymi projektowymi i zdefiniowanymi kryteriami oceny.<br>4. System generuje ocene raportu oraz szczegółowy feedback dotyczacy mocnych i słabych stron pracy.<br>5. System zapisuje ocene i feedback w bazie danych oraz powiadamia nauczyciela o zakończonej ocenie. |
| **Warunki końcowe** | • Nauczyciel musi być zalogowany do systemu.<br>• Raport zostaje oceniony i zapisany w systemie.<br>• Nauczyciel otrzymuje powiadomienie o zakończonej ocenie raportu. |
| **Alternatywna kolejność kroków** | 1. System wykrywa nowy przesłany raport oczekujacy na ocene.<br>2. System analizuje treść raportu przy użyciu zaawansowanych modeli jezykowych.<br>3. System napotyka problem z analiza (np. nieczytelny tekst, błedy w formacie).<br>4. System powiadamia nauczyciela o problemie i prosi o weryfikacje raportu.<br>5. Nauczyciel weryfikuje i poprawia raport, a nastepnie ponownie przesyła do systemu.<br>6. System ponownie analizuje treść raportu, sprawdza zgodność z wytycznymi i generuje ocene oraz feedback. |

| | |
|---|---|
| **Komentarze i uwagi** | • System powinien być w stanie obsługiwać różne problemy z analiza i odpowiednio informować użytkownika o napotkanych błedach.<br>• System powinien zapewniać wysoka dokładność i spójność ocen, aby użytkownicy mogli ufać wynikowym ocenom. |

## Przypadek użycia 3: Odczyt i konwersja plików

Table 4: Przypadek użycia: Odczyt i konwersja plików

| | |
|---|---|
| **Typ** | Operacyjny |
| **Użytkownicy** | Nauczyciel akademicki |
| **Warunki wstepne** | • Nauczyciel musi być zalogowany do systemu.<br>• Plik raportu musi być przesłany do systemu.<br>• System musi obsługiwać format pliku (Word, PDF, LaTeX). |
| **Typowa kolejność kroków** | 1. Nauczyciel przesyła raport do systemu w jednym z obsługiwanych formatów (Word, PDF, LaTeX).<br>2. System rozpoznaje format pliku i przystepuje do jego odczytu.<br>3. System przetwarza plik, konwertujac jego zawartość na format zrozumiały dla modeli jezykowych.<br>4. System przeprowadza analize zawartości przetworzonego pliku, identyfikujac istotne sekcje i elementy raportu.<br>5. System zapisuje przetworzona i skonwertowana wersje pliku w bazie danych oraz informuje nauczyciela o zakończeniu procesu konwersji. |
| **Warunki końcowe** | • Plik raportu jest poprawnie odczytany i skonwertowany.<br>• System zapisuje skonwertowana wersje pliku w bazie danych.<br>• Nauczyciel otrzymuje powiadomienie o zakończeniu procesu konwersji. |

| | |
|---|---|
| **Alternatywna kolejność kroków** | 1. Nauczyciel przesyła raport do systemu w jednym z obsługiwanych formatów (Word, PDF, LaTeX).<br>2. System rozpoznaje format pliku i przystepuje do jego odczytu.<br>3. System napotyka problem z odczytem pliku (np. uszkodzony plik, nieczytelny format).<br>4. System powiadamia nauczyciela o problemie i prosi o weryfikacje pliku.<br>5. Nauczyciel poprawia plik i ponownie przesyła do systemu.<br>6. System ponownie przetwarza plik, konwertujac jego zawartość na format zrozumiały dla modeli jezykowych. |
| **Komentarze i uwagi** | • System powinien być w stanie rozpoznać i poprawnie przetwarzać różne formaty plików używane przez nauczycieli.<br>• System powinien informować użytkowników o wszelkich problemach napotkanych podczas odczytu i konwersji plików, umożliwiajac szybkie rozwiazanie problemów. |

## Przypadek użycia 4: Personalizacja kryteriów oceny

Table 5: Przypadek użycia: Personalizacja kryteriów oceny

| | |
|---|---|
| **Typ** | Operacyjny |
| **Użytkownicy** | Nauczyciel akademicki |
| **Warunki wstepne** | • Nauczyciel musi być zalogowany do systemu. |
| **Typowa kolejność kroków** | 1. Nauczyciel loguje sie do systemu StudentReportLLMs.<br>2. Przechodzi do sekcji ustawień ocen.<br>3. Modyfikuje istniejace kryteria oceny lub dodaje nowe według własnych preferencji. Zapisuje wprowadzone zmiany.<br>4. System automatycznie uwzglednia nowe kryteria podczas oceny kolejnych raportów studenckich. |

| | |
|---|---|
| **Warunki końcowe** | • Zmodyfikowane kryteria oceny sa zapisane i uwzgledniane podczas kolejnych ocen raportów studenckich.<br>• Nauczyciel może dostosować proces oceniania do swoich potrzeb lub wymagań projektowych. |
| **Alternatywna kolejność kroków** | 1. Nauczyciel loguje sie do systemu StudentReportLLMs.<br>2. Przechodzi do sekcji ustawień ocen.<br>3. Nauczyciel próbuje dodać nowe kryteria oceny, ale napotyka problem z zapisaniem zmian.<br>4. System wyświetla komunikat o błedzie i prosi nauczyciela o ponowne zapisanie ustawień.<br>5. Nauczyciel ponownie próbuje zapisać zmiany, a system potwierdza poprawne zastosowanie nowych kryteriów oceny. |
| **Komentarze i uwagi** | • Interfejs użytkownika systemu powinien być intuicyjny i łatwy w obsłudze, aby nauczyciel mógł bezproblemowo dostosować kryteria oceny.<br>• System powinien zapewnić odpowiednie komunikaty o błedach w przypadku problemów podczas zapisywania zmian w ustawieniach ocen.<br>• Warto umożliwić nauczycielowi podglad zmian przed ich zatwierdzeniem, aby uniknać niezamierzonych modyfikacji kryteriów oceny. |

### Przypadek użycia 5: Interakcja użytkownika z systemem

Ten przypadek użycia obejmuje wszystkie kroki niezbedne do interakcji użytkownika z systemem oceniania LLM, w tym przesyłanie sprawozdania, ustawianie kryteriów oceny oraz pobieranie wyników oceny. Diagram przypadku użycia jest przedstawiony na figurze 4.

### 1. Przesyłanie sprawozdania

- **Otwórz aplikacje**: Upewnij sie, że interfejs użytkownika systemu oceniania LLM jest uruchomiony.
- **Prześlij sprawozdanie**:
  - Kliknij przycisk Upload w głównym oknie.
  - Wybierz plik sprawozdania (PDF, Word lub LaTeX) z lokalnego systemu.
  - Kliknij Open, aby przesłać plik.

- **Potwierdzenie**:
  - Po pomyślnym przesłaniu sprawozdania zostanie wyświetlony komunikat potwierdzajacy.
  - Zanotuj `report_id`, ponieważ bedzie potrzebny do kolejnych operacji.

## 2. Ustawianie kryteriów oceny

- **Otwórz okno kryteriów**:
  - Kliknij przycisk Set Criteria w głównym oknie.
- **Definiowanie kryteriów**:
  - Wprowadź kryteria oceny. Każde kryterium powinno mieć nazwe i wage.
  - Przykład:

```
{
  "criteria": [
    {
      "name": "Jakość treści",
      "weight": 0.5
    },
    {
      "name": "Zgodność z wytycznymi",
      "weight": 0.3
    },
    {
      "name": "Oryginalność",
      "weight": 0.2
    }
  ]
}
```

- **Zapisywanie kryteriów**:
  - Kliknij przycisk Save, aby zastosować kryteria.
  - Wyświetlony zostanie komunikat potwierdzajacy pomyślne ustawienie kryteriów.

## 3. Pobieranie wyników oceny

- **Sprawdź wyniki**:
  - Kliknij przycisk Check Results w głównym oknie.
  - Wprowadź `report_id` sprawozdania, dla którego chcesz pobrać wyniki.
- **Wyświetlanie wyników**:
  - Wyniki oceny zostana wyświetlone, pokazujac oceny jakości, zgodności i oryginalności.
  - Przykładowy format wyników:

```
{
  "report_id": "<report_id>",
  "grading": {
    "quality": 85,
    "compliance": 90,
    "originality": 95,
    "overall": 90
  }
}
```

- **Szczegółowa analiza**:
  - Klikajac na poszczególne kryteria, można zobaczyć szczegółowe informacje zwrotne i obszary do poprawy.

## Dodatkowe funkcje

- **Analiza wspólnych błedów**: System oferuje analize czestych błedów we wszystkich sprawozdaniach.
- **Eksportowanie wyników**: Można eksportować wyniki oceny do pliku CSV lub JSON dla dalszej analizy.

## Czesto wystepujace problemy

- **Błedy przy przesyłaniu plików**:
  - Upewnij sie, że format pliku jest obsługiwany (PDF, Word, LaTeX).
  - Sprawdź połaczenie internetowe i spróbuj ponownie.
- **Problem z zapisem kryteriów**:
  - Upewnij sie, że wszystkie kryteria maja poprawne nazwy i wagi.
  - Sprawdź, czy nie ma żadnych komunikatów o błedach i popraw dane według potrzeb.
- **Brak wyników**:
  - Zweryfikuj poprawność `report_id`.
  - Upewnij sie, że sprawozdanie zostało ocenione przed próba pobrania wyników.

Te kroki zapewniaja kompleksowa instrukcje użytkowania systemu oceniania LLM, obejmujaca wszystkie kluczowe funkcje i możliwości interakcji z systemem.

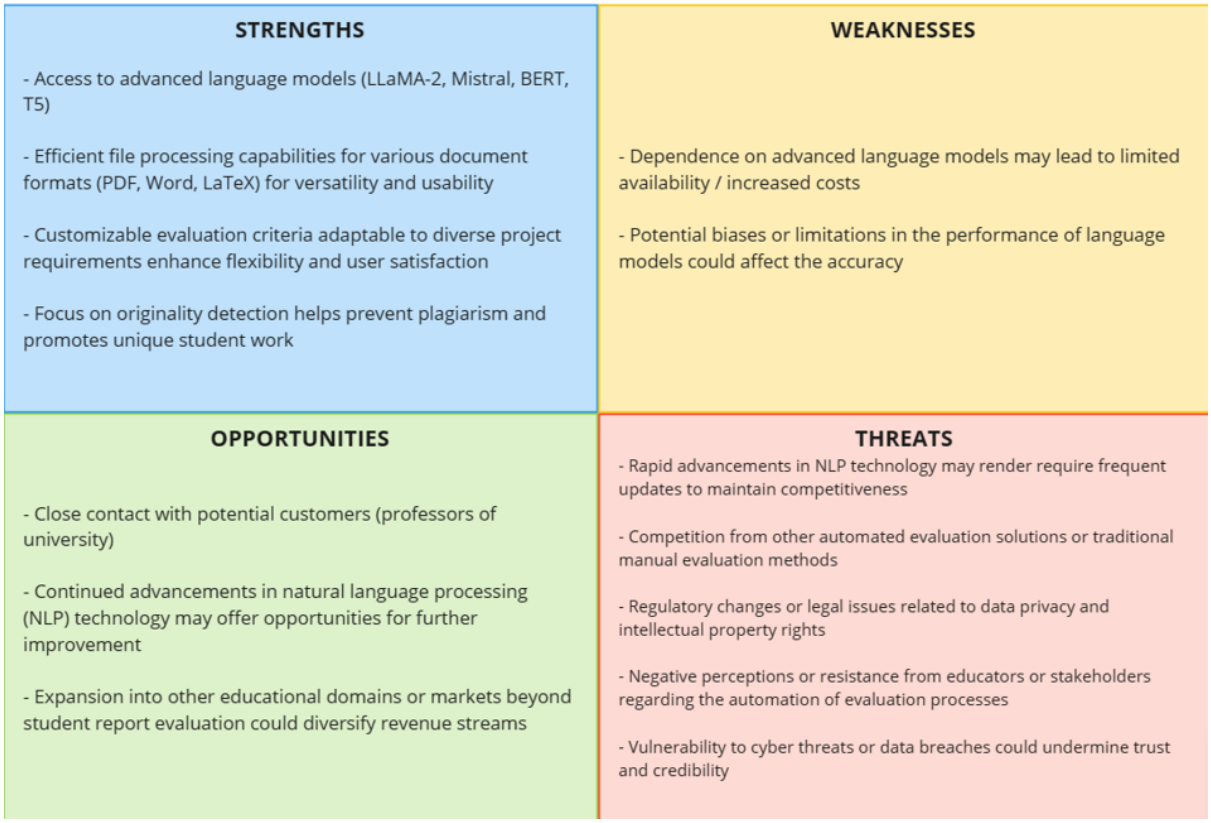


Figure 3: Diagram SWOT systemu *StudentReportLLMs*

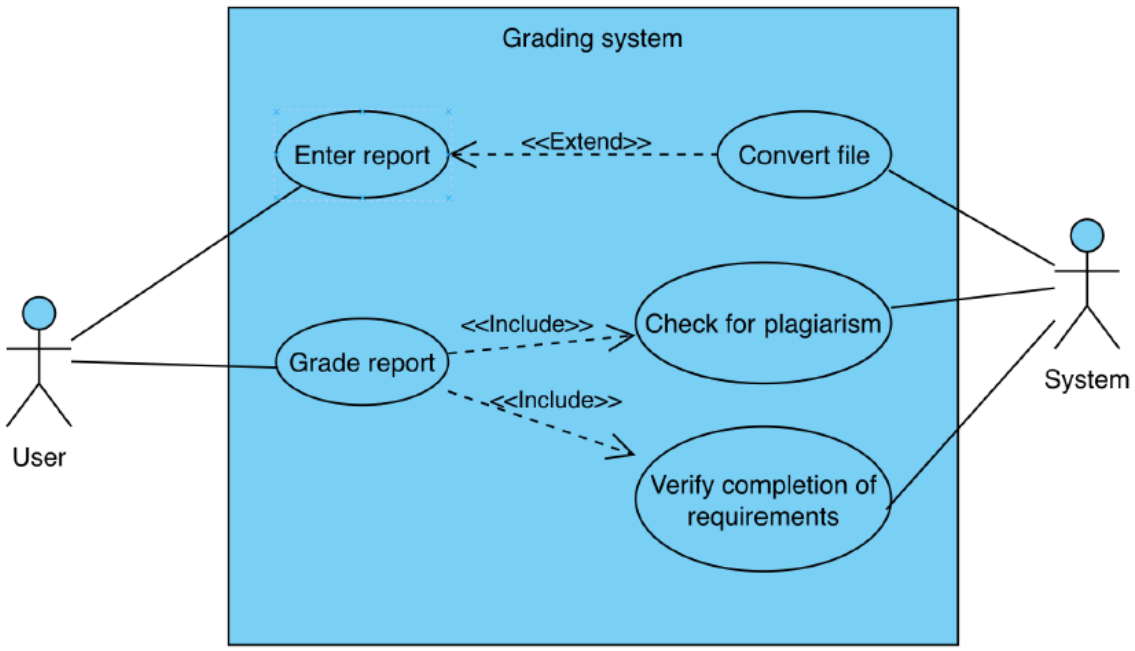


Figure 4: Diagram przypadku użycia: Interakcja użytkownika z systemem w systemie *StudentReportLLMs*

# 4 Architektura Systemu

W tej sekcji przedstawiono architekture systemu *StudentReportLLMs*, który ma na celu automatyzacje oceny sprawozdań studenckich przy użyciu dużych modeli jezykowych.

## 4.1 Architektura ogólna

Architektura systemu oparta jest na mikroserwisach, co umożliwia łatwa skalowalność i niezależność poszczególnych komponentów. Główne komponenty architektoniczne to:

- **Frontend**: Interfejs użytkownika (UI), dostepny jako aplikacja webowa umożliwiajaca studentom przesyłanie swoich sprawozdań oraz przegladanie ocen.
- **Backend**: Centralny serwis zarzadzajacy, który obsługuje logike biznesowa, integracje z modelami jezykowymi oraz zarzadzanie zadaniami oceny.
- **Serwisy modeli jezykowych**: Oddzielne serwisy odpowiedzialne za integracje z różnymi modelami jezykowymi (np. GPT-4, BERT), które wykonuja analize i ocene tekstu.
- **Baza danych**: Centralna baza danych przechowujaca skonwertowane sprawozdania oraz baza dancyh do analizy i przetwarzania sprawozdań.
- **System odczytu i analizy plików**: Serwis do przetwarzania plików w formatach PDF, Word, LaTeX, konwertujacy je na tekst, który jest nastepnie przekazywany do serwisów modeli jezykowych.

Architektura opiera sie na komunikacji asynchronicznej oraz zastosowaniu API do integracji poszczególnych komponentów.

## Diagram komponentów

Poniżej, na figurze 5, przedstawiono diagram komponentów architektonicznych systemu *StudentReportLLMs*, który ilustruje interakcje miedzy poszczególnymi serwisami:

Diagram pokazuje strukture systemu oraz przepływ danych miedzy serwisami. Frontend komunikuje sie z backendem poprzez API, a backend zarzadza komunikacja z serwisami modeli jezykowych oraz baza danych.

## Diagram klas

Poniżej, na figurze 6, przedstawiono diagram klas systemu *StudentReportLLMs*, który ilustruje klasy i ich relacje:

## 4.2 Technologie i narzedzia

Do realizacji projektu *StudentReportLLMs* wykorzystano szereg nowoczesnych technologii i narzedzi:

- **Frontend**: React.js
- **Backend**: Node.js, Python (do integracji z modelami jezykowymi)
- **Baza danych**: MongoDB, QDRANT
- **Serwisy modeli jezykowych**: Python, TensorFlow, Hugging Face Transformers
- **System kontroli wersji**: Git, GitHub
- **Konteneryzacja**: Docker
- **Zarzadzanie zadaniami**: RabbitMQ

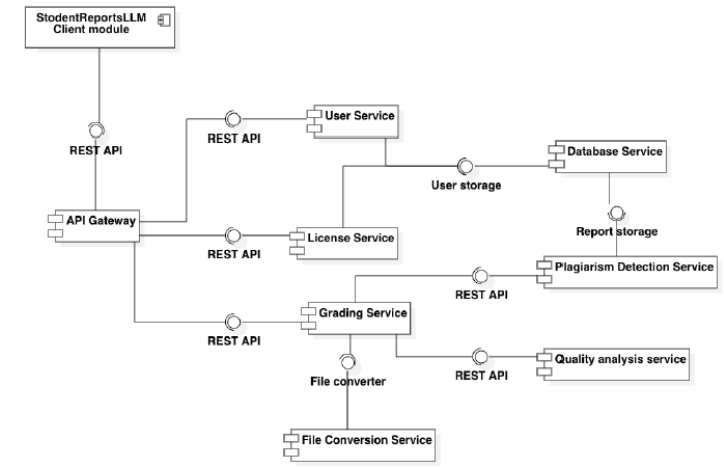


Figure 5: Diagram komponentów architektonicznych systemu *StudentReportLLMs*

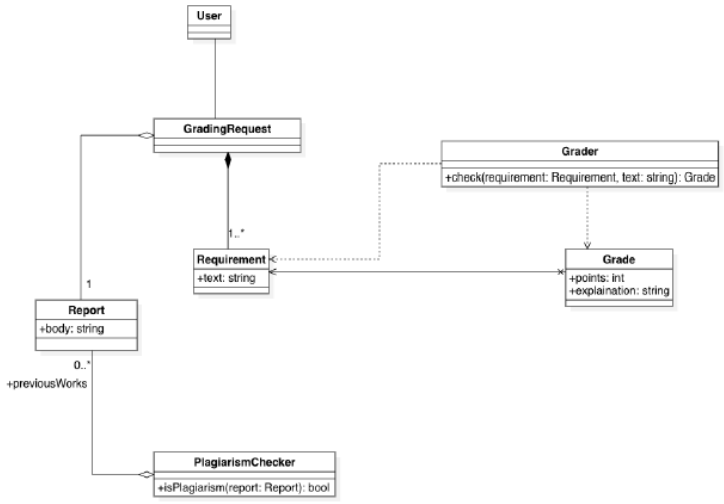


Figure 6: Diagram klas h systemu *StudentReportLLMs*

## 4.3 Bezpieczeństwo

Bezpieczeństwo systemu *StudentReportLLMs* zapewniono poprzez:

- Uwierzytelnianie i autoryzacje użytkowników przy użyciu JWT (JSON Web Token).
- Zabezpieczenia API przed atakami typu *SQL injection* oraz *cross-site scripting (XSS)*.
- Szyfrowanie danych przechowywanych w bazie danych oraz w transmisji miedzy serwisami.

Podjeto także środki ostrożności, aby zapewnić zgodność z przepisami o ochronie danych osobowych (GDPR).

# 5 Projekt Interfejsu

Projekt interfejsu użytkownika (UI) systemu *StudentReportLLMs* ma na celu zapewnienie intuicyjnej, efektywnej i przyjaznej dla użytkownika platformy, umożliwiajacej łatwa interakcje studentów oraz nauczycieli z systemem automatycznej oceny sprawozdań studenckich. Interfejs powinien być dostosowany do potrzeb różnych użytkowników, zapewniajac transparentność oceniania oraz dostep do szczegółowych informacji na temat wyników i zaleceń.

## 5.1 Założenia projektowe

Główne założenia projektowe interfejsu użytkownika obejmuja:

- Prostota obsługi: Interfejs powinien być intuicyjny i łatwy w nawigacji, co umożliwi szybkie i efektywne korzystanie zarówno studentom, jak i nauczycielom.
- Responsywność: Interfejs powinien być responsywny, dostosowujac sie automatycznie do różnych urzadzeń (komputery, tablety, telefony), co zapewni pełen dostep do funkcjonalności systemu niezależnie od używanego urzadzenia.
- Personalizacja: System powinien umożliwiać personalizacje doświadczenia użytkownika, na przykład poprzez dostosowanie preferencji wyświetlania danych lub ustawień powiadomień.
- Bezpieczeństwo: Interfejs powinien zapewniać wysoki poziom bezpieczeństwa danych osobowych i wyników oceniania, zgodnie z obowiazujacymi przepisami o ochronie danych.
- Wizualizacja danych: System powinien oferować czytelne i przejrzyste prezentowanie wyników oceniania, statystyk oraz raportów, aby użytkownicy mogli łatwo analizować i interpretować zgromadzone dane.

## 5.2 Elementy interfejsu

Interfejs użytkownika systemu *StudentReportLLMs* bedzie zawierać nastepujace kluczowe elementy:

- Panel logowania: Dedykowany dla studentów i nauczycieli, umożliwiajacy autentykacje i dostep do odpowiednich funkcji systemu.
- Panel administracyjny: Dla administratorów systemu, zapewniajacy zarzadzanie użytkownikami, ustawieniami systemu oraz raportami.
- Formularz wysyłania sprawozdań: Umożliwiajacy studentom przesyłanie swoich prac w różnych formatach (PDF, Word, LaTeX).
- Wyświetlanie wyników: Interaktywne wyświetlanie wyników oceniania, z możliwościa szczegółowego przegladu komentarzy i zaleceń.
- Panel konfiguracyjny kryteriów oceny: Dla nauczycieli, umożliwiajacy definiowanie specyficznych kryteriów oceniania dostosowanych do wymagań projektowych.
- Powiadomienia: System powinien obsługiwać powiadomienia w czasie rzeczywistym, informujace o statusie ocen, komunikatach od nauczycieli oraz innych istotnych wydarzeniach.

Projekt interfejsu bedzie stale ewoluował na podstawie feedbacku użytkowników oraz zmieniajacych sie potrzeb edukacyjnych, zapewniajac wysoka użyteczność i satysfakcje z jego użytkowania.

## 5.3 Specyfikacja API

Ta subsekcja określa punkty końcowe API dla systemu oceny LLM. API umożliwia użytkownikom przesyłanie sprawozdań studenckich, ustawianie kryteriów oceny oraz pobieranie wyników oceniania.

### Base URL

`http://StudentReportGrader/api/`

## Endpoints

### 1. Prześlij raport

**Endpoint:** `/upload`
**Method:** `POST`
**Description:** Uploads a student report for grading.
**Request:**

**Headers:**
`Content-Type: multipart/form-data`
**Body:**
`file`: The report file to be uploaded (PDF, Word, LaTeX).

**Response:**

**Success (200):**
{
"message": "Report uploaded successfully.",
"report_id": "¡report_id¿"
}
**Error (400/500):**
{
"error": "Description of the error."
}

### 2. Ustaw kryteria oceniania

**Endpoint:** `/criteria`
**Method:** `POST`
**Description:** Sets the grading criteria for evaluating reports.
**Request:**

**Headers:**
`Content-Type: application/json`
**Body:**
{
"criteria": [
{
"name": "Criterion 1",
"weight": 0.5
},
{
"name": "Criterion 2",
"weight": 0.5
}
]
}

**Response:**

**Success (200):**
{
"message": "Criteria set successfully."
}
**Error (400/500):**
{
"error": "Description of the error."
}

### 3. Pobierz wyniki oceniania

**Endpoint:** `/results/<report_id>`
**Method:** `GET`
**Description:** Retrieves the grading results for a specific report.

**Response:**

**Success (200):**

{
"report_id": "¡report_id¿",
"grading": {
"quality": 85,
"compliance": 90,
"originality": 95,
"overall": 90
}
}

**Error (400/500):**

{
"error": "Description of the error."
}

## 4. Kody błedów

Powszechne błedy:

- **400 Bad Request:** The request could not be understood or was missing required parameters.
- **401 Unauthorized:** Authentication failed or user does not have permissions for the requested operation.
- **404 Not Found:** The requested resource could not be found.
- **500 Internal Server Error:** An error occurred on the server.

# 6 Testowanie

Testowanie systemu *StudentReportLLMs* odgrywa kluczowa role w zapewnieniu jego niezawodności, funkcjonalności oraz zgodności z wymaganiami. Proces testowania obejmuje szeroki zakres działań majacych na celu sprawdzenie każdego aspektu systemu, zarówno pod katem jego działania technicznego, jak i użyteczności.

## 6.1 Cele testowania

Główne cele testowania systemu *StudentReportLLMs* obejmuja:

- Weryfikacja poprawności funkcjonalnej: Testowanie ma na celu sprawdzenie, czy wszystkie funkcjonalności systemu działaja zgodnie z założeniami i spełniaja oczekiwania użytkowników.
- Ocena wydajności: Testy wydajnościowe pomagaja określić, jak system radzi sobie z różnym obciażeniem, zapewniajac odpowiednia reaktywność i szybkość działania.
- Bezpieczeństwo: Testowanie bezpieczeństwa ma na celu zidentyfikowanie potencjalnych luk oraz zagrożeń zwiazanych z dostepem do danych osobowych i bezpieczeństwem systemu jako całości.
- Testy integracyjne: Sprawdzaja, czy poszczególne komponenty systemu współpracuja ze soba zgodnie z założeniami projektowymi, zapewniajac pełna funkcjonalność.
- Testy użyteczności: Ocena interfejsu użytkownika pod katem intuicyjności, łatwości obsługi oraz zgodności z oczekiwaniami użytkowników końcowych.
- Testy wydajności modeli jezykowych: Sprawdzenie, czy zastosowane duże modele jezykowe działaja zgodnie z oczekiwaniami pod wzgledem generacji tekstu, analizy treści i oceny oryginalności.

## 6.2 Metodologia testowania

Proces testowania systemu *StudentReportLLMs* bedzie oparty na zintegrowanych metodologiach testowych, obejmujacych:

- Testy jednostkowe: Sprawdzenie poprawności działania poszczególnych komponentów systemu oraz ich izolowanych funkcji.
- Testy integracyjne: Weryfikacja współpracy różnych komponentów systemu i ich interakcji, aby zapewnić spójność i pełna funkcjonalność systemu.
- Testy akceptacyjne: Przeprowadzenie testów z udziałem użytkowników końcowych, aby ocenić, czy system spełnia ich oczekiwania oraz czy jest łatwy w użyciu.
- Testy wydajnościowe: Ocena wydajności systemu pod wzgledem czasu odpowiedzi, obciażenia i stabilności działania w różnych warunkach.
- Testy bezpieczeństwa: Weryfikacja odporności systemu na ataki typu *SQL injection*, *cross-site scripting* oraz inne potencjalne zagrożenia.

## 6.3 Plan testów

Plan testów obejmuje szczegółowe scenariusze testowe dla każdej funkcjonalności systemu, uwzgledniajac różne przypadki użycia, typowe błedy użytkowników oraz nieoczekiwane sytuacje. Każdy test bedzie dokumentowany, a wyniki beda analizowane i raportowane, co umożliwi identyfikacje i szybkie rozwiazywanie napotkanych problemów.

1. **Sprawdzenie czasu przetwarzania sprawozdań studenckich przez model jezykowy**
   - **Opis przypadku testowego:** Czas przetwarzania sprawozdań studenckich nie powinien przekraczać 15-20 sekund.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Przesłanie sprawozdania studenckiego do systemu i oczekiwanie na przetworzenie.

(b) Rozpoczecie przetwarzania sprawozdania studenckiego przez model jezykowy oraz rozpoczecie odmierzania czasu.
(c) Weryfikacja czy czas przetwarzania jest akceptowalny.

- **Oczekiwane rezultaty:** System przetwarza sprawozdanie studenckie w akceptowalnym czasie.

2. **Sprawdzenie czy system poprawnie analizuje jakość treści sprawozdania**
   - **Opis przypadku testowego:** Jakość treści jest analizowana, a system generuje raport końcowy.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Przesłanie sprawozdania studenckiego do analizy jakości treści.
     (b) Uruchomienie funkcji analizy jakości treści.
     (c) Sprawdzenie wygenerowanego raportu oceny jakości.
   - **Oczekiwane rezultaty:** System prawidłowo analizuje jakość treści sprawozdania i generuje raport po zakończeniu procesu.
3. **Sprawdzenie czy system poprawnie ocenia zgodność z wytycznymi projektowymi**
   - **Opis przypadku testowego:** Zgodność sprawozdania z wytycznymi jest oceniana.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Przesłanie sprawozdania studenckiego do oceny zgodności z wytycznymi.
     (b) Zdefiniowanie specyficznych kryteriów oceny zgodnie z wytycznymi projektowymi.
     (c) Uruchomienie funkcji oceny zgodności z wytycznymi.
     (d) Sprawdzenie wyników działania funkcji wzgledem wytycznych.
   - **Oczekiwane rezultaty:** System prawidłowo ocenia zgodność sprawozdania z wytycznymi projektowymi.
4. **Sprawdzenie funkcji oceny oryginalności tekstu**
   - **Opis przypadku testowego:** Wykrywanie plagiatu.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Przesłanie sprawozdania studenckiego podejrzanego o plagiat.
     (b) Uruchomienie funkcji oceny oryginalności tekstu.
     (c) Sprawdzenie raportu oryginalności.
   - **Oczekiwane rezultaty:** System identyfikuje zduplikowane fragmenty i generuje wynik oceny oryginalności.
5. **Sprawdzenie kompatybilności systemu z różnymi formatami plików**
   - **Opis przypadku testowego:** Kompatybilność z formatami plików PDF, Word, LaTeX - rezultaty powinny być te same dla każdego formatu.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Przesłanie sprawozdań studenckich w formatach PDF, Word i LaTeX.
     (b) Uruchomienie funkcji analizy jakości i oceny zgodności dla każdego formatu pliku.
     (c) Porównanie wyników dla każdego formatu.
   - **Oczekiwane rezultaty:** System poprawnie odczytuje i przetwarza sprawozdania we wszystkich obsługiwanych formatach.
6. **Testowanie interfejsu użytkownika pod katem łatwości użytkowania**
   - **Opis przypadku testowego:** Intuicyjny interfejs użytkownika.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Nawigacja po interfejsie użytkownika w celu uzyskania dostepu do różnych funkcji.

(b) Formułowanie zapytań oceny za pomoca dostepnego interfejsu.

- **Oczekiwane rezultaty:** Interfejs użytkownika jest intuicyjny i umożliwia łatwa interakcje z systemem.

7. **Testowanie integracji systemu z modelami jezykowymi**

   - **Opis przypadku testowego:** Integracja z modelami jezykowymi bez potrzeby dodatkowego szkolenia.
   - **Kroki potrzebne do przeprowadzenia testu:**
     (a) Upewnienie sie, że modele jezykowe sa zintegrowane bez potrzeby dodatkowego szkolenia.
     (b) Uruchomienie analizy jakości i oceny zgodności za pomoca tych modeli.
   - **Oczekiwane rezultaty:** System skutecznie integruje i wykorzystuje modele jezykowe do analizy i oceny.

## 6.4 Ewaluacja i raportowanie

Po zakończeniu testów zostanie przeprowadzona ocena ich wyników oraz przygotowany raport z rekomendacjami dotyczacymi dalszych działań. Wszystkie ustalone nieprawidłowości beda korygowane, a system bedzie testowany ponownie w celu potwierdzenia poprawności wprowadzonych zmian.

# 7 Wdrożenie

Proces wdrożenia systemu *StudentReportLLMs* obejmuje szereg kluczowych działań majacych na celu zapewnienie płynnego uruchomienia i użytkowania systemu przez jego użytkowników końcowych.

## 7.1 Przygotowanie środowiska wdrożeniowego

Pierwszym krokiem w procesie wdrożenia jest przygotowanie odpowiedniego środowiska do uruchomienia systemu. Bedzie to obejmować:

- Instalacja i konfiguracja serwerów: Zapewnienie odpowiedniej infrastruktury serwerowej, która bedzie obsługiwać aplikacje oraz przechowywać dane.
- Konfiguracja baz danych: Utworzenie i skonfigurowanie odpowiednich baz danych potrzebnych do przechowywania danych użytkowników, sprawozdań studenckich oraz wyników ocen.
- Instalacja zewnetrznych zależności: Zapewnienie, że wszystkie zewnetrzne biblioteki, narzedzia i modele jezykowe, które sa niezbedne do działania systemu, sa poprawnie zainstalowane i skonfigurowane.

W celu przedstawienia procesu wdrożenia, poniższy rusunek na 7 przedstawia diagram wdrożenia.

## 7.2 Przygotowanie środowiska lokalnego

Przed przystapieniem do dalszych kroków, należy przygotować środowisko lokalne.

### Krok 1: Klonowanie repozytorium

Rozpocznij od sklonowania repozytorium z GitHub:

```
git clone https://github.com/magdalenpakula/yourproject.git
cd yourproject
```

### Krok 2: Utworzenie środowiska wirtualnego

Zaleca sie użycie środowiska wirtualnego do zarzadzania zależnościami. Możesz je utworzyć za pomoca virtualenv:

```
python -m venv venv
source venv/bin/activate   # Na Windows użyj ‘venv\Scripts\activate‘
```

### Krok 3: Instalacja zależności

Zainstaluj wymagane pakiety Pythona za pomoca pip:

```
pip install -r requirements.txt
```

### Krok 4: Konfiguracja MongoDB i Qdrant

- MongoDB: Upewnij sie, że MongoDB jest zainstalowane i uruchomione na Twoim lokalnym komputerze lub zdalnym serwerze. Instrukcje instalacji znajdziesz na oficjalnej stronie MongoDB.
- Qdrant: Zainstaluj i uruchom Qdrant za pomoca Dockera.

```
docker pull qdrant/qdrant
docker run -p 6333:6333 qdrant/qdrant
```

### Krok 5: Konfiguracja aplikacji

Utwórz plik .env w katalogu głównym i dodaj nastepujace konfiguracje:

```
MONGO_URI=mongodb://localhost:27017
QDRANT_HOST=localhost
QDRANT_PORT=6333
```

Dostosuj wartości w zależności od konfiguracji swojego środowiska.

### Krok 6: Uruchomienie aplikacji

Aby uruchomić aplikacje, wykonaj nastepujace polecenie:

```
python src/gui/__init__.py
```

### Krok 7: Uruchomienie testów

Aby upewnić sie, że wszystko jest poprawnie skonfigurowane, uruchom testy jednostkowe:

```
pytest src/tests/
```

### Dodatkowa konfiguracja (opcjonalna)

**Instalacja LaTeX (do przetwarzania plików LaTeX)**: Do przetwarzania plików LaTeX może być konieczna instalacja dystrybucji LaTeX.

## 7.3 Testowanie w środowisku produkcyjnym

Przed przejściem do pełnego wdrożenia, system bedzie poddany intensywnym testom w środowisku produkcyjnym. Celem jest upewnienie sie, że wszystkie funkcjonalności działaja zgodnie z oczekiwaniami oraz że system jest odporny na obciażenie i gotowy do użytku.

## 7.4 Szkolenie użytkowników

Kolejnym krokiem bedzie przeprowadzenie szkoleń dla użytkowników końcowych systemu, w tym dla wykładowców i studentów. Szkolenia te maja na celu zapoznanie użytkowników z interfejsem systemu, jego funkcjonalnościami oraz procedurami postepowania, takimi jak przesyłanie sprawozdań, formułowanie kryteriów oceny oraz interpretacja wyników ocen.

## 7.5 Pełne wdrożenie systemu

Gdy wszystkie powyższe kroki zostana zakończone pomyślnie, system *StudentReportLLMs* zostanie w pełni wdrożony do użytku. Proces ten bedzie monitorowany i wspierany przez zespół techniczny, który bedzie gotowy reagować na ewentualne problemy i pytania użytkowników.

## 7.6 Monitorowanie i utrzymanie

Po wdrożeniu systemu bedzie kontynuowane monitorowanie jego działania oraz regularne aktualizacje i utrzymywanie systemu. Celem jest zapewnienie wysokiej dostepności, bezpieczeństwa oraz efektywności działania systemu w dłuższym okresie.

# 8 Utrzymanie i wsparcie

Utrzymanie i wsparcie systemu *StudentReportLLMs* jest kluczowe dla zapewnienia jego stabilności, bezpieczeństwa oraz efektywności w długim okresie użytkowania. Proces utrzymania i wsparcia obejmuje szereg działań majacych na celu monitorowanie, utrzymanie oraz ewentualne rozwijanie systemu w odpowiedzi na potrzeby użytkowników.

## 8.1 Monitorowanie systemu

Podstawowym elementem utrzymania systemu jest ciagłe monitorowanie jego działania. Proces monitorowania powinien obejmować:

- Monitorowanie wydajności: Śledzenie wydajności systemu, w tym czasu odpowiedzi, zużycia zasobów oraz obciażenia serwera, aby zapewnić płynne działanie bez opóźnień.
- Monitorowanie bezpieczeństwa: Analiza logów w celu wczesnego wykrywania potencjalnych zagrożeń bezpieczeństwa, takich jak próby nieautoryzowanego dostepu czy ataki typu DDoS.
- Monitorowanie błedów: Zbieranie informacji o błedach systemowych oraz aplikacyjnych, aby szybko identyfikować i naprawiać ewentualne problemy.

## 8.2 Wsparcie użytkowników

System *StudentReportLLMs* powinien zapewniać efektywne wsparcie dla swoich użytkowników, w tym dla wykładowców oraz studentów. Wsparcie użytkowników może obejmować:

- Pomoc techniczna: Udzielanie szybkiej odpowiedzi na pytania techniczne oraz rozwiazywanie problemów zwiazanych z działaniem systemu.
- Wsparcie użytkowe: Szkolenie użytkowników w zakresie korzystania z systemu, w tym omówienie procedur, interpretacji wyników ocen oraz wyjaśnienie funkcjonalności.
- Obsługa zgłoszeń: Zapewnienie efektywnego procesu obsługi zgłoszeń użytkowników, w tym bieżacego informowania o postepie rozwiazania problemu.

## 8.3 Aktualizacje i rozwój systemu

Aby system *StudentReportLLMs* pozostał aktualny i efektywny, konieczne jest regularne wprowadzanie aktualizacji oraz rozwijanie jego funkcjonalności. Proces ten powinien być oparty na:

- Analizie feedbacku: Analiza opinii użytkowników oraz ich sugestii dotyczacych usprawnień i nowych funkcjonalności systemu.
- Planowaniu aktualizacji: Określenie harmonogramu i zakresu kolejnych aktualizacji systemu, aby zapewnić minimalne zakłócenia w jego działaniu.
- Rozwój funkcjonalności: Implementacja nowych funkcji oraz usprawnień zgodnie z wymaganiami użytkowników oraz rozwojem technologii.

## 8.4 Zarzadzanie dokumentacja

Aby zapewnić przejrzystość i dostepność informacji, niezbedne jest prowadzenie aktualnej dokumentacji systemu, która zawiera opisy funkcjonalności, procedur obsługi, historie zmian oraz plany rozwoju.

# 9 Zarzadzanie projektem

## 9.1 Harmonogram

| Name / Description | Priority | Size estimate (Points) | State | Target iteration |
|---|---|---|---|---|
| Określenie celów oraz założeń | 5 | 1 | Zrobione | 1 |
| Określenie kosztów z uwzglednieniem amortyzacji oraz wstepnego planu realizacji | 4 | 2 | Zrobione | 1 |
| Wyznaczenie głównych funkcji dla członków drużyny projektowej | 5 | 1 | W trakcie | 1 |
| Określenie wymagań funkcjonalnych oraz niefunkcjonalnych | 5 | 2 | Nie zrobione | 1 |
| Zdefiniowanie środowiska oraz kluczowych komponentów | 4 | 3 | Nie zrobione | 1 |
| Podzielenie projektu na zadania | 4 | 4 | W trakcie | 1 |
| Utworzenie harmonogramu z pewna niepewnościa czasowa | 4 | 4 | Zrobione | 1 |
| Przydzielenie zadań dla developerów | 3 | 2 | Nie zrobione | 1 |
| Głeboka analiza rynku wraz ze sprawdzeniem podobnych już rozwiazań | 4 | 8 | Nie zrobione | 2 |
| Wyłuskanie wad i zalet z aplikacji | 3 | 3 | Nie zrobione | 2 |
| Poszukiwania odpowiednich modeli jezykowych | 5 | 5 | Nie zrobione | 2 |
| Znalezienie danych testowych dla modelu, sprawdzenia dokładności | 4 | 4 | Nie zrobione | 2 |
| Zbudowanie interfejsu użytkownika GUI | 5 | 16 | Nie zrobione | 2 |
| Zaprogramowanie interfejsu dla plików wejściowych różnego rozszerzenia | 4 | 16 | Nie zrobione | 2 |
| Dodanie funkcjonalności manipulacji opcja sprawdzenia | 4 | 16 | Nie zrobione | 2 |
| Zaprogramowanie modułu sprawdzajacego (edycja modelu) | 5 | 32 | Nie zrobione | 2 |
| Testowanie oprogramowania | 4 | 16 | Nie zrobione | 2 |
| Sprawdzanie spełnionych założeń | 5 | 8 | Nie zrobione | 3 |
| Naprawienie potrzebnych niedociagnieć | 5 | 8 | Nie zrobione | 3 |
| Dostarczenie dokumentacji technicznej | 5 | 24 | Nie zrobione | 3 |

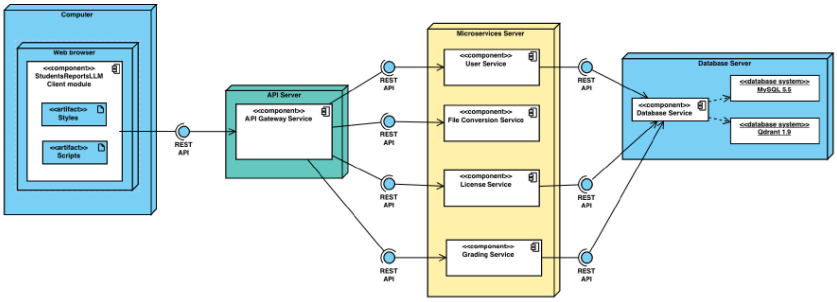


Figure 7: Diagram wdrożenia systemu *StudentReportLLMs*