Оглавление

1. Общее описание

1.1 Цель работы

Изучить теоретические принципы унифицированного процесса разработки СОИУ и составляющих его этапов. Получить практические навыки применения шаблонов при проектировании и разработке СОИУ. Освоить применение CASE средств для разработки СОИУ.

1.2 Описание задания

Выполнить проектирование СОИУ в соответствии с описанием ее функциональности (определяется вариантом). Для проектирования использовать этапы и модели унифицированного процесса. По результатам проектирования получить работающую программу с паттернами (по варианту). Для построения диаграмм использовать среду StarUML

1.3 Задание варианта АСУ/ИС:

Тема: АИС IT фирмы
паттерн бизнес-логики: service layer
паттерн работы с БД: active record
паттерн GOF: наблюдатель
Система предназначена для автоматизации управления задачами разработчиков в IT фирме

2. Этап анализа и планирования требований

2.1 Спецификация основных проектных требований

2.1.1 Функциональные требования

Система должна:

1. Обеспечить возможность идентификации пользователя
2. Обеспечить возможность управления задачами пользователей
3. Обеспечить возможность ввести учет задач пользователей
4. Обеспечить возможность распределения задач между пользователями

2.1.2 Нефункциональные требования

Система должна:

1. Иметь удобный интерфейс клиентской части
2. Предоставлять клиентскую часть через десктопное приложение
3. Обеспечивать постоянное и стабильное соединение с базой данных
4. Обеспечивать одновременную работу с 100 пользователями
5. Масштабироваться и обеспечивать возможность добавления новых типов задач и статусов
6. Обеспечивать ответ в течении 500мс при стабильном соединении
7. Потреблять до 2Gb ОП

2.2 Модель предметной области

Диаграмма классов предметной области приведена на рисунке 2.1.

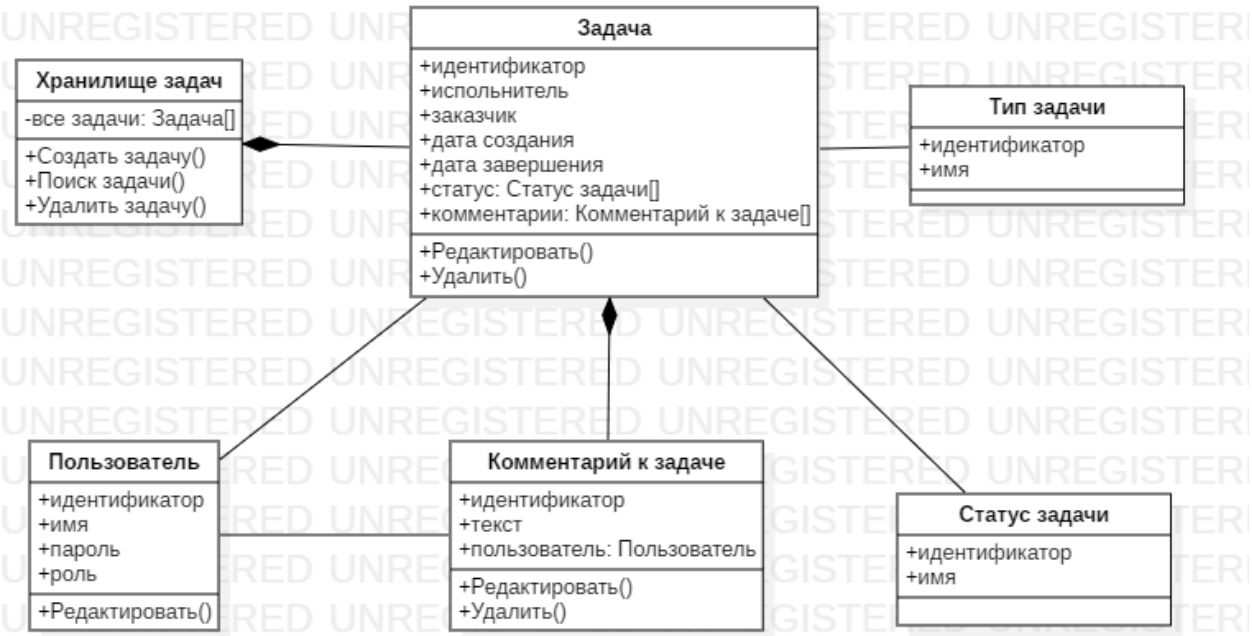


Рис 2.1 – Диаграмма классов предметной области

2.3 Выявленные актеры

В рамках проекты выявлено 2 актера:

1. Пользователь, занимается созданием задач и редактированием задач, к которым он имеет доступ
2. Администратор, может делать все, что и пользователь, только имеет доступ ко всем задачам и имеет право на создание новых типов задач и статусов задач

2.4. Прецеденты

2.5.1 Выявленные прецеденты

Диаграмма прецедентов приведена на рисунке 2.2

По результатам проведенного анализа выявлены следующие основные прецеденты:

1. Создание пользователя
2. Проверка прав доступа пользователя
3. Добавление задачи
4. Редактирование задачи
5. Добавление комментария к задаче
6. Редактирование комментария
7. Добавление новых типов задач
8. Добавление новых статусов задач
9. Построение календаря задач
10. Поиск задач

Прецедент **Создание пользователя** заключается в наполнении БД информацией о новом пользователе (регистрация пользователя)

Прецедент **Проверка прав доступа пользователя** заключается в проверке того имеет ли пользователь право на редактирование данных, которые он пытается изменить

Прецедент **Добавление задачи** заключается в наполнении БД информацией о новой задаче (установка сроков задачи, выбор исполнителя, добавление описания)

Прецедент **Редактирование задачи** заключается в изменении записи задачи в БД (подразумевается, как изменение отдельных полей, так и удаление задачи целиком)

Прецедент **Добавление комментария к задаче** заключается в наполнении БД данными о комментарии, который будет отображаться вместе с задачей

Прецедент **Редактирование комментария** заключается в редактировании записи комментария в БД (подразумевается, как изменение отдельных полей, так и удаление целиком)

Прецедент **Добавление новых типов задач** заключается наполнение БД информацией о новом типе задач, который будет отображаться при редактировании задачи

Прецедент **Добавление новых статусов задач** заключается наполнение БД информацией о новом статусе задач, который будет отображаться при редактировании задачи

Прецедент **Построение календаря задач** заключается в выборке задач по параметрам и представления их в виде календаря задач, на котором отображается краткая информация о задаче

Прецедент **Поиск задач** заключается в поиске задач по параметрам и представлении их в виде списка

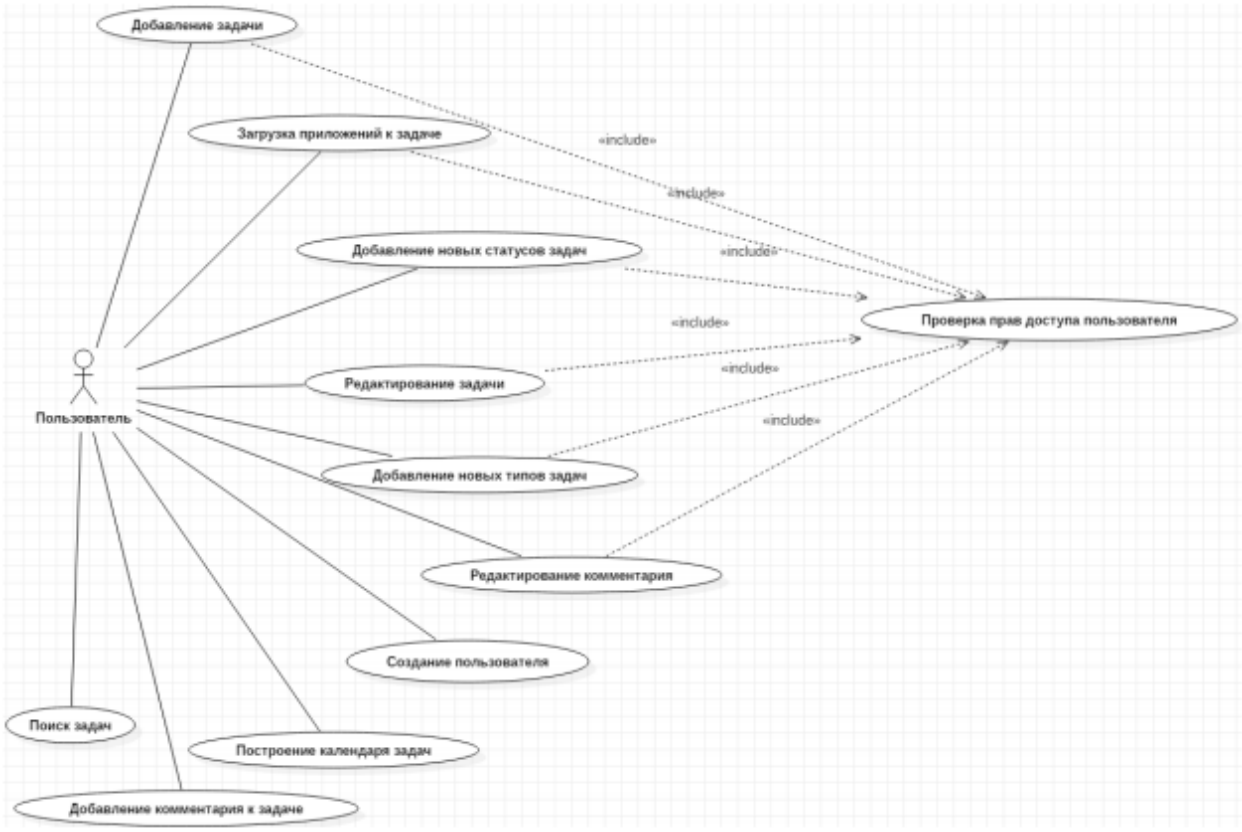


Рис 2.2 – Диаграмма прецедентов

2.5.2 Список приоритетов прецедентов

Приоритеты прецедентов приведены в таблице 2.1

Таблица 2.1 – Приоритеты прецедентов

Прецедент	**Приоритет**
Создание пользователя	Средний
Проверка прав доступа пользователя	Высокий
Добавление задачи	Высокий
Редактирование задачи	Средний
Добавление комментария к задаче	Средний
Редактирование комментария	Средний
Добавление новых типов задач	Низкий
Добавление новых статусов задач	Низкий
Поиск задач	Высокий

2.6 Перечень рисков

На этапе Начало была проведена инициация рисков с разделением на проектные, технические и коммерческие риски

2.6.1 Проектные риски

1. **Риски управления требованиями.** На этапах Конструирование и Переход проектная команда может понять, что изначальные требования не соответствовали реальности. Для минимизации риска следует детально провести интервьюирование заказчика, наиболее полно составить ТЗ и согласовать его с заказчиком.
2. **Риски, связанные с персоналом**. Участники проектной команды могут уйти на больничный, быть уволены или показать низкую производительность, что увеличит сроки разработки проекта. Для минимизации риска следует предусмотреть возможность растягивания графика разработки.

2.6.2 Технические риски

1. **Риски недостаточной оценки сложности**. На этапах Начало и Конструирование могут возникнуть проблемы проектирования и реализации каких-либо модулей или компонентов, из-за чего следует заложить в бюджет проекта средства на найм дополнительного персонала или экспертов–консультантов.
2. **Риски неверной реализации.** На этапах Конструирование и Переход может выясниться, что какие-либо части системы реализованы неправильно, с ошибками в логике работы или с полной неработоспособностью. Для минимизации рисков следует проводить постоянное тестирование на соответствие требованиям, интеграционное тестирование и нагрузочное тестирование системы
3. **Риски несовместимости технологий.** На этапах Конструирование и Переход может выясниться, что какие-либо технологии, используемые системой могут оказаться несовместимы, как друг с другом, так и с используемой платформой. Для минимизации рисков следует провести анализ технологий, которые будут использоваться в проекте

2.6.3 Коммерческие риски

1. **Риски потери финансирования**. В течение разработки проекта может возникнуть недостаток денежных средства для продолжения работ. Для минимизации риска следует выделить отдельную статью расходов в бюджете одного или нескольких ключевых партнёров на срок реализации проекта, которая позволит покрыть превышение стоимости разработки в случае его возникновения

2.7 Описание возможной архитектуры

При составлении архитектуры системы был проведён анализ основных проектных требований, модели предметной области и выявленных прецедентов. Результат приведён на рисунке 2.3.

Принято решение использовать 3х-уровневую архитектуру состоящую из БД, сервера приложения и клиентского приложения, поскольку такой подход позволит разделить работу над клиентской и серверной частью, что приведет к снижению влиянию одного на другое.

Для разработки серверной части был выбран язык программирования Golang по той причине, что он является высокоуровневым ЯП, ориентирован на разработку серверных приложений.

Принято решение использовать при разработке клиентской части язык C# в связке с кроссплатформенным фреймворком Avalonia UI и паттерном разработки MVVM, т.к использование данных технологий позволит ускорить разработку десктопного приложения для всех актуальных платформ

Серверная ОС может быть принадлежать как семействам Windows ОС, так и ОС основанных на ядре Linux.

Клиентская часть является кроссплатформенной, таким образом осуществляется поддержка актуальных на данный момент пользовательских ОС (Linux, Windows, OS X)

Принято решение использовать ОС Debian 9.1 в качестве серверной ОС, поскольку она является свободно распространяемой и довольно проста в первоначальной настройке и администрировании.

Для развертывания сервера приложения будет использоваться Docker для минимизации риска несовместимости технологий и ОС сервера, а также позволит упростить повторную развертку на другом сервере.

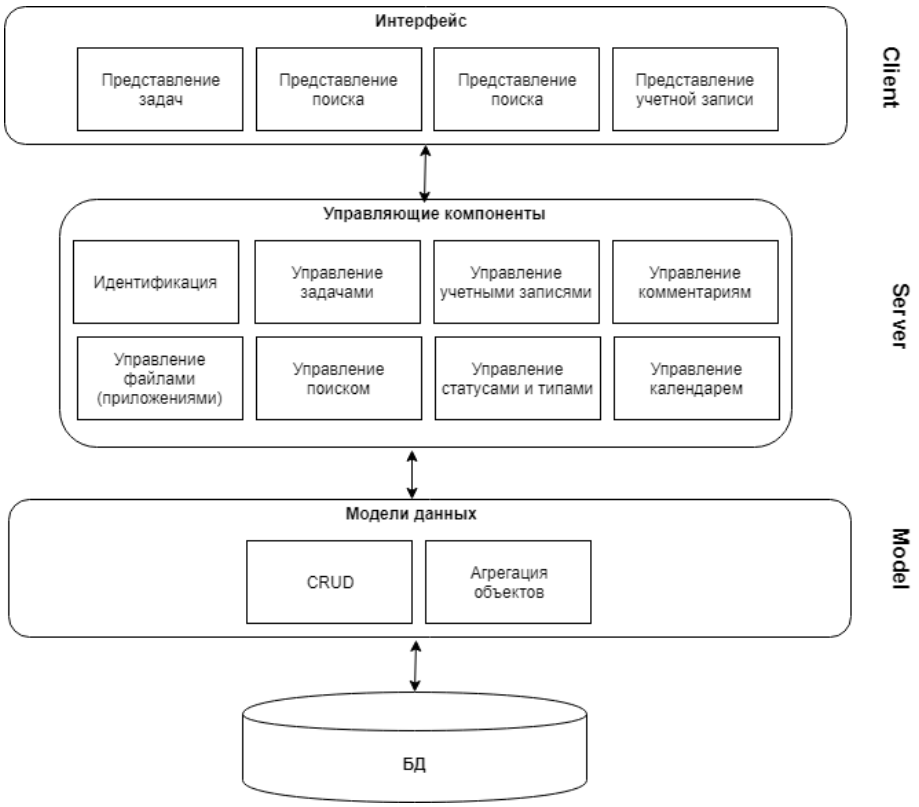


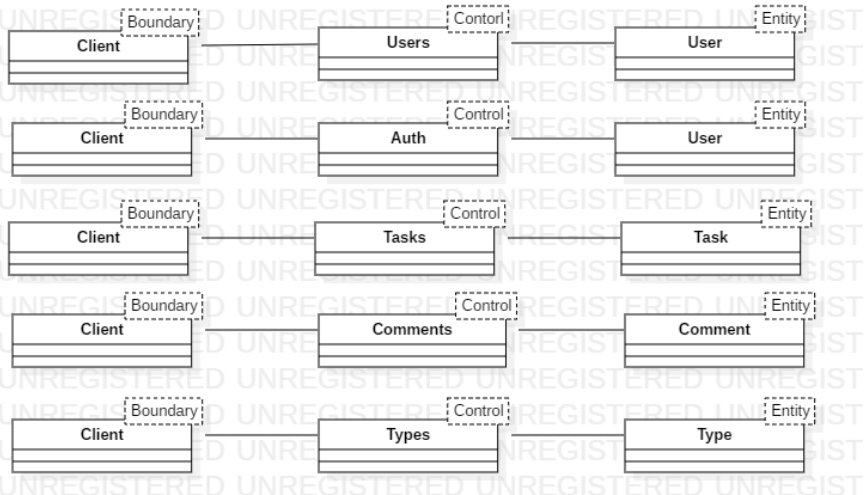
Рис 2.3 – Базовая архитектура системы

Для обеспечения работы сервера с HTTP– и HTTPS–запросами используются веб-серверы Nginx и Apache.

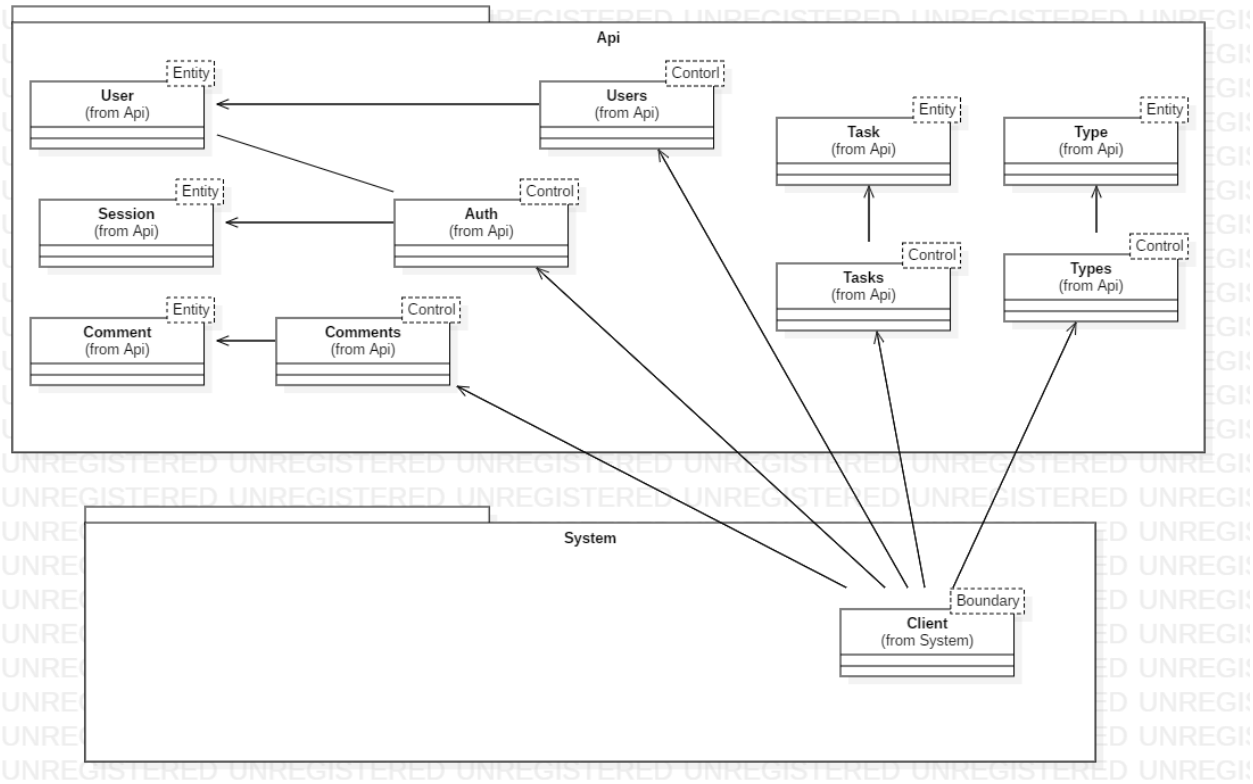
Благодаря использованию ORM выбранный язык разработки нечувствителен к используемой СУБД. Выбрана СУБД PostgreSQL, поскольку её основными особенностями являются поддержка БД практически неограниченного размера и лёгкая расширяемость.

2.8. диаграммы классов анализа

2.8.1. Граничные классы



2.8.2. Пакеты анализа



2.9. Оценка проекта по COCOMO II

Для оценка стоимости, затрат и длительности проекта используется конструктивная модель стоимости этапа композиции приложения на основе объектных указателей. Оценка сложности экранов приведена в таблице 2.2. Оценка сложности отчётов приведена в таблице 2.3.

Таблица 2.2 – Оценка сложности экранов

Экран	**Количество представлений**	**Сложность**
Экран входа	1	Простая
Главный экран	2	Средняя
Экран задачи	4	Средняя
Экран поиска	2	Простая
Экран статусов	1	Простая
Экран типов	1	Простая
Экран профиля	2	Простая

Таблица 2.3 – Оценка сложности отчетов

Экран	**Количество представлений**	**Сложность**
Отчет об изменении задачи	3	Средняя

Язык C# является языком программирования четвёртого поколения, поэтому в расчёте количества объектных указателей появится объект 4GL. Но будет использоваться вес 10, такой же, как и для 3GL компонентов. Оценка количества объектных указателей приведена в таблице 2.4.

Таблица 2.4 – Оценка количества объектных указателей

Тип объекта	**Количество**	**Вес**	**Итого**
Экран простой	5	1	5
Экран средний	2	2	4
Отчет средний	1	3	3
4GL компонент	1	10	10
Итого			22

Для проектной команды разработка подобного рода систем является абсолютно новой задачей, поэтому процент повторного использования кода $\%REUSE = 0$.
Новые объектные указатели рассчитаны в формуле 2.1.

$$NOP = OP \times \frac{100-\%REUSE}{100} = 22 \times \frac{100-0}{100} = 22 \quad (2.1)$$

Примем, что опытность разработчиков и зрелость среды разработки номинальные, тогда $PROD = 13$. Затраты рассчитаны в формуле 2.2.

$$\text{ЗАТРАТЫ} = \frac{NOP}{PROD} = \frac{22}{13} = 1.69 \text{ чел.} - \text{мес.} \quad (2.2)$$

Примем, что среднее значение рабочего коэффициента является номинальным и равно 15000$. Стоимость рассчитана в формуле 2.3

$$\text{СТОИМОСТЬ} = \text{ЗАТРАТЫ} \times \text{РАБ}_{\text{КОЭФ}} = 90\,000\text{Р} \times 1{,}69 = 152\,100\text{Р} \quad (2.3)$$

Параметры вычисления длительности разработки приведены в таблице 2.5.

Таблица 2.5 – Параметры вычисления длительности разработки

Масштабный фактор	**Значение**	**Описание**
Предсказуемость PREC	3	Отчасти непредсказуемый
Гибкость разработки FLEX	3	Редкое расслабление в работе
Разрешение архитектуры- /риска RESL	4	Разрешение риска 40%
Связность группы TEAM	1	Высокая кооперативность
Зрелость процесса PMAT	3	Номинальная зрелость

Показатель степени B рассчитан в формуле 2.4

$$B = 1.01 + 0.01 \times \sum_{i=1}^{5} W_i = 1.01 + 0.14 = 1.15 \quad (2.4)$$

Процент увеличения номинального графика примем за 50
Тогда $SCEDPercentage$ = 150.
Длительность разработки оценена в формуле 2.5.

$$TDEV = \left[3 \times \text{ЗАТРАТЫ}^{(0{,}33+0{,}2 \times [B-1.01])}\right] \times \frac{\text{SCEDPercentage}}{100} = 6.138 \text{ мес} \qquad (2.5)$$

3.Этап проектирования

3.1.Уточненное описание контекста системы

3.1.1. Глоссарий понятий

- Пользователь: человек, имеющий аккаунт в системе
- Задача: Текстовое описания работы, которую пользователю предстоит выполнить или проследить ее выполнение. Содержит в себе сроки выполнения работы и текстовые комментарии пользователей
- Статус задачи: Описание текущего состояния задачи

3.2. Применяемые паттерны

3.2.1. Service Layer

Определяет границу между приложением и слоем сервисов, который образует набор доступных операций и управляет ответом приложения в каждой операции.

Паттерн Service Layer определяет для приложения границу и набор допустимых операций с точки зрения взаимодействующих с ним клиентских. Он инкапсулирует бизнес-логику приложения, управляя транзакциями и управляя ответами в реализации этих операций.

3.2.2. Active Record

Один объект управляет и данными, и поведением. Большинство этих данных постоянны и их надо хранить в БД. Этот паттерн использует наиболее очевидный подход - хранение логики доступа к данным в объекте сущности.

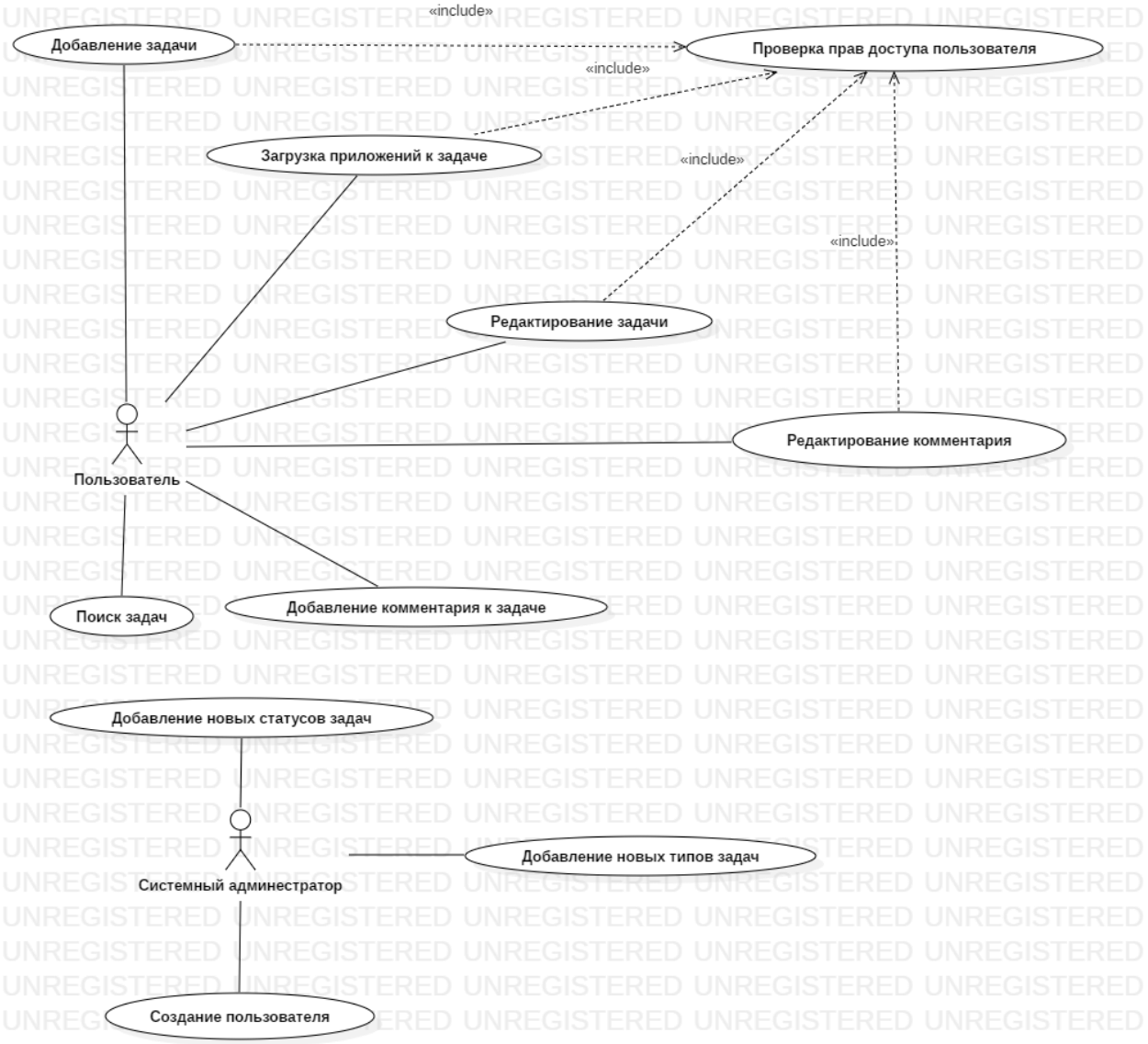
Объект является "обёрткой" одной строки из БД или представления, включает в себя доступ к БД и логику обращения с данными.

3.2.3. Observer

Паттерн "Наблюдатель" (Observer) представляет поведенческий шаблон проектирования, который использует отношение "один ко многим". В этом отношении есть один наблюдаемый объект и множество наблюдателей. И при изменении наблюдаемого объекта автоматически происходит оповещение всех наблюдателей.

Данный паттерн еще называют Publisher-Subscriber (издатель-подписчик), поскольку отношения издателя и подписчиков характеризуют действие данного паттерна: подписчики подписываются email-рассылку определенного сайта. Сайт-издатель с помощью email-рассылки уведомляет всех подписчиков о изменениях. А подписчики получают изменения и производят определенные действия: могут зайти на сайт, могут проигнорировать уведомления и т.д.

3.3. Диаграмма прецедентов

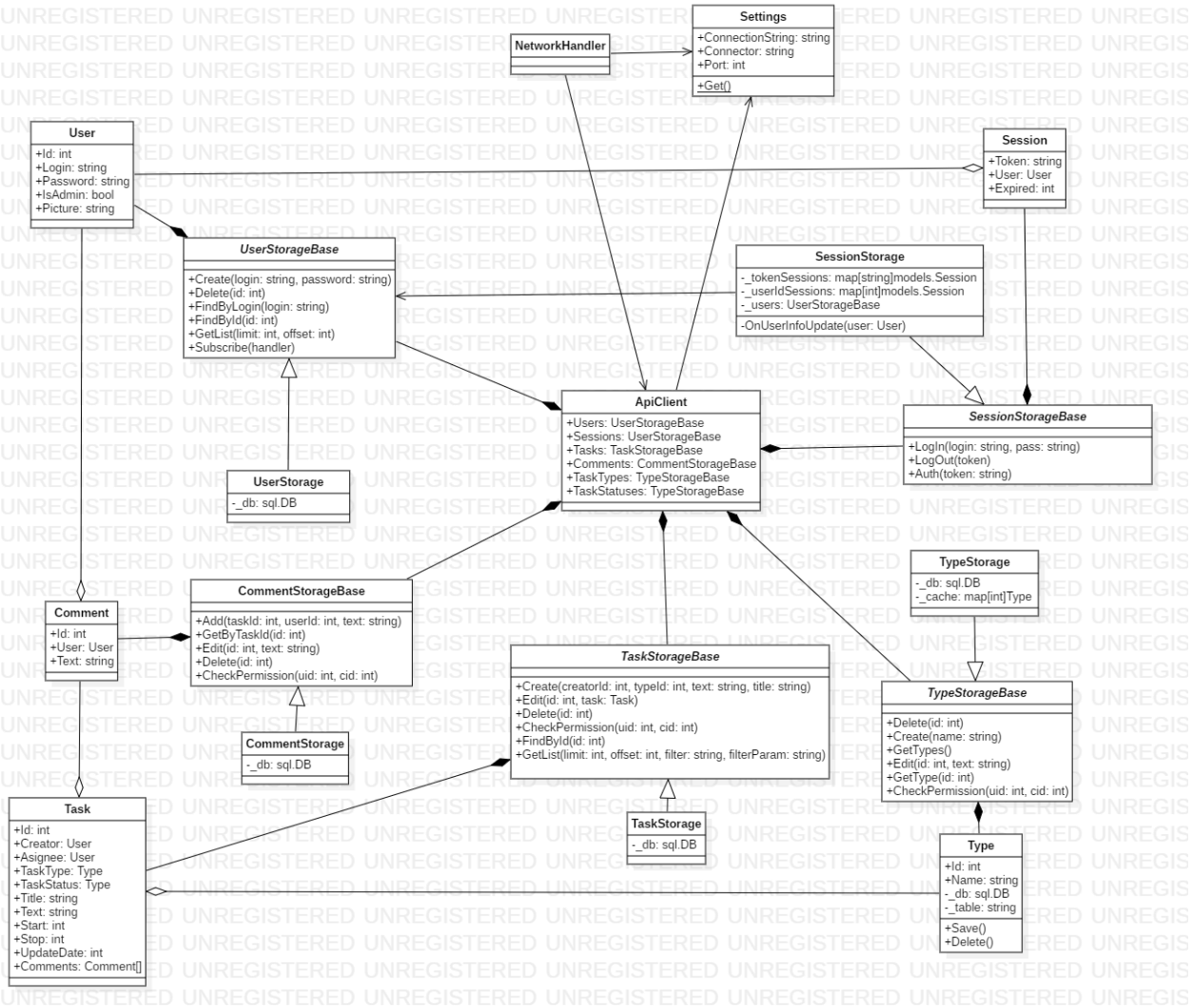


3.3.1. Расширенные описания прецедентов

Редактирование задачи

Описание	Прецедент дает возможность изменить данные о задаче
Субъекты	Пользователь
Предусловия	Клиент изъявил желания на изменение задачи при условии, что клиент имеет право на данную задачу
Основной поток	Для изменения задачи клиент заполняет данные в специальной форме задачи. В форму заносятся параметра, которые пользователь желает изменить
Альтернативный поток	Отказ в изменение задачи при отсутствие прав на нее
Постусловия	Данные задачи будут изменены

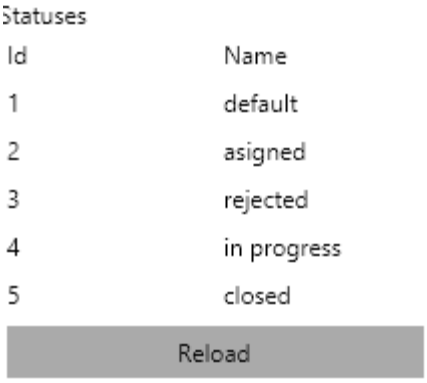
3.4. Диаграмма классов с использованием паттернов



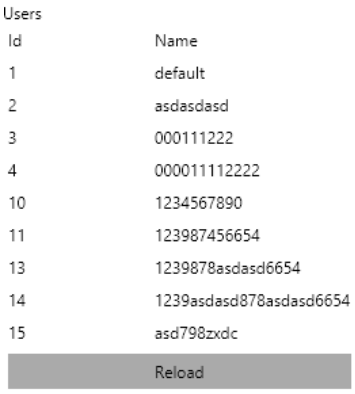
3.5. Прототип пользовательского интерфейса

3.5.1. Форма входа

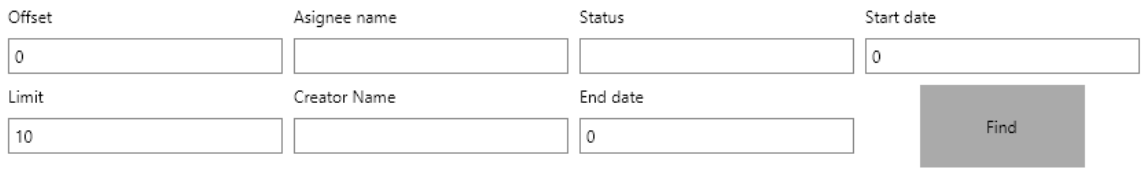
3.5.2. Список статусов задач



3.5.3. Список пользователей



3.5.4. Форма поиска задач

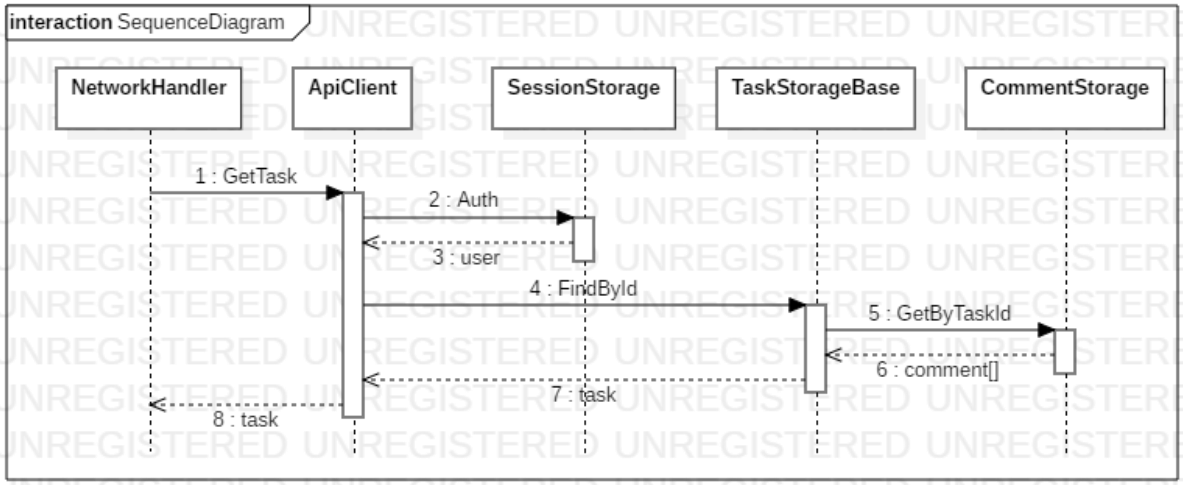


Id	Title	Asignee	Creator	Start date	End date	Status	Comments
10	new	000111222	000111222	42	1707	rejected	20
15	new	000111222	000111222	0	12345	default	16
17	new	000111222	000111222	0	0	default	0
19	test 1	000111222	000011112222	0	0	default	0
20	yey	default	000011112222	0	0	default	5
21	qweqwe	default	000111222	0	0	default	0

3.5.5. Форма просмотра задачи

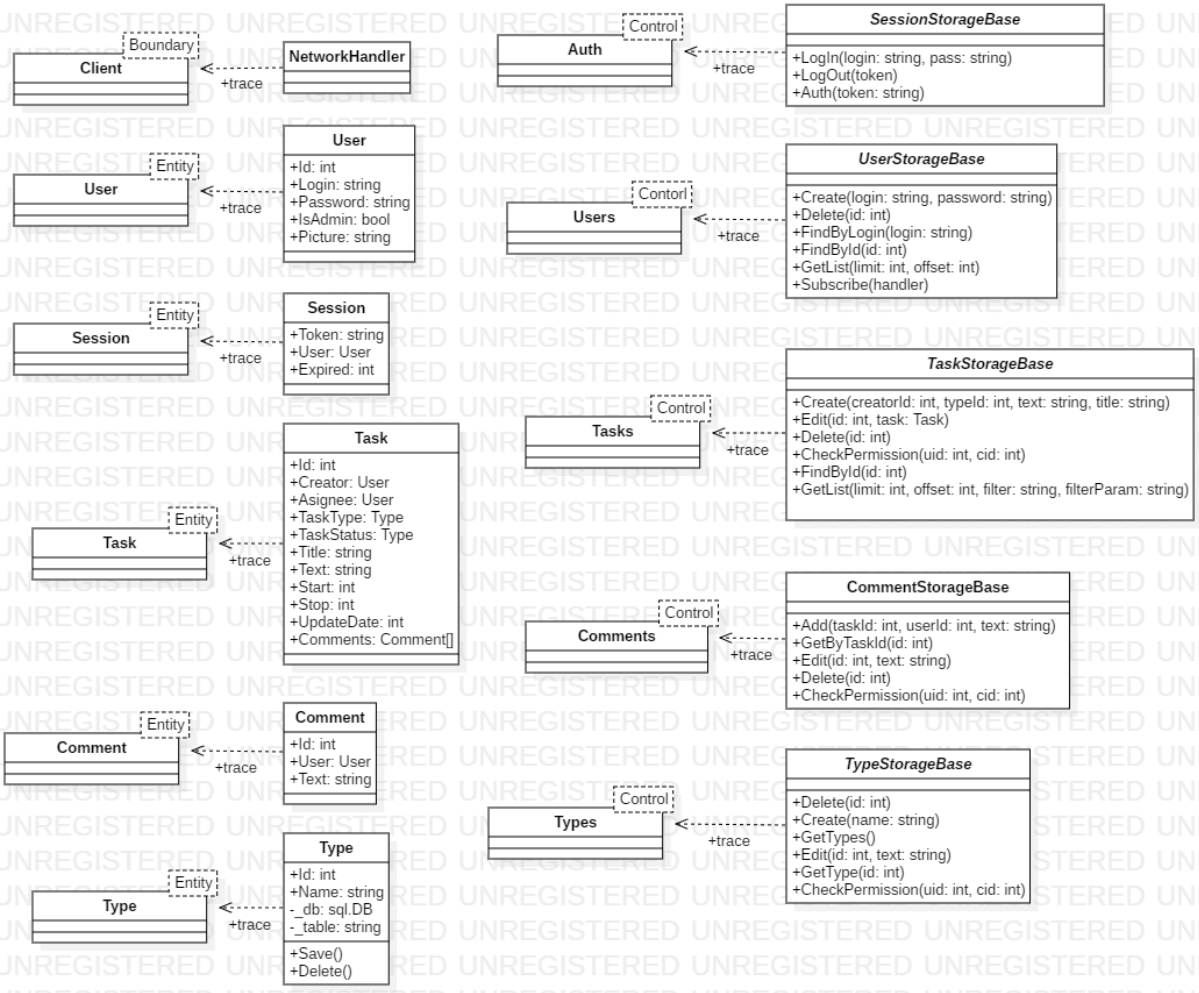
3.5.6. Форма создания задачи

3.6. Диаграммы последовательностей

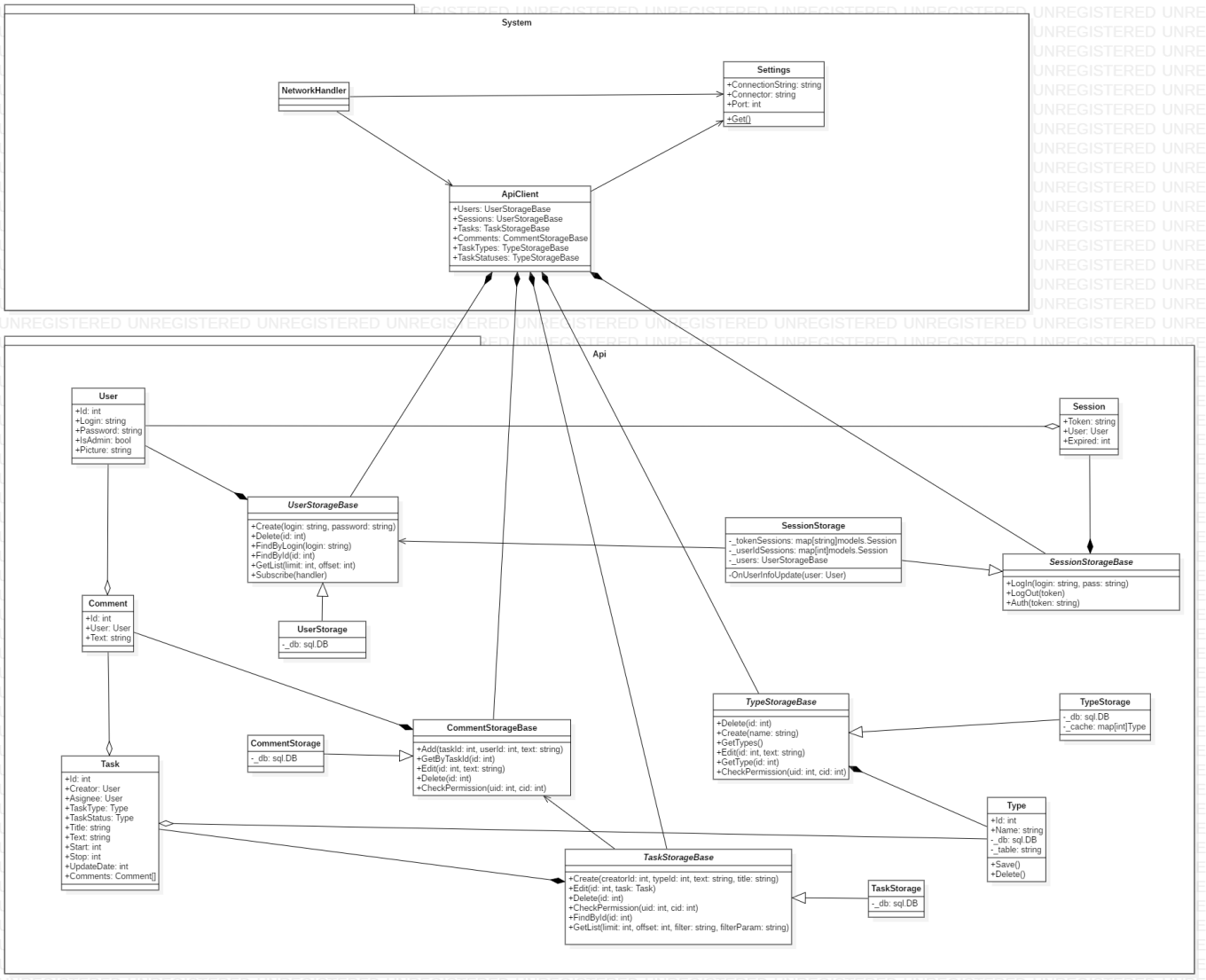


3.7. Трассировка

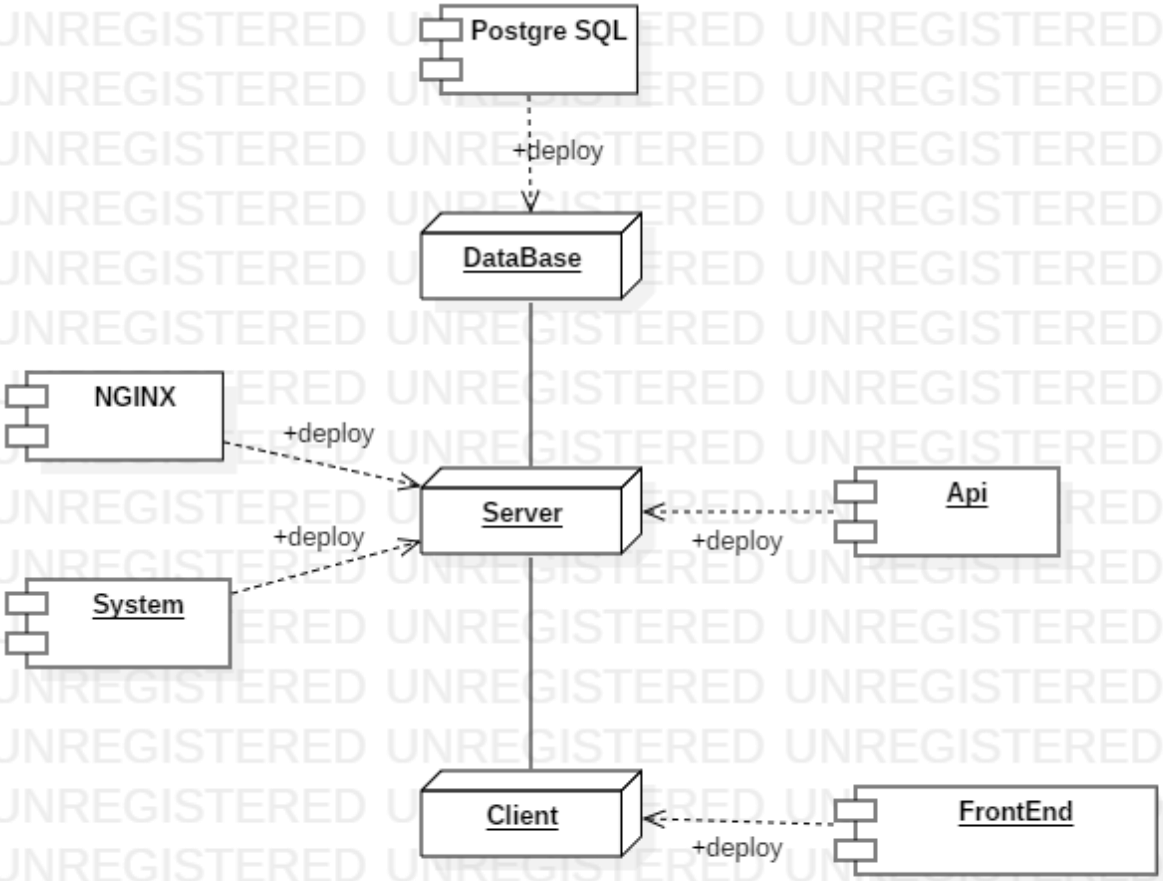
Классы проектирования в классы проектирования



3.8. Пакеты системы



3.9. Архитектура системы



3.10. Итеративная разработка

В рамках рамках работы будут реализованы две итерации проекта:

1. Реализация серверной части системы: разработка схемы базы данных, разработка api для взаимодействия клиентской части с серверной, аутентификация, выборка и изменение данных системы
2. Реализация клиентской части: реализация поддержки взаимодействия клиента с api сервера, проекция данных, полученных через api на классы представлений, реализация пользовательского интерфейса

Первая итерация включает в себя разработку всех прецедентов на стороне сервера
Первая итерация включает в себя разработку всех прецедентов на стороне клиента

3.11. Тестирование

3.11.1. Функциональное тестирование

Тестирование интерфейса авторизации

Действие	Ожидаемый результат	Результат
Запустить приложение	Приложение запустится на странице авторизации	Приложение открывается на странице авторизации
Ввести некорректные пользовательские данные и/или адрес хоста	Приложение останется на странице авторизации и на отобразит сообщение об ошибке	Приложение остается на странице авторизации и отображает ошибку
Ввести корректные пользовательские данные и адрес хоста	Приложение перейдет на страницу поиска задач	Приложение переходит на страницу поиска задач

Тестирование интерфейса поиска задач

Действие	Ожидаемый результат	Результат
Ввести лимит и сдвиг задач	Задачи найдены в соответствие с заданными лимитом и сдвигом	Задачи найдены в соответствие с заданными лимитом и сдвигом
Ввести имя пользователя в поле фильтра по создателю задачи	Задачи найдены в соответствие с пользователем, создавшим их	Задачи найдены в соответствие с пользователем, создавшим их
Ввести имя пользователя в поле фильтра по исполнителю задачи	Задачи найдены в соответствие с пользователем, назначенным на их выполнение	Задачи найдены в соответствие с пользователем, назначенным на их выполнение
Ввести название статуса задачи в поле фильтра задачи по статусу	Задачи найдены в соответствие с их статусом	Задачи найдены в соответствие с их статусом
Ввести дату начала задачи в поле фильтра поиска по дате начала	Задачи найдены в соответствие с датой их начала	Задачи найдены в соответствие с датой их начала
Ввести дату конца задачи в поле фильтра поиска по дате конца	Задачи найдены в соответствие с датой их конца	Задачи найдены в соответствие с датой их конца

Тестирование интерфейса редактирования задач

Действие	Ожидаемый результат	Результат
Выбрать id задачи и открыть ее	Отобразится задача в соответствии с выбранным id	Отображается задача в соответствии с выбранным id
Отредактировать задачу, не обладая правами на нее	Задача изменена не будет, выведется сообщение об ошибке	Задача не меняется и выводится сообщение об ошибке
Отредактировать задачу обладая правами на нее	Задача будет изменена	Задача изменена
Удалить задачу, не обладая правами на нее	Задача не удалится	Задача не удалилась
Удалить задачу, обладая правами на нее	Задача удалится	Задача удалилась

Тестирование интерфейса создания задачи

Действие	Ожидаемый результат	Результат
Заполнить форму	Задача создаться	Задача создалась и добавилась в список

Тестирование интерфейса статусов и пользователей

Действие	Ожидаемый результат	Результат
Нажать на кнопку обновления	Данные обновятся	Данные обновляются

4.Этап внедрения

4.1. Рекомендации по установке

Для установки серверной части системы необходимо наличие операционная система, поддерживаемая компилятором ЯП Golang версии 11 и совместимая с СУБД postgresql. Для запуска серверной части необходимо создать БД из образа, поставляемого вместе с системой и указать в файле конфигурации сервера строку подключения к БД.

Установка на стороне клиента не требуется, все что нужно это ОС, совместимая с .net core 2.2

4.2. Перечень документации для пользователей и заказчиков

- Расчетно-пояснительная записка

4.3. Рекомендации по внедрению

- Провести обучение сотрудников по использованию системы
- Для клиентов сделать интерактивный обучающий ролик для работы с системой