# 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 23**](#_b0leqrmq4xm)

[**Спецификация и ТЗ Api 24**](#_jxb7xexpxsmt)

[**Модели данных 24**](#_lun4towya9bo)

[**Er 24**](#_cgovtscgs07)

[**Прочее 25**](#_xdsbseoee192)

[Концепт 25](#_u6efp2fe8jh)

# 

# 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 к каждому заявлению иметь данные: дата создания заявления, автора, адресата, сроки (для отпуска, больничного не в прошедшем времени).
* 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 100 запросов в секунду.
* Система должна поддерживать высокую производительность при обработке заявлений, обеспечивая минимальные задержки.

### Безопасность

* Размер прикрепляемых файлов к заявлению должен быть из доступных форматов и не должен превышать лимит в 10мб.
* 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 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.

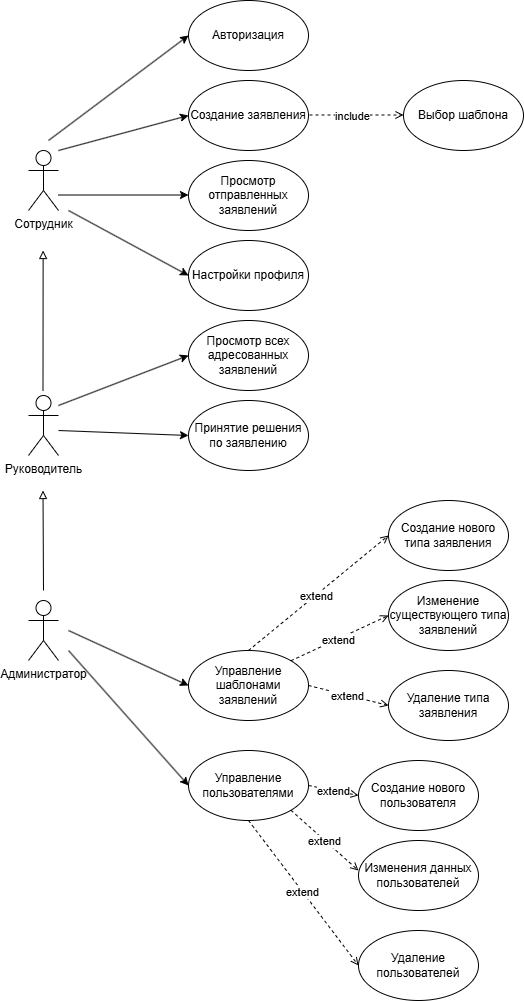
## 

## 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 | Авторизация UseCase\_U1 |
| --- | --- |
| Участники | Сотрудник, система |
| Предусловие | Пользователь перешел на адрес системы и находится на странице авторизации |
| Сценарий | 1. Пользователь вводит данные для входа: логин и пароль 2. Нажимает на кнопку “Вход” 3. Система отправляет запрос на сервер для проверки данных пользователя (POST /auth)    1. 500 Internal Server Error – Если система не отвечает. Ошибка сервера.       1. Вывод ошибки в интерфейс “Ошибка сервера. Попробуйте позже.”    2. Если система отвечает кодом ошибки 4xx:       1. 400 Bad Request – Неверный формат запроса          1. Система подсветит в интерфейсе пустые поля       2. 401 Unauthorized – “Неверные логин или пароль”          1. Вывод ошибки в интерфейс       3. 403 Forbidden – Пользователь заблокирован          1. Вывод ошибки в интерфейс “Доступ запрещен, обратитесь к Администратору”       4. 404 Not found – Пользователь не найден          1. Вывод ошибки в интерфейс “”       5. 429 Too Many Requests – Превышено количество попыток входа          1. Блокировка формы на 5 минут    3. 200 OK – Если система отвечает       1. Система проверила данные пользователя и вернула токен, который действует 24 часа 4. Система перенаправляет пользователя на главный экран приложения и дает доступ к функциям относительно его роли |
| Результаты | Пользователь имеет доступ к системе |

| Название UC | Создание заявления UseCase\_U2 |
| --- | --- |
| Участники | Сотрудник, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится на главном экране веб приложения |
| Сценарий | 1. Пользователь выбирает раздел “Создать заявление” (/statements) 2. Система отображает список доступных шаблонов 3. Пользователь нажимает на кнопку с названием нужного шаблона и нажимает продолжить    1. 404 Not found – Если пользователь нажал продолжить и не выбрал шаблон       1. Система выводит сообщение: “Шаблон не найден”       2. Переход к пункту 2 4. Система отображает форму с полями ввода, соответствующими выбранному шаблону 5. Пользователь заполняет поля    1. Если пользователь передумал заполнять заявление:       1. Пользователь нажимает кнопку “Отмена”    2. Пользователь заполнил данные и нажал кнопку “Отправить заявление”       1. 400 Bad request – Если обязательные поля не заполнены или в полях введены некорректные данные          1. Подсветить поля, вывести сообщение об ошибке введенных данных. Сценарий возвращается к шагу 5.    3. Если пользователь хочет отложить заполнение документа:       1. Нажимает кнопку сохранения 6. Пользователь прикрепляет документы, при условии необходимости сторонних документов для рассмотрения заявления...    1. Система отвечает 413 Payload Too Large – В случае размера файла > 10 МБ       1. Уведомить пользователя о лимите размера файла    2. Система отвечает 415 Unsupported Media Type – Если система не поддерживает формат файла       1. Уведомить пользователя о поддерживаемых форматах файлов    3. Система отвечает 500 Internal Server Error – Ошибка сохранения, сервер не отвечает       1. Предложить повторить попытку    4. Система отвечает 201 Created – Система сохраняет заявление, назначает статус “На рассмотрении” и уведомляет руководителя. Сценарий завершен |
| Результаты | Заявление создано и отправлено адресату |

| Название UC | Просмотр отправленных заявлений UseCase\_User\_3 |
| --- | --- |
| Участники | Сотрудник, система |
| Предусловие | Пользователь успешно прошел авторизацию и находится на главном экране веб приложения |
| Сценарий | 1. Сотрудник открывает раздел “Мои заявления”. (/statements/sent) 2. Система отправляет запрос в серверную часть для получения списка заявлений    1. 500 Internal Sever Error – Если система не отвечает       1. Система выводит сообщение об ошибке сервера    2. 204 No Content – Если система ответила и данных нет:       1. Отображение сообщения в интерфейсе: “У вас нет заявлений”    3. 200 OK – Если сервер ответил и данные найдены       1. Система отображает таблицу с заявлениями 3. Пользователь применяет фильтры по сортировке заявлений (например, по статусу или по дате)    1. Система(Fe) обновляет список согласно выбранным фильтрам 4. 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”. (/statements/received) 2. Система совершает запрос в серверную часть для получения входящих заявлений    1. 403 Forbidden – Если у пользователя недостаточно прав       1. Вывести сообщение о недостатке прав    2. 204 No content – Если серверная часть отвечает, но заявлений нет:       1. Система отвечает: “Нет заявлений, требующих вашей обработки”       2. Предлагает действия: “Обновить список” или “Вернуться в главное меню”    3. 500 Internal Server Error – Если серверная часть не отвечает:       1. Система отображает ошибку сервера    4. 200 OK – Если серверная часть 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    2. Сценарий завершен |
| 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. Пользователь нажимает на кнопку “Одобрить” или “Отклонить” (/statements/decision/{id})    1. 400 Bad Request – Некорректный ID заявлений       1. Уведомление пользователя: “Заявление не найдено”    2. 403 Forbidden – Нет прав на решение по заявлению       1. Вывод сообщения о недостатке прав    3. 409 Conflict – Заявление уже обработано       1. Обновить статус на экране    4. 200 OK – Действие пользователя обработано       1. Если действие “Отклонить”, то система отображает поле для ввода комментария.          1. Пользователь отправляет комментарий          2. 400 Bad Request – Некорректный ID заявлений             1. Уведомление пользователя: “Заявление не найдено”          3. 403 Forbidden – Нет прав на одобрение             1. Вывод сообщения о недостатке прав          4. 409 Conflict – Заявление уже обработано             1. Обновить статус на экране          5. 200 OK – Действие пользователя обработано       2. Система обновляет статус       3. Система отправляет уведомление сотруднику       4. Сценарий завершен |
| 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” (/templates (get))    1. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    2. 204 No Content – Серверная часть ответила, но шаблонов нет       1. Вывести пустой список шаблонов и другой интерфейс    3. 200 OK – Серверная часть ответила       1. Вывести список шаблонов и другой интерфейс 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. Администратор нажимает на кнопку “Добавить шаблон” (/templates (post))       1. 403 Forbidden – Нет прав администратора          1. Вывести сообщение: “Доступ запрещен”       2. 409 Conflict – Шаблон с таким названием существует          1. Предложить пользователю переименовать название шаблона или проверить список существующих шаблонов       3. 201 Created – Шаблон успешно создан          1. Вывести сообщение: “Шаблон успешно добавлен”. Таймаут на 5 секунд.          2. Вернуться к списку шаблонов и обновить его          3. Сценарий завершен |
| Результаты | Шаблоны заявлений обновлены |

| Название UC | Редактирование 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”.    1. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    2. 204 No Content – Серверная часть ответила, но шаблонов нет       1. Вывести пустой список шаблонов и другой интерфейс       2. Сценарий завершен    3. 200 OK – Серверная часть ответила       1. Вывести список шаблонов и другой интерфейс 2. Система отображает список существующих шаблонов. 3. Администратор выбирает интересующий шаблон и напротив названия шаблона нажимает на кнопку “Изменить шаблон” (/templates/{id} (post)):    1. Администратор изменяет название, описание шаблона    2. Администратор в виде конструктора изменяет поля, заполняет их описание и свойства. Нажимает на кнопку “Изменить”    3. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    4. 400 Forbidden – Некорректные данные       1. Вывести сообщение: “Проверьте введенные данные”    5. 200 OK – Изменения сохранены       1. Система принимает изменения и обновляет список       2. Сценарий завершен |
| Результаты | Шаблоны заявлений обновлены |

| Название UC | Удаление 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. 403 Forbidden – Нет прав администратора    1. Вывести сообщение: “Доступ запрещен” 3. 204 No Content – Серверная часть ответила, но шаблонов нет    1. Вывести пустой список шаблонов и другой интерфейс    2. Сценарий завершен 4. 200 OK – Серверная часть ответила    1. Вывести список шаблонов и другой интерфейс 5. Система отображает список существующих шаблонов. 6. Администратор выбирает интересующий шаблон и напротив названия шаблона нажимает на кнопку “Удалить шаблон” подтверждая его действия (/templates/{id} (delete)):    1. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    2. 200 OK – Изменения сохранены       1. Система удаляет шаблон и обновляет список       2. Сценарий завершен |
| 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”    1. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    2. 204 No Content – Серверная часть ответила, но пользвателей нет       1. Вывести пустой список пользователей и другой интерфейс    3. 200 OK – Серверная часть ответила       1. Вывести список пользователей и другой интерфейс 2. Администратор нажимает на кнопку “Добавить пользователя” (/users (post))    1. Система отображает форму с полями: email, логин, пароль , роль (сотрудник, руководитель, администратор), ФИО, отдел    2. Администратор заполняет поля и нажимает кнопку “Сохранить”       1. Пароль генерируется системой или вручную администратором и данные отправляются в серверную часть          1. 400 Bad Request – Некорректные данные             1. Сообщение пользователю о проверке введенных данных          2. 409 Conflict – Пользователь с таким email существует             1. Проверить имя пользователя          3. 201 Created – Пользователь создан             1. Система вернет сообщение об успехе операции и обновит список пользователей |
| 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. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    2. 204 No Content – Серверная часть ответила, но пользвателей нет       1. Вывести пустой список пользователей и другой интерфейс       2. Сценарий завершен    3. 200 OK – Серверная часть ответила       1. Вывести список пользователей и другой интерфейс 2. Администратор выбирает интересующего пользователя и напротив данных пользователя нажимает на кнопку “Изменить данные пользователя” (/user/{id} (put)):    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)    6. Система обработает ввод и отправит изменения в серверную часть       1. 403 Forbidden – Попытка изменить роль пользователя выше своих прав       2. 200 OK – Данные обновлены          1. Система сохраняет изменение данных и возвращает пользователя на страницу со списком всех пользователей          2. Сценарий завершен |
| 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. 403 Forbidden – Нет прав администратора       1. Вывести сообщение: “Доступ запрещен”    2. 204 No Content – Серверная часть ответила, но пользвателей нет       1. Вывести пустой список пользователей и другой интерфейс       2. Сценарий завершен    3. 200 OK – Серверная часть ответила       1. Вывести список пользователей и другой интерфейс 2. Администратор выбирает интересующего пользователя и нажимает кнопку “Удалить” (/user/{id} (delete)):    1. 409 Conflict – У пользователя есть активные заявления       1. Вывод сообщения: “Отклонить все заявления пользователя или переназначить на другого”    2. 204 No Content – Пользователь удален       1. Уведомление: “Пользователь обновлен”       2. Обновить список пользователей       3. Сценарий завершен |
| Результаты | Пользователь не имеет доступа к системе |

# Требования и описания системы

Система представляет собой монолитное веб-приложение, реализованное как единое целое без выделенных микросервисов.

Архитектура локальная и закрытая: предполагается развертывание в инфраструктуре организации (Доступа из сети интернет не будет).

Взаимодействие с пользователями осуществляется через веб интерфейс. Аутентификация и авторизация реализованы с использованием JWT.

Интеграция с ЭЦП

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

[ClassesDiagramm](https://drive.google.com/file/d/1ba73aRH303PdnpV3B6ZXoHui7k36fOHp/view?usp=sharing)

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

# Спецификация и ТЗ Api

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

[api.pdf](https://drive.google.com/file/d/1CWJtwJq3Sgh6wY7QWc26CgOI-9mKQhsS/view?usp=drive_link)

# Модели данных

[stm.svg](https://drive.google.com/file/d/1ohcvgGR0JnBwhXmJA58m828e4KxmaC0w/view?usp=sharing)

[dbdiagram.io/stm](https://dbdiagram.io/d/stm-677a76465406798ef74ff4b5)

### 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").

**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.

**Statements**: 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.

**Documents**: Таблица документов прикрепляется к заявлениям, хранит всю необходимую информацию о прикрепленном файле: название, тип, путь, вес, дату загрузки, заявление к которому относится.

**Tokens**: Таблица хранящая токены пользователей, необходима затем чтобы при каждом запросе к системе не вводить данные для входа.