

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

Magdalena Pakuła Jakub Pawlak Piotr Hynasiński
Artur Pietrzak Rafał Górniak

June 16, 2024

Contents

1	Wstęp	4
1.1	Cel dokumentu	4
1.2	Zakres projektu	4
2	Terminy i definicje	4
2.1	Opis potrzeby	5
2.2	Zidentyfikowani osbiorcy	6
3	Opis systemu	6
3.1	Zakres funkcjonalny	6
3.2	Zakres niefunkcjonalny	6
4	Wymagania	7
4.1	Wymagania funkcjonalne	7
4.2	Wymagania niefunkcjonalne	7
4.3	Opis przypadków użycia	9
4.3.1	Przypadek użycia 1: Przesyłanie raportu do systemu	9
4.3.2	Przypadek użycia 2: Automatyczna ocena treści raportów	9
4.3.3	Przypadek użycia 3: Odczyt i konwersja plików	9
4.3.4	Przypadek użycia 4: Personalizacja kryteriów oceny	9
4.3.5	Przypadek użycia 5: Ogólne korzystanie z sytemu przez użytkownika	9
5	Architektura Systemu	9
5.1	Architektura ogólna	9
5.2	Diagram komponentów	9
5.3	Diagram klas	14
5.4	Technologie i narzędzia	14
5.5	Bezpieczeństwo	14
6	Projekt Interfejsu	14
6.1	Założenia projektowe	15
6.2	Elementy interfejsu	15
7	Testowanie	16
7.1	Cele testowania	16
7.2	Metodologia testowania	16
7.3	Plan testów	17
7.4	Ewaluacja i raportowanie	18
8	Wdrożenie	19
8.1	Przygotowanie środowiska wdrożeniowego	19
8.2	Testowanie w środowisku produkcyjnym	19
8.3	Szkolenie użytkowników	19
8.4	Pełne wdrożenie systemu	19
8.5	Monitorowanie i utrzymanie	19

9	Utrzymanie i wsparcie	20
9.1	Monitorowanie systemu	20
9.2	Wsparcie użytkowników	20
9.3	Aktualizacje i rozwój systemu	20
9.4	Zarządzanie dokumentacją	21
10	Zarządzanie projektem	21
10.1	Harmonogram	21

1 Wstęp

1.1 Cel dokumentu

Celem niniejszego dokumentu jest przedstawienie szczegółowej dokumentacji projektowej dla systemu Automatyzacja Oceny Sprawozdań Studenckich przy Użyciu Dużych Modeli Językowych (StudentReportLLMs). Dokument ma na celu opisanie zakresu projektu, jego celów, oraz definicji związanych 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 implementację systemu wykorzystującego zaawansowane modele językowe 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:

- Analizę i projektowanie systemu, w tym zdefiniowanie głównych funkcji oraz wymagań technicznych.
- Wybór odpowiednich technologii i modeli językowych.
- Implementację systemu, w tym rozwój interfejsu użytkownika oraz integrację modeli językowych.
- Testowanie i ewaluację systemu.

2 Terminy i definicje

Termin	Definicja
LLMs (Large Language Models)	Zaawansowane modele językowe, takie jak GPT-4, BERT, T5, wykorzystywane do analizy i generowania tekstu.
Ocena jakości treści	Proces analizy sprawozdań pod kątem poprawności językowej, 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 umożliwiająca interakcje z systemem przez studentów i nauczycieli.
Information Retrieval	Proces wyszukiwania informacji w dużych zbiorach danych, który w projekcie będzie wspierany przez bazy danych i technologie zarządzania wektorami.
Duże modele językowe (LLMs)	Duże modele językowe to zaawansowane systemy sztucznej inteligencji, które zostały wytrenowane na ogromnych zbiorach danych tekstowych. LLMs są w stanie rozumieć i generować ludzką mowę w sposób, który imituje ludzkie pisanie.
Automatyzacja	Automatyzacja odnosi się do procesu zastępowania lub uzupełniania zadań wykonywanych przez ludzi zadaniami wykonywanymi przez maszyny lub programy komputerowe. Celem automatyzacji jest zwiększenie wydajności, obniżenie kosztów i minimalizacja błędów poprzez eliminację ludzkiego udziału lub zautomatyzowanie powtarzalnych zadań.
Tokenizacja	Proces podziału tekstu na mniejsze jednostki, zwane tokenami, co jest często używane w przetwarzaniu języka naturalnego, aby lepiej zrozumieć i analizować zawartość tekstu.
OCR (Optical Character Recognition)	Technologia pozwalająca na konwersję obrazów lub zeskanowanych dokumentów na edytowalny tekst, co jest istotne dla odczytu i analizy zawartości plików PDF.
Skalowalność systemu	Skalowalność systemu odnosi się do zdolności systemu do efektywnego dostosowywania się do wzrostu obciążenia lub zasobów. System jest uznawany za skalowalny, jeśli może rosnąć lub kurczyć się w zależności od potrzeb, zachowując przy tym swoją wydajność, niezawodność i funkcjonalność.
API (Application Programming Interface)	Interfejs programowania aplikacji, który umożliwia komunikację między różnymi komponentami oprogramowania, na przykład między systemem automatyzacji oceny a silnikami modeli językowych.
System odczytu i analizy plików	System zdolny do przetwarzania plików w formatach PDF, Word, LaTeX w celu analizy ich treści przez modele językowe.

2.1 Opis potrzeby

Głównym założeniem, a zarazem potrzebą jest wiarygodne, dokładne oraz z jak najmniejszym błędem sprawdzanie prac naukowych napisanych przez usługobiorców akademii, czyli studentów przy pomocy określonego narzędzia. Przy danych z góry kryteriach, program ma zachowywać się w należyty sposób. Potrzeba zaistniała ze względu na możliwą stronniczość wykładowcy. Oznacza to, że osoby, które chodziły na wykłady, krewni bądź znajomi, osoby faworyzowane przez różne względy mogłyby być oceniane wyżej w porównaniu do reszty, za taką samą pracę. Następnym możliwym zagrożeniem jest błąd ludzki. W danej chwili, przy braku stu-procentowej pewności osoba sprawdzająca może niesłusznie odjąć punkty, podczas gdy rozwiązania, fraza zaproponowana przez studenta jest zupełnie poprawna i vice versa. Następna coraz większa "plaga" jest plagiat oraz paragrafy generowane przez sztuczną inteligencję. To również powinno być

rozróżnione oraz wyłapane. Dlatego więc, aby każdy uczeń był równy wobec systemu sprawdzania, potrzebna jest technika, program to zapewniający. Ponadto każde konto uczelniane powinno mieć dostęp do tego rozwiązania, a zatem potrzebna jest autentykacja użytkownika dla studenta oraz nauczyciela, jako pośredniego administratora narzędzia.

2.2 Zidentyfikowani osbiorcy

Nazwa	Opis
Wykładowca (Nauczyciel aka- demicki)	Osoba pracująca w systemie oświaty w sektorze publicznym lub prywatnym w zależności od rodzaju miejsca pracy. Jednym z obowiązków zadawanie, a w konsekwencji sprawdzanie prac pisemnych, napisanych przez klientów uczelni lub kadry akademickiej w roli recenzenta.
Student (Uczeń)	Osoba realizująca ofertę uczelni, na której się znajduje w charakterze usługobiorcy. Píše zadany przez nauczyciela raport, a następnie według kryteriów oceniania dostaje należyta informację zwrotną w postaci oceny.

3 Opis systemu

Projekt *StudentRaportLLMs* polega na stworzeniu kompleksowego systemu opartego na zaawansowanych modelach językowych, który automatycznie oceni sprawozdania studenckie. System będzie przetwarzał tekst sprawozdań, analizując treść i sprawdzając zgodność z wytycznymi projektowymi lub innymi kryteriami. Na podstawie analizy, system będzie przyznawał oceny, dostosowane do konkretnego zadania lub przedmiotu. Aby zapewnić wiarygodność ocen, wyniki automatycznej oceny będą porównywane z wynikami ręcznie ocenionych sprawozdań, co pozwoli na dostosowanie parametrów oceniania i udoskonalenie procesu. Ponadto, system będzie integrowany z istniejącymi platformami edukacyjnymi, ułatwiając przesyłanie prac i wymianę informacji między studentami a nauczycielami. Projekt będzie elastyczny i przystosowany do zmian w technologiach oraz przepisach prawnych dotyczących ochrony danych osobowych.

3.1 Zakres funkcjonalny

System ma wykorzystywać duże modele językowe do oceny sprawozdań studenckich. Ocena ma dotyczyć dwóch kluczowych aspektów: jakości treści i zgodności z wytycznymi projektowymi (konkretnego zadania, którego dotyczy sprawozdanie). Dzięki temu możliwe będzie szybkie, obiektywne i dokładne ocenianie treści sprawozdań oraz ich zgodności z wytycznymi projektowymi. To rozwiązanie ma potencjał zrewolucjonizować proces oceniania, poprawiając jakość edukacji poprzez wprowadzenie bardziej obiektywnych i zgodnych z wytycznymi ocen. Wymagane będzie zintegrowanie i wykorzystanie zaawansowanych modeli językowych, 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.

3.2 Zakres niefunkcjonalny

Ważnym elementem będzie stworzenie intuicyjnego interfejsu użytkownika, który pozwoli na łatwe formułowanie zapytań dotyczących specyficznych kryteriów oceny dla ogółu

dostępnych prac, takich jak często popełniane błędy czy powtarzające się braki w sprawozdaniach. Kluczowymi elementami będą efektywność, precyzja oraz użyteczność systemu. Ważne będzie również stworzenie szczegółowej dokumentacji projektowej.

4 Wymagania

Poniżej znajdują się wszystkie wymagania funkcjonalne i нефункционаłne projektu:

4.1 Wymagania funkcjonalne

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

4.2 Wymagania нефункционаłne

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

4.3 Diagram przepływu danych

DFD są szczególnie przydatne do przedstawienia przepływu danych i procesów w ramach systemu. Poniżej na fig. ?? znajduje się diagram przedstawiający, jak przetwarzane są dane w systemie.

4.4 Diagram Business Process Model and Notation

BPMN idealnie nadaje się do modelowania procesów biznesowych. Poniżej na fig. ?? znajduje się diagram, który wskaże kluczowe procesy biznesowe związane z funkcjonalnością systemu.

4.5 Diagram SWOT

Diagram SWOT (Strengths, Weaknesses, Opportunities, Threats) pokazany niżej na fig. ?? pomoże zidentyfikować wewnętrzne mocne strony, słabości oraz zewnętrzne szanse i zagrożenia.

4.6 Opis przypadków użycia

4.6.1 Przypadek użycia 1: Przesyłanie raportu do systemu

4.6.2 Przypadek użycia 2: Automatyczna ocena treści raportów

4.6.3 Przypadek użycia 3: Odczyt i konwersja plików

4.6.4 Przypadek użycia 4: Personalizacja kryteriów oceny

4.6.5 Przypadek użycia 5: Ogólne korzystanie z systemu przez użytkownika

W tym przypadku przedstawiony będzie diagram przypadku użycia na figurze 1.

5 Architektura Systemu

W tej sekcji przedstawiono architekturę systemu *StudentReportLLMs*, który ma na celu automatyzację oceny sprawozdań studenckich przy użyciu dużych modeli językowych.

5.1 Architektura ogólna

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

- **Frontend:** Interfejs użytkownika (UI), dostępny jako aplikacja webowa umożliwiająca studentom przysyłanie swoich sprawozdań oraz przeglądanie ocen.
- **Backend:** Centralny serwis zarządzający, który obsługuje logikę biznesową, integracje z modelami językowymi oraz zarządzanie zadaniami oceny.
- **Serwisy modeli językowych:** Oddzielne serwisy odpowiedzialne za integracje z różnymi modelami językowymi (np. GPT-4, BERT), które wykonują analizę i ocenę tekstu.

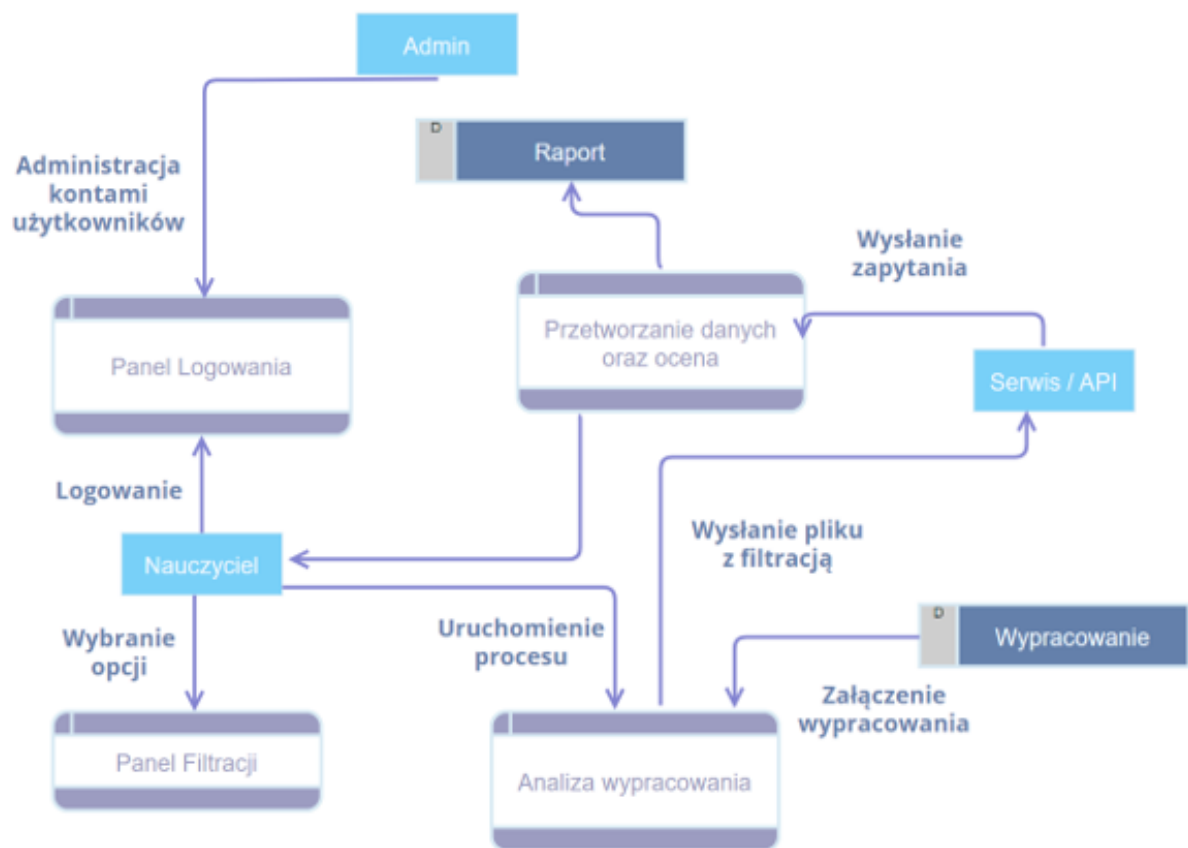


Figure 1: Diagram przepływu danych w systemie *StudentReportLLMs*

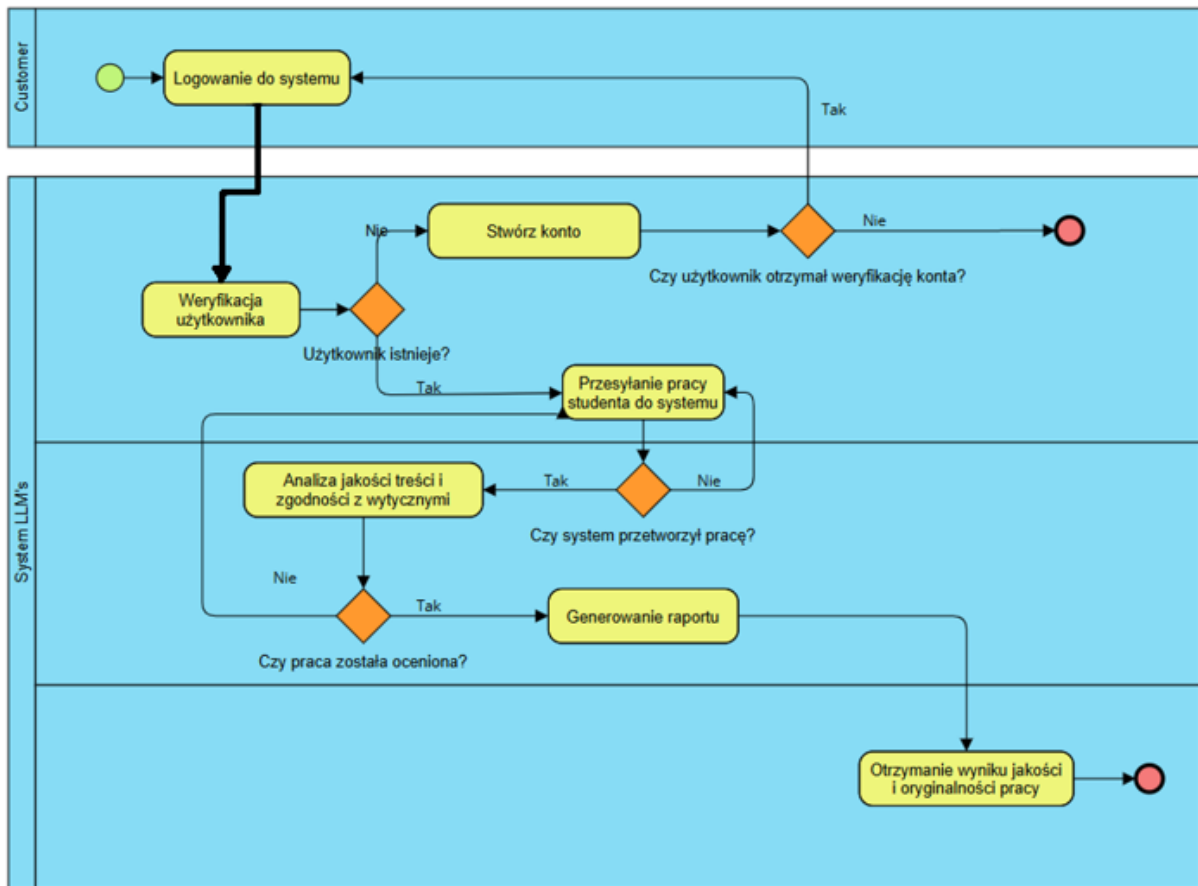


Figure 2: Diagram Business Process Model and Notation systemu *StudentReportLLMs*

<p>STRENGTHS</p> <ul style="list-style-type: none"> - Access to advanced language models (LLaMA-2, Mistral, BERT, T5) - Efficient file processing capabilities for various document formats (PDF, Word, LaTeX) for versatility and usability - Customizable evaluation criteria adaptable to diverse project requirements enhance flexibility and user satisfaction - Focus on originality detection helps prevent plagiarism and promotes unique student work 	<p>WEAKNESSES</p> <ul style="list-style-type: none"> - Dependence on advanced language models may lead to limited availability / increased costs - Potential biases or limitations in the performance of language models could affect the accuracy
<p>OPPORTUNITIES</p> <ul style="list-style-type: none"> - Close contact with potential customers (professors of university) - Continued advancements in natural language processing (NLP) technology may offer opportunities for further improvement - Expansion into other educational domains or markets beyond student report evaluation could diversify revenue streams 	<p>THREATS</p> <ul style="list-style-type: none"> - Rapid advancements in NLP technology may require frequent updates to maintain competitiveness - Competition from other automated evaluation solutions or traditional manual evaluation methods - Regulatory changes or legal issues related to data privacy and intellectual property rights - Negative perceptions or resistance from educators or stakeholders regarding the automation of evaluation processes - Vulnerability to cyber threats or data breaches could undermine trust and credibility

Figure 3: Diagram SWOT systemu *StudentReportLLMs*

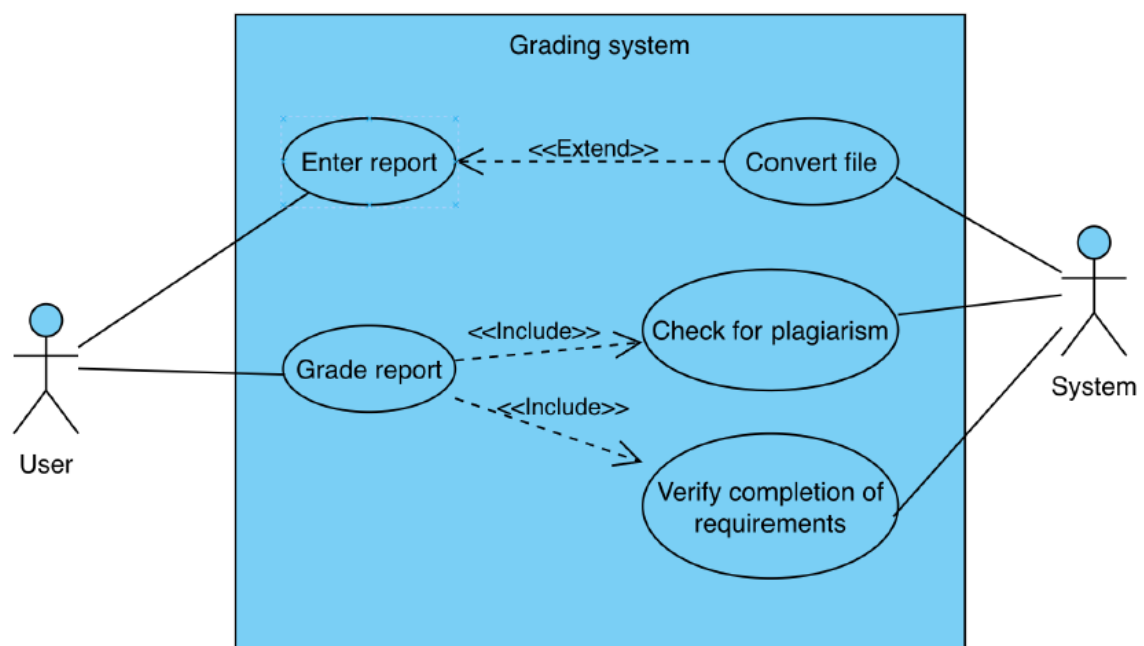


Figure 4: Diagram przypadku użycia *StudentReportLLMs*

Table 1: Przypadek użycia: Przesyłanie raportu do systemu

Typ	Operacyjny
Użytkownicy	Nauczyciel akademicki
Warunki wstępne	<ul style="list-style-type: none"> • Nauczyciel musi być zalogowany do systemu. • Raport studencki musi być w odpowiednim formacie (Word, PDF, LaTeX).
Typowa kolejność kroków	<ol style="list-style-type: none"> 1. Nauczyciel loguje się do systemu "StudentReportLLMs". 2. Nauczyciel wybiera opcje przesyłania nowego raportu. 3. System wyświetla formularz do przesyłania pliku. 4. Nauczyciel wypełnia formularz, dodając odpowiedni plik raportu oraz wszelkie dodatkowe informacje wymagane przez system (np. kategoria, autor, przedmiot). 5. Nauczyciel przesyła formularz, a system zapisuje raport do bazy danych i wyświetla potwierdzenie poprawnego przesłania.
Warunki końcowe	<ul style="list-style-type: none"> • Raport jest zapisany w systemie i oczekuje na automatyczną ocenę. • Nauczyciel otrzymuje powiadomienie o poprawnym przesłaniu raportu.
Alternatywna kolejność kroków	<ol style="list-style-type: none"> 1. Nauczyciel loguje się do systemu "StudentReportLLMs". 2. Nauczyciel wybiera opcje przesyłania nowego raportu. 3. System wyświetla formularz do przesyłania pliku. 4. Nauczyciel próbuje przesłać plik w nieprawidłowym formacie. 5. System wyświetla komunikat o błędzie i prosi o poprawienie formatu pliku. 6. Nauczyciel poprawia format pliku i ponownie przesyła formularz. 7. System zapisuje raport do bazy danych i wyświetla potwierdzenie poprawnego przesłania.
Komentarze i uwagi	<ul style="list-style-type: none"> • System powinien obsługiwać różne formaty plików i weryfikować ich poprawność przed przesłaniem. • System powinien być intuicyjny i łatwy w użyciu, aby nauczyciele mogli szybko

Table 2: Przypadek użycia: Automatyczna ocena treści raportów

Typ	Operacyjny
Użytkownicy	Nauczyciel akademicki
Warunki wstępne	<ul style="list-style-type: none"> • Nauczyciel musi być zalogowany do systemu. • Raport studencki musi być w odpowiednim formacie (Word, PDF, LaTeX). • System musi mieć zdefiniowane kryteria oceny.
Typowa kolejność kroków	<ol style="list-style-type: none"> 1. System wykrywa nowy przesłany raport oczekujący na ocene. 2. System analizuje treść raportu przy użyciu zaawansowanych modeli językowych. 3. System sprawdza zgodność treści raportu z wytycznymi projektowymi i zdefiniowanymi kryteriami oceny. 4. System generuje ocene raportu oraz szczegółowy feedback dotyczący mocnych i słabych stron pracy. 5. System zapisuje ocene i feedback w bazie danych oraz powiadamia nauczyciela o zakończonej ocenie.
Warunki końcowe	<ul style="list-style-type: none"> • Nauczyciel musi być zalogowany do systemu. • Raport zostaje oceniony i zapisany w systemie. • Nauczyciel otrzymuje powiadomienie o zakończonej ocenie raportu.
Alternatywna kolejność kroków	<ol style="list-style-type: none"> 1. System wykrywa nowy przesłany raport oczekujący na ocene. 2. System analizuje treść raportu przy użyciu zaawansowanych modeli językowych. 3. System napotyka problem z analiza (np. nieczytelny tekst, błędy w formacie). 4. System powiadamia nauczyciela o problemie i prosi o weryfikację raportu. 5. Nauczyciel weryfikuje i poprawia raport, a następnie ponownie przesyła do systemu. 6. System ponownie analizuje treść raportu, sprawdza zgodność z wytycznymi i generuje ocene oraz feedback.
Komentarze i uwagi	<ul style="list-style-type: none"> • System powinien być w stanie obsługiwać różne problemy z analiza i

Table 3: Przypadek użycia: Odczyt i konwersja plików

Typ	Operacyjny
Użytkownicy	Nauczyciel akademicki
Warunki wstępne	<ul style="list-style-type: none"> • Nauczyciel musi być zalogowany do systemu. • Plik raportu musi być przesłany do systemu. • System musi obsługiwać format pliku (Word, PDF, LaTeX).
Typowa kolejność kroków	<ol style="list-style-type: none"> 1. Nauczyciel przesyła raport do systemu w jednym z obsługiwanych formatów (Word, PDF, LaTeX). 2. System rozpoznaje format pliku i przystępuje do jego odczytu. 3. System przetwarza plik, konwertując jego zawartość na format zrozumiały dla modeli językowych. 4. System przeprowadza analizę zawartości przetworzonego pliku, identyfikując istotne sekcje i elementy raportu. 5. System zapisuje przetworzoną i skonwertowaną wersję pliku w bazie danych oraz informuje nauczyciela o zakończeniu procesu konwersji.
Warunki końcowe	<ul style="list-style-type: none"> • Plik raportu jest poprawnie odczytany i skonwertowany. • System zapisuje skonwertowaną wersję pliku w bazie danych. • Nauczyciel otrzymuje powiadomienie o zakończeniu procesu konwersji.
Alternatywna kolejność kroków	<ol style="list-style-type: none"> 1. Nauczyciel przesyła raport do systemu w jednym z obsługiwanych formatów (Word, PDF, LaTeX). 2. System rozpoznaje format pliku i przystępuje do jego odczytu. 3. System napotyka problem z odczytem pliku (np. uszkodzony plik, nieczytelny format). 4. System powiadamia nauczyciela o problemie i prosi o weryfikację pliku. 5. Nauczyciel poprawia plik i ponownie przesyła do systemu. 6. System ponownie przetwarza plik, konwertując jego zawartość na format zrozumiały dla modeli językowych.
Komentarze i uwagi	<ul style="list-style-type: none"> • System powinien być w stanie rozpoznać i obsłużyć różne formaty plików.

Table 4: Przypadek użycia: Personalizacja kryteriów oceny

Typ	Operacyjny
Użytkownicy	Nauczyciel akademicki
Warunki wstępne	<ul style="list-style-type: none"> • Nauczyciel musi być zalogowany do systemu.
Typowa kolejność kroków	<ol style="list-style-type: none"> 1. Nauczyciel loguje się do systemu "StudentReportLLMs". 2. Przechodzi do sekcji ustawień ocen. 3. Modyfikuje istniejące kryteria oceny lub dodaje nowe według własnych preferencji. Zapisuje wprowadzone zmiany. 4. System automatycznie uwzględni nowe kryteria podczas oceny kolejnych raportów studenckich.
Warunki końcowe	<ul style="list-style-type: none"> • Zmodyfikowane kryteria oceny są zapisane i uwzględniane podczas kolejnych ocen raportów studenckich. • Nauczyciel może dostosować proces oceniania do swoich potrzeb lub wymagań projektowych.
Alternatywna kolejność kroków	<ol style="list-style-type: none"> 1. Nauczyciel loguje się do systemu "StudentReportLLMs". 2. Przechodzi do sekcji ustawień ocen. 3. Nauczyciel próbuje dodać nowe kryteria oceny, ale napotyka problem z zapisaniem zmian. 4. System wyświetla komunikat o błędzie i prosi nauczyciela o ponowne zapisanie ustawień. 5. Nauczyciel ponownie próbuje zapisać zmiany, a system potwierdza poprawne zastosowanie nowych kryteriów oceny.
Komentarze i uwagi	<ul style="list-style-type: none"> • Interfejs użytkownika systemu powinien być intuicyjny i łatwy w obsłudze, aby nauczyciel mógł bezproblemowo dostosować kryteria oceny. • System powinien zapewnić odpowiednie komunikaty o błędach w przypadku problemów podczas zapisywania zmian w ustawieniach ocen. • Warto umożliwić nauczycielowi podgląd zmian przed ich zatwierdzeniem, aby uniknąć niezamierzonych modyfikacji kryteriów oceny.

- **Baza danych:** Centralna baza danych przechowująca skonwertowane sprawozdania oraz baza danych do analizy i przetwarzania sprawozdań.
- **System odczytu i analizy plików:** Serwis do przetwarzania plików w formatach PDF, Word, LaTeX, konwertujący je na tekst, który jest następnie przekazywany do serwisów modeli językowych.

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

5.2 Diagram komponentów

Poniżej, na figurze 2, przedstawiono diagram komponentów architektonicznych systemu *StudentReportLLMs*, który ilustruje interakcje między poszczególnymi serwisami:

Diagram pokazuje strukturę systemu oraz przepływ danych między serwisami. Frontend komunikuje się z backendem poprzez API, a backend zarządza komunikacją z serwisami modeli językowych oraz bazą danych.

5.3 Diagram klas

Poniżej, na figurze 3, przedstawiono diagram klas systemu *StudentReportLLMs*, który ilustruje klasy i ich relacje:

5.4 Technologie i narzędzia

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

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

5.5 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)*.

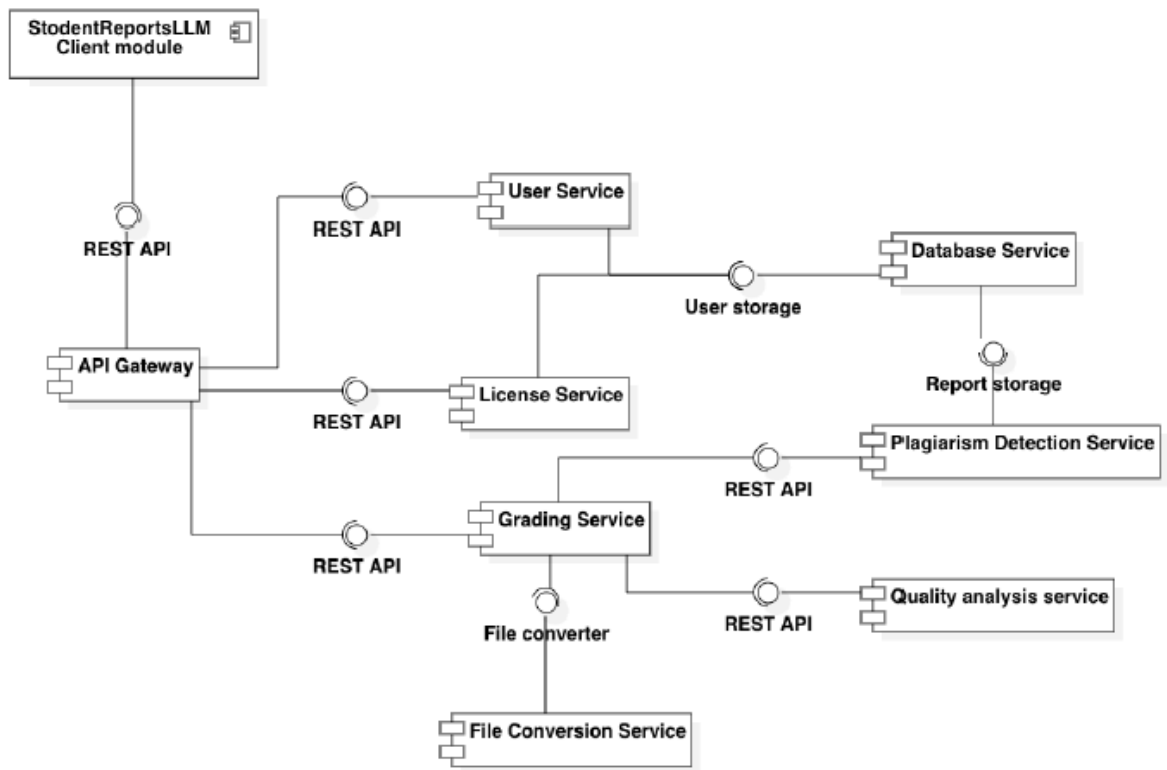


Figure 5: Diagram komponentów architektonicznych systemu *StudentReportLLMs*

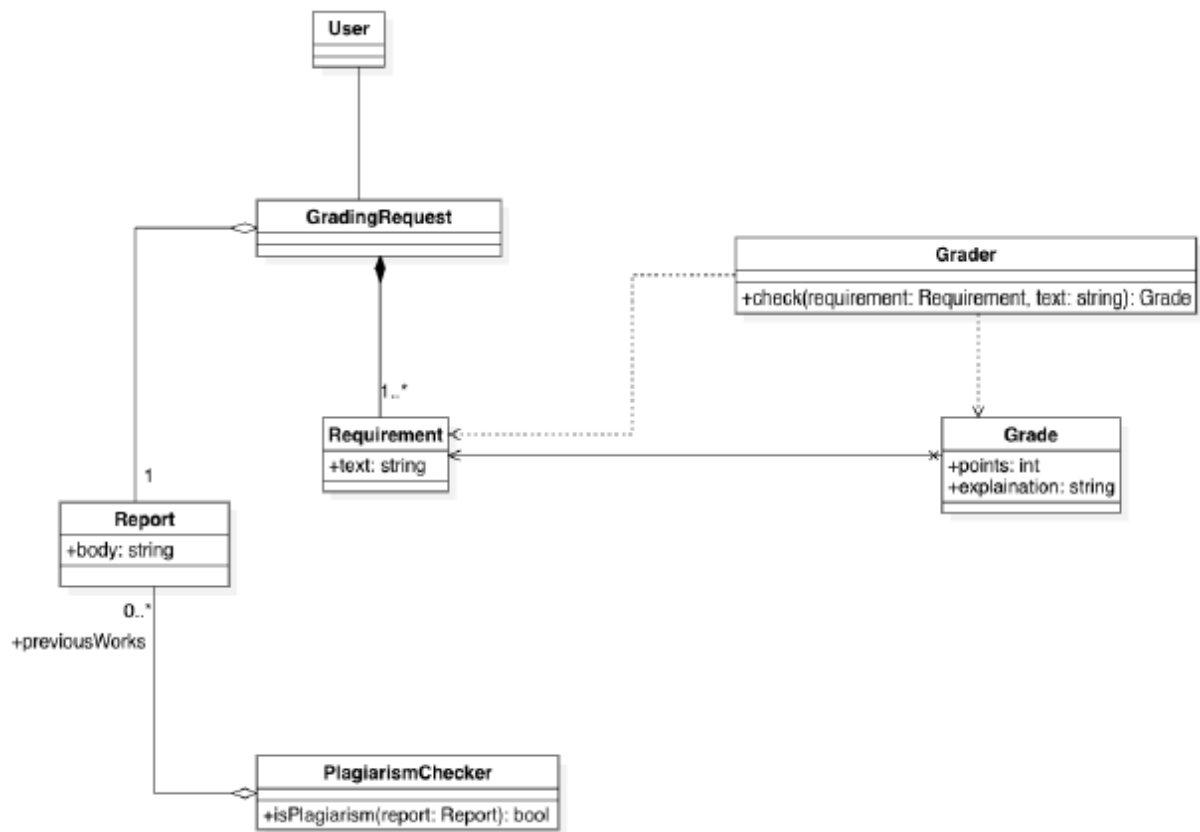


Figure 6: Diagram klas h systemu *StudentReportLLMs*

- Szyfrowanie danych przechowywanych w bazie danych oraz w transmisji między serwisami.

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

6 Projekt Interfejsu

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

6.1 Założenia projektowe

Główne założenia projektowe interfejsu użytkownika obejmują:

- 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, dostosowując się automatycznie do różnych urządzeń (komputery, tablety, telefony), co zapewni pełen dostęp do funkcjonalności systemu niezależnie od używanego urządzenia.
- Personalizacja: System powinien umożliwiać personalizację 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 obowiązującymi 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.

6.2 Elementy interfejsu

Interfejs użytkownika systemu *StudentReportLLMs* będzie zawierać następujące kluczowe elementy:

- Panel logowania: Dedykowany dla studentów i nauczycieli, umożliwiający autentykację i dostęp do odpowiednich funkcji systemu.
- Panel administracyjny: Dla administratorów systemu, zapewniający zarządzanie użytkownikami, ustawieniami systemu oraz raportami.
- Formularz wysyłania sprawozdań: Umożliwiający studentom przysyłanie swoich prac w różnych formatach (PDF, Word, LaTeX).

- Wyświetlanie wyników: Interaktywne wyświetlanie wyników oceniania, z możliwością szczegółowego przeglądu komentarzy i zaleceń.
- Panel konfiguracyjny kryteriów oceny: Dla nauczycieli, umożliwiający definiowanie specyficznych kryteriów oceniania dostosowanych do wymagań projektowych.
- Powiadomienia: System powinien obsługiwać powiadomienia w czasie rzeczywistym, informujące o statusie ocen, komunikatach od nauczycieli oraz innych istotnych wydarzeniach.

Projekt interfejsu będzie stale ewoluował na podstawie feedbacku użytkowników oraz zmieniających się potrzeb edukacyjnych, zapewniając wysoka użyteczność i satysfakcję z jego użytkowania.

7 Testowanie

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

7.1 Cele testowania

Główne cele testowania systemu *StudentReportLLMs* obejmują:

- Weryfikacja poprawności funkcjonalnej: Testowanie ma na celu sprawdzenie, czy wszystkie funkcjonalności systemu działają zgodnie z założeniami i spełniają oczekiwania użytkowników.
- Ocena wydajności: Testy wydajnościowe pomagają określić, jak system radzi sobie z różnym obciążeniem, zapewniając odpowiednią reaktywność i szybkość działania.
- Bezpieczeństwo: Testowanie bezpieczeństwa ma na celu zidentyfikowanie potencjalnych luk oraz zagrożeń związanych z dostępem do danych osobowych i bezpieczeństwem systemu jako całości.
- Testy integracyjne: Sprawdzają, czy poszczególne komponenty systemu współpracują ze sobą zgodnie z założeniami projektowymi, zapewniając pełną funkcjonalność.
- Testy użyteczności: Ocena interfejsu użytkownika pod kątem intuicyjności, łatwości obsługi oraz zgodności z oczekiwaniami użytkowników końcowych.
- Testy wydajności modeli językowych: Sprawdzenie, czy zastosowane duże modele językowe działają zgodnie z oczekiwaniami pod względem generacji tekstu, analizy treści i oceny oryginalności.

7.2 Metodologia testowania

Proces testowania systemu *StudentReportLLMs* będzie oparty na zintegrowanych metodologiach testowych, obejmujących:

- 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 pełną 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 względem czasu odpowiedzi, obciążenia i stabilności działania w różnych warunkach.
- Testy bezpieczeństwa: Weryfikacja odporności systemu na ataki typu *SQL injection*, *cross-site scripting* oraz inne potencjalne zagrożenia.

7.3 Plan testów

Plan testów obejmuje szczegółowe scenariusze testowe dla każdej funkcjonalności systemu, uwzględniając różne przypadki użycia, typowe błędy użytkowników oraz nieoczekiwane sytuacje. Każdy test będzie dokumentowany, a wyniki będą analizowane i raportowane, co umożliwi identyfikację i szybkie rozwiązywanie napotkanych problemów.

1. Sprawdzenie czasu przetwarzania sprawozdań studenckich przez model językowy

- **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) Rozpoczęcie przetwarzania sprawozdania studenckiego przez model językowy oraz rozpoczęcie odmierzenia 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 względem 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 kątem łatwości użytkowania**
- **Opis przypadku testowego:** Intuicyjny interfejs użytkownika.
 - **Kroki potrzebne do przeprowadzenia testu:**
 - (a) Nawigacja po interfejsie użytkownika w celu uzyskania dostępu do różnych funkcji.
 - (b) Formułowanie zapytań oceny za pomocą dostępnego interfejsu.

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

7. Testowanie integracji systemu z modelami językowymi

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

7.4 Ewaluacja i raportowanie

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

8 Wdrożenie

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

8.1 Przygotowanie środowiska wdrożeniowego

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

- Instalacja i konfiguracja serwerów: Zapewnienie odpowiedniej infrastruktury serwerowej, która będzie 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 zewnętrznych zależności: Zapewnienie, że wszystkie zewnętrzne biblioteki, narzędzia i modele językowe, które są niezbędne do działania systemu, są poprawnie zainstalowane i skonfigurowane.

W celu przedstawienia procesu wdrożenia, poniższy rysunek na ?? przedstawia diagram wdrożenia.

8.2 Testowanie w środowisku produkcyjnym

Przed przejściem do pełnego wdrożenia, system będzie poddany intensywnym testom w środowisku produkcyjnym. Celem jest upewnienie się, że wszystkie funkcjonalności działają zgodnie z oczekiwaniami oraz że system jest odporny na obciążenie i gotowy do użytku.

8.3 Szkolenie użytkowników

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

8.4 Pełne wdrożenie systemu

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

8.5 Monitorowanie i utrzymanie

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

9 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 mających na celu monitorowanie, utrzymanie oraz ewentualne rozwijanie systemu w odpowiedzi na potrzeby użytkowników.

9.1 Monitorowanie systemu

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

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

9.2 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 rozwiązywanie problemów związanych 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 bieżącego informowania o postępie rozwiązania problemu.

9.3 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 dotyczących 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.

9.4 Zarządzanie dokumentacją

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

10 Zarządzanie projektem

10.1 Harmonogram

Name / Description	Priority	Size estimate (Points)	State	Target iteration
Określenie celów oraz założeń	5	1	Zrobione	1
Określenie kosztów z uwzględnieniem amortyzacji oraz wstępnego 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 pewną niepewnością czasową	4	4	Zrobione	1
Przydzielenie zadań dla developerów	3	2	Nie zrobione	1
Głęboka analiza rynku wraz ze sprawdzeniem podobnych już rozwiązań	4	8	Nie zrobione	2
Wyłuskanie wad i zalet z aplikacji	3	3	Nie zrobione	2
Poszukiwania odpowiednich modeli językowych	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 opcją sprawdzenia	4	16	Nie zrobione	2
Zaprogramowanie modułu sprawdzającego (edycja modelu)	5	32	Nie zrobione	2
Testowanie oprogramowania	4	16	Nie zrobione	2
Sprawdzanie spełnionych założeń	5	8	Nie zrobione	3
Naprawienie potrzebnych niedociągnień	5	8	Nie zrobione	3
Dostarczenie dokumentacji technicznej	5	24	Nie zrobione	3

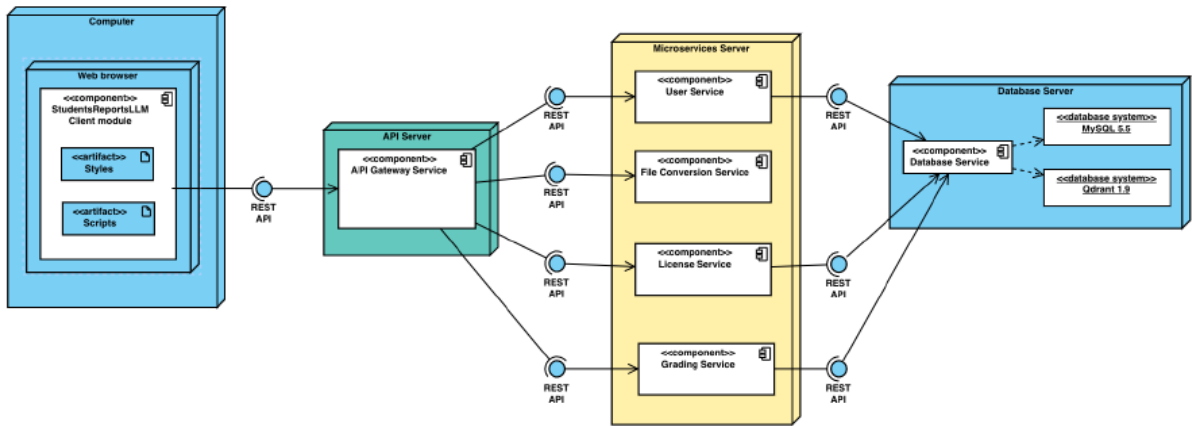


Figure 7: Diagram wdrożenia systemu *StudentReportLLMs*