

Система антиплагиата (КПО ДЗ-3)

Цель: Реализация микросервисной системы для проверки студенческих работ на плагиат с использованием Docker, REST API и принципов распределённой архитектуры.

⌚ Соответствие критериям задания

№	Критерий	Реализация	Макс. балл
1	Основные требования <ul style="list-style-type: none">— Отправка работы— Сохранение в СУБД + файл— Анализ и отчёт— API для получения отчёта	<input checked="" type="checkbox"/> Полностью реализовано через ApiGateway → MetadataService + FileStoringService → FileAnalysisService <ul style="list-style-type: none">— POST /api/submit-work— GET /api/works/{id}/reports	2
2	Микросервисная архитектура <ul style="list-style-type: none">— ≥2 бизнес-микросервиса— Обработка ошибок	<input checked="" type="checkbox"/> 3 бизнес-микросервиса: <ol style="list-style-type: none">1. MetadataService (PostgreSQL)2. FileStoringService (файловая система)3. FileAnalysisService (анализ текста, хеширование, облако слов) <input checked="" type="checkbox"/> Глобальный обработчик ошибок в ApiGateway (<code>UseExceptionHandler</code>)	2
3	Контейнеризация <ul style="list-style-type: none">— Dockerfile для каждого сервиса— docker-compose.yml— Запуск через <code>docker compose up</code>	<input checked="" type="checkbox"/> Все сервисы имеют многоступенчатую сборку <input checked="" type="checkbox"/> HEALTHCHECK в каждом Dockerfile <input checked="" type="checkbox"/> Единый <code>docker-compose.yml</code> <input checked="" type="checkbox"/> Команда <code>docker compose up --build</code> разворачивает всю систему	2
4	Swagger / Postman <ul style="list-style-type: none">— Документация всех API	<input checked="" type="checkbox"/> Swagger UI доступен по http://localhost:8080 <ul style="list-style-type: none">— Документированы все эндпоинты:<ul style="list-style-type: none">• POST /api/submit-work• GET /api/works/{workId}/reports— Формат совместим с Postman (REST, JSON)	1
5	Качество кода и документация <ul style="list-style-type: none">а) Чистый, модульный кодб) Описание архитектуры и сценариев	<input checked="" type="checkbox"/> Код: record-типы, dependency injection, явная маршрутизация, Serilog <input checked="" type="checkbox"/> Этот README содержит описание архитектуры и пользовательские сценарии (см. ниже)	2

№	Критерий	Реализация	Макс. балл
6	Бонус: облако слов	<input checked="" type="checkbox"/> Интеграция с quickchart.io <input checked="" type="checkbox"/> Ссылка возвращается в отчёте <input checked="" type="checkbox"/> Возможность открыть в браузере из ConsoleClient	1
ИТОГО		10 / 10	<input checked="" type="checkbox"/>

Архитектура системы

Система состоит из следующих компонентов:

1. API Gateway (api-gateway:8080)

- Единая точка входа.
- Агрегирует вызовы к микросервисам.
- Обеспечивает маршрутизацию, обработку ошибок и документацию (Swagger).

2. Metadata Service (metadata-service:8081)

- Хранит метаданные работ в PostgreSQL:
 - WorkId , StudentId , AssignmentId
 - FileId , ReportId , TextHash , SubmittedAt
- Предоставляет REST API для CRUD-операций.

3. File Storing Service (file-storing:8082)

- Хранит **файлы работ** на диске (volume file_storage).
- Не обрабатывает содержимое — только сохранение и отдача.

4. File Analysis Service (file-analysis:8083)

- Скачивает файл по FileId .
- Извлекает текст (.txt / .docx).
- Нормализует и вычисляет **SHA256 хеш**.
- Ищет совпадения по хешу в MetadataService .
- Генерирует **облако слов** через https://quickchart.io/wordcloud .
- Сохраняет отчёт в локальном хранилище (volume report_storage).
- Обновляет ReportId и TextHash в MetadataService .

5. Console Client

- Интерактивный клиент на Spectre.Console .

- Поддерживает:
 - Отправку файлов
 - Получение отчётов
 - Тестирование API
 - Полный end-to-end тест
 - Проверку состояния сервисов
-

Пользовательские сценарии

Сценарий 1: Отправка работы

1. Пользователь выбирает файл в `ConsoleClient`.
2. `ConsoleClient` → `ApiGateway` (`POST /api/submit-work`).
3. `ApiGateway` :
 - Вызывает `FileStoringService` → получает `FileId`.
 - Вызывает `MetadataService` → создаёт запись → получает `WorkId`.
 - Вызывает `FileAnalysisService` → запускает анализ (`WorkId` , `FileId`).
4. `FileAnalysisService` :
 - Скачивает файл.
 - Анализирует текст.
 - Сохраняет отчёт → возвращает `ReportId`.
 - Обновляет `ReportId` и `TextHash` в `MetadataService`.
5. `ApiGateway` возвращает `WorkId` , `FileId` , `AnalysisStarted`.

Сценарий 2: Получение отчёта

1. Пользователь запрашивает отчёт по `WorkId`.
2. `ConsoleClient` → `ApiGateway` (`GET /api/works/{WorkId}/reports`).
3. `ApiGateway` :
 - Запрашивает `WorkMetadata` из `MetadataService`.
 - Запрашивает `AnalysisReport` из `FileAnalysisService` по `ReportId`.
4. Возвращает агрегированный JSON с полной информацией.

Сценарий 3: Обнаружение плагиата

1. При анализе `FileAnalysisService` вычисляет хеш текста.
 2. Запрашивает `MetadataService` (`GET /submissions/by-hash/{hash}`).
 3. Если найдены работы другого студента с тем же хешем → `plagiarism = true`.
 4. Добавляет найденные работы в `plagiarismEvidence`.
-

Запуск системы

В MetadataService выполните:

```
dotnet ef database update
```

```
# Сборка и запуск
docker compose up --build

# Доступные сервисы
http://localhost:8080          # API Gateway + Swagger UI
http://localhost:8081/health      # Metadata Service
http://localhost:8082/health      # File Storing Service
http://localhost:8083/health      # File Analysis Service
```

Swagger UI: `http://localhost:8080`

Postman: импортируйте `curl` из Swagger или используйте `/api/submit-work` и `/api/works/{id}/reports` напрямую.

Тестирование

Запустите `ConsoleClient` и используйте:

- **Полный тест** → автоматическая проверка всех сервисов.
- **Создание тестовых данных** → генерация `work1.txt`, `work2.txt` (частичное совпадение), `work3.txt` (уникальный).

```
dotnet run --project src/ConsoleClient
```

Технические детали

- **Язык:** C# / .NET 9
- **База данных:** PostgreSQL (встроен в `docker-compose`)
- **Логирование:** Serilog (консоль)
- **Извлечение текста:** TextExtractor с поддержкой `.txt` и `.docx`
- **Алгоритм антиплагиата:** сравнение SHA256 хешей нормализованного текста
- **Облако слов:** <https://quickchart.io/wordcloud?text=...>