# Предисловие

Цель проекта:

Создание приложения для защищенного электронного документооборота (ЭДО), позволяющего пользователям безопасно обмениваться документами, обрабатывать их и хранить без необходимости подключения к сети Интернет.

Технический результат проекта:

Разработка приложения для защищенного ЭДО в рамках закрытого информационного контура, обеспечивающего сбор, хранение, обработку, обмен и предоставление конфиденциальной информации.

Из требований к закрытому контуру можно выделить несколько основных особенностей разрабатываемого приложения, которые нужно согласовывать и на которые нужно уделять больше внимания:

1. Количество одновременных пользователей системы
2. Насколько распределена система по серверам
3. Необходимая скорость развертки приложения

# Решение на основе микросервисов

1. Описание и функции программ

Технологический стек проекта:

| **Компонент** | **Технология / Решение** |
| --- | --- |
| Язык | Python 3.10+ (типизация, async/await) |
| API‑framework | FastAPI (+ Uvicorn / Gunicorn) |
| ORM | SQLAlchemy core + Alembic (миграции) |
| БД (реляционная) | PostgreSQL |
| Хранилище файлов | MinIO (S3‑совместимое) |
| Криптография | PyCA Cryptography (AES‑256‑GCM) |
| Auth / SSO | FastAPI OAuth2PasswordBearer + JWT |
| Контейнеризация | Docker |
| Оркестрация | Kubernetes |
| API‑Gateway | Kong / FastAPI |
| Логирование | EFK-stack (Elasticsearch + Fluentd + Kibana) |
| Метрики / Tracing | Prometheus + Grafana + Jaeger |
| Desktop-клиент (опц.) | Electron или PyQt для офлайн‑режима |

Ключевые модули и алгоритмы:

1. Модуль аутентификации пользователей:

- Форма ввода логина и пароля с использованием корпоративной почты.

- Одноразовый шестизначный код, отправляемый на почту для аутентификации.

- Интеграция с LDAP для централизованного управления учетными записями.

2. Модуль управления доступом:

- Реализация трех уровней доступа: пользователь, руководитель и администратор.

- Разграничение прав и привилегий на каждом уровне.

- Возможность гибко настраивать роли и права пользователей через веб-интерфейс.

3. Модуль безопасности передачи данных:

- Применение методов криптографического шифрования (AES-256) для защиты данных при передаче

- Использование TLS для шифрования каналов связи.

4. Модуль проверки целостности документов:

- Использование технологии блокчейн (или иного механизма контроля целостности) для проверки изменений и обеспечения неизменности данных.

- Валидация подписей документов с использованием цифровых сертификатов.

5. Модуль управления журналом и статусом документов:

- Возможность просмотра журнала действий и статусов документов с функциями сортировки и поиска.

- Автоматическая генерация отчетов о деятельности пользователей.

6. Модуль аналитики и мониторинга:

- Реализация инструментов для анализа активности пользователей и выявления аномалий.

- Интеграция с системами мониторинга (например, Prometheus и Grafana) для отслеживания производительности системы.

1. Стек технологий и микро сервисов

Описание сервисов

API Gateway (FastAPI)

* Единая точка входа, роутинг на downstream‑сервисы.
* Валидация JWT, rate‑limiting.

Auth & 2FA Service

* Логин/пароль, выдача JWT (access + refresh).
* TOTP‑коды (PyOTP) + отправка на корпоративную почту (SMTP).

User Service

* CRUD пользователей, хранение ролей (user/manager/admin).
* RBAC‑пространство через декораторы Depends в FastAPI.

Document Service

* Загрузка/скачивание документов.
* Хранение метаданных в PostgreSQL, сам файл — MinIO.
* Подпись документов (PyCA).
* Хэш каждой версии для проверки целостности (SHA‑256).

Task Service

* Celery + Redis: фоновые задачи (OCR с Tesseract, отправка уведомлений, генерация отчётов).

модули и алгоритмы

1. Аутентификация и 2FA
   * JWT (PyJWT), хранение «refresh tokens» в БД.
   * TOTP (PyOTP): пользователю при входе присылается код на email.
2. Управление доступом
   * Простая RBAC через роли в таблице users.role.
   * Декораторы-Depends в FastAPI для ограничения эндпойнтов.
3. Шифрование и безопасность
   * AES‑256‑GCM шифрование файлов перед сохранением (ключи в ENV или Vault).
   * TLS на уровне nginx (в Docker Compose конфиге).
4. Целостность документов
   * В таблице версий документов хранится prev\_hash + current\_hash.
   * Для верификации строим цепочку хэшей.
5. Журнал и аудит
   * Каждый запрос к документам логируется (user\_id, endpoint, timestamp).
   * Отчёты формируются по этим логам через SQL-запросы.
6. Мониторинг и метрики
   * FastAPI‑middleware для Prometheus (prometheus‑fastapi‑instrumentator).
   * Базовые метрики: latency, error rate, task queue length.