# Система электронного учета заявлений

[**Система электронного учета заявлений 1**](#_f80e64mmw1q5)

[**Введение 3**](#_5hdigu515am7)

[Цель 3](#_nm1ati6o172d)

[Описание 3](#_s0iswiga5zwg)

[Основные задачи, которые решает система. 3](#_p5eao2hjh1ay)

[**Требования к разрабатываемой системе 4**](#_x319jtrixmrj)

[User stories 4](#_o34vcm55an7x)

[Пользователя 4](#_pboi4un5h4cv)

[Создание заявления 4](#_6yr6776bay8h)

[История заявлений 4](#_lygxmjkiet0s)

[Прикрепление документов 4](#_ro99qc1pmwhh)

[Получение уведомлений о статусе заявления 4](#_esmzuqivwn4t)

[Руководителя 4](#_7epjcei0x03e)

[Рассмотрение заявлений сотрудников 4](#_8f6cdkr0kou2)

[Написание заявлений 4](#_y98xlo3b0o8u)

[История заявлений 5](#_yn4i6swysguu)

[Прикрепление документов 5](#_bs8guyoqb8b3)

[Уведомления о новых заявлениях 5](#_xm3wsux6i5h6)

[Администратора 5](#_nfwu1sdshx85)

[Управление шаблонами заявлений 5](#_hkq5s6e6yl9f)

[Управление пользователями 5](#_usbjn210877m)

[Мониторинг системы 5](#_50giasmn5m8)

[Ролевая модель 6](#_kauihiwhng5j)

[Пользовательские требования 7](#_vvpkwdg6n1v7)

[Функциональные требования 7](#_j0tfz65pujnc)

[Виды заявлений 8](#_iyyoanmylezf)

[Виды статусов заявлений 8](#_673t941cedzq)

[Нефункциональные требования 8](#_lczv1ac6f53i)

[Поддержка и время отклика 8](#_pepcucreyuxd)

[Производительность 8](#_px3vr7u5c8a8)

[Безопасность 9](#_sb4fl9wqgpk)

[Надежность и доступность 9](#_wnme4vjhjmqt)

[Масштабируемость 9](#_hkeouf94ysw)

[Поддержка языков и платформ 9](#_9d84rghf7ngc)

[Доступность и удобство использования 9](#_3s8388b0fhih)

[Требования к интерфейсу пользователя 10](#_nlemnena3m4)

[Use case diagram 12](#_shokza8rirw5)

[Пользователя 13](#_1igus1dqrvog)

[Руководитель 15](#_srym15m8koe)

[Администратор 17](#_3lcb7bu9mbsn)

[**Требования и описание системы 19**](#_b0leqrmq4xm)

[**Спецификация и ТЗ Api 19**](#_jxb7xexpxsmt)

[**Прочее 20**](#_xdsbseoee192)

[Sequence diagram 20](#_hqhy8g86h029)

[Подробности из UseCase’a 21](#_2w63j1f2958x)

[Модели данных 21](#_vad712dq6bs)

[Структура таблиц 21](#_yofykepl5871)

[Связи между таблицами 22](#_fxz01kcdx95v)

[Типы данных 22](#_wmdsdqpvdtp)

[Концепт 23](#_u6efp2fe8jh)

[Обсуждение 23](#_cggwbcqlfdby)

# 

# Введение

## Цель

Автоматизация процесса создания, обработки и отслеживания заявлений для сотрудника и его руководителя.

## Описание

Система служит платформой для оформления и обработки заявлений между сотрудниками и их непосредственно руководителями.

Сотрудники:

* Могут быстро создавать заявления через готовые шаблоны (отпуск, командировка, расходы и т.д.), внося персональные данные (даты, причина, суммы).
* Автоматически направляет заявление руководителю и отправляет уведомление на его email.
* Дает возможность отслеживать статус в реальном времени.

Руководители:

* Представляет единый интерфейс с перечнем завялений, требующих решения.
* Позволяет одобрить или отклонить заявление в один клик, с возможностью указания комментариев.
* Автоматически уведомляет сотрудника о решении через email и обновляет статус в системе.

## Основные задачи, которые решает система.

Автоматизация создания и обработки заявлений, управление шаблонами заявлений, отслеживание статусов заявлений и уведомление на email пользователей об изменениях.

# Требования к разрабатываемой системе

## User stories

### Пользователя

#### Создание заявления

**Как** сотрудник,

**я хочу** иметь возможность создавать заявления по шаблонам,

**чтобы** автоматизировать этот процесс.

#### История заявлений

**Как** сотрудник,

**я хочу** иметь возможность видеть историю всех своих заявлений,

**чтобы** отслеживать их статус и решения.

#### Прикрепление документов

**Как** сотрудник,

**я хочу** прикреплять документы (pdf, word, docx) к заявлению,

**чтобы** предоставить всю необходимую информацию для его обработки.

#### Получение уведомлений о статусе заявления

**Как** сотрудник,  
**я хочу** получать уведомления на электронную почту о изменениях статуса моего заявления,  
**чтобы** быть в курсе прогресса обработки заявления и своевременно реагировать на изменения.

### Руководителя

#### Рассмотрение заявлений сотрудников

**Как** руководитель,

**я хочу** просматривать заявления, которые мне отправили подчиненные,

**чтобы** автоматизировать процесс рассмотрения заявлений.

#### Написание заявлений

**Как** руководитель направления,

**я хочу** иметь возможность создавать заявления по шаблонам адресованные своему руководству,

**чтобы** автоматизировать этот процесс.

#### История заявлений

**Как** руководитель,

**я хочу** иметь возможность видеть историю всех своих заявлений,

**чтобы** отслеживать их статус и решения.

#### Прикрепление документов

**Как** руководитель ,

**я хочу** прикреплять документы (pdf, word, docx) к заявлению,

**чтобы** предоставить всю необходимую информацию для его обработки.

#### Уведомления о новых заявлениях

**Как** руководитель,

**я хочу** получать уведомления о новых объявлениях на электронную почту,

**чтобы** быть в курсе актуальных заявлений на мое имя.

### Администратора

#### Управление шаблонами заявлений

**Как** администратор системы,  
**я хочу** создавать, удалять, редактировать и удалять шаблоны заявлений,  
**чтобы** обеспечить актуальность и правильность форматов заявлений для пользователей.

#### Управление пользователями

**Как** администратор,

**я хочу** управлять пользователями(просматривать, редактировать, удалять),

**чтобы** поддерживать актуальность и безопасность данных.

#### Мониторинг системы

**Как** администратор,

**я хочу** видеть статистистику и отчеты по заявлениям,

**чтобы** анализировать производительность системы и выявлять узкие места.

## 

## Ролевая модель

| Роль | Функции | Пример |
| --- | --- | --- |
| Пользователь (сотрудник компании) | * Создания заявлений по шаблонам * Просмотр статуса заявлений * Просмотр решения по заявлению * Возможность дополнить заявление | Целевой пользователь авторизуется в системе, выбирает из доступного перечня необходимую форму для заполнения, по надобности прикрепляет документы, в случае их необходимости, может дополнить недостающие детали и в заключении видит результат запроса, может скачать результирующий документ. |
| Администратор | * Создает, редактирует, удаляет шаблоны заявлений * Управляет системой обработки заявлений * Изменяет статус заявлений | Администратор после авторизации имеет возможность управления шаблонами, отслеживает систему обработки заявлений, может на нее повлиять, при необходимости. |
| Руководитель отдела (конкретного подразделения или глава компании) | * Создает, редактирует, удаляет шаблоны заявлений * Управляет системой обработки заявлений * Изменяет статус заявлений | Руководитель отдела может управлять заявлениями своей команды, одобрять или отказывать, указывать комментарии к решению. |

## Пользовательские требования

* Пользователь должен иметь возможность регистрации.
* Пользователь должен иметь возможность аутентификации, авторизации.
* Пользователь должен иметь возможность видеть подсказки по наведению на элементы.
* Пользователь должен иметь оставить обратную связь.
* Пользователь должен иметь выбрать шаблон заявления, заполнить требуемые поля, отправить заявление.
* Пользователь должен иметь возможность просматривать оставленные заявления.
* Пользователь должен получать уведомления на почту о изменении статуса заявления.
* Пользователь должен иметь возможность предусмотреть отображение перечня заявлений пользователя с возможностью группировки по системам обработки.

## Функциональные требования

* Система должна иметь возможность оставить обратную связь для пользователей, включая отправку сообщения администратору для решения проблемы, с указанием темы, содержания сообщения и возможностью прикрепить скриншот.
* Система должна иметь удобный и интуитивный интерфейс.
* Система должна иметь подсказки у основных элементов взаимодействия с пользователем.
* Система должна поддерживать авторизацию пользователя с тремя ролями (у ролей разные права): руководитель, сотрудник и администратор, включая идентификацию по логину и паролю, авторизацию, аутентификацию, а также регистрацию новых пользователей.
* Система должна позволять заполнение заявлений на основе шаблонов, привязанных к системам обработки, с полями, зависящими от выбранного шаблона.
* Система должна предоставлять администратору возможность управления шаблонами и системами обработки, включая создание, редактирование и удаление шаблонов заявлений, а также управление системами обработки заявлений.
* Система должна проверять корректность введенных шаблонных данных.
* Система должна отправлять пользователю уведомления о изменении статуса заявления асинхронно, включая информацию о новом статусе заявления.
* Система должна иметь интерфейс, адаптированный для слабовидящих пользователей.
* Система должна предусмотреть отображение перечня заявлений пользователя с возможностью группировки по системам обработки.

### Виды заявлений

* Командировка
* Отпуск
* Увольнение
* Больничный

### Виды статусов заявлений

* На рассмотрении
* Одобрено
* Отклонено

## Нефункциональные требования

### Поддержка и время отклика

* Система должна обеспечивать ответ на запросы поддержки в течение 24 часов.

### Производительность

* Система должна обрабатывать до 1000 запросов в секунду.
* Система должна поддерживать высокую производительность при обработке заявлений, обеспечивая минимальные задержки.

### Безопасность

* Прежде чем пользоваться системой, необходимо зарегистрироваться. Email пользователя должен быть уникальным.
* После регистрации пользователь получает письмо с подтверждением на почту.
* Система обеспечивает аутентификацию пользователей через логин и пароль.
* Пароли должны храниться в хешированном виде.
* Данные пользователя должны быть защищены: использование HTTPS и шифрования при передаче данных.
* Система должна обеспечивать защиту от несанкционированного доступа и атак, таких как DDoS.

### Надежность и доступность

* Система должна быть доступна 99,9% времени.
* Система должна поддерживать механизмы резервного копирования и восстановления данных.

### Масштабируемость

* Система должна поддерживать горизонтальное масштабирование для увеличения производительности при росте нагрузки.
* Архитектура системы должна позволять легко добавлять новые серверы или узлы для обработки запросов.

### Поддержка языков и платформ

* Интерфейс системы должен поддерживать несколько языков: Русский, Английский.
* Система должна быть доступна на всех популярных браузерах: Google Chrome, Safari, Mozilla Firefox, Microsoft Edge.
* Поддержка актуальных версий браузера не ниже актуальной и актуальной 10.
* Поддержка разрешений мобильных телефонов: минимальное 320x480, рекомендуемое 1080x1920.
* Поддержка разрешений десктопных устройств: минимальное 1024x768, рекомендуемое 1920x1080.

### Доступность и удобство использования

* Система должна иметь интуитивно понятный интерфейс, доступный для пользователей с ограниченными возможностями.
* Поддержка интерфейса для слабовидящих пользователей.

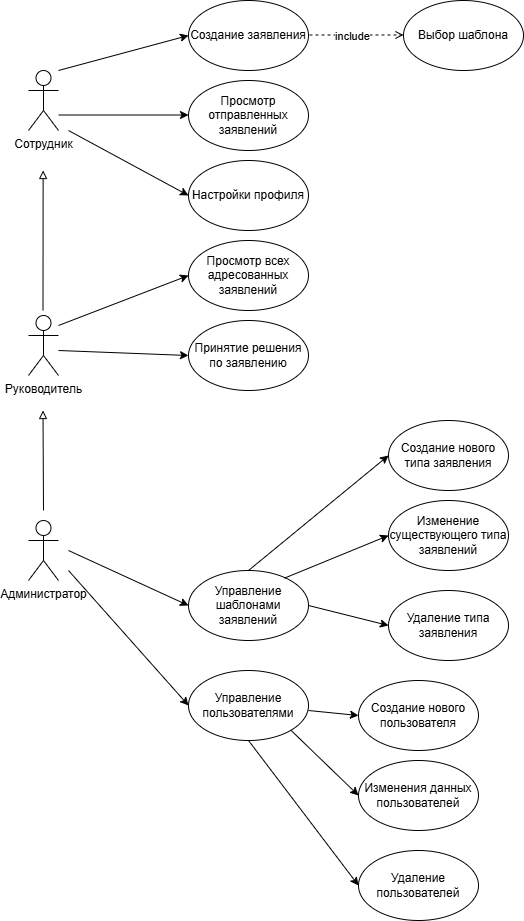
## 

## Требования к интерфейсу пользователя

1. Интерфейс администратора:
   1. Панель управления: Администратор должен иметь доступ к централизованной панели управления, где можно просматривать и управлять всеми заявлениями, пользователями, системами обработки.
   2. Управление пользователями: Возможность добавления, редактирования, удаления пользователей, а также управления их ролями.
   3. Мониторинг активность: Интерфейс должен предоставить инструменты для мониторинга активности пользователей и систем обработки, включая логи и отчеты.
2. Интерфейс руководителя:
   1. Просмотр подчиненных: Руководитель должен иметь доступ к просмотру его сотрудников.
   2. Список заявлений: Руководитель должен иметь доступ к просмотру всех заявлений (рассмотренных, на рассмотрении).
   3. Заполнение заявления: Руководитель должен иметь возможность написать заявление своему руководителю.
3. Интерфейс сотрудника:
4. Доступность и отзывчивость системы:
   1. Интуитивно понятный дизайн: Интерфейс должен быть простым и интуитивно понятным, чтобы пользователи могли легко находить нужные функции и выполнять задачи.
   2. Обратная связь: Система должна предоставлять мгновенную обратную связь на действия пользователя, например, подтверждение отправки заявления или сохранения изменений.
5. Адаптивный дизайн:
   1. Поддержка различных устройств: Интерфейс должен корректно отображаться на разных устройствах, включая мобильные телефоны и компьютеры.
   2. Разрешение экранов: Поддержка самый популярных разрешений устройств.
6. Интеграция с внешними сервисами:
   1. Внешняя Api имеющая доступ к данным: Система должна предоставлять данные клиентов, которые получили одобрения заявлений(только на чтение).
   2. Безопасность данных: Взаимодействие с внешними сервисами должно быть защищено, с использованием аутентификации и шифрования данных.
7. Управление заявлениями:
   1. Создание и отправка заявлений: Пользователи должны иметь возможность безпроблемно создавать и отправлять заявления через интерфейс, с поддержкой конкретных шаблонов.
   2. Проверка введенных данных: В случае, когда пользователь ввел в паспортные данные кириллицу интерфейс системы должен обозначить этот момент подсвечивая поле.

[Wireframe UI UX](https://wireframe.cc/pro/pp/a7730c52a886707)

## Use case diagram



### Пользователя

| Название UC | Создание заявления UseCase\_U1 |
| --- | --- |
| Участники | Сотрудник, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится на главном экране веб приложения |
| Сценарий | 1. Пользователь выбирает раздел “Создать заявление” 2. Система отображает список доступных шаблонов 3. Пользователь нажимает на кнопку с названием нужного шаблона и нажимает продолжить    1. Если пользователь нажал продолжить и не выбрал шаблон       1. Система выводит сообщение: “Выберите шаблон для продолжения”       2. Переход к пункту 2 4. Система отображает форму с полями ввода, соответствующими выбранному шаблону 5. Пользователь заполняет поля    1. Если обязательные поля не заполнены       1. Сценарий возвращается к шагу 5.    2. Если в полях введены некорректные данные       1. Система подсвечивает поле и выводит уведомление: “Проверьте корректность введенных данных”       2. Сценарий возвращается к шагу 5    3. Если пользователь передумал заполнять заявление:       1. Пользователь нажимает кнопку “Отмена”    4. Если пользователь хочет отложить заполнение документа:       1. Нажимает кнопку сохранения 6. Пользователь прикрепляет документы, при условии необходимости сторонних документов для рассмотрения заявления...    1. Если файл не удовлетворяет условиям: формат файла из допустимых (pdf, docx) и размер файла не больше 10 мб       1. Система выводит ошибку: “Неверный формат файл или его размер больше 10мб” 7. Пользователь нажимает кнопку “Отправить” 8. Система сохраняет заявление, назначает статус “На рассмотрении” и уведомляет руководителя |
| Результаты | Заявление создано и отправлено адресату |

| Название UC | Просмотр отправленных заявлений UseCase\_User\_2 |
| --- | --- |
| Участники | Сотрудник, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится на главном экране веб приложения |
| Сценарий | 1. Сотрудник открывает раздел “Мои заявления”. 2. Система отправляет запрос в базу данных для получения списка заявлений    1. Если база данных не отвечает       1. Система выводит сообщение об ошибке (5xx)       2. Сценарий завершается    2. Если база данных ответила и данных в базе нет:       1. Система отображает: “У вас нет заявлений”       2. Сценарий завершается    3. Если база данных ответила и данные найдены 3. Система отображает таблицу с заявлениями 4. Пользователь применяет фильтры по сортировке заявлений (например, по статусу или по дате)    1. Система обновляет список согласно выбранным фильтрам 5. Пользователь может нажать на конкретное заявление и узнать детали |
| Результаты | Отображение актуального списка заявлений с возможностью фильтрации |

### Руководитель

| Название UC | Просмотр входящих заявлений UseCase\_Manager\_1 |
| --- | --- |
| Участники | Руководитель, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится на главном экране веб приложения |
| Сценарий | 1. Руководитель выбирает раздел “Входящие заявления”. 2. Система совершает запрос в базу данных    1. Если база данных не отвечает (таймаут или ошибка подключения):       1. Система отображает ошибку (5xx)       2. Сценарий завершается    2. Если база данных отвечает, но заявлений нет:       1. Система отвечает: “Нет заявлений, требующих вашей обработки”       2. Предлагает действия: “Обновить список” или “Вернуться в главное меню”       3. Сценарий завершен    3. Если база данных ответила и заявления найдены:       1. Система выводит пользователю страницу с заявлениями 3. Система отображает таблицу с колонками: Сотрудник, Тип заявления, Дата подачи, Статус 4. Руководитель сортирует заявления по дате, статусу, сотруднику    1. Система отображает список по заданным фильтрам 5. Для просмотра деталей руководитель нажимает на строчку с заявлением    1. Система открывает карточку заявления с:       1. Полными данными по заявлени, по сотруднику       2. Прикрепленными документами          1. Руководитель нажимает на документ и скачивает его в нужном формате для ознакомления       3. Кнопками “Одобрить” / ”Отклонить” |
| Результаты | Отображение списка входящих заявлений для конкретного руководителя |

| Название UC | Принятие решения по заявлению UC\_Руководитель\_2 |
| --- | --- |
| Участники | Руководитель, система |
| Предусловие | Руководитель успешно прошел авторизацию и находится на странице конкретного заявления |
| Сценарий | 1. Руководитель выбирает действие:    1. “Одобрить”       1. Система обновляет статус       2. Система отправляет уведомление сотруднику    2. “Отклонить”       1. Система выводит поле для комментирования       2. Пользователь заполняет комментарий и отклоняет заявление       3. Система изменяет статус заявления       4. Система отправляет уведомление сотруднику |
| Результаты | Заявление обработано, статус заявления изменен, сотрудник уведомлен |

### 

### Администратор

| Название UC | Создание шаблона UseCase\_Admin\_StatementTemplates\_1 |
| --- | --- |
| Участники | Администратор, система |
| Предусловие | Администратор успешно прошел авторизацию и находится в админ панели веб приложения |
| Сценарий | 1. Администратор открывает раздел “Шаблоны заявлений” 2. Администратор нажимает на кнопку “Добавить шаблон”:    1. Система отображает форму с полями: название, описание, тип завления(отпуск, командировка, …) и конструктор для дополнительных полей    2. Администратор заполняет обязательные поля (название, предназначение шаблона) и добавляет поля необходимые для этого типа заявлений, заполняет их описание и свойства    3. Система проверяет корректность полученных данных и сохраняет 3. Система принимает данные и обновляет список |
| Результаты | Шаблоны заявлений обновлены |

| Название UC | Создание шаблона UseCase\_Admin\_Templates\_2 |
| --- | --- |
| Участники | Администратор, система |
| Предусловие | Администратор успешно прошел авторизацию и находится в админ панели веб приложения |
| Сценарий | 1. Администратор открывает раздел “Шаблоны заявлений”. 2. Система отображает список существующих шаблонов. 3. Администратор выбирает интересующий шаблон и напротив названия шаблона нажимает на кнопку “Изменить шаблон”:    1. Администратор изменяет название, описание шаблона    2. Администратор в виде конструктора изменяет поля, заполняет их описание и свойства    3. Система проверяет корректность изменений полученных данных 4. Система принимает изменения и обновляет список |
| Результаты | Шаблоны заявлений обновлены |

| Название UC | Создание шаблона UseCase\_Admin\_Templates\_3 |
| --- | --- |
| Участники | Администратор, система |
| Предусловие | Администратор успешно прошел авторизацию и находится в админ панели веб приложения |
| Сценарий | 1. Администратор открывает раздел “Шаблоны заявлений”. 2. Система отображает список существующих шаблонов. 3. Администратор выбирает интересующий шаблон и напротив названия шаблона нажимает на кнопку “Удалить шаблон”:    1. Система спрашивает подтверждения намерений    2. Система удаляет шаблон 4. Система обновляет список с учетом последних изменений |
| Результаты | Шаблоны заявлений обновлены |

| Название UC | Добавление нового пользователя UseCase\_Admin\_UserManagment\_1 |
| --- | --- |
| Участники | Администратор, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится в админ панели |
| Сценарий | 1. Администратор открывает раздел “Пользователи” 2. Администратор нажимает на кнопку “Добавить пользователя”    1. Система отображает форму с полями: email, логин, пароль , роль (сотрудник, руководитель, администратор), ФИО, отдел    2. Администратор заполняет поля       1. Пароль генерируется системой или вручную администратором       2. Выбрав роль Администратор получит описания доступных функций для этого пользователя    3. Система проверяет уникальность данных: email и логина       1. Если данные корректны, создает пользователя и отправляет ему на электронную почту приветственное письмо с инструкцией       2. Если данные не уникальны, система возвращает на сценарий 2b 3. Система примет изменения и обновит список сотрудников |
| Результаты | Добавлен новый пользователь в системе |

| Название UC | Редактирование данных пользователями UseCase\_Admin\_UserManagment\_2 |
| --- | --- |
| Участники | Администратор, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится в админ панели |
| Сценарий | 1. Администратор открывает раздел “Пользователи”.    1. Система отображает список существующих пользователей в системе. 2. Администратор выбирает интересующего пользователя и напротив данных пользователя нажимает на кнопку “Изменить данные пользователя”:    1. Система отображает форму с текущими данными пользователя 3. Администратор выбирает на изменение:    1. Роль       1. Администратор выбирает новую роль сотрудника    2. Отдел       1. Администратор выбирает новый отдел (из существующих в системе) сотрудника (при переводе с проекта на проект)    3. ФИО       1. Администратор заполняет вручную данные пользователя    4. Email       1. Администратор вручную вводит новую электронную почту сотрудника       2. Система проверяет корректность введенных данных    5. Пароль       1. Автоматический сброс          1. Администратор нажимает “Сбросить пароль”          2. Система генерирует новый пароль и отправляет его на почту пользователя       2. Ручный сброс          1. Администратор вводит новый пароль          2. Система проверяет сложность пароля по критериям (минимальная длина, наличие цифр/символов) 4. Система сохраняет изменение данных и возвращает пользователя на страницу со списком всех пользователей |
| Результаты | Данные пользователей изменены |

| Название UC | Удаление пользователей UseCase\_Admin\_UserManagment\_3 |
| --- | --- |
| Участники | Администратор, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится в админ панели |
| Сценарий | 1. Администратор открывает раздел “Пользователи”.    1. Система отображает список существующих пользователей в системе. 2. Администратор выбирает интересующего пользователя и нажимает кнопку “Удалить”:    1. Система проверяет, есть ли у пользователя       1. Активные заявления (в статусе “на рассмотрении”)       2. История заявлений (архивные или завершенные)    2. Если активных заявлений нет       1. Система запрашивает подтверждение на удаление пользователя       2. При подтверждении:          1. Система помечает пользователя как удален или полностью удаляет пользователя из базы данных.          2. Уведомление администратора об успешном удалении    3. Если есть активные заявления       1. Система блокирует удаление и предлагает          1. Переназначить заявление другому пользователю (из списка руководителей)             1. Возврат к сценарию 2b          2. Отклонить все заявления удаляемого пользователя             1. Возврат к сценарию 2b 3. Завершение сценария    1. Система обновляет список пользователей |
| Результаты | Пользователь не имеет доступа к системе |

# Требования и описание системы

[bpmn drawio](https://drive.google.com/file/d/1TlCTe0rMsgWnmWKs9xKKFrmRYm1eMVr8/view?usp=sharing)

[диаграмма классов](https://drive.google.com/file/d/1RhpYPQjgm2GLz9wHzGctmTQPAMvu9S7v/view?usp=sharing)

[С4](https://drive.google.com/file/d/12eqH-Wcgs6auzPGa-mc_CwKLV5Ay2AYa/view?usp=sharing)

С4 или uml

# Спецификация и ТЗ Api

[Swaggerhub](https://app.swaggerhub.com/apis/mkhl/stm/1.0.0)

# 

# Прочее

## Sequence diagram

# 

## 

## Подробности из UseCase’a

Быстрая и безошибочная подача заявления на получение сертификата и проверку пожарной безопасности.

**Как** юридическое лицо,

**я хочу** подать заявление на проверку пожарной безопасности,  
**чтобы** получить сертификат, подтверждающий соответствие требованиям.

Критерий приема:

1. Я могу зарегистрировать в системе заполнив необходимые данные
2. Я могу авторизоваться в системе с использованием логина и пароля.
3. После авторизации я вижу список доступных шаблонов заявлений.
4. Я выбираю шаблон “Проверка пожарной безопасности”.
5. Система предлагает заполнить необходимые поля, такие как: адрес помещения, контакные данные и так далее.
6. Я заполняю все обязательные поля и прикрепляю все необходимые документы.
7. После заполнения я отправляю заявление на рассмотрение.
8. Я получаю уведомления о том, что заявление отправлено и ожидает обработки.
9. Я могу отслеживать статус заявления и решения по нему в личном кабинете.
10. При изменении статуса, я получаю уведомление на электронную почту.

## Модели данных

### Структура таблиц

**Users**: Таблица пользователей хранит основную информацию о каждом пользователе, включая имя, email, пароль и роль. Это позволяет системе управлять доступом и аутентификацией пользователей.

**Statuses**: Таблица статусов определяет возможные состояния заявлений (например, "в обработке", "одобрено", "отклонено"). Это обеспечивает стандартизацию и упрощает отслеживание прогресса заявлений.

**Templates**: Таблица шаблонов содержит информацию о доступных шаблонах заявлений, включая их описание и структуру полей. Это позволяет администраторам управлять форматами заявлений и обеспечивать их актуальность.

**Applications**: Таблица заявлений связывает пользователей, шаблоны и статусы, храня информацию о каждом заявлении, включая дату создания и обновления. Это центральная таблица, которая позволяет отслеживать жизненный цикл каждого заявления.

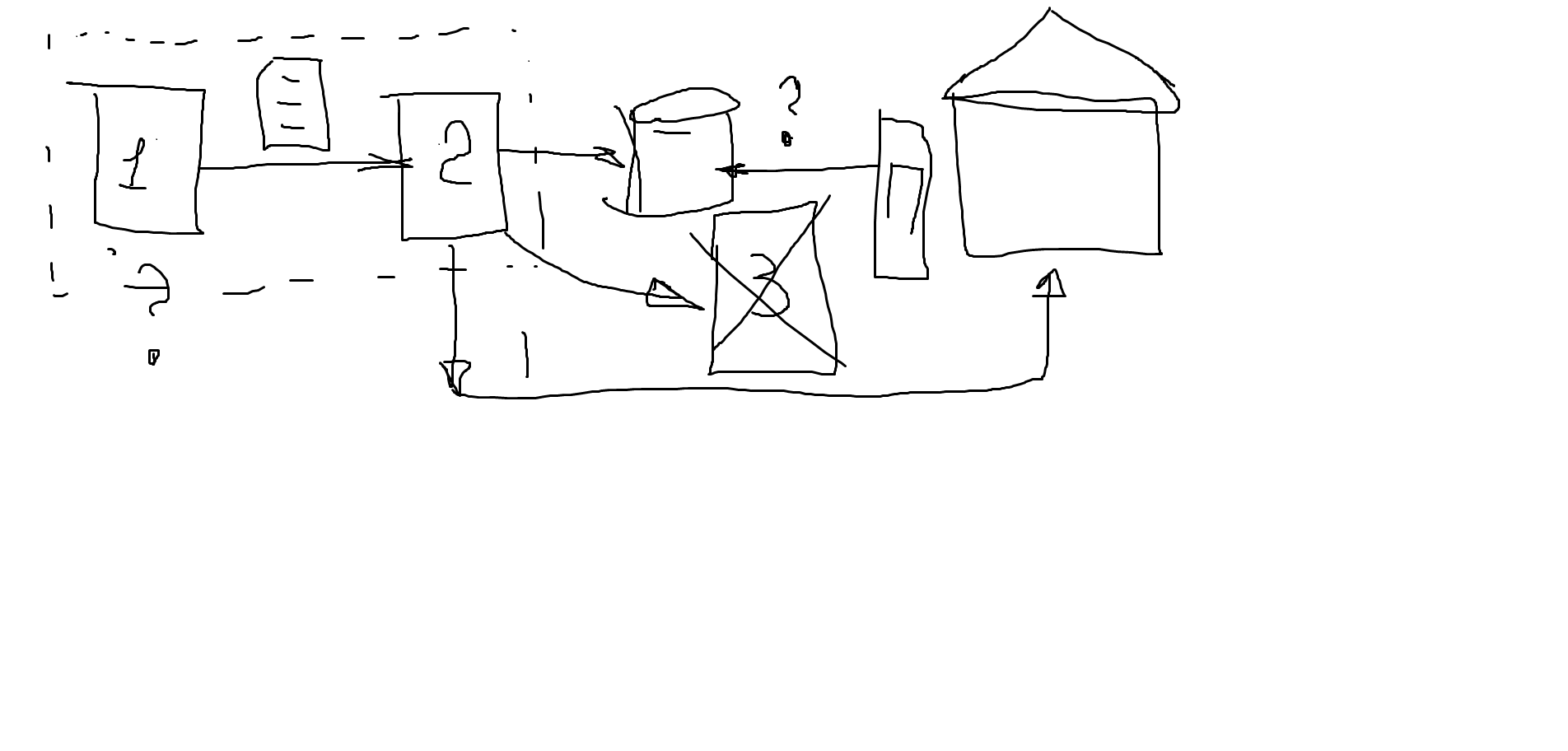
### Связи между таблицами

Внешние ключи в таблице `Applications` (`userId`, `templateId`, `statusId`) обеспечивают целостность данных и позволяют легко связывать заявления с соответствующими пользователями, шаблонами и статусами. Это упрощает навигацию и управление данными.

### Типы данных

* Использование подходящих типов данных (например, INT для идентификаторов, VARCHAR для текстовых данных, TIMESTAMP для даты и времени) обеспечивает эффективное хранение и обработку данных.
* Поле `fields` в таблице `Templates` имеет тип JSON, что позволяет гибко хранить и изменять структуру полей шаблонов без необходимости изменения схемы базы данных.

## Концепт



## Обсуждение

Обнаружил, один пункт из заданий или требований в системе, а именно “Предусмотреть отображение перечня заявлений пользователя с возможность группировки по системам обработки.”, это означает что систем обработки несколько, гмгм. Я пока это вставил в функциональные требования. Кажется, что это значит, администратор должен иметь возможность просмотра заявлений конкретного пользователя и еще сортировать их по системам обработки. Вот, что думаешь? Правильно ли я понял? Я пока это вставил как система должна в раздел функциональные требования.

Вопрос про: администратор владеет правами пользователя. К этому есть вопросы потому что пользователи юр лица, у них там свое поля методы, корректно ли администратору давать возможности пользователя. Может на данном этапе у нас при регистрации нет требования вводить данные присущие юридическим лицам, а если потом этим обрастем, что в это поле добавлять администратору, если он по факту не юр лицо, да и вообще зачем админу создавать заявления? Если только для проверки.. Вообщем вопрос на подумать.

Еще получается нужно ли в общие требования добавлять что у нас две несколько систем обработки? Или пока так оставим..

Требования к интерфейсу пользователя как то бедновато выглядят по сравнению с другими заголовками.

На фигму я бы не смотрел, так как там еще старая версия видения интерфейса и уже есть апдейты.