Software Architecture Document
(Описание Архитектуры)

Version control

Дата	Кто изменил	Какие изменения внесены
16.09	Артем	Разделы документа 1-3
18.09	Анна	Разделы 4.5, 4.6,
31.09	Михаил	Раздел 4.1, 4.2, 4.3, 4.4, 5.6,
30.09	Анна	Исправления 4.3, раздел 5.1, 5.2, 5.4, 6
01.11	Михаил	Раздел 5.5, 8.3, 8.4, 8.5, 8.6
05.11	Артем	Раздел 7.1, 8.1, 8.2
07.11	Михаил	Исправление диаграммы 8.6
13.11	Артем	Исправление диаграммы 8.2,
20.11	Анна	Исправление диаграмм 4.5, 5.3

1. Introduction (Введение)

[Введение представляет собой обзор на весь документ в целом и включает в себя следующие разделы - назначение, область применения, определения и аббревиатуры, ссылки и обзор.]

1.1 Purpose

[Данный документ описывает архитектуру приложения как набор точек зрения на неё - use case view, logical view, process view, deployment view, implementation view (может быть стоит перевести названия?). Взаимодействие элементов разных точек зрения представлено в виде UML-диаграмм.]
[Укажите назначение данного документа.]

В данном документе описывается архитектура разрабатываемого продукта в виде набора точек зрения: use case view (сценарии использования), logical view (структура и взаимодействие компонентов), process view (выполняющиеся процессы и их временная привязка), deployment view (схема развертывания и конфигурация инфраструктуры), implementation view (организация кода и детали реализации).

1.2 Scope (Область применения)

[Приведите краткое описание области применения данного документа, к какому(им) проекту(ам) он относится, кем будет использоваться и т.д.]

Документ относится к проекту информационной системы “Бросайте курить!” и относится к:

1. Аналитики и Product-менеджеры — используют документ, чтобы связать бизнес-требования с техническими ограничениями, оценивать влияние изменений и управлять сроками/стоимостью.
2. Заказчик — получает понятные границы поставки, внешние зависимости, ключевые риски и ожидаемые нефункциональные показатели.
3. Разработчики — используют документ как источник единых стандартов и диаграмм взаимодействий, чтобы уверенно реализовывать и развивать модули/сервисы системы.
4. Тестировщики — формируют стратегию и приоритеты тестирования на основе описанной архитектуры продукта, ее границ и ключевых сценариев.

1.3 Definitions, Acronyms and Abbreviations (Определения и аббревиатуры)

[Укажите значение терминов и аббревиатур, которые употребляются в данном документе. Возможно указание ссылки на Глоссарий проекта.]

ССЫЛКА на глоссарий, содержащий значение терминов и аббревиатур, которые употребляются в данном документе.

1.4 References (Ссылки)

[Перечислите списком названия документов, на которые ссылаетесь в данном, укажите их источники.]

1. Книга “Корпорация “Бросайте курить”, автор Стивен Кинг;
2. Vision;
3. Software Requirements Specification (SRS);
4. Use Case template (UC);
5. Risk List (RL);
6. SDP (Software Development Plan).
7. Глоссарий

1.5 Overview (Обзор документа)

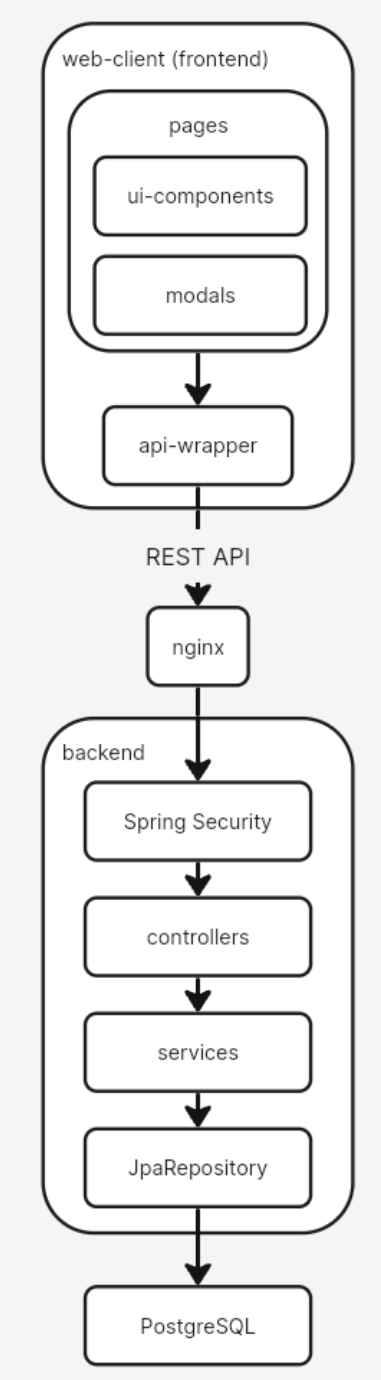
[Приведите краткое описание остальных разделов документа.]

Краткое описание последующих разделов документа:

1. Представление архитектуры - описание архитектуры системы с содержанием того, из каких точек зрения состоит архитектура.
2. Цели и ограничения архитектуры - описание архитектурно-значимых факторов.
3. Use case view - описание основных сценариев использования.
4. Logical view - описание структуры и взаимодействия компонентов.
5. Process view - описание выполняющихся процессов и их временная привязка.
6. Deployment view - описание схемы развертывания и конфигурации инфраструктуры.
7. Implementation view - описание организация кода и детали реализации.
8. Производительность - описание основных характеристик измерения производительности системы и их границы.
9. Качество - описание, как архитектура обеспечивает качество системы.

2. Architectural Representation (Представление архитектуры)

[Данный раздел описывает в общем архитектуру системы. Укажите, какие типы диаграмм необходимы для описания разных точек зрения. Рекомендуется воспользоваться следующей таблицей:]



https://app.holst.so/board/ac6c1ab0-d8d5-43d1-bfd7-4aa9065ebb0f

Diagram\View	Use Case View	Logical View	Implementation view	Process view**	Deployment View
Use Case Diagram	+	-	-		-

Class Diagram	+ (Взаимодействие сущностей)	+ (Описание основных классов и интерфейсов их взаимодействия)	+ (Полное описание классов с указанием их методов/полей, указать типы связей между классами)		-
Activity Diagram	+ (Абстрактное описание)	+ (Более подробное описание, уровни взаимодействия должны совпадать с диаграммой пакетов)	+ (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов).		-
State Machine Diagram	+ (Абстрактное описание)	+ (Более подробное описание, уровни взаимодействия должны совпадать с диаграммой пакетов)	+ (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов).		-
Sequence Diagram	+ (Абстрактное описание)	+ (Более подробное описание, уровни взаимодействия должны совпадать с диаграммой пакетов)	+ (Полное описание прецедента с указанием вызываемых методов, используемых классов и объектов).		-
Cooperative Diagram	+ (Абстрактное описание)	+ (Более подробное описание, уровни взаимодействия должны совпадать с	+ (Полное описание прецедента с указанием вызываемых методов,		-

		диаграммой пакетов)	используемых классов и объектов).		
Package Diagram	-	+	-		-
Data Base Diagram	-	-	+ (Полная ER модель базы данных + её даталогическая модель)		-
Deployment Diagram	-	-	-		+ (Подробная диаграмма развертывания с указанием характеристик машин и интерфейсов взаимодействия)
Timeline diagram				+	

**Activity, Sequence, Cooperative и State Machine диаграммы составляются на основе одного прецедента (каждый тип диаграмм - на основе своего).*
***Всё представление описывается только в случае, если в системе есть процессы, жестко привязанные к определенным моментам времени (пример - наступление нового месяца, времени суток и т.д.)*

3. Architectural Goals and Constraints (Цели и ограничения архитектуры)

[Перечислите здесь все архитектурно-значимые факторы - важные прецеденты, специфичные требования к работе системы и т.д.]

Цели:

1. **Масштабируемость:** система должна расти по числу пользователей/устройств без переработки базовой архитектуры.
2. **Модульность:** функционал разделен на модули: IoT-мониторинг нарушений, управление персоналом, работа с клиентом.
3. **Производительность:** достигнуть пропускной способности не менее 10 000 запросов в минуту на ключевых сценариях.
4. **Доступность:** Обеспечить целевой uptime ≥ 90%.
5. **Интеграционность:** Использовать REST API для взаимодействия с IoT-устройствами.

Ограничения:
Технологические:

1. **Платформа развёртывания:** Система должна запускаться на **ЭВМ Helios**.
2. **Бэкенд:** Java (LTS 8+) и Spring Boot 3.4.4; стиль — гексагональная архитектура, связь между компонентами — RESTful API.
3. **Фронтенд:** React 19, react-router-dom 6, Axios, ESLint/Prettie.
4. **Стандарты взаимодействия:** внутрисервисная коммуникация — HTTP/HTTPS; БД — реляционная (PostgreSQL).

Производительность:

1. **Нагрузка:** система должна выдерживать суммарно ≈10 000 запросов/мин.
2. **Время отклика:** для внутренних ручек — менее 300 мс; для клиентского веб-интерфейса — до 3 сек.
3. **Доступность:** целевой uptime ≥ 90%; окно профилактики 02:00–06:00; восстановление после сбоя ≤ 10 мин.

Функциональные:

1. **Обязательные модули:** реализация трёх модулей: IoT-мониторинг нарушений, управление персоналом, работа с клиентом.
2. **Интеграции и внешние интерфейсы:** REST-API для фронтенда и взаимодействия с IoT.

Временные:

1. **Общий дедлайн и ключевые вехи:** целевое завершение проекта — **29.12.2025,** предусмотрены поэтапные релизы/версии и тестирование.

Ресурсные:

1. **Состав команды:** 3 человека.
2. **Бюджет:** бюджет проекта зафиксирован в Business Case, контроль исполнения — согласно SDP.

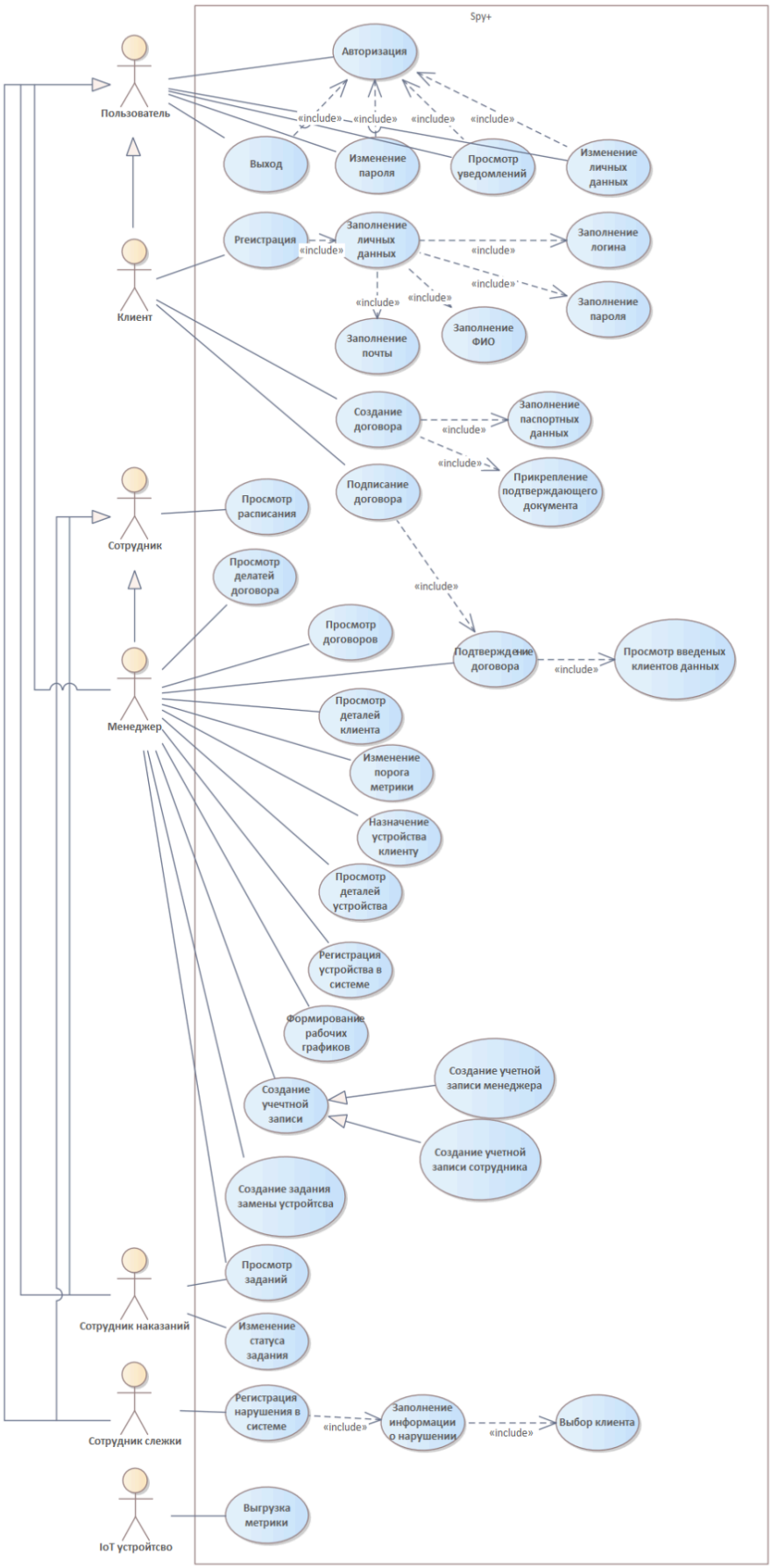
Важные прецеденты:

1. **Сотрудник слежки / IoT-мониторинг** — приём метрик с устройств, обработка порогов и генерация алертов.
2. **Менеджер / Управление персоналом** — назначение и перераспределение заданий, контроль выполнения.
3. **Сотрудник исполнения наказаний** — фиксация результата в системе.
4. **Клиент/внешний пользователь** — заключение/просмотр договора, отправка заявок и получение уведомлений.

4. Use-Case View

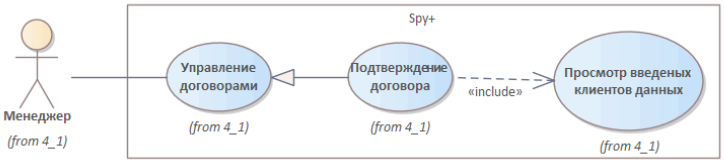
[Данный раздел содержит описание основных сценариев использования системы разными типами пользователей. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

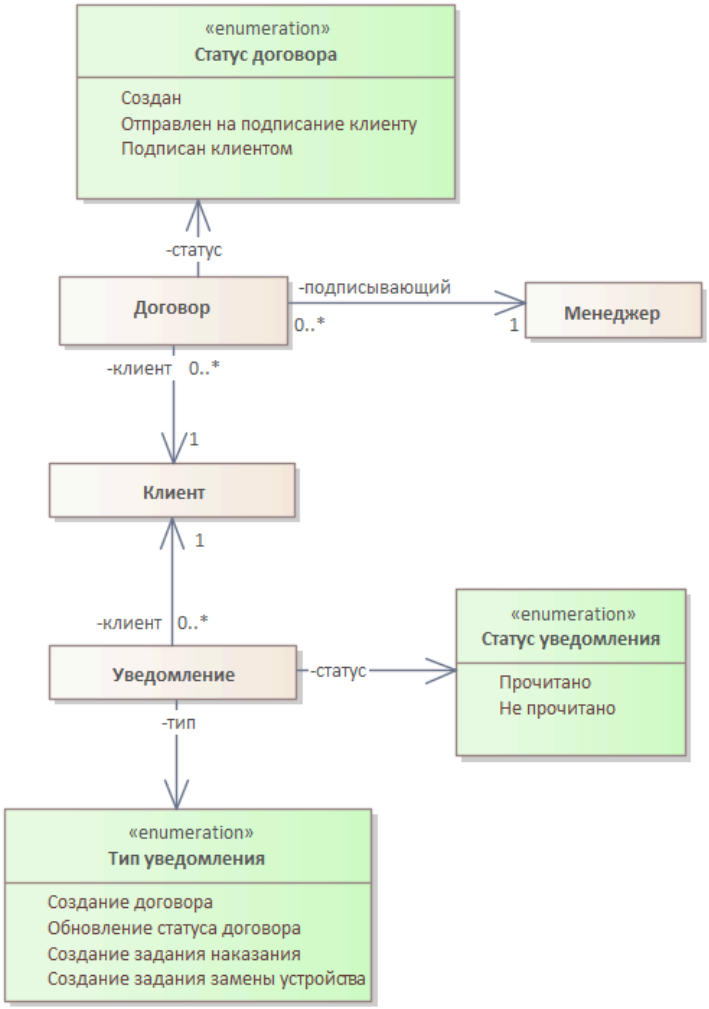
4.1 Use Case Diagram


Spy+
Пользователь
Авторизация
«include»
«include»
«include»
«include»
Выход
Изменение пароля
Просмотр уведомлений
Изменение личных данных
Клиент
Ргеистрация
«include»
Заполнение личных данных
«include»
Заполнение логина
«include»
Заполнение пароля
«include»
«include»
Заполнение почты
Заполнение ФИО
Создание договора
«include»
Заполнение паспортных данных
«include»
Прикрепление подтверждающего документа
Подписание договора
Сотрудник
Просмотр расписания
Просмотр делатей договора
«include»
Менеджер
Просмотр договоров
Подтверждение договора
«include»
Просмотр введеных клиентов данных
Просмотр деталей клиента
Изменение порога метрики
Назначение устройства клиенту
Просмотр деталей устройства
Регистрация устройства в системе
Формирование рабочих графиков
Создание учетной записи менеджера
Создание учетной записи
Создание учетной записи сотрудника
Создание задания замены устройтсва
Сотрудник наказаний
Просмотр заданий
Изменение статуса задания
Сотрудник слежки
Регистрация нарушения в системе
«include»
Заполнение информации о нарушении
«include»
Выбор клиента
IoT устройтсво
Выгрузка метрики

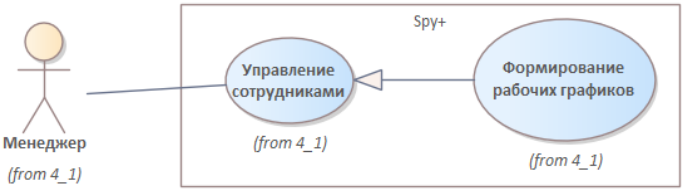
4.2 Class Diagram

Подтверждение договора менеджером



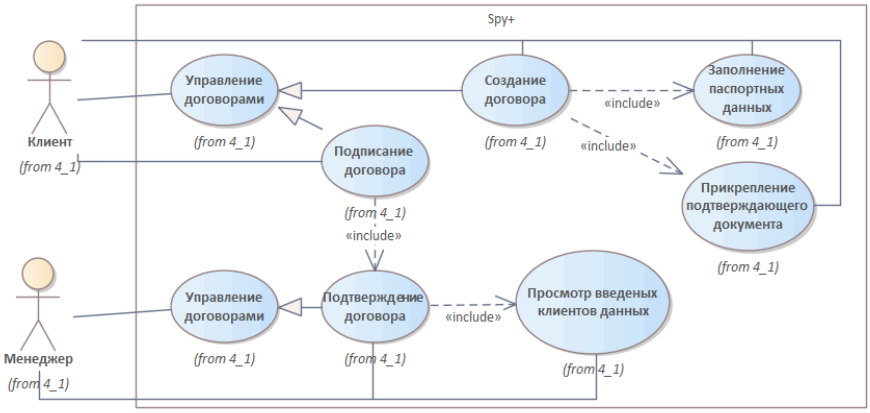


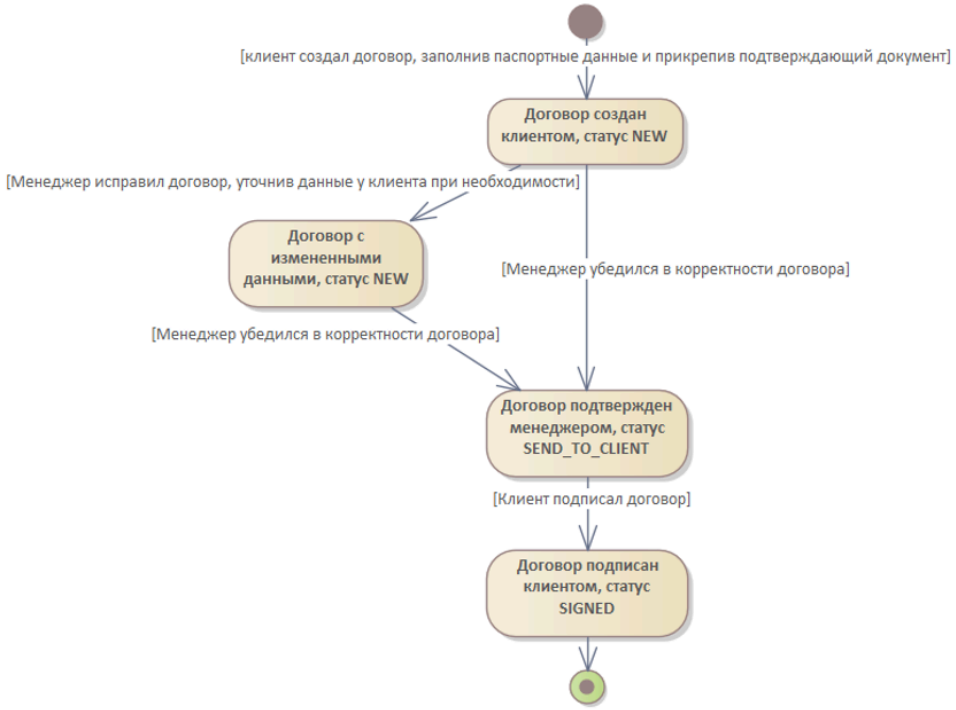
4.3 Activity Diagram



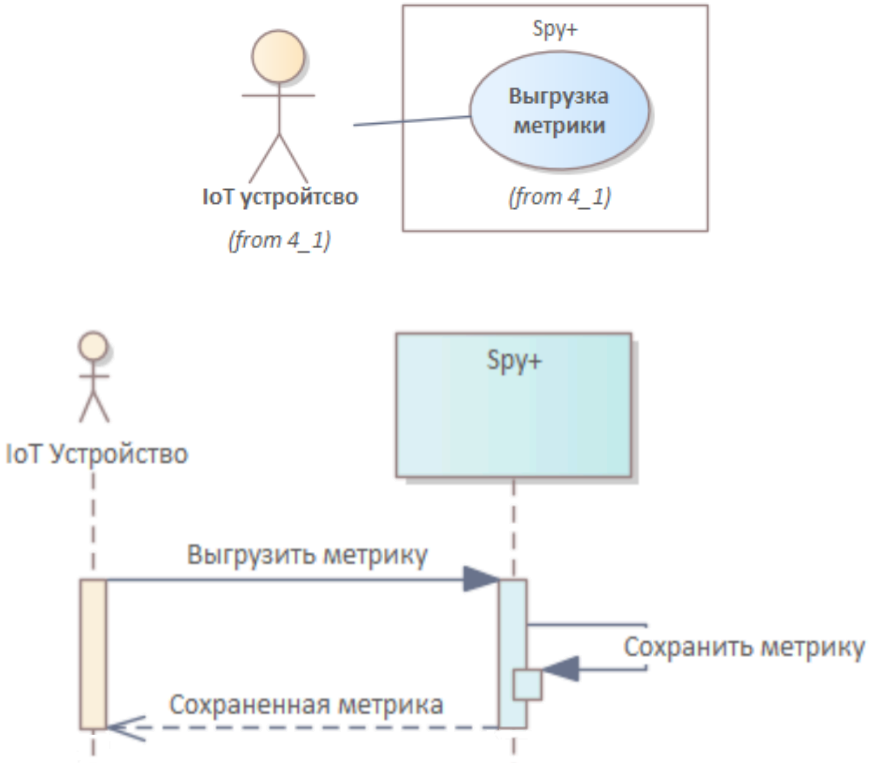
4.4 State Machine Diagram

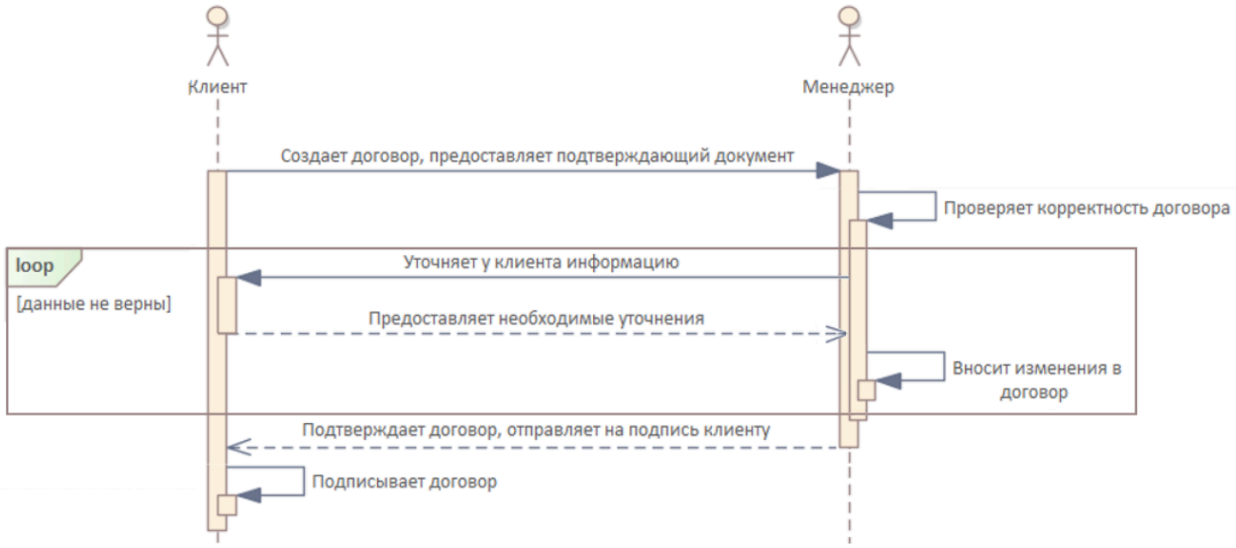
Изменение статуса договора



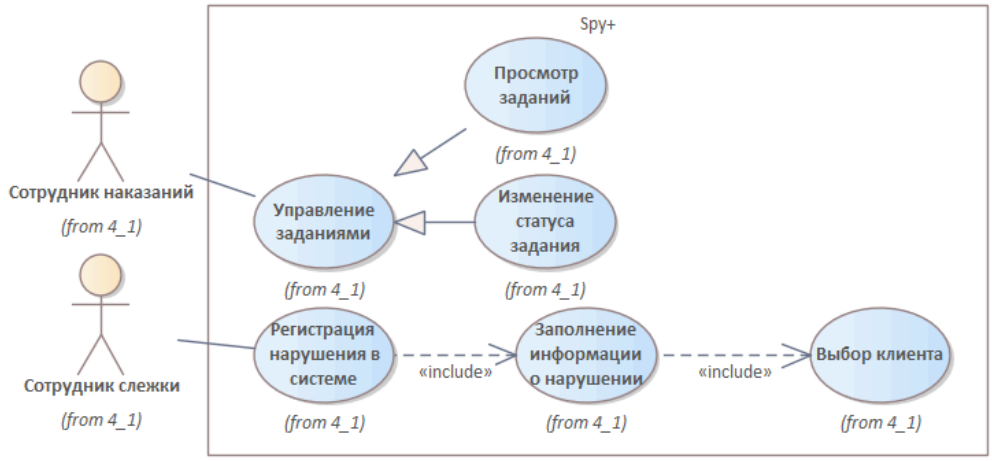


4.5 Sequence Diagram (Отправка метрик IOT-устройством)





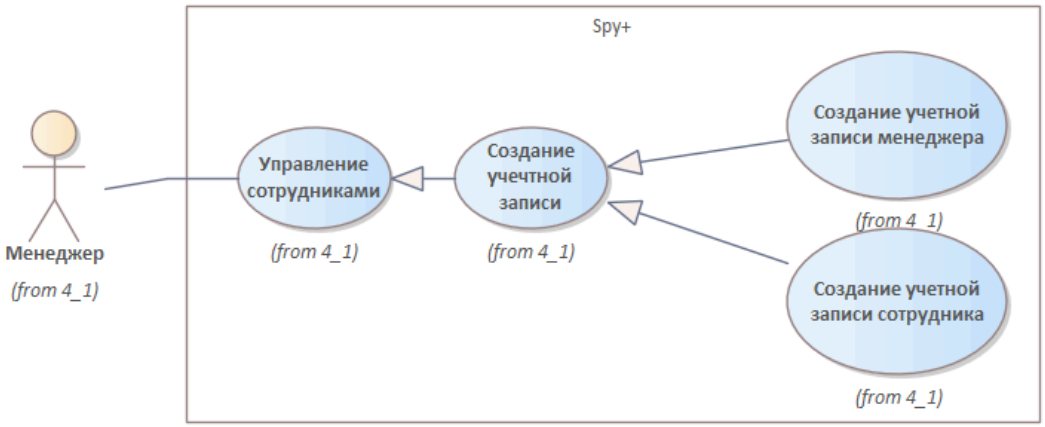
4.6 Cooperative Diagram
Регистрация задания в системе

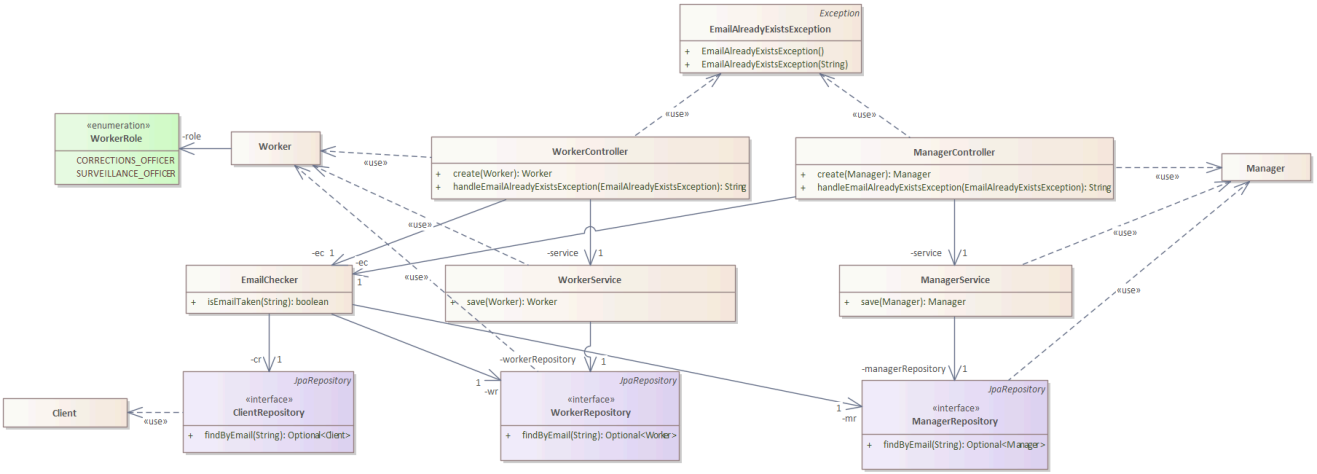


5. Logical View

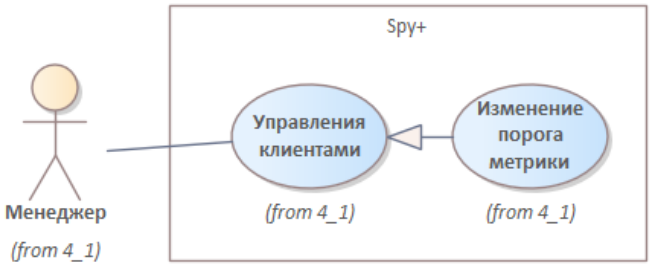
[Данный раздел содержит описание слоев, на которые делится приложение, а также интерфейсов их взаимодействия. Приведите описание каждого из слоев, как они связаны между собой, их назначение. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

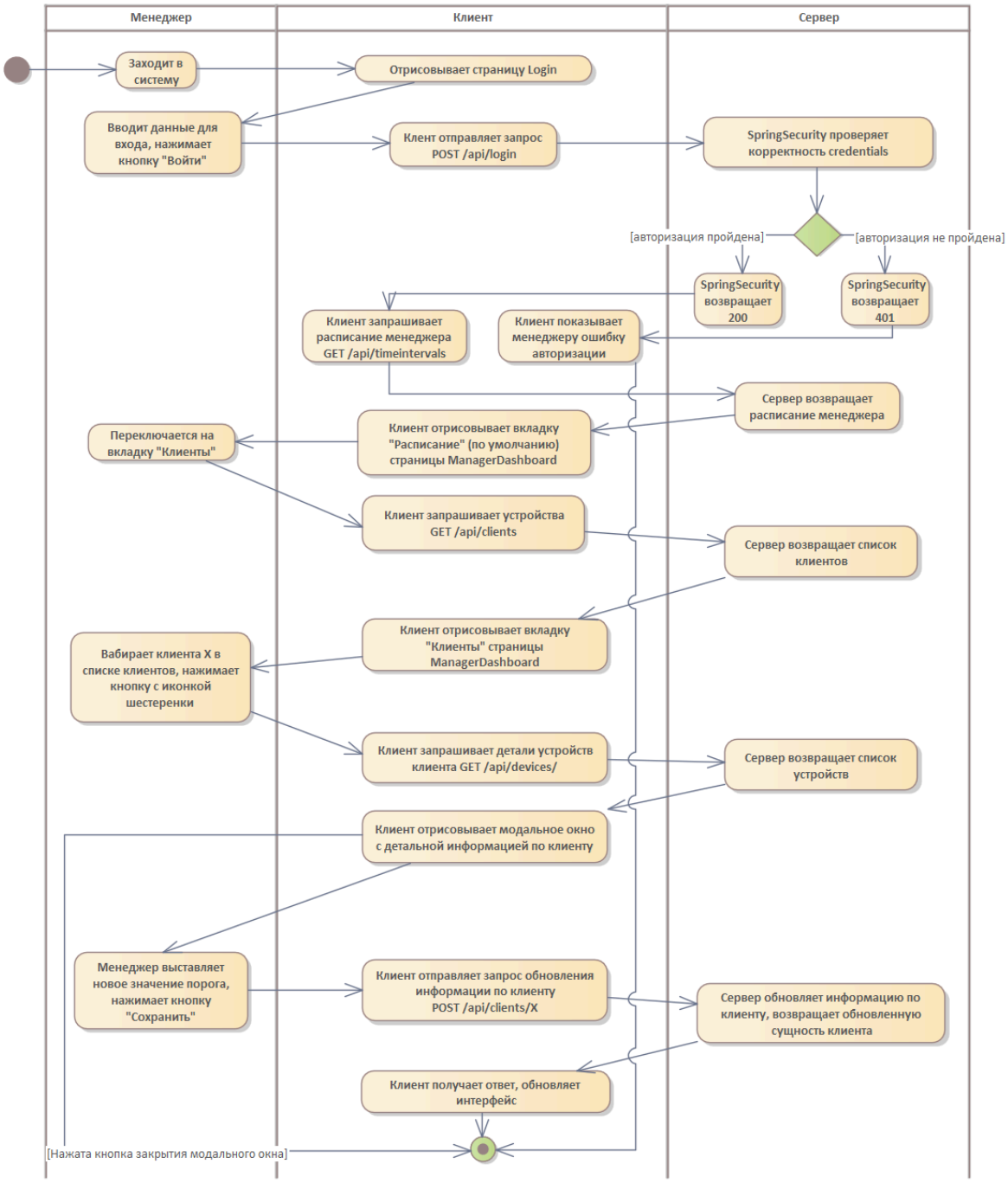
5.1 Class Diagram



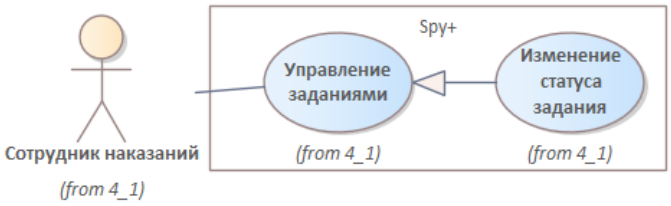


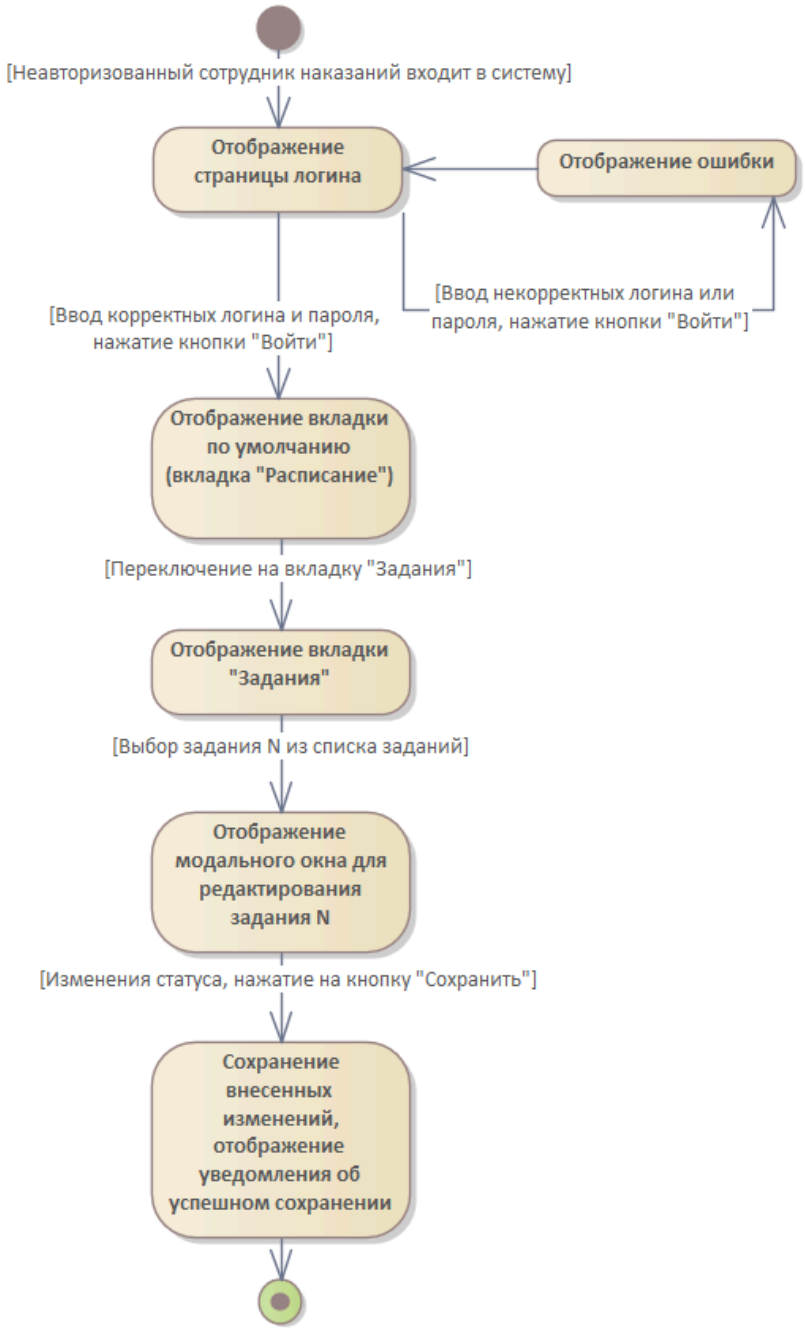
5.2 Activity Diagram



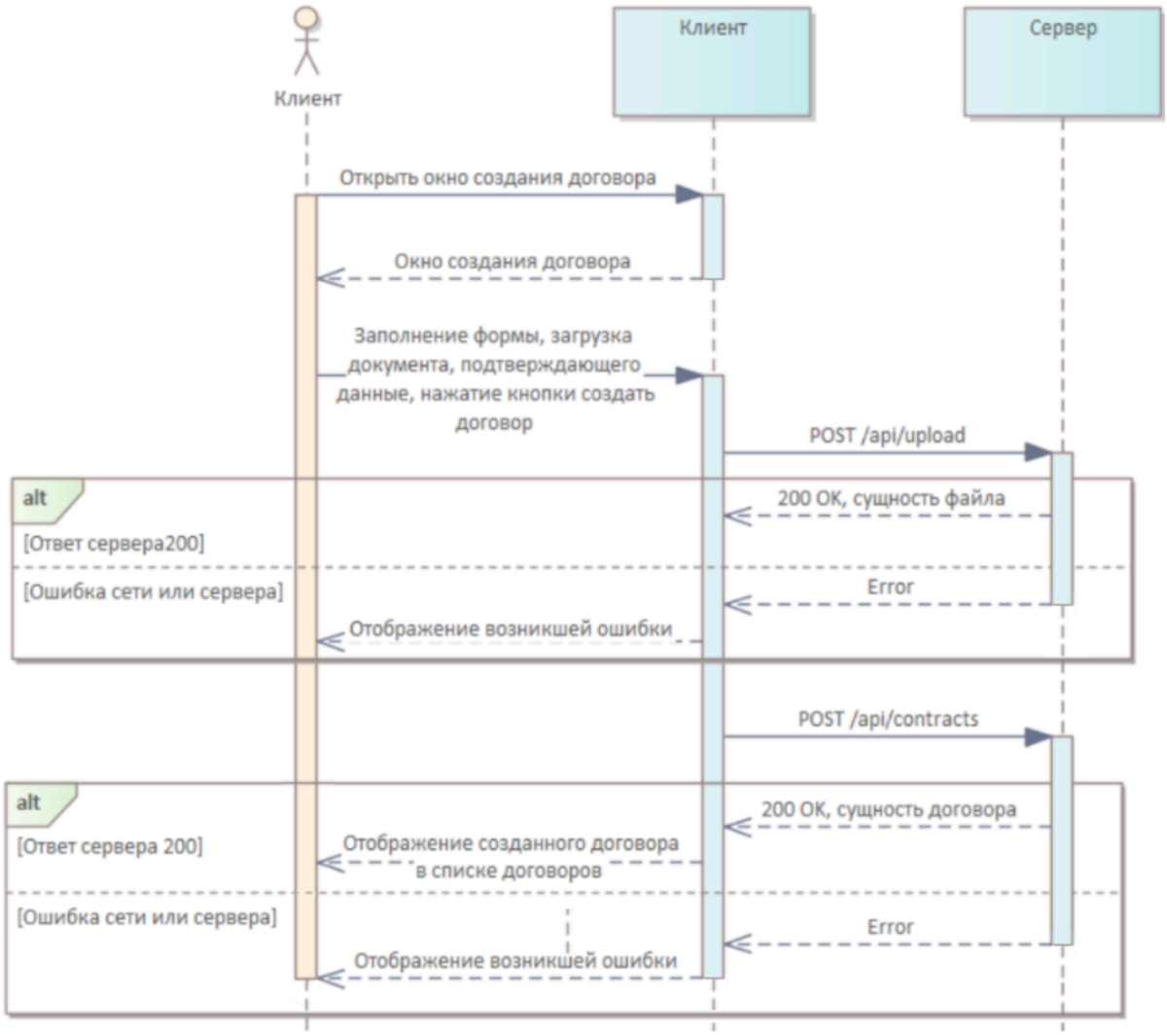


5.3 State Machine Diagram

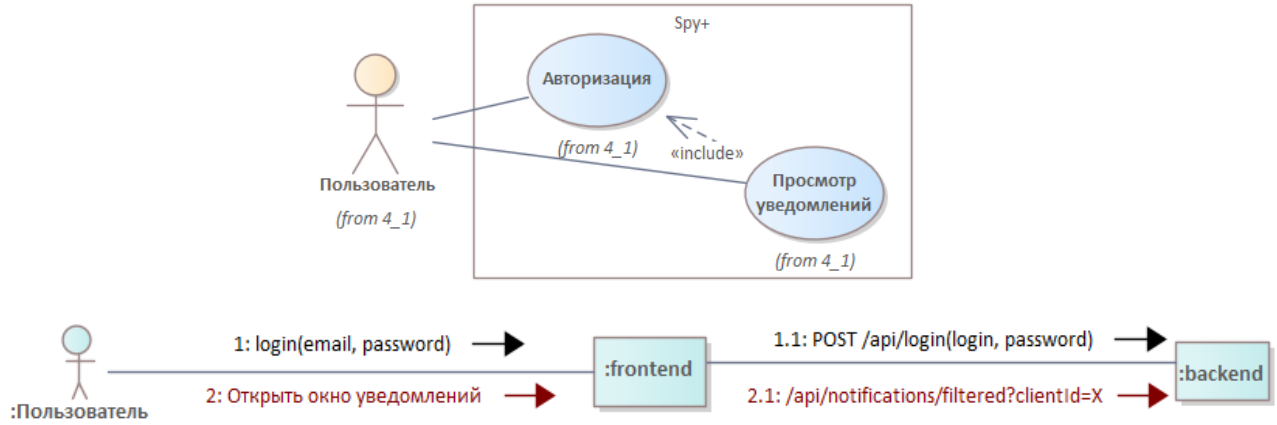




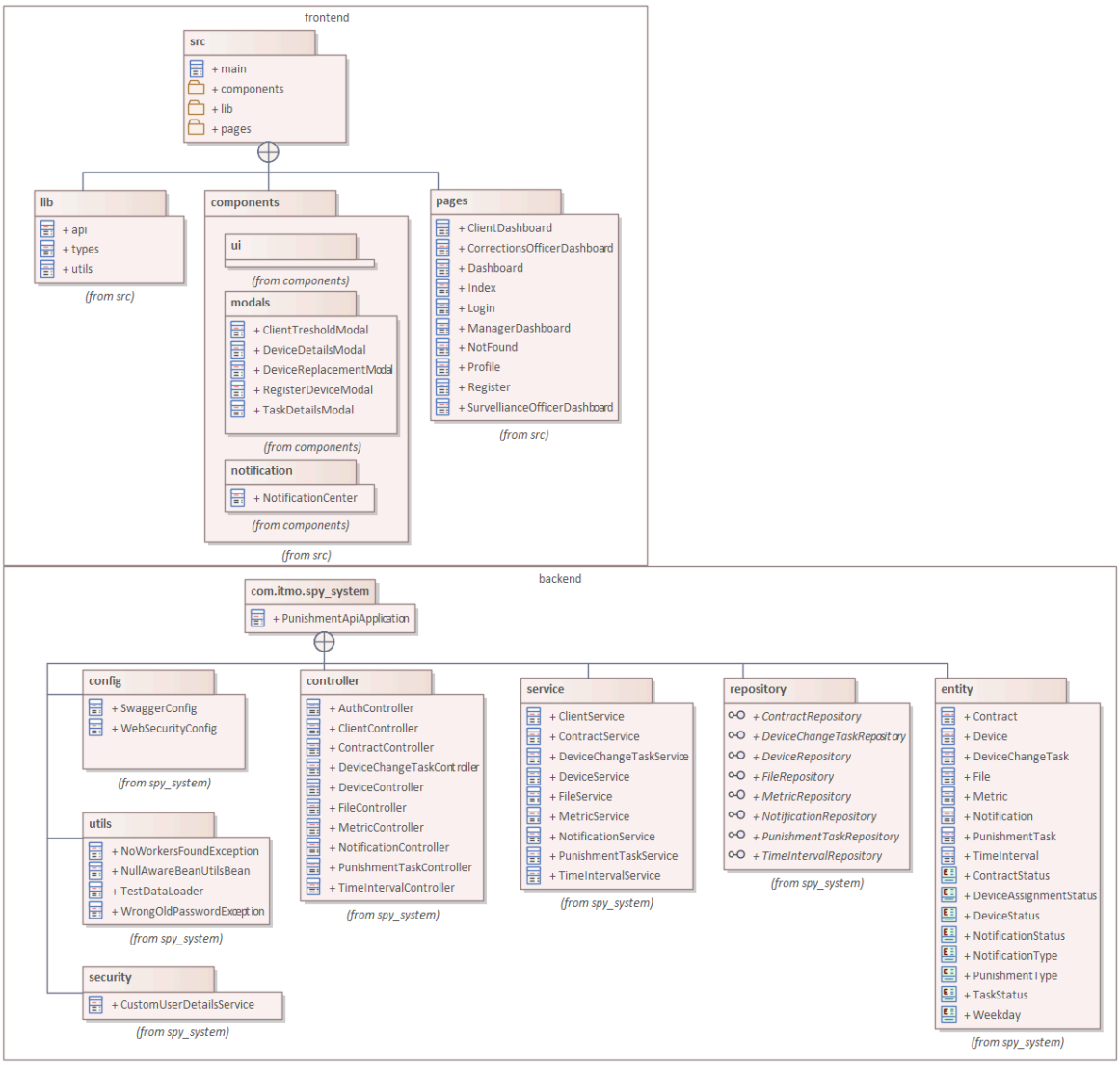
5.4 Sequence Diagram (Создание договора клиентом)



5.5 Cooperative Diagram
Просмотр уведомлений пользователем



5.6 Package Diagram



6. Process View

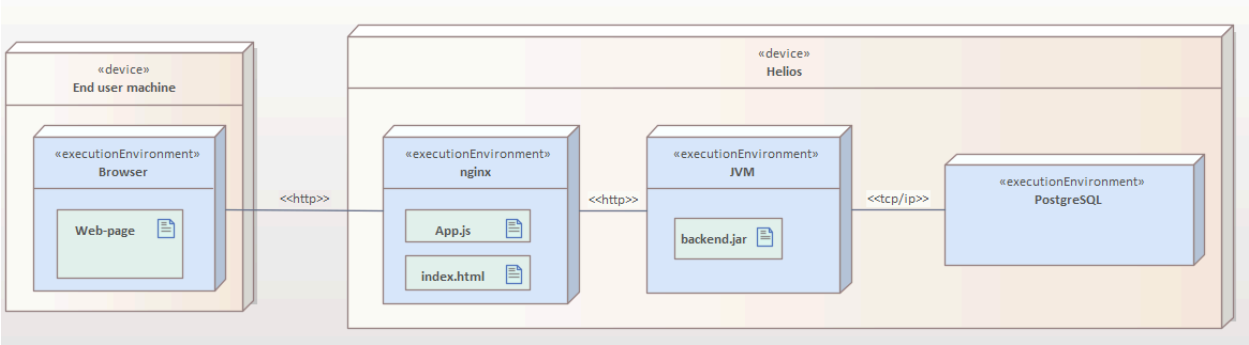
[Данный раздел описывает систему как совокупность взаимодействующих процессов, привязанных к определенным моментам времени. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

Все процессы не имеют жесткой привязки к определенным моментам времени.

7. Deployment View

[Данный раздел содержит описание конфигурации файлов, из которых состоит система, мест их расположения и описание взаимодействия их друг с другом. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

7.1 Deployment Diagram

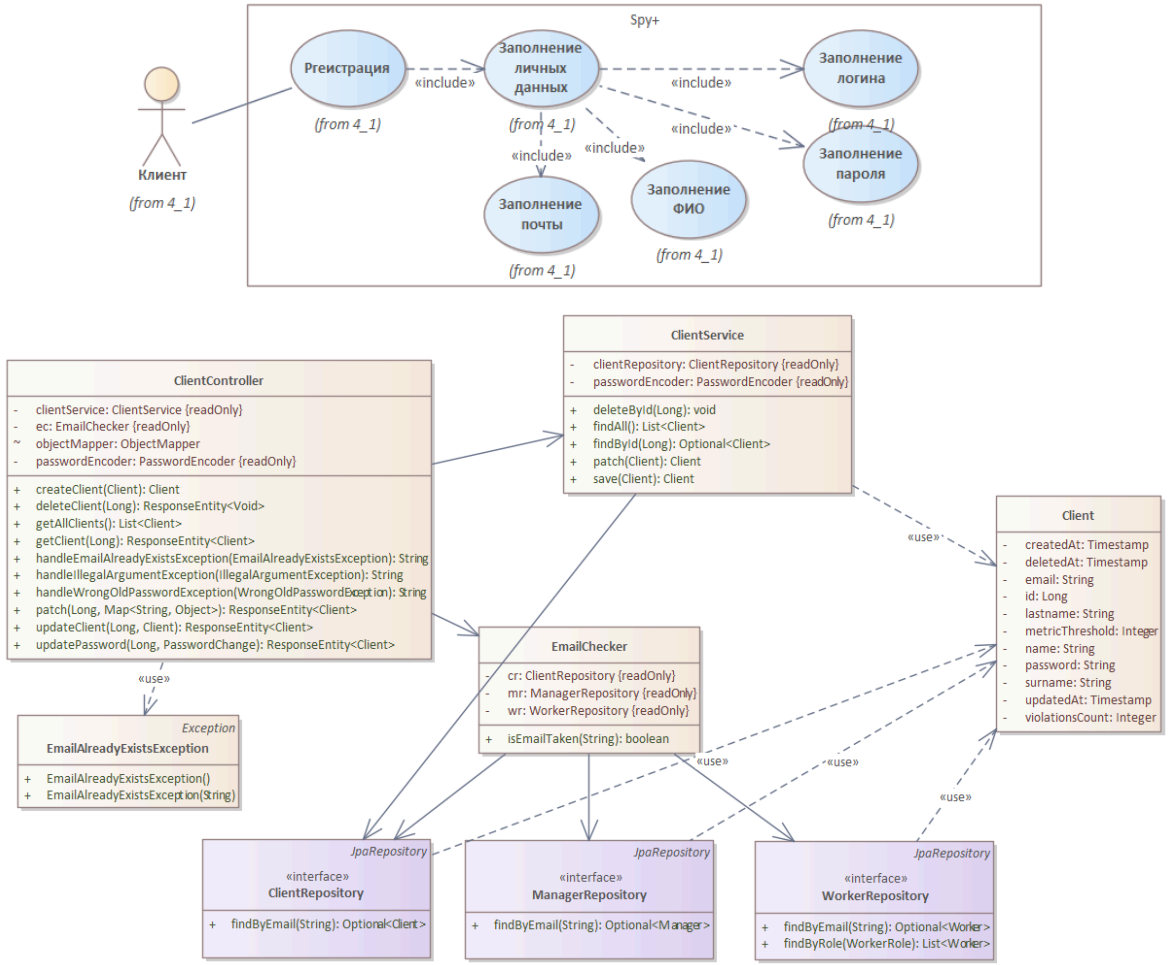


8. Implementation View

[Данный раздел содержит описание системы в уже реализованном виде. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

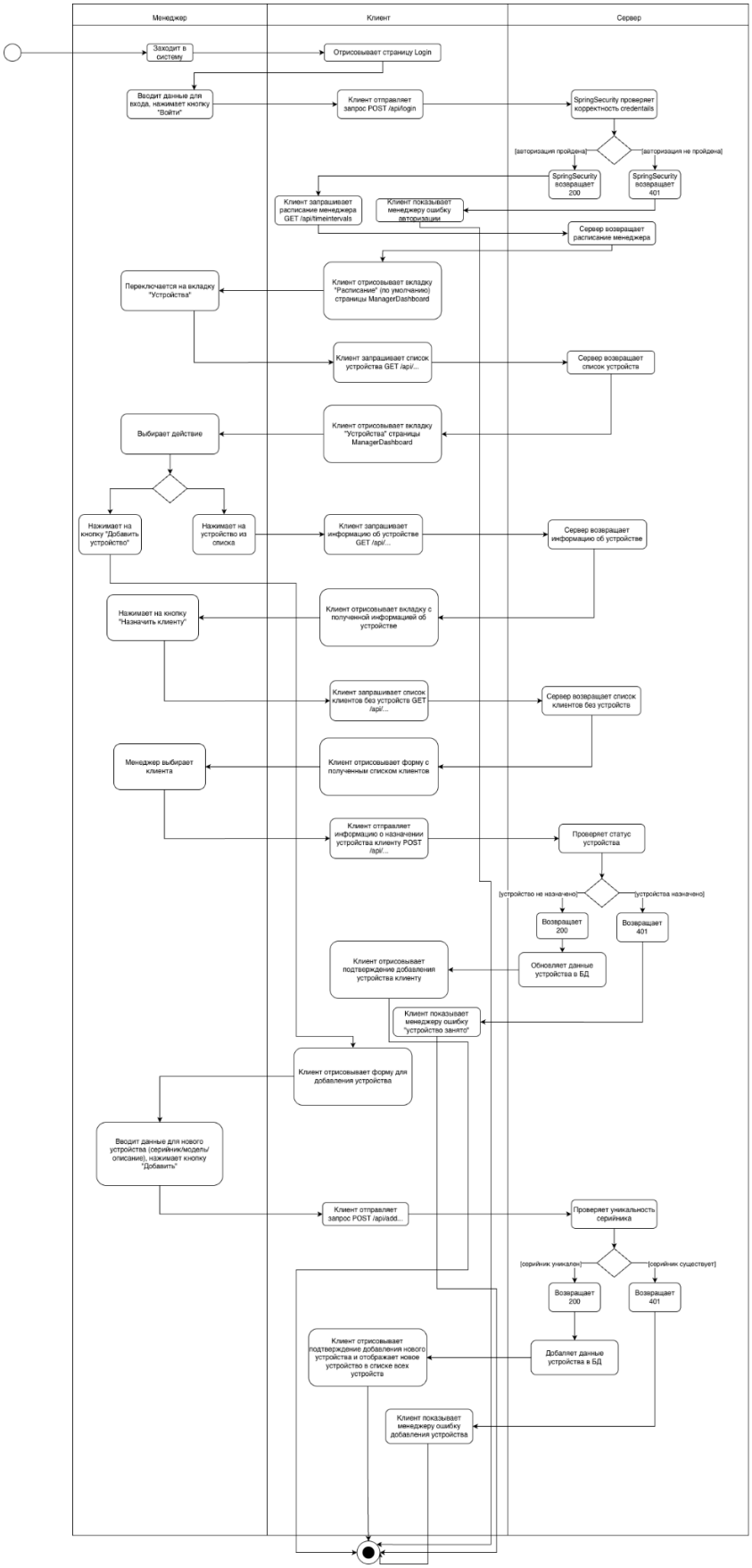
8.1 Class Diagram

Регистрация клиента

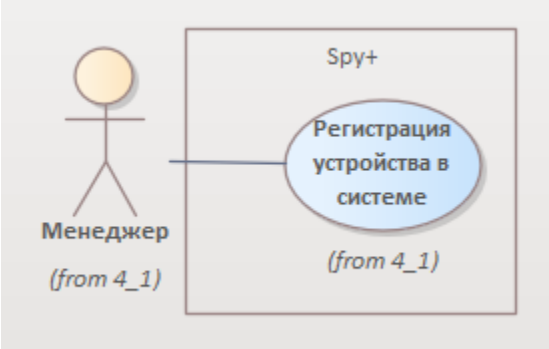


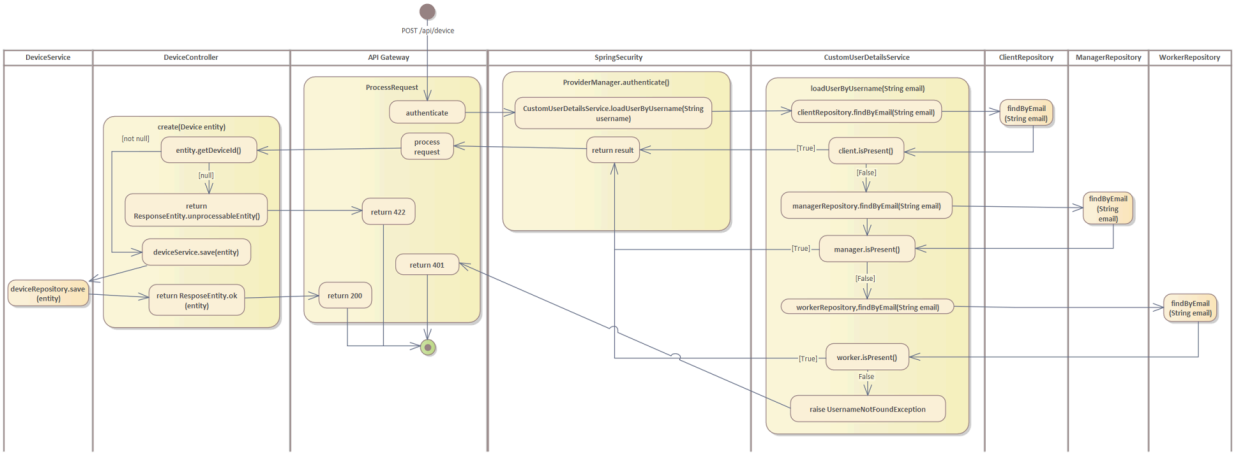
8.2 Activity Diagram

Управление устройствами менеджером

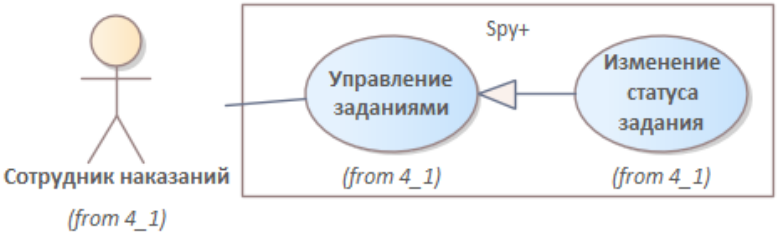

Менеджер
Клиент
Сервер
Заходит в систему
Отрисовывает страницу Login
Вводит данные для входа, нажимает кнопку "Войти"
Клиент отправляет запрос POST /api/login
SpringSecurity проверяет корректность credentials
[авторизация пройдена]
[авторизация не пройдена]
SpringSecurity возвращает 200
SpringSecurity возвращает 401
Клиент запрашивает расписание менеджера GET /api/timeintervals
Клиент показывает менеджеру ошибку авторизации
Сервер возвращает расписание менеджера
Переключается на вкладку "Устройства"
Клиент отрисовывает вкладку "Расписание" (по умолчанию) страницы ManagerDashboard
Клиент запрашивает список устройства GET /api/...
Сервер возвращает список устройств
Выбирает действие
Клиент отрисовывает вкладку "Устройства" страницы ManagerDashboard
Нажимает на кнопку "Добавить устройство"
Нажимает на устройство из списка
Клиент запрашивает информацию об устройстве GET /api/...
Сервер возвращает информацию об устройстве
Нажимает на кнопку "Назначить клиенту"
Клиент отрисовывает вкладку с полученной информацией об устройстве
Клиент запрашивает список клиентов без устройств GET /api/...
Сервер возвращает список клиентов без устройств
Менеджер выбирает клиента
Клиент отрисовывает форму с полученным списком клиентов
Клиент отправляет информацию о назначении устройства клиенту POST /api/...
Проверяет статус устройства
[устройство не назначено]
[устройства назначено]
Возвращает 200
Возвращает 401
Клиент отрисовывает подтверждение добавления устройства клиенту
Обновляет данные устройства в БД
Клиент показывает менеджеру ошибку "устройство занято"
Клиент отрисовывает форму для добавления устройства
Вводит данные для нового устройства (серийник/модель/описание), нажимает кнопку "Добавить"
Клиент отправляет запрос POST /api/add...
Проверяет уникальность серийника
[серийник уникален]
[серийник существует]
Возвращает 200
Возвращает 401
Клиент отрисовывает подтверждение добавления нового устройства и отображает новое устройство в списке всех устройств
Добавляет данные устройства в БД
Клиент показывает менеджеру ошибку добавления устройства

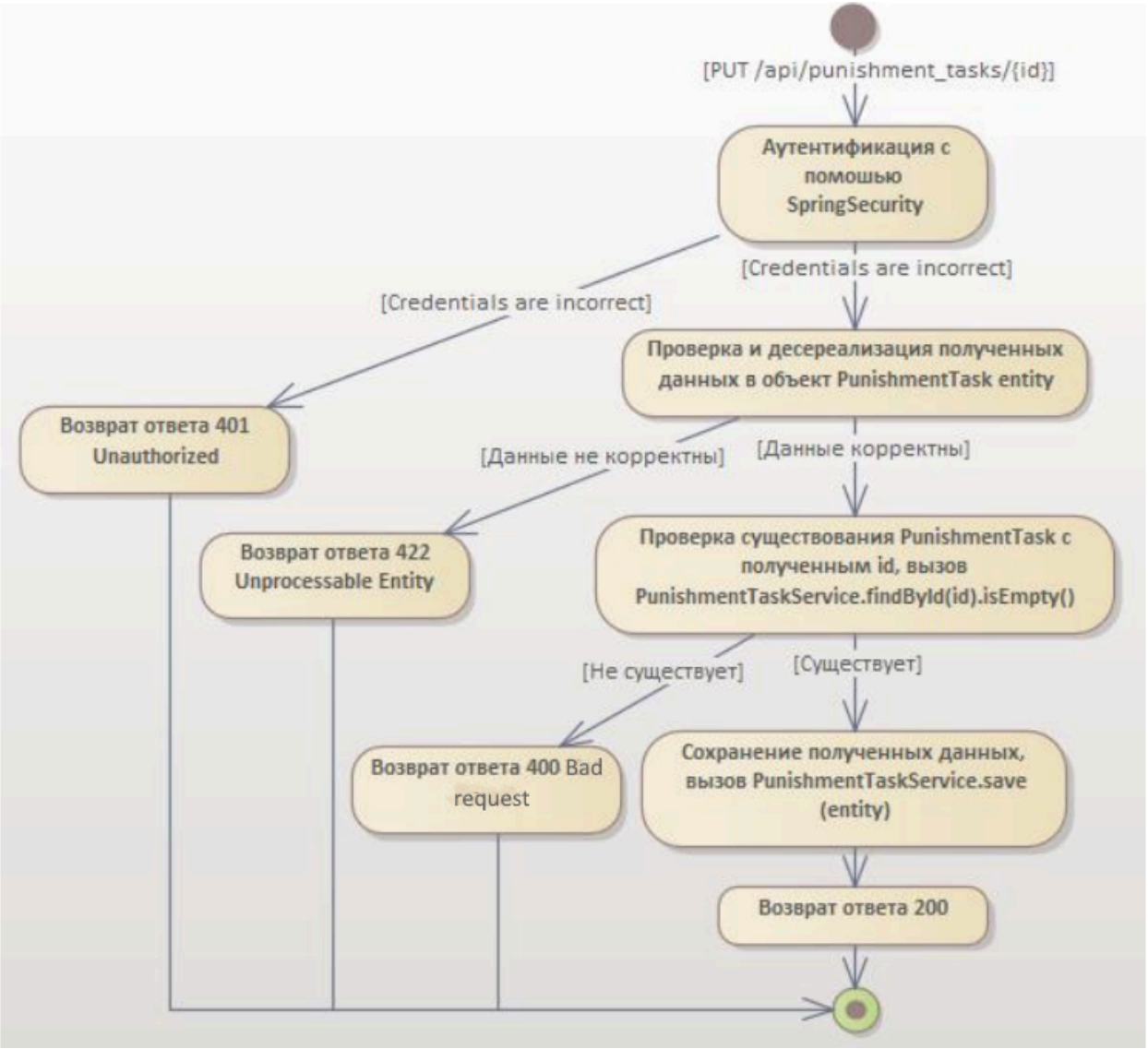
Регистрация устройства менеджером



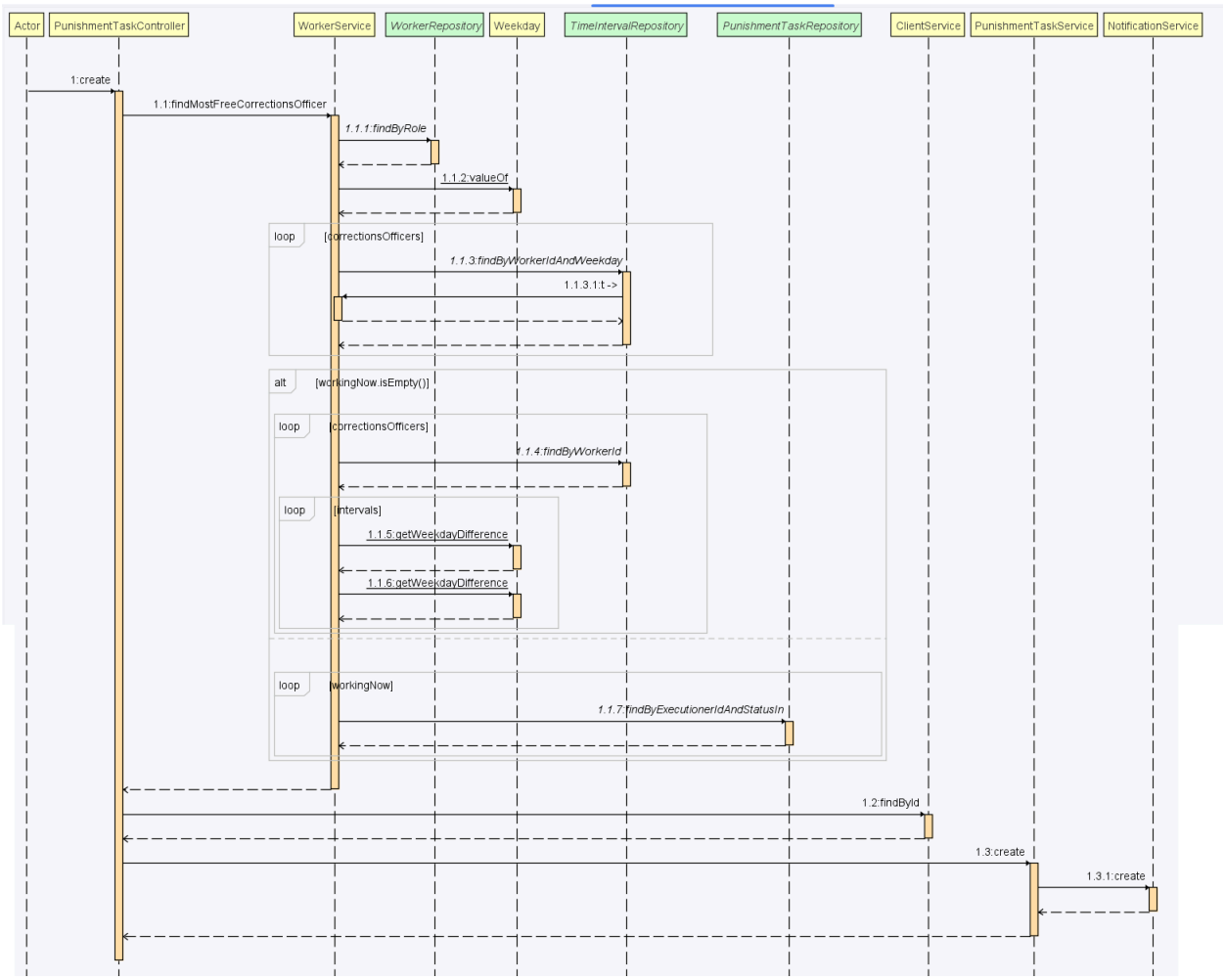


8.3 State Machine Diagram

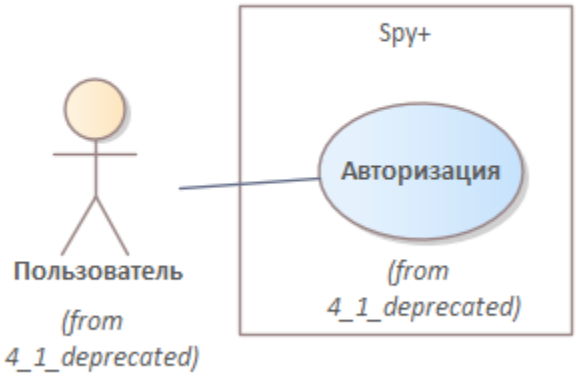


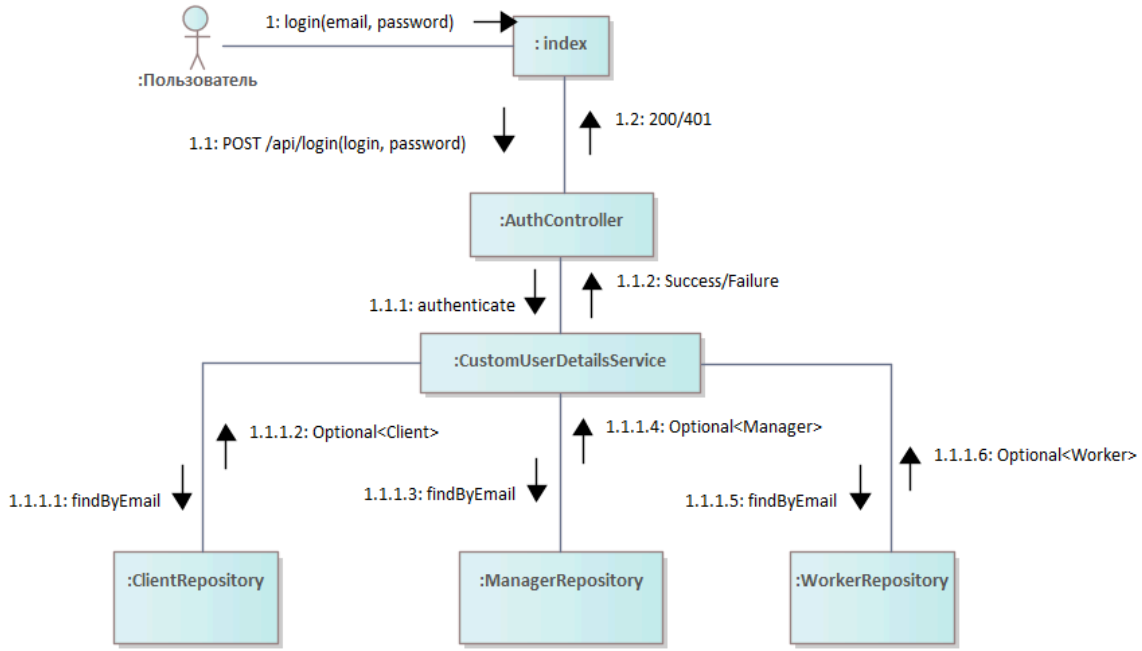


8.4 Sequence Diagram
регистрация сотрудником слежки нарушения в системе

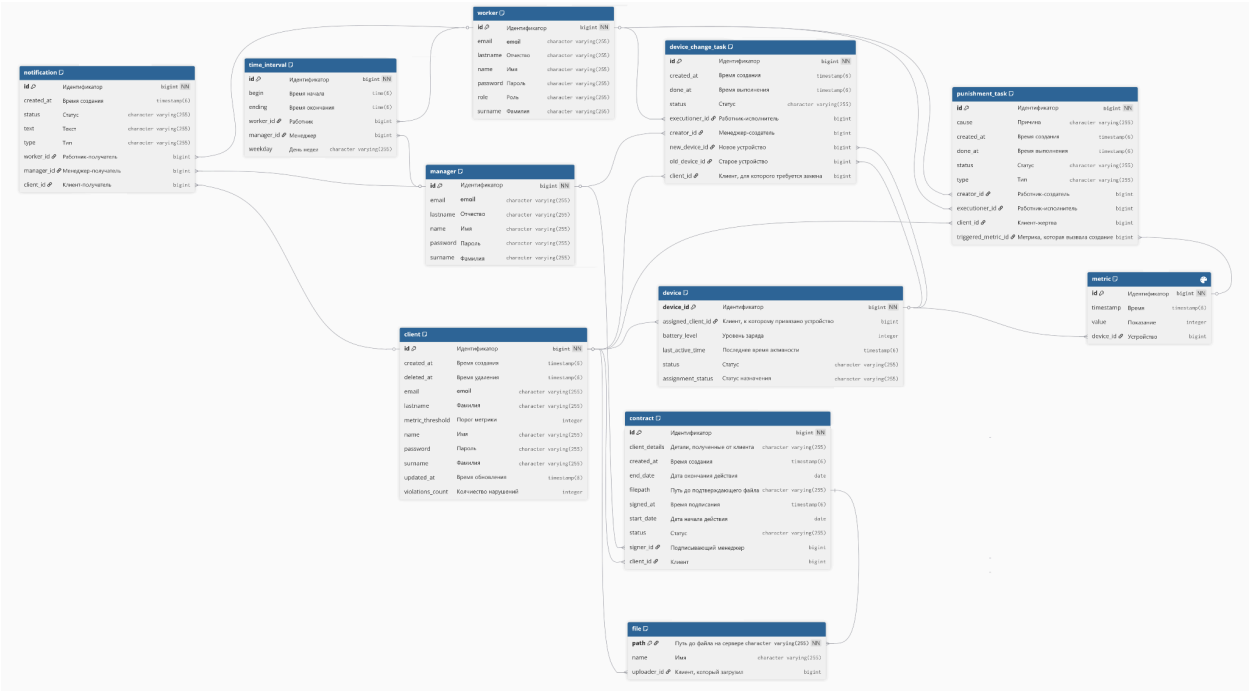


8.5 Cooperative Diagram





8.6 DB Diagram



9. Size and Performance (Производительность)

Данный раздел описывает основные характеристики измерения производительности системы и их границы, которые могут оказать влияние на архитектуру,]

Система должна выдерживать суммарно ≈10 000 запросов/мин (~167 rps).

Ограничение по времени ответа для внутренних ручек — менее 300 мс, для клиентского веб-интерфейса — до 3 сек.

10. Quality (Качество)

Данный раздел описывает, каким образом архитектура системы удовлетворяет её показателям качества - масштабируемости, надежности, мобильности, безопасности и т.д.]

Максимальное время восстановления — не более 10 минут.

Доступность системы (uptime ≥ 90%). Допустимый суммарный простой:

1. в день — не более 2 ч 24 мин;
2. в месяц — не более 72 часов;
3. окно профилактики — 02:00–06:00 (по необходимости).

Система хранит пароли пользователей только в виде хэшей в базе данных PostgreSQL.