АВТОРЫ ДОКУМЕНТА
КСЕНИЯ КАМЫШАНСКАЯ
ЕВГЕНИЙ ГУРИН
ПАВЕЛ ЕФАРИНОВ

Software Architecture Document

1. Introduction (Введение)

1.1 Purpose

Назначение данного документа описать архитектуру проекта САС-ИП-СА.

1.2 Scope (Область применения)

Данный документ описывает архитектуру приложения *САС-ИП-СА* как набор точек зрения на неё - представление use case, логическое представление, представление процессов, представление развертывания и представление реализации. Взаимодействие элементов разных точек зрения представлено в виде UML-диаграмм.

Данный документ относится к проекту “Производство передвижных бань в Княжестве Новгородском”, разрабатываемого командой из студентов ИТМО. Проект автоматизирует производственный процесс бань, включающий в себя создание каркаса, взаимодействие с подрядчиками, установку оборудования

1.3 Definitions, Acronyms and Abbreviations (Определения и аббревиатуры)

Определения и аббревиатуры перечислены в Glossary

1.4 References (Ссылки)

Лекции Клименкова С.В.
UML 2 and the Unified Process
Сказка Царевна Лягушка
Гуси Лебеди
Фольклор с участием Бабы Яги

2. Architectural Representation (Представление архитектуры)

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 диаграммой пакетов)	+ (Полное описание прецедента с указанием вызываемых методов, 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 diagramm				+	

**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.)*

** в каждом представлении по одной диаграмме*

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.]

Существуют некоторые ключевые требования и системные ограничения, которые оказывают существенное влияние на архитектуру:

2.4.1 Сервер будет разворачиваться на оборудовании с подключенным интернет соединением со скоростью от 10 МБ\с

*2.4.2 Программное обеспечение будет разворачиваться на сервере **Helios.** В связи с этим*

2.4.3 Программное обеспечение будет состоять из серверного модуля и модуля пользовательского интерфейса

*2.4.4 Серверная часть программного обеспечения будет написана на языке **Python***

*2.4.4 Пользовательский интерфейс программного обеспечения будет написан на языке **JavaScript***

*2.4.5 В качестве базы данных будет использоваться **Postgres***

*2.4.6 Пользователи будут пользоваться с браузера **Firefox >= 110.0.1**, **Google Chrome >= 110.0***

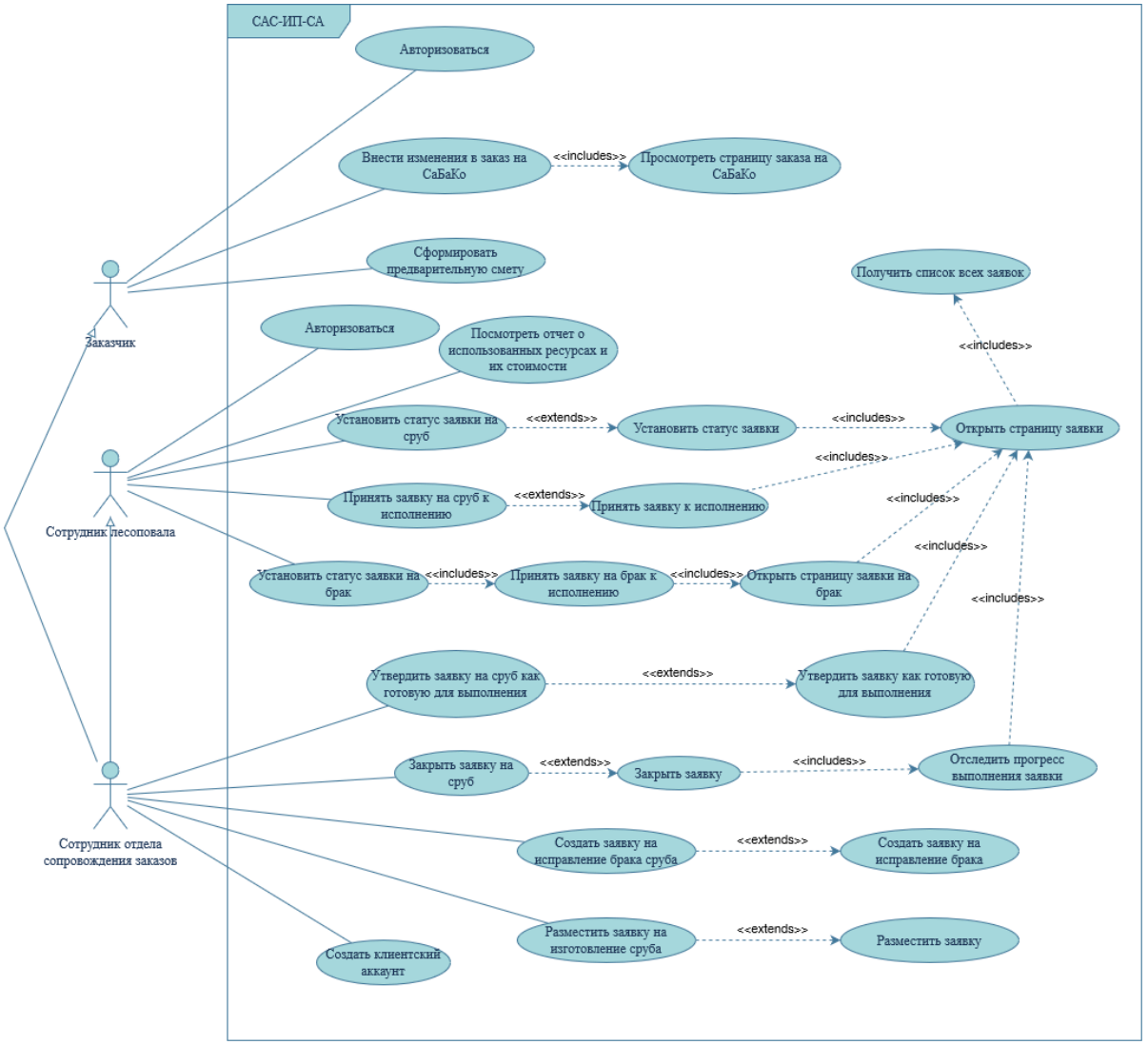
2.4.7 Для использования системы требуется стабильное интернет соединение

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.]

Тут видимо диаграммы каждого юз кейса

4.1. Use Case Diagram



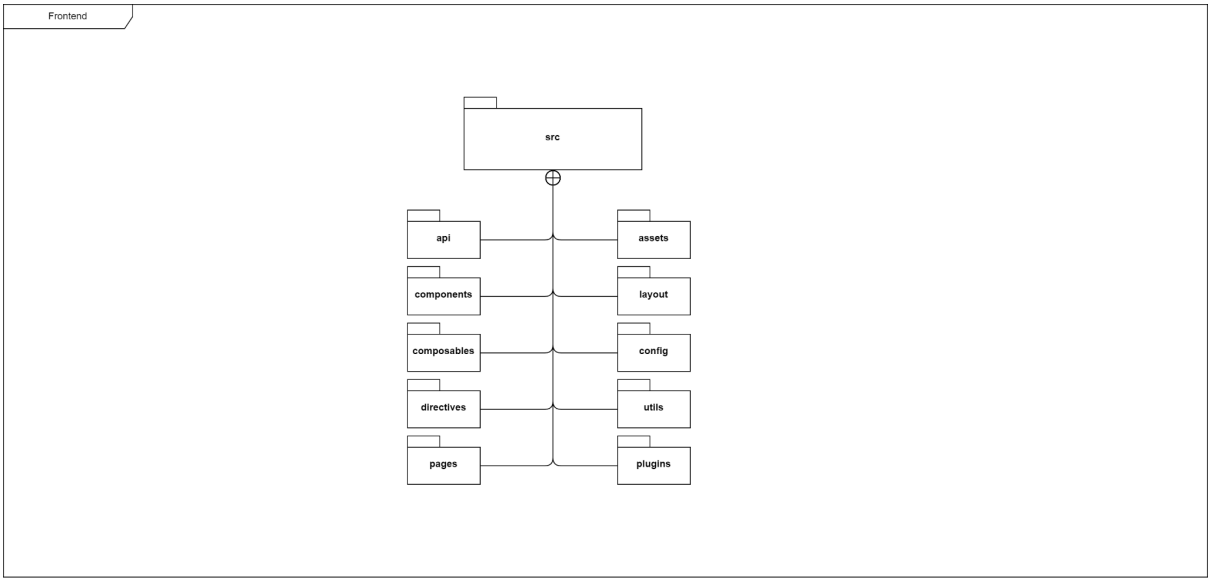
Описание use-cases архитектуры программного обеспечения. В данной главе описываются сценарии использования, на основе которых разрабатывается функциональность. Описывается набор сценариев, которые влияют на разработку архитектуры или на конкретный её аспект.

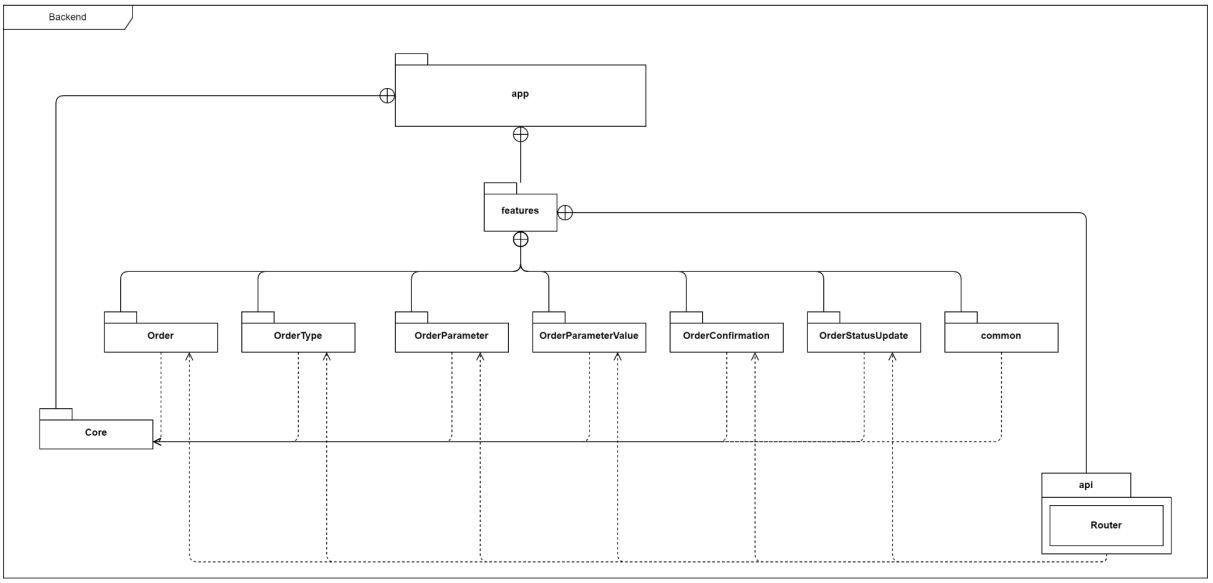
Набор сценариев:

- Авторизация
- Создать клиентский аккаунт
-
- Создание заказа на СаБаКо и указание требований
- Изменение заказа СаБаКо

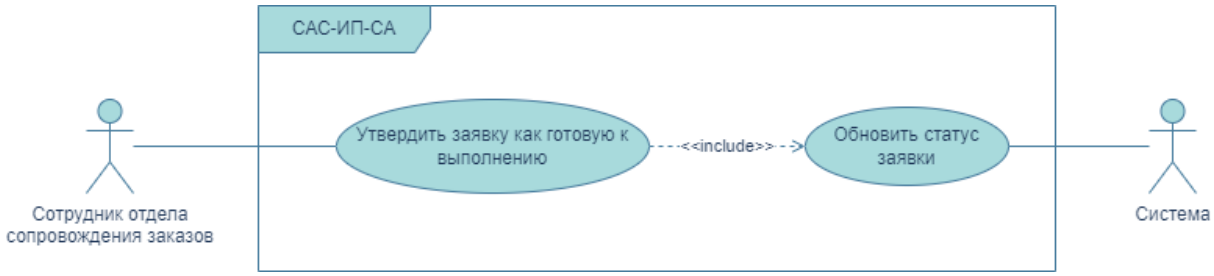
- Получить информацию о существующих заказах СаБаКо
- Просмотреть заказ
-
- Создание заявки на компонент\брак СаБаКо
- Добавить материалы (в т.ч указать стоимость)
- Изменение заявки на компонент СаБаКо
- Указать материалы, используемые при выполнении заявки
- Утверждение заявки на компонент СаБаКо как готовой для выполнения
- Установить статус заявки на компонент СаБаКо
- Принять к исполнению
- Просмотреть заявку
- Посмотреть все заявки
- Получить прогресс по заявке
-
- Передача изделия в доставку
- Согласовать время и дату доставки
- Просмотр отчета об использованных ресурсах

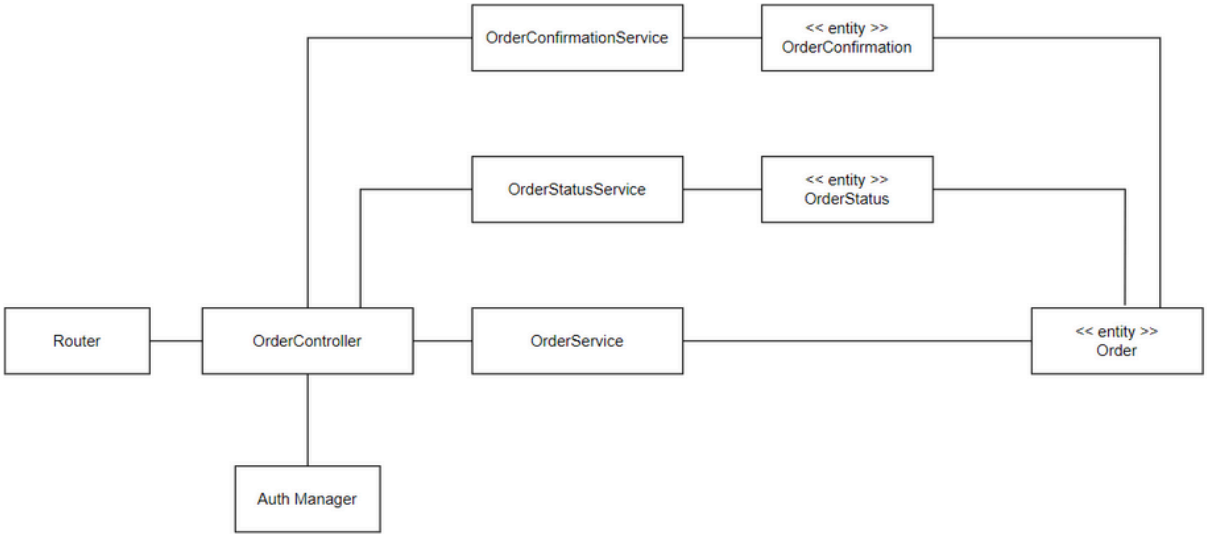
4.2. Package Diagram



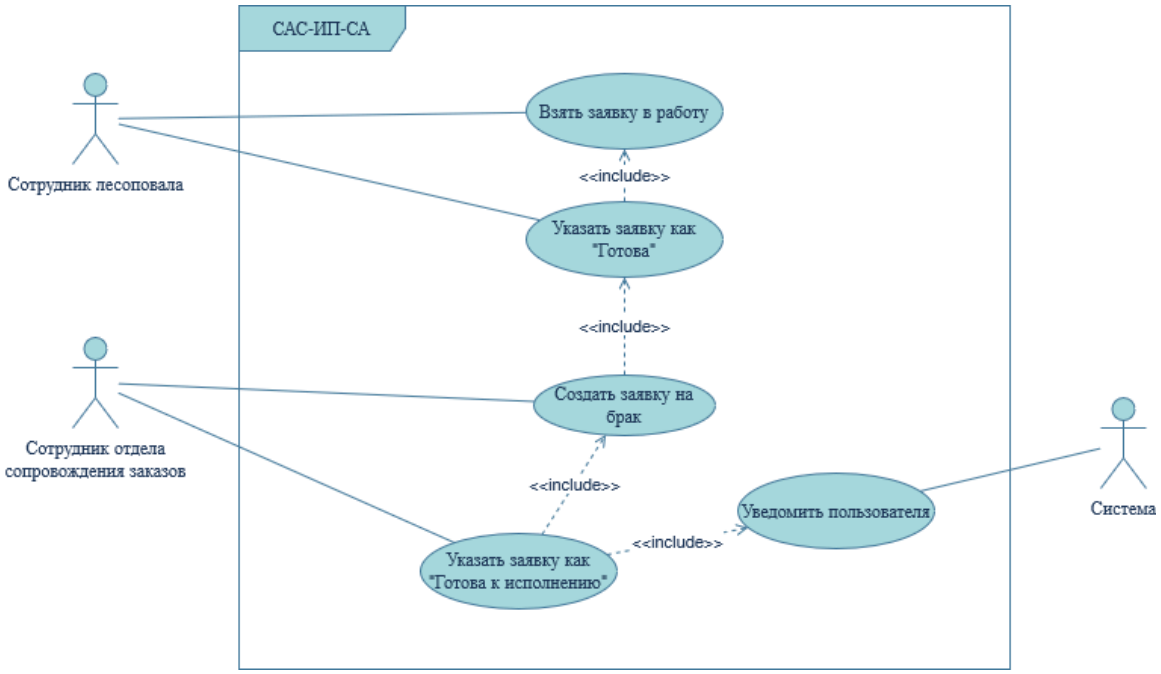


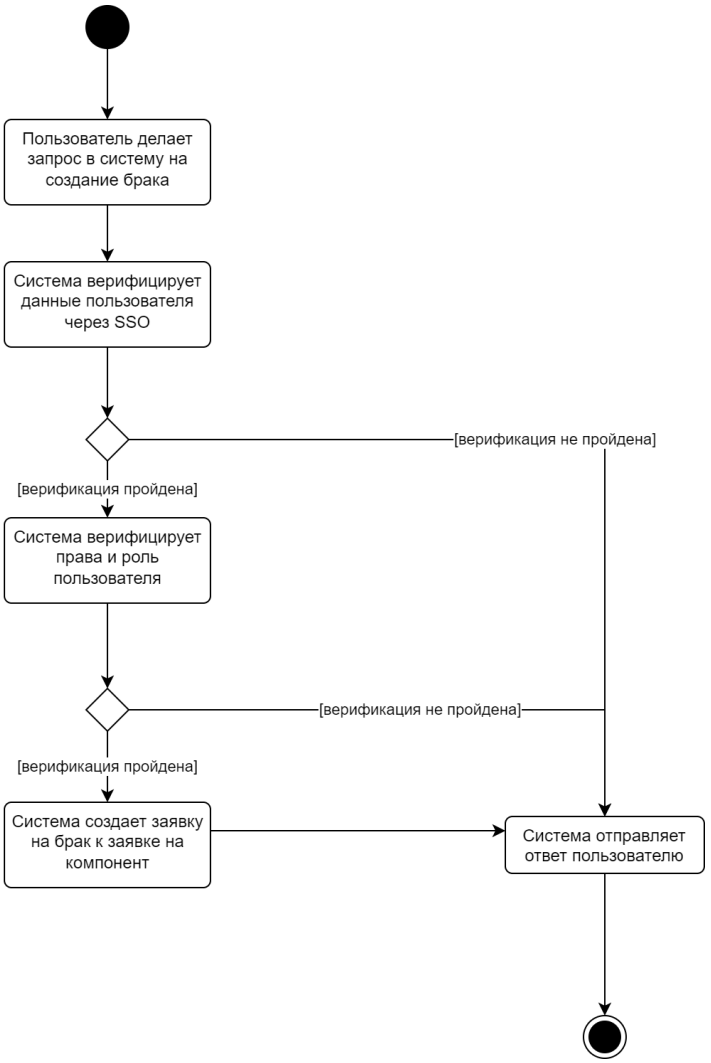
4.3. Class Diagram (Утвердить заявку как готовую для выполнения)



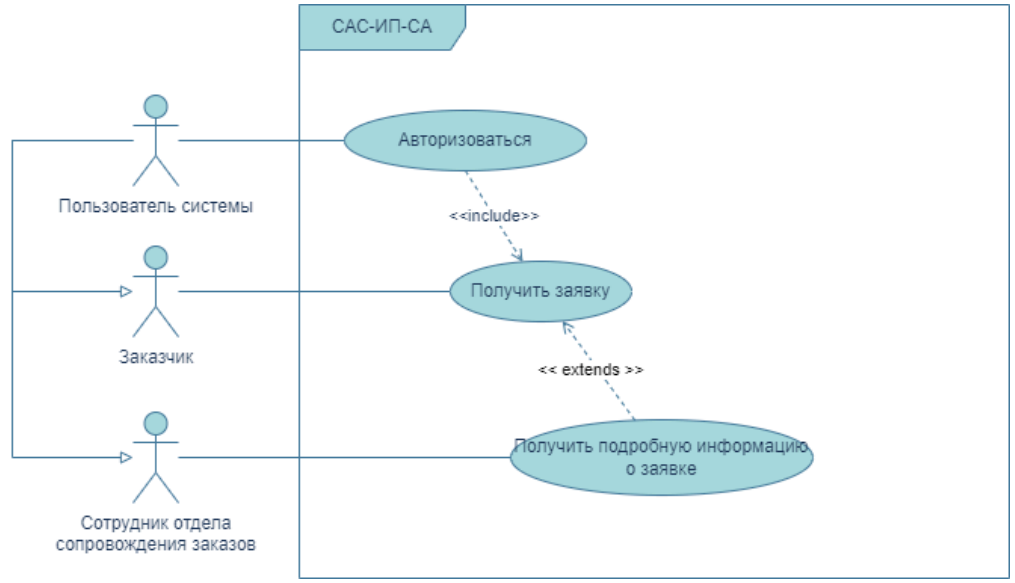


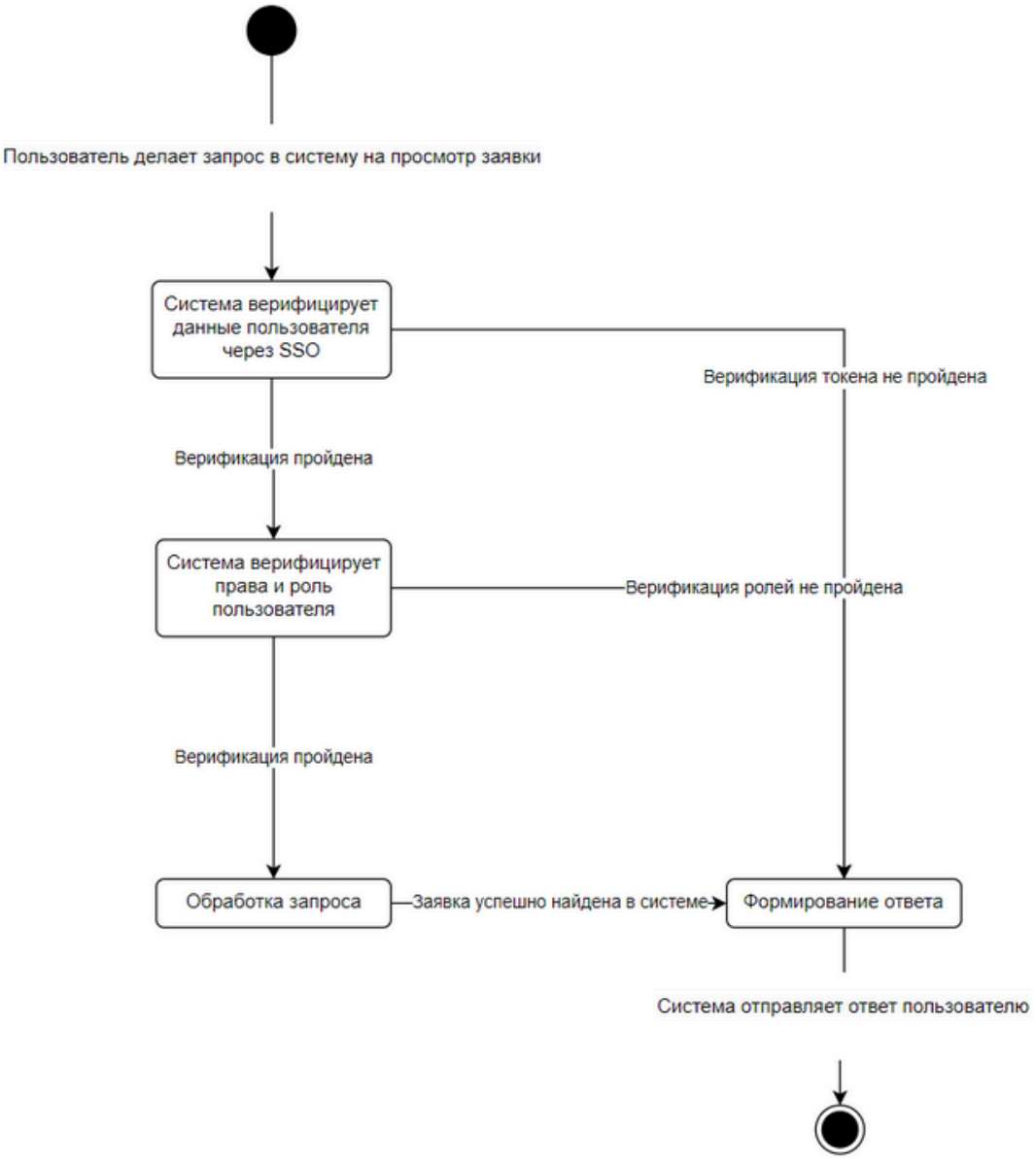
4.4. Activity Diagram (Создание заявки на брак)



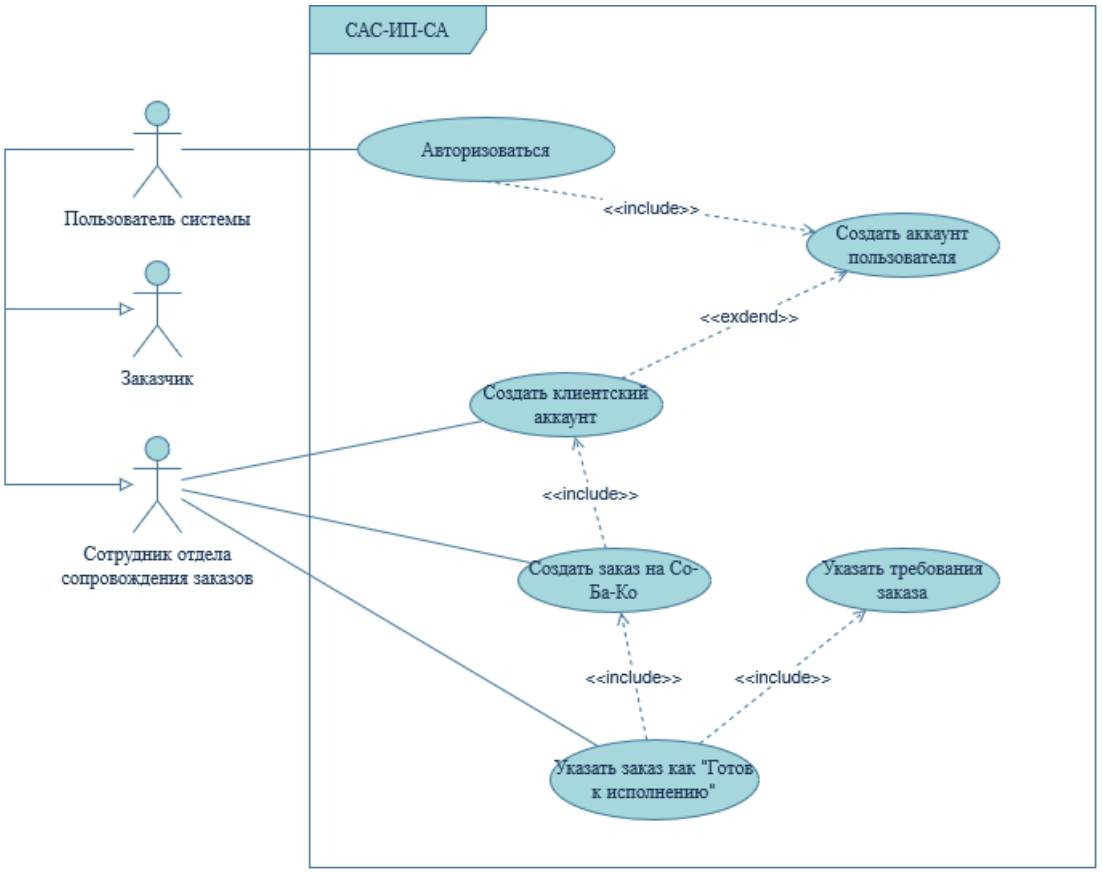


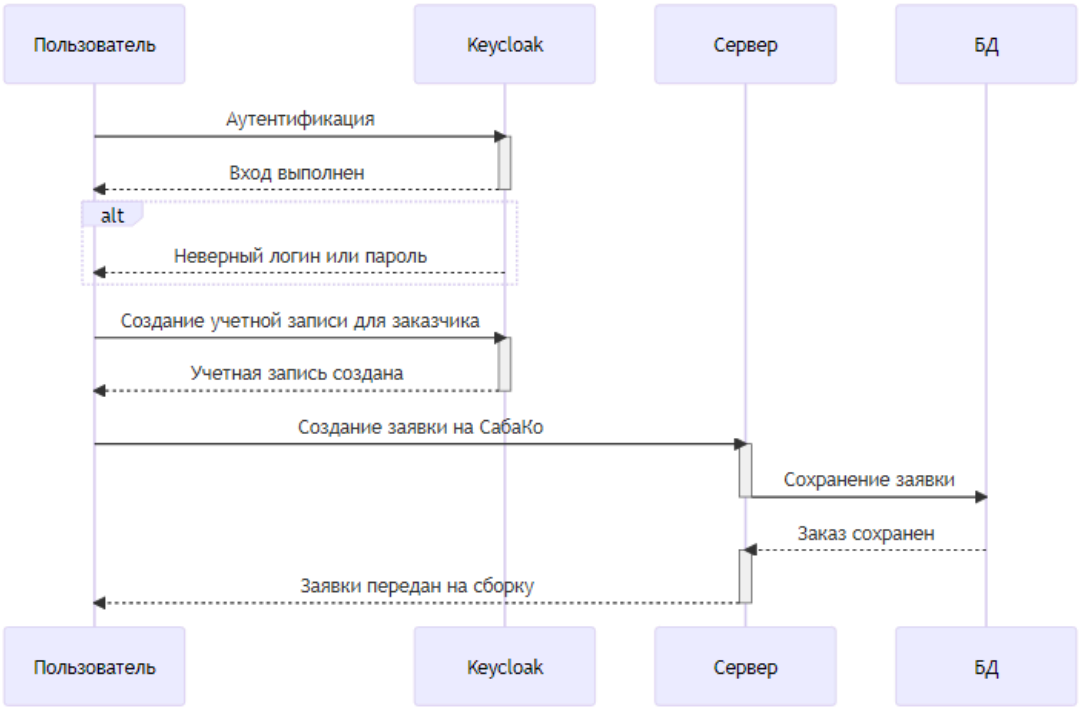
4.4. State Machine Diagram (Получить заявку)



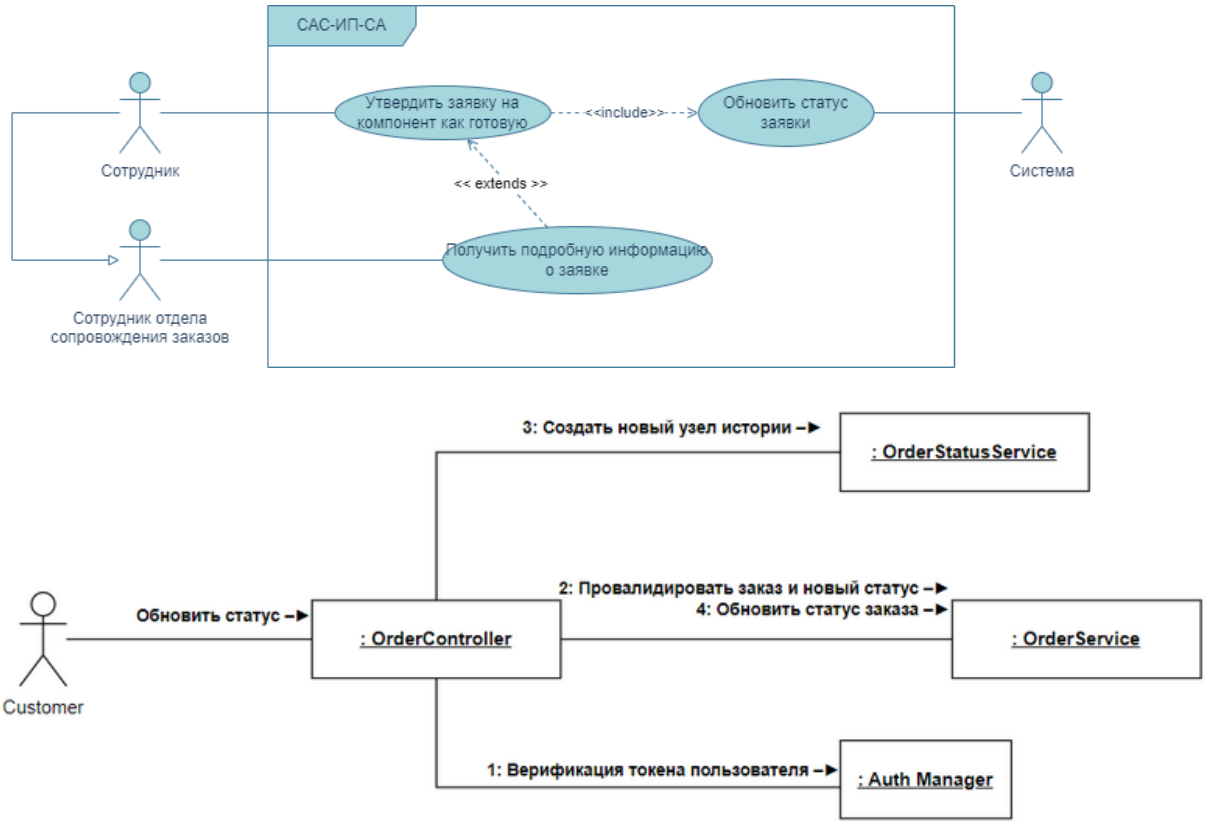


4.5. Sequence Diagram (Создание заявки)





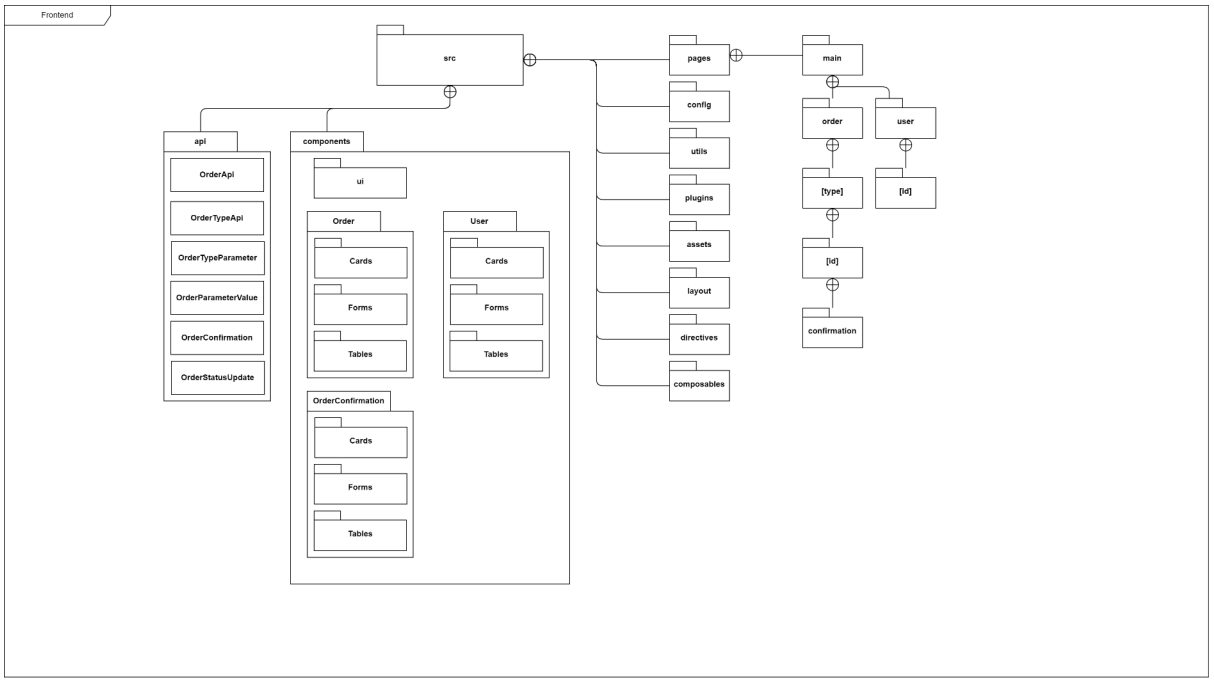
4.6. Cooperative Diagram (Установить статус Готово на заявке)

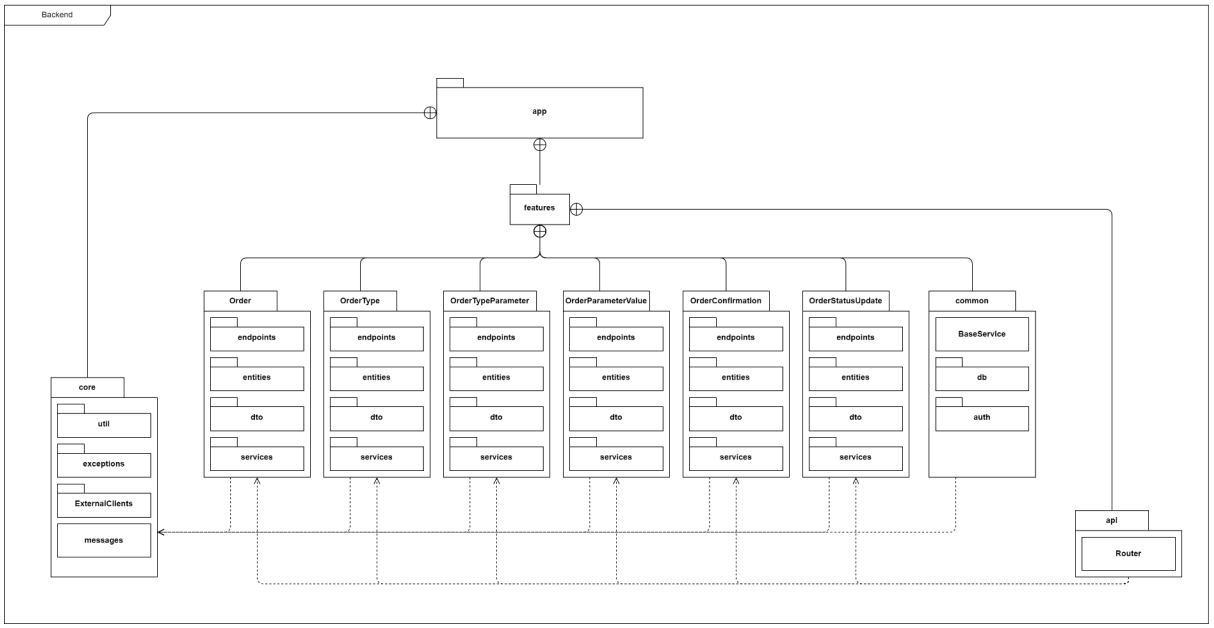


5. Logical View

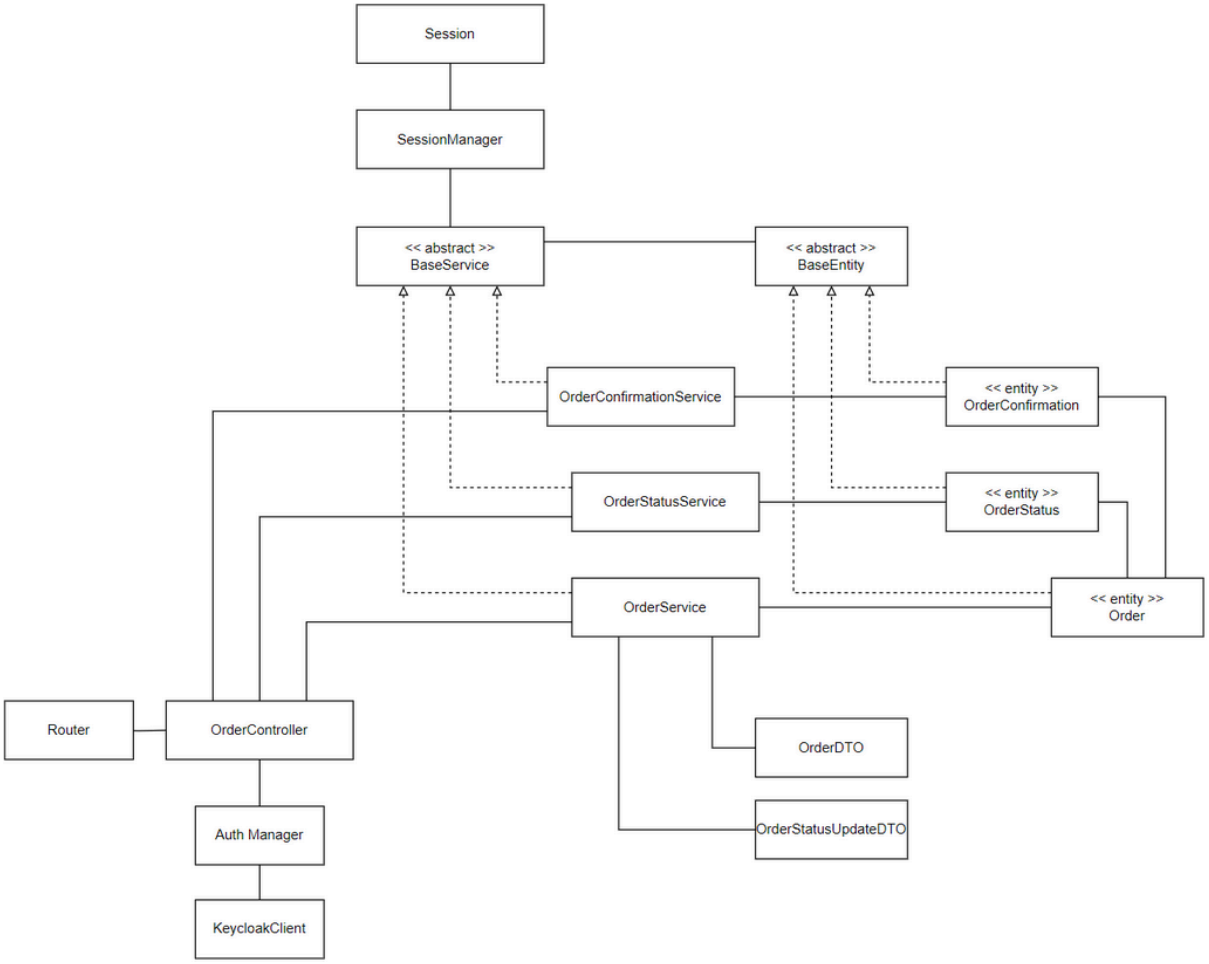
[Данный раздел содержит описание слоев, на которые делится приложение, а также интерфейсов их взаимодействия. Приведите описание каждого из слоев, как они связаны между собой, их назначение. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

5.1. Package Diagram

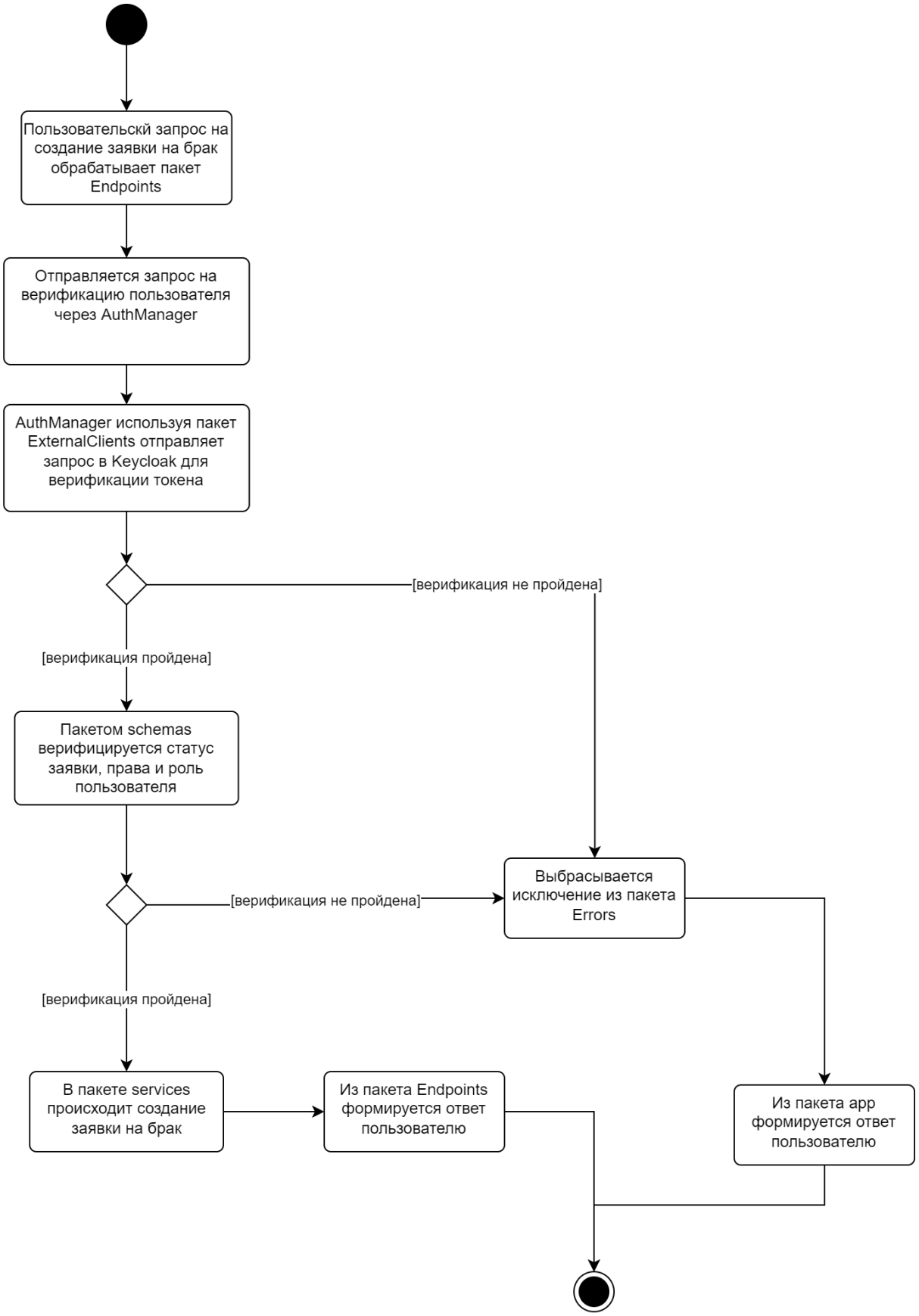




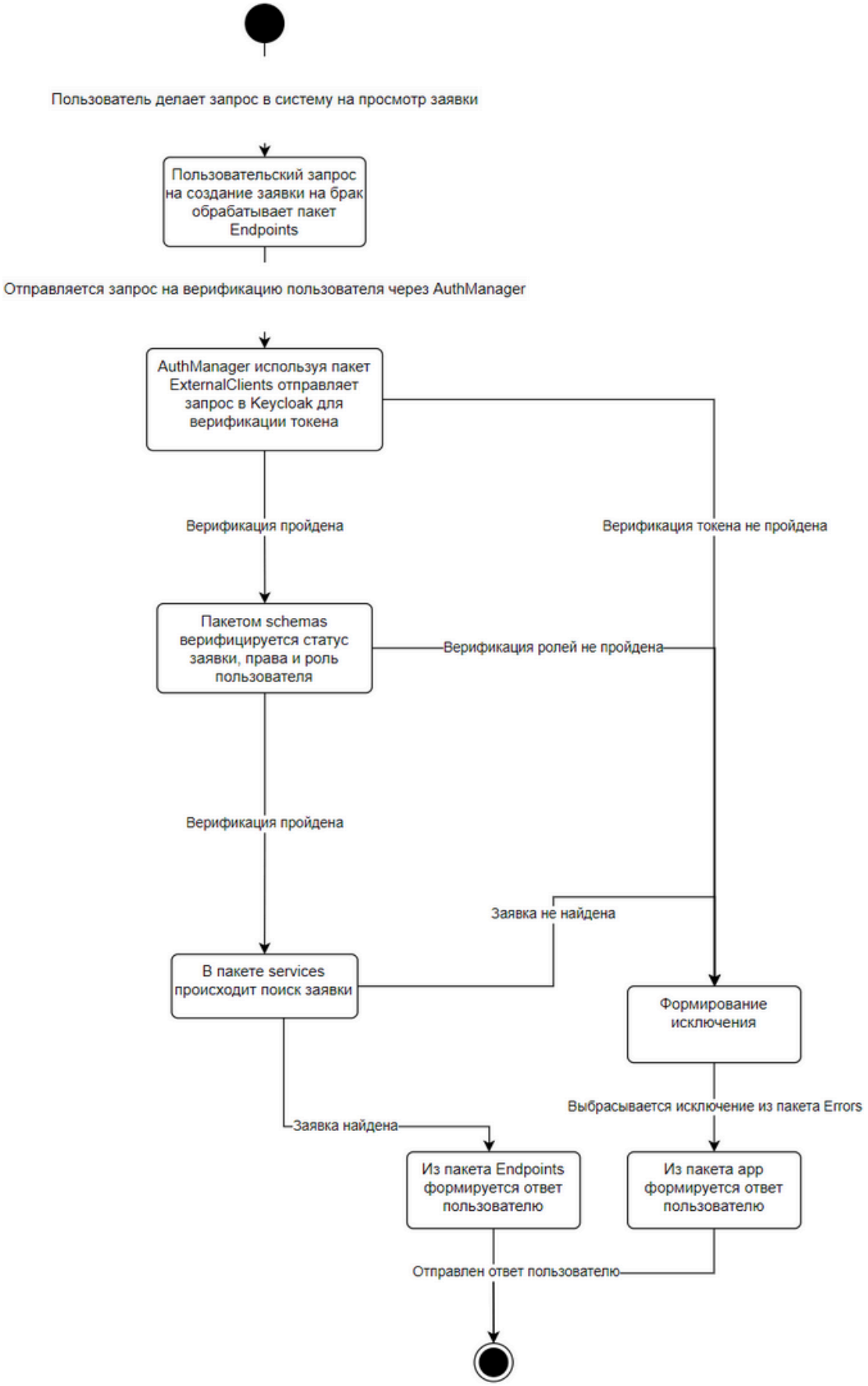
5.2. Class Diagram (Утвердить заявку как готовую к исполнению)



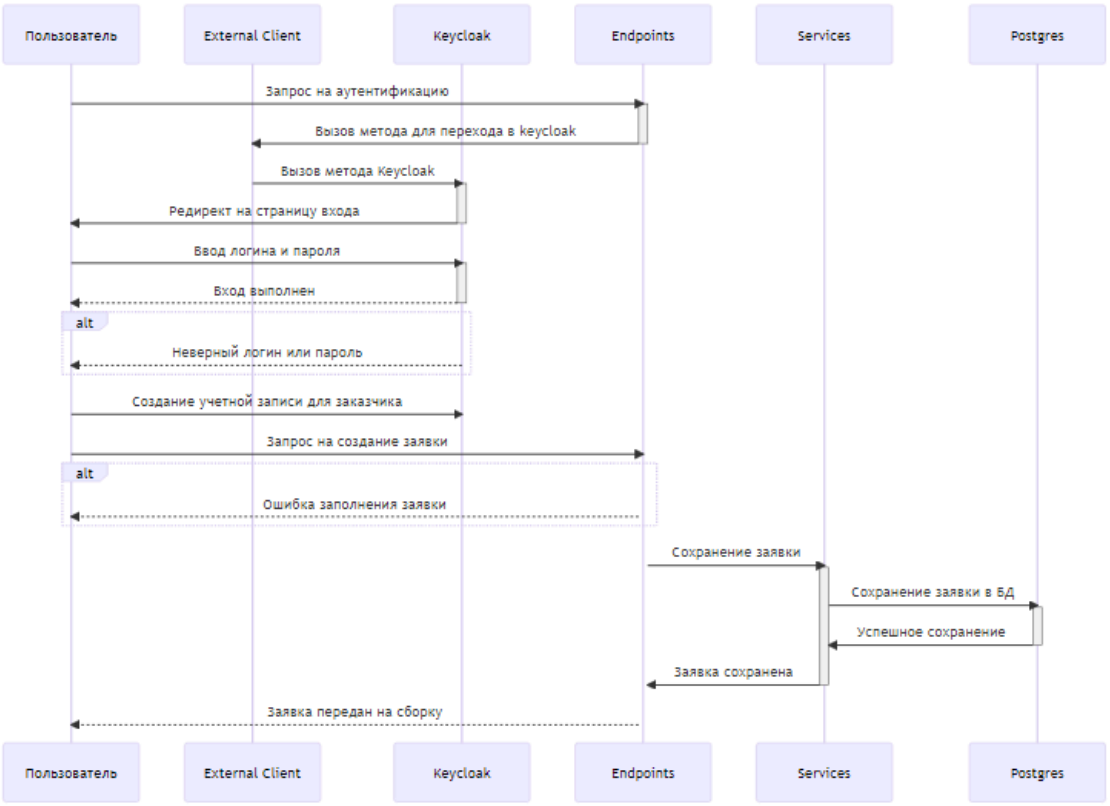
5.3. Activity Diagram (Создание заявки на брак)



5.4. State Machine Diagram (Получить заявку)



5.5. Sequence Diagram (Создание заявки)



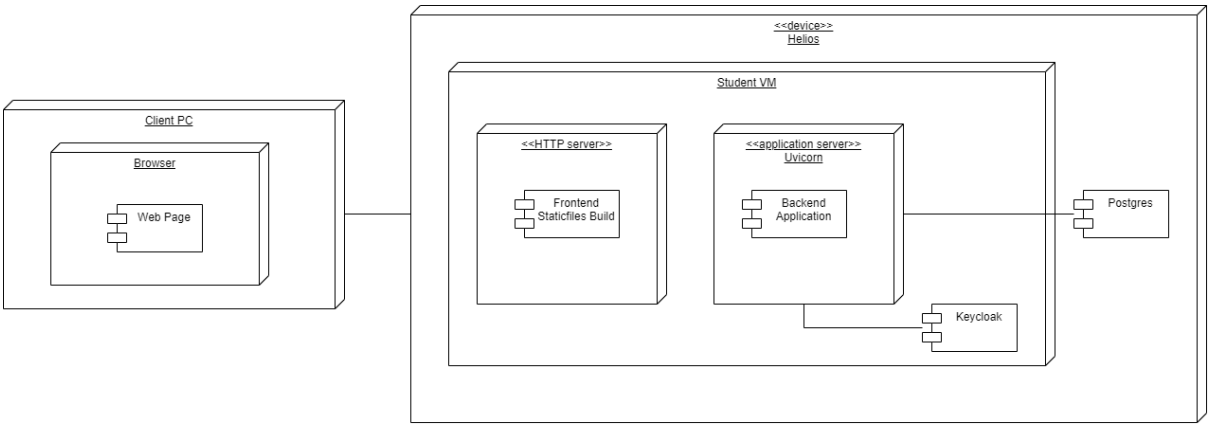
5.6. Cooperative Diagram (Установить статус Готово на заявке)

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.]

**диаграмма размещения по серверам*

7.1. Deployment Diagram

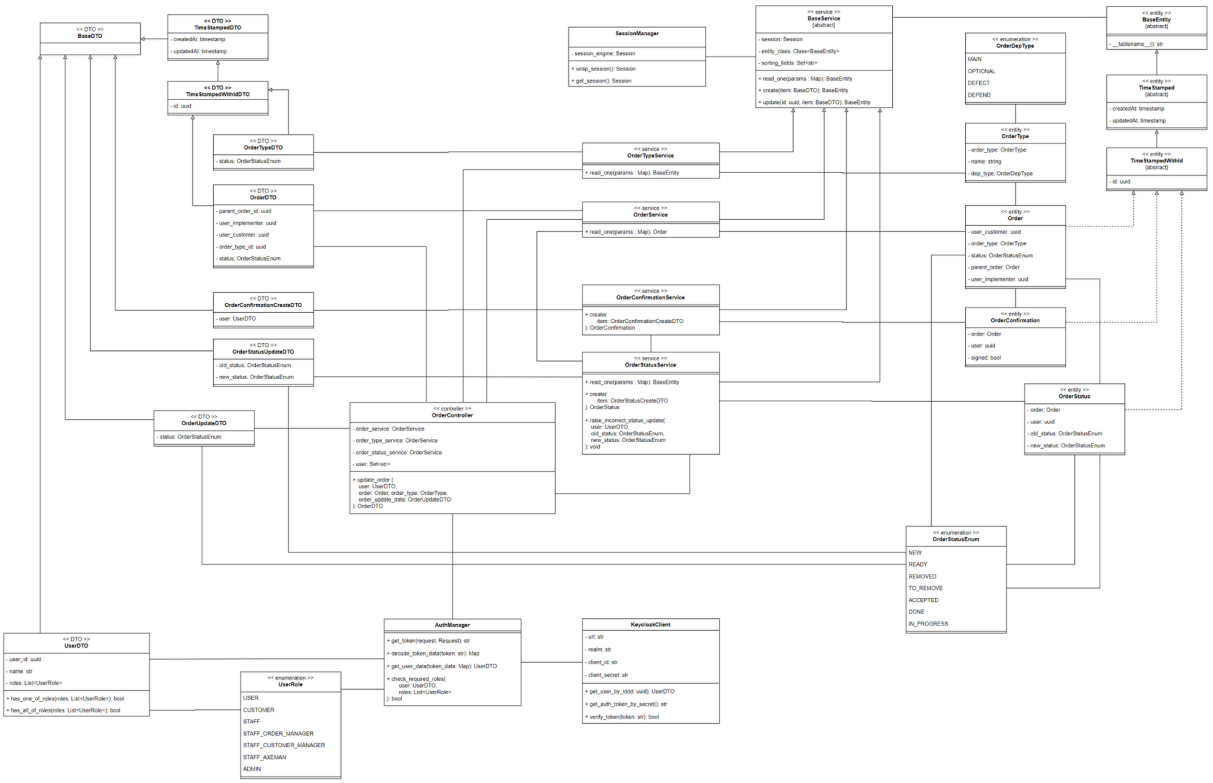


8. Implementation View

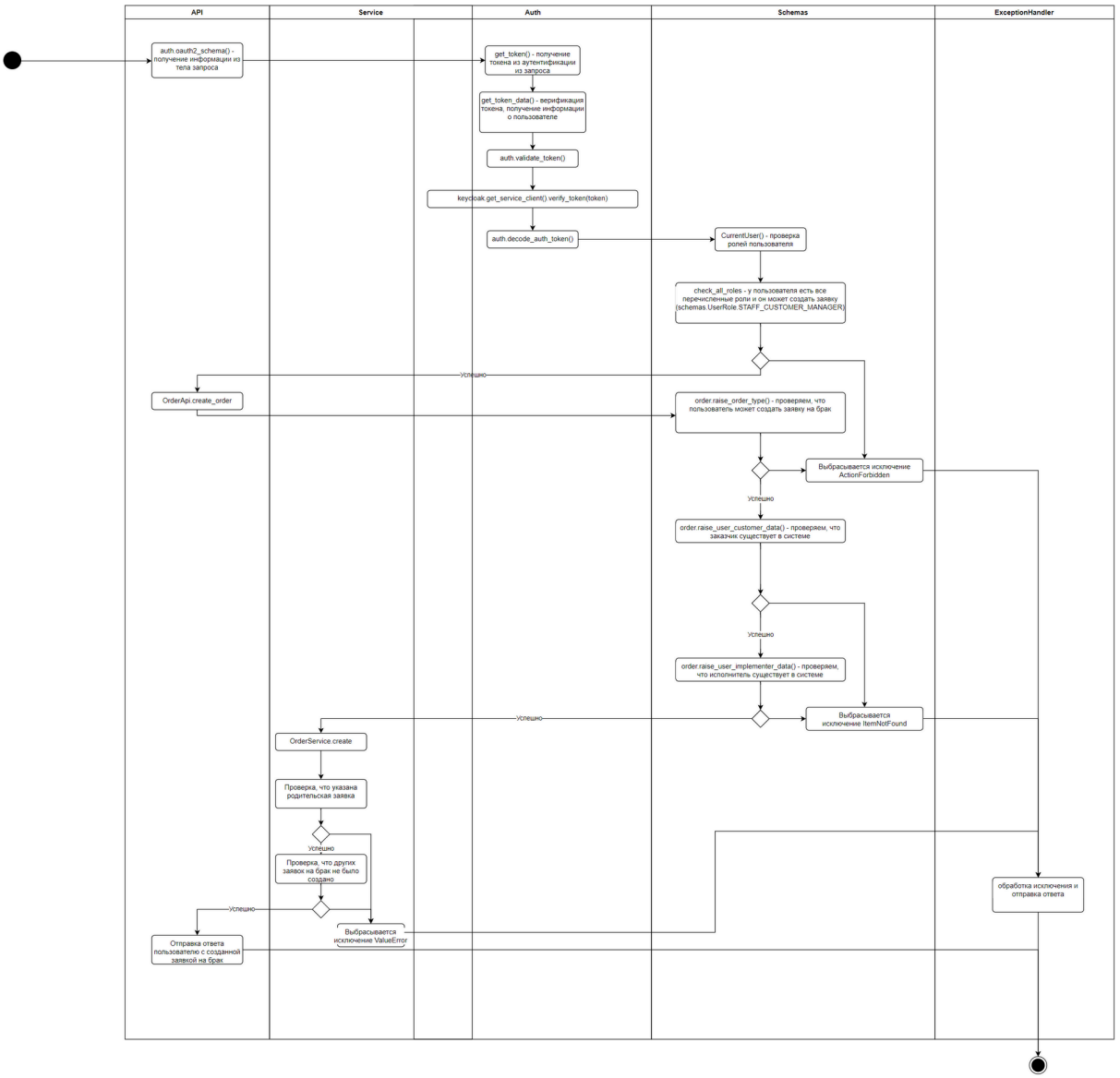
[Данный раздел содержит описание системы в уже реализованном виде. Включите сюда необходимые диаграммы, указанные в п.2, приведите краткое описание каждой диаграммы.]

**детали реализации, классовая диаграмма*

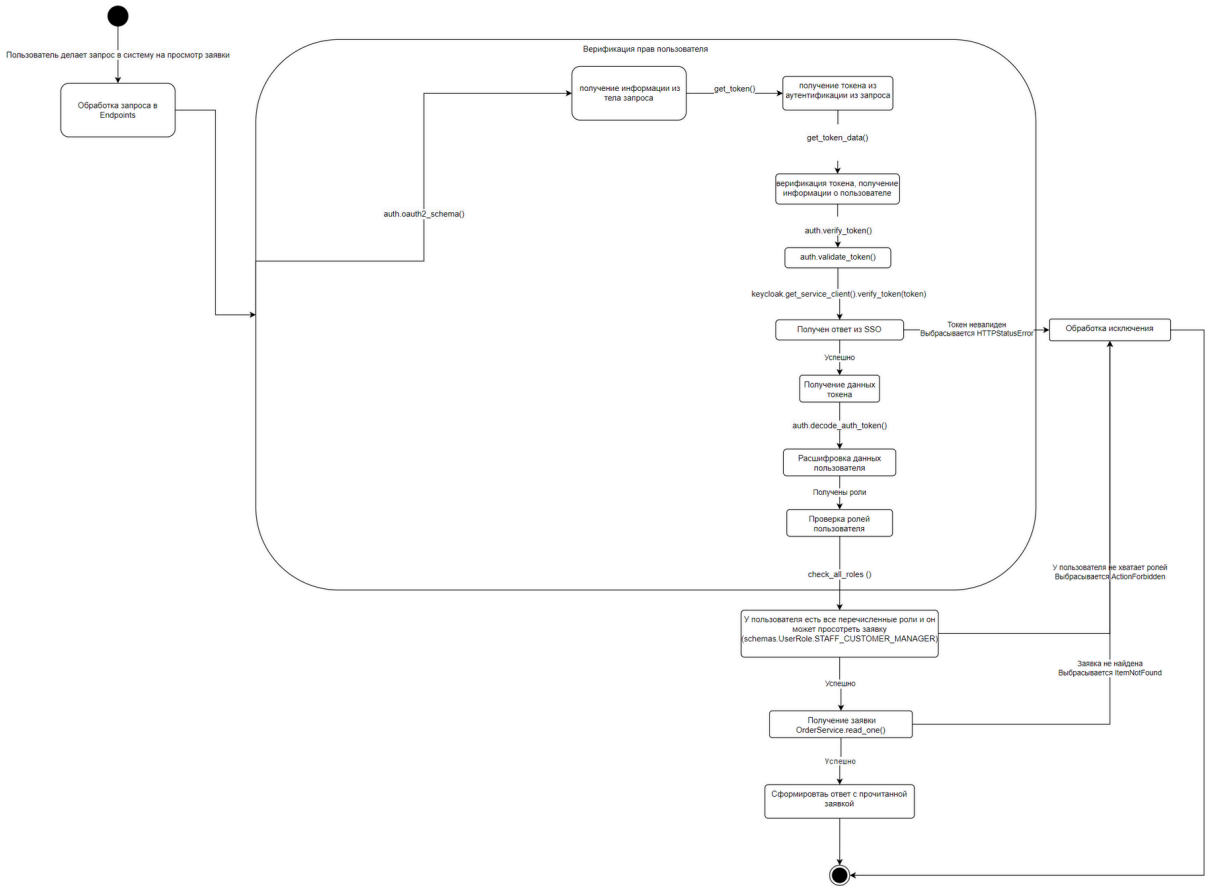
8.1. Class Diagram (Создание заявки на компонент)



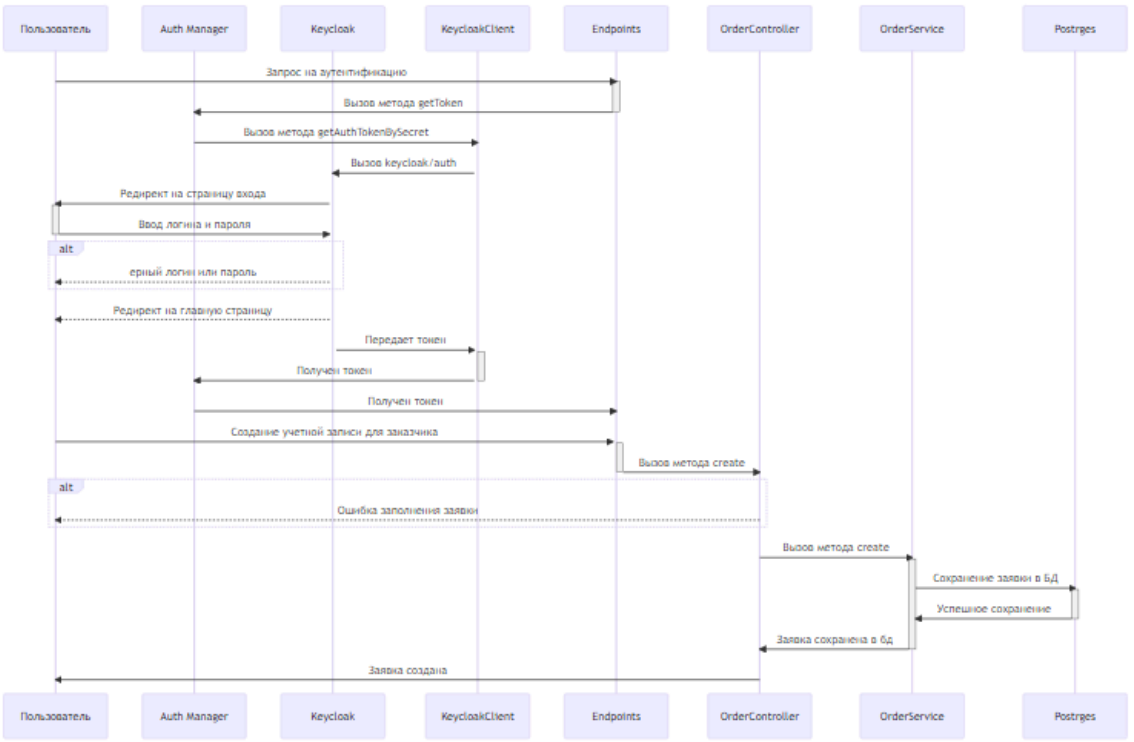
8.2. Activity Diagram (Создание заявки на брак)



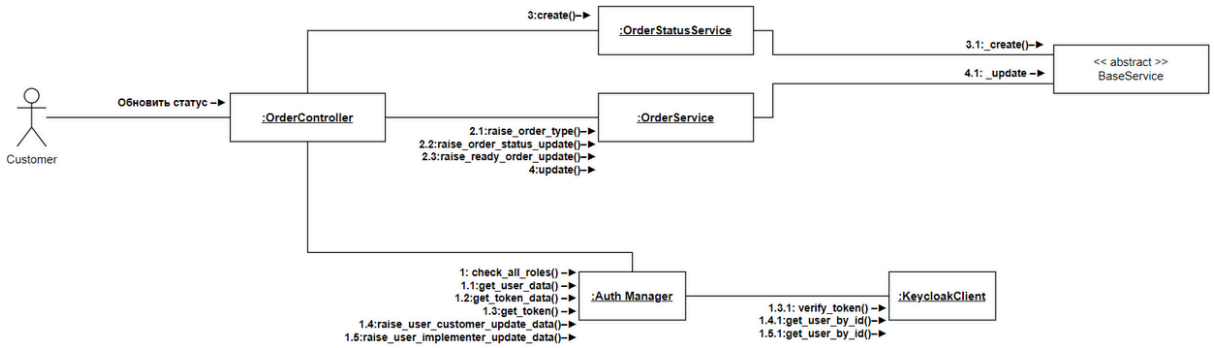
8.3. State Machine Diagram



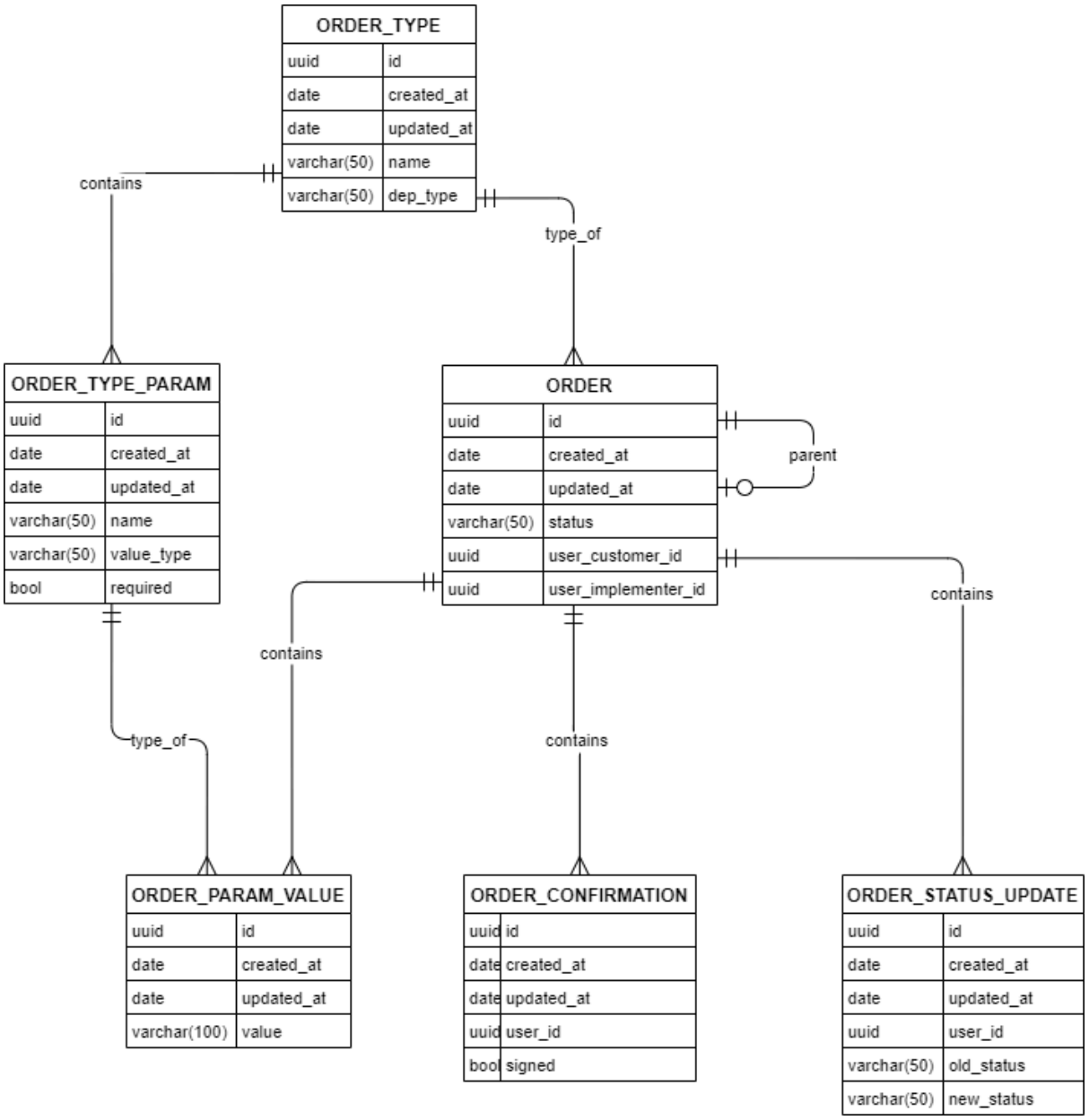
8.4. Sequence Diagram (Создание заявки)



8.5. Cooperative Diagram (Установить статус Готово на заявке)



8.6. Data Base Diagram



9. Size and Performance (Производительность)

- Среднее время ответа сервера - менее 1 минуты
    - система разделена на сервисы и использует асинхронные методы
- Среднее количество одновременно работающих пользователей в день - около 20
    - нет привязки к сессиям
    - независимость состояния приложения и состояния пользователей
    - использование keycloak
- Использование ресурсов сервера ограничивается со стороны Helios, что может приводить к не существенным задержкам в обработке запросов

- Отправка уведомлений пользователям внутри и за границами системы ограничивается 50 сообщениями в сутки
    - в системе не выделяются отдельные модули для высокой производительности
- Система не должна использовать более 8 ГБ оперативной памяти
    - не читаем файлы в память целиком
    - динамически не выделяем память вручную
    - придерживаемся итеративных парадигм
- Система не должна использовать более 100 ГБ постоянной памяти, без учета размера БД
    - не создаём дополнительных файлов

10. Quality (Качество)

Система позволяет обрабатывать не менее 1000 активных процессов в сутки

- система использует асинхронный фреймворк и разделена на модули для отдельных сущностей

Сервисный интервал системы не более 1.5 часов

- система разделена на независимые логические модули и пакеты, а также миграции, которые позволяют делать быстрое обновление и не требуют долгой ручной работы
- скрипты запуска автоматизируют процесс старта приложения

Пользователю с разными ролями системы доступны различные действия в интерфейсе

- для выполнения данного требования используется сервис авторизации и аутентификации Keycloak, позволяющий разграничивать роли пользователей
- модули ExternalClients, AuthManager и KeycloakClient реализуют взаимодействие с keycloak