# 3. Эскизный проект

## Разработать программный комплекс FX2 — простой и безопасный web-сервис для обмена файлами без регистрации. Сервис должен обеспечивать загрузку и скачивание файлов с автоматической очисткой по истечении срока хранения. Пользователь взаимодействует с системой исключительно через web-интерфейс, получая уникальную ссылку для доступа к каждому файлу.

## Компоненты системы:

| **Компонент** | **Назначение** | **Краткое описание** |
| --- | --- | --- |
| **Web-интерфейс** | Взаимодействие с пользователем | Предоставляет форму для загрузки файла, отображает ссылку на скачивание, уведомления об ошибках или успехе операции. Возможна реализация с использованием HTML/CSS/JavaScript и фреймворка Bootstrap. |
| **Сервер приложений** | Обработка логики системы | Обрабатывает HTTP-запросы на загрузку и скачивание. Выполняет проверку ограничений, сохраняет файл, формирует уникальные ссылки. Предполагаемая реализация: Python (FastAPI/Flask) или Node.js. |
| **Файловое хранилище** | Физическое хранение пользовательских файлов | Сохраняет файлы на диске сервера или в S3-совместимом облаке. Поддержка автоматического удаления по таймеру. Возможна реализация через локальную ФС или облачные API (MinIO, AWS S3). |
| **База данных** | Хранение метаданных | Хранит информацию о файлах: путь, размер, дата загрузки, срок хранения, статус. Возможна реализация на PostgreSQL или SQLite. |
| **Планировщик задач** | Управление сроками хранения и автоматическое удаление файлов | Периодически проверяет базу на истекшие сроки и удаляет файлы. Возможна реализация средствами cron или через встроенный планировщик фреймворка. |

## Взаимодействие компонентов:

1. Пользователь через **Web-интерфейс** загружает файл.

2. **Сервер приложений** проверяет размер файла и общий объём, формирует уникальный идентификатор.

3. **Файловое хранилище** сохраняет файл по сгенерированному пути.

4. В **базу данных** вносится запись: имя файла, путь, идентификатор, дата загрузки, срок хранения.

5. Пользователь получает уникальную ссылку, содержащую идентификатор.

6. При обращении к ссылке **сервер приложений** находит файл в хранилище и отдаёт пользователю.

7. По истечении срока хранения **планировщик задач** удаляет файл и соответствующую запись в базе данных.

## Предварительная оценка затрат на разработку :

| **Этап** | **Оценка трудозатрат (чел.-дни)** | **Ответственные исполнители** |
| --- | --- | --- |
| Проектирование архитектуры | 3 | Системный архитектор |
| Разработка Web-интерфейса | 3 | Frontend-разработчик |
| Реализация сервера приложений | 5 | Backend-разработчик |
| Развёртывание файлового хранилища | 2 | DevOps-инженер |
| Интеграция базы данных | 2 | Backend-разработчик |
| Планировщик удаления файлов | 1 | Backend-разработчик |
| Тестирование компонентов и интеграции | 3 | QA-инженер |
| Подготовка документации | 1 | Технический писатель |
| **ИТОГО:** | **20 чел.-дней** |  |

**Общая оценка трудозатрат : средняя ставка от 8000 - 12000 рублей на человека за один день. В сумме 500000 рублей.**

## Набросок плана реализации:

1. Подготовка технической документации (ТП, ТЗ) — 1–2 дня.
2. Проектирование архитектуры и интерфейсов взаимодействия — 1 день.
3. Разработка backend-части: сервер приложений, хранилище, база данных — 5–6 дней.
4. Разработка frontend-части — 2–3 дня.
5. Интеграция компонентов и отладка — 2 дня.
6. Разработка системы удаления по сроку хранения — 1 день.
7. Проведение функционального и нагрузочного тестирования — 3 дня.
8. Подготовка пользовательской и эксплуатационной документации — 1 день.