|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Информатика и системы управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_Программное обеспечение ЭВМ и информационные технологии\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОМУ ПРОЕКТУ***

***НА ТЕМУ:***

Разработка приложения для ведения журнала расходов и заметок

Студент \_\_\_\_\_ИУ7-23М\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_**В. Г. Горячев **\_\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_**Г. А. Щетинин **\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2023 г.*

Оглавление

[ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 5](#_Toc137830349)

[Введение 6](#_Toc137830350)

[1. Аналитический раздел 7](#_Toc137830351)

[1.1 Описание системы 7](#_Toc137830352)

[1.2 Ролевая система 7](#_Toc137830353)

[1.3 Сценарии использования 8](#_Toc137830354)

[1.4 Существующие аналоги 10](#_Toc137830355)

[1.5 Модель сущностей системы 11](#_Toc137830356)

[Вывод 16](#_Toc137830357)

[2. Конструкторский раздел 17](#_Toc137830358)

[2.1 Концептуальный дизайн системы 17](#_Toc137830359)

[2.2 Архитектура системы 18](#_Toc137830360)

[2.3 Структура базы данных 20](#_Toc137830361)

[2.4 Входные данные 24](#_Toc137830362)

[2.5 Выходные данные 25](#_Toc137830363)

[2.6 Проектирование взаимодействия 25](#_Toc137830364)

[3. Технологический раздел 27](#_Toc137830365)

[3.1 Выбор языка и среды программирования 27](#_Toc137830366)

[3.2 Описание реализации серверной части 27](#_Toc137830367)

[3.3 Развёртывание системы 28](#_Toc137830368)

[3.4 Тестирование 28](#_Toc137830369)

[3.5 Программный интерфейс системы (API) 29](#_Toc137830370)

[Заключение 31](#_Toc137830371)

[Список использованных источников 32](#_Toc137830372)

[Приложение А 33](#_Toc137830373)

# ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| Ведение бюджета | Систематическое ведение записей доходов и расходов с последующим автоматизированным или ручным подсчётом сумм доходов и расходов, баланса, трат по категориям и других метрик, в зависимости от требуемой информации. |
| WEB-интерфейс | Интерфейс пользователя, предоставляемой системой через Web-браузер. В разрабатываемой системе только один веб-интерфейс. |
| Область видимости | В рамках Системы может быть личной или групповой. Используется как способ организации доступа к данным и его разграничения. Этот термин описывает только логику Системы и отличается от определения «области видимости» в программировании |
| СОА (SOA) | Сервис-ориентированная архитектура (Service Oriented Architecture), [модульный](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) подход к разработке [программного обеспечения](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), основанный на использовании [распределённых](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%91%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F), [слабо связанных](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D0%BB%D0%B0%D0%B1%D0%BE%D0%B5_%D1%81%D0%B2%D1%8F%D0%B7%D1%8B%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) заменяемых компонентов, оснащённых стандартизированными [интерфейсами](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9) для взаимодействия по стандартизированным [протоколам](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) |
| Профиль пользователя | Информация о пользователе портала, хранящаяся в портале |
| Сессия | Сессия на сайте – серия запросов к порталу, сделанных одним пользователем в заданный промежуток времени |
| Фронтенд | Серверное приложение, принимающее запросы от пользователя портала. На каждый из типов запросов от пользователя фронтенд определяет, как организовать выполнение запроса. Фронтенд принимает запросы от пользователя, анализирует их и в соответствии с заложенным алгоритмом выполняет запросы к бекендам |
| Бекенд | Серверное приложение, выполняющее определенную задачу, например, взаимодействие с СУБД. Бекенды принимают запросы от фронтенда |
| Сборка | Процесс формирования из исходных программных кодов и скриптов, в соответствии с заданными настройками, готового к запуску программного продукта |
| Разворачивание | Развёртывание и запуск веб-приложения или сайта в его рабочей среде, то есть на сервере или хостинге, первичная инициализация веб-приложения (если требуется) |
| OpenID Connect | Открытый стандарт децентрализованной системы аутентификации, предоставляющей пользователю возможность создать единую учётную запись для аутентификации на множестве не связанных друг с другом интернет-ресурсов, используя услуги третьих лиц (приложение 4) |

# Введение

Распределённая учётная система «Бюджет и задачи» (далее просто Система) предназначена для автоматизации таких действий, как ведение заметок, бюджета (журналирование доходов и расходов) и текущих задач, а также обеспечение многопользовательского доступа к этим возможностям. Это позволяет применять Систему как для личного или семейного пользования, так и для небольших организаций, для которых программные системы, обладающие расширенным функционалом, избыточны или слишком дороги (не обеспечивают заметной выгоды по сравнению со стоимостью приобретения или подписки).

Целью курсового проекта является разработка системы для ведения бюджета, задач и работы с заметками на основе микросервисной архитектуры. Для решения поставленной задачи необходимо:

* формализовать задание и определить необходимый функционал;
* провести анализ существующих решений;
* описать архитектуру системы;
* описать структуру базы данных;
* реализовать прототип системы «Бюджет и задачи»

# Аналитический раздел

## 1.1 Описание системы

Предлагаемый формат системы – веб-приложение, доступ к которому с помо-щью браузера может осуществляться как со стационарных, так и с мобильных устройств, и которое основано на микросервисной архитектуре. Распределённый характер Системы призван решать задачи синхронизации данных между пользователями и не привязывать работу с Системой к конкретному устройству.

В рассматриваемой предметной области требуется, в первую очередь, возможность быстро и централизованно создавать записи о расходах или доходах, задачи и заметки, при этом заметка может являться прототипом задачи, а задача может сопровождаться определёнными изменениями бюджета. Добавление возможностей по анализу бюджета или истории задач может стать естественным путём дальнейшего совершенствования Системы.

Разрабатываемая система может предоставляться пользователю как услуга (хранение данных и работа на облачных серверах, которыми пользователь не распоряжается), так и как коробочное решение.

## 1.2 Ролевая система

Ролевая система должна включать в себя две роли, обладающие следующими правами и возможностями:

1. Пользователь:
   1. имеет доступ к сервисам Системы в рамках зоны его личной и групповых зон видимости (видит и может изменять только то, к чему относится сам);
   2. может получать доступ к открытой информации всех других пользователей (в рамках разрабатываемой системы это только имя и логин);
2. Администратор:
   1. как и пользователь, имеет доступ к сервисам Системы в рамках зоны его личной и групповых зон видимости;
   2. может получать доступ к открытой информации всех других пользователей;
   3. имеет доступ ко всем групповым зонам видимости, то есть способен просматривать записи всех групп, созданных другими пользователями;
   4. не имеет доступа к личным зонам видимости;
   5. имеет доступ к странице администратора.

Гостевая роль не рассматривается ввиду того, что единственные доступные ей действия – это авторизация, если такой пользователь зарегистрирован в системе.

## 1.3 Сценарии использования

Рассмотрим Use Cases диаграммы (диаграммы использования). В них описываются действующие лица и прецеденты – конкретные действия, которые выполняет актор при работе с системой.

На рис. 1 и 2 представлены диаграммы для обеих рассматриваемых ролей.

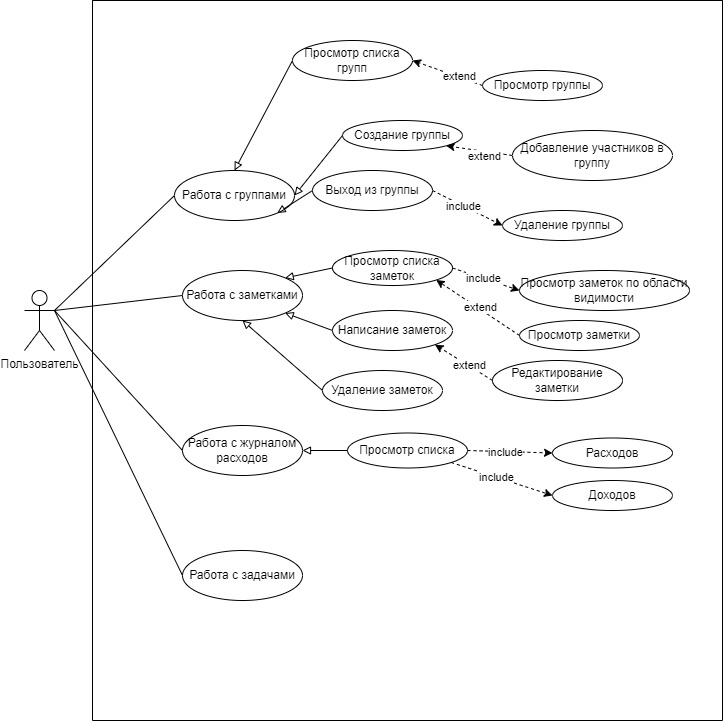


Рисунок 1 — Диаграмма прецедентов с точки зрения пользователя



Рисунок 2 — Диаграмма прецедентов с точки зрения администратора

## 1.4 Существующие аналоги

Аналогами Системы в области отдельных задач, выполняемых ею, являются все сервисы, позволяющие:

* делать заметки – бумажные блокноты, Google Keeps и аналоги, личный диалог (“Избранное”, где оставлять и читать какие-то сообщения может только пользо-ватель) и каналы в Telegram и других мессенджерах;
* вести бюджет – таблицы Excel, онлайн-таблицы Google, приложение “Журнал расходов” на Android и аналоги;
* записывать задачи/планы – то же, что и для ведения заметок, канбан-доски в Github и Gitlab, отдельные реализации канбан-досок, календари, встроенные приложения почты (например, в почте Mail.ru) и др.

Таким образом, у Системы достаточно много аналогов на разных платформах и в разных областях её функционала, однако её преимуществом станет объединение возможностей в рамках одного приложения.

## 1.5 Модель сущностей системы

Отношения сущностей предметной области представлены в виде ER-диаграммы в нотации Crow’s foot на рис. 3.

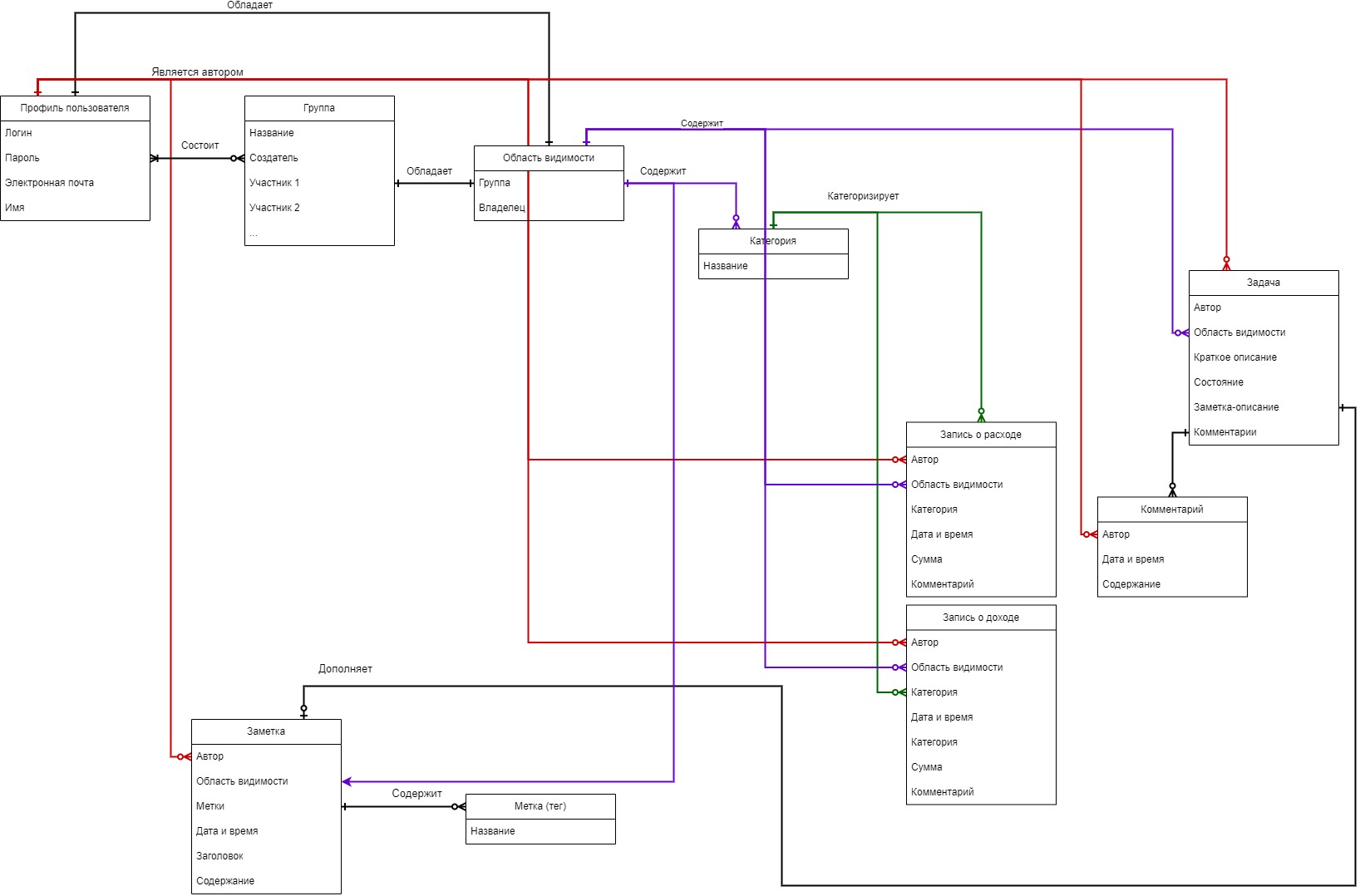


Рисунок 3 – ER-диаграмма системы

Система должна предоставлять следующие возможности, представленные в виде таблиц 1-4. Содержащиеся в них функции приводятся с точки зрения Пользователя. Отличия версии для Администратора приводятся в отдельной строке внизу таблицы.

Одним из важных понятий является область видимости (далее ОВ), определение которой приводится в глоссарии. В Системе ОВ может быть личной (все создаваемые в этой ОВ записи видны только самому пользователю, который их создал) и групповой (записи в этой ОВ доступны только участникам выбранной группы).

Работа с пользовательскими учётными записями и группами

|  |  |
| --- | --- |
| Функции | 1) регистрация нового пользователя с валидацией вводимых данных;  2) аутентификация пользователей;  3) ролевое разграничение на пользователей и администраторов (требования к ролевой системе изложены в отдельном пункте);  4) работа с группами:  а) создание группы;  б) просмотр состава в виде списка логинов и имён участников группы;  в) просмотр собственного списка групп, в которых пользователь состоит;  г) приглашать другого пользователя по логину в группу;  д) выход из группы;  е) автоматическое удаление группы при выходе всех участников;  5) справка по предоставляемым функциям. |
| Основные объекты, их характеристика | **Профиль пользователя**: логин, пароль, имя Пользователя, электронная почта.  **Группа:** название, один или несколько Пользователей. |
| Связи объектов | *Один Пользователь может состоять во множестве Групп.*  *Одна Группа может включать множество Пользователей.* |
| Отличия у администратора | 1. Администратор не может зарегистрироваться, можно только создать нового или назначить существующего пользователя через базу данных; 2. Администратор может просматривать список и состав всех созданных пользователями групп. |

Работа с ведением бюджета

|  |  |
| --- | --- |
| Функции | 1. работа с категориями расходов и доходов:    1. создание категории с областью видимости:       * личная;       * для выбранной группы;    2. переименовать категорию;    3. удалить категорию;    4. при удалении категории все записи… (ещё не решил, что должно происходить); 2. работа с записями о доходах и расходах:    1. просмотр записи;    2. создание записи с выбором группы и категории;    3. изменение записи;    4. просмотр списка всех записей в сортировке по убыванию даты:       * по категориям;       * по группам;       * за всё время или за период: день, неделя, текущий месяц, год;    5. удаление записи; 3. вычисление баланса (по личным записям пользователя или по выбранной группе); 4. справка по предоставляемым функциям. |
| Основные объекты, их характеристика | **Категория:** название, принадлежность к ОВ (к личной ОВ текущего пользователя или к одной из его групп). Категории не должны повторяться (  **Запись о расходе или доходе:** комментарий произвольной длины, сумма в рублях, выбранная ОВ, категория в выбранной ОВ, автор, дата и время. |
| Связи объектов | *Одна Группа имеет одну ОВ.*  *Один Пользователь имеет одну ОВ.*  *Одна ОВ имеет множество Категорий.*  *Один Пользователь является автором множества Записей.*  *Одна Запись имеет одну ОВ, Категорию и одного автора.* |
| Отличия у администратора | 1. Администратор имеет возможность просматривать ОВ всех групп. |

Работа с заметками

|  |  |
| --- | --- |
| Функции | 1. просмотр списка всех заметок в сортировке по убыванию даты:    1. по выбранной группе;    2. личные; 2. просмотр заметки 3. создать заметку с выбором группы:    1. при создании заметки с меткой «Задача» должна автоматически создаваться задача, содержащая данную заметку в качестве вложения; 4. изменить заметку; 5. удалить заметку. |
| Основные объекты, их характеристика | **Заметка**: автор, дата и время, выбранная ОВ, заголовок, текст-содержание, опциональные теги (метки). |
| Связи объектов | *Одна Группа имеет одну ОВ.*  *Один Пользователь имеет одну ОВ.*  *Одна ОВ имеет множество Заметок.*  *Один Пользователь является автором множества Заметок.*  *Одна Заметок может иметь несколько тегов (меток).* |
| Отличия у администратора | 1. Администратор имеет возможность просматривать ОВ всех групп. |

Работа с задачами

|  |  |
| --- | --- |
| Функции | 1. просмотр списка всех задач в сортировке по убыванию даты:    1. по группе;    2. личные задачи; 2. просмотр задачи; 3. создание задачи (по умолчанию всем новым задачам задаётся статус «Создана» с выбором группы; 4. изменение задачи; 5. изменение статуса задачи; 6. просмотр комментариев к задаче в сортировке по убыванию даты; 7. добавление комментария к задаче; 8. удаление комментария к задаче; 9. удаление задачи. |
| Основные объекты, их характеристика | **Задача**: автор, дата и время, название, краткое описание, ссылка на Заметку (может быть пустой), статус задачи, который может принимать одно из заданных состояний (приложение 1).  **Комментарии**: имя автора, текст, время публикации. Комментарии видны только вместе с задачей.  **Статус:** строка из заданного разработчиками набора (приложение 1). |
| Связи объектов | *Одна Группа имеет одну ОВ.*  *Один Пользователь имеет одну ОВ.*  *Одна ОВ имеет множество Заметок.*  *Одна Задача может иметь один статус, множество Задач могут принимать один Статус.*  *Один Пользователь является автором множества Задач.*  *Одна Задача может ссылаться на одну Заметку.*  *Одна Задача может иметь множество Комментариев.* |
| Отличия у администратора | 1. Администратор имеет возможность просматривать ОВ всех групп. |

## Вывод

В результате проведённого анализа были сформулированы основные требования к системе в виде диаграмм прецедентов, её сущностная и ролевая системы, а также рассмотрены аналоги.

# 2. Конструкторский раздел

## 2.1 Концептуальный дизайн системы

Концептуальный дизайн системы содержит наиболее общие схемы описания функционала приложения с точки зрения пользователей. На рис. 4 представлена диаграмма IDEF0 первого уровня. Данный вид диаграммы позволяет формализовать описание запросов пользователя и ответов системы на них, отобразив систему в виде «черного» ящика.

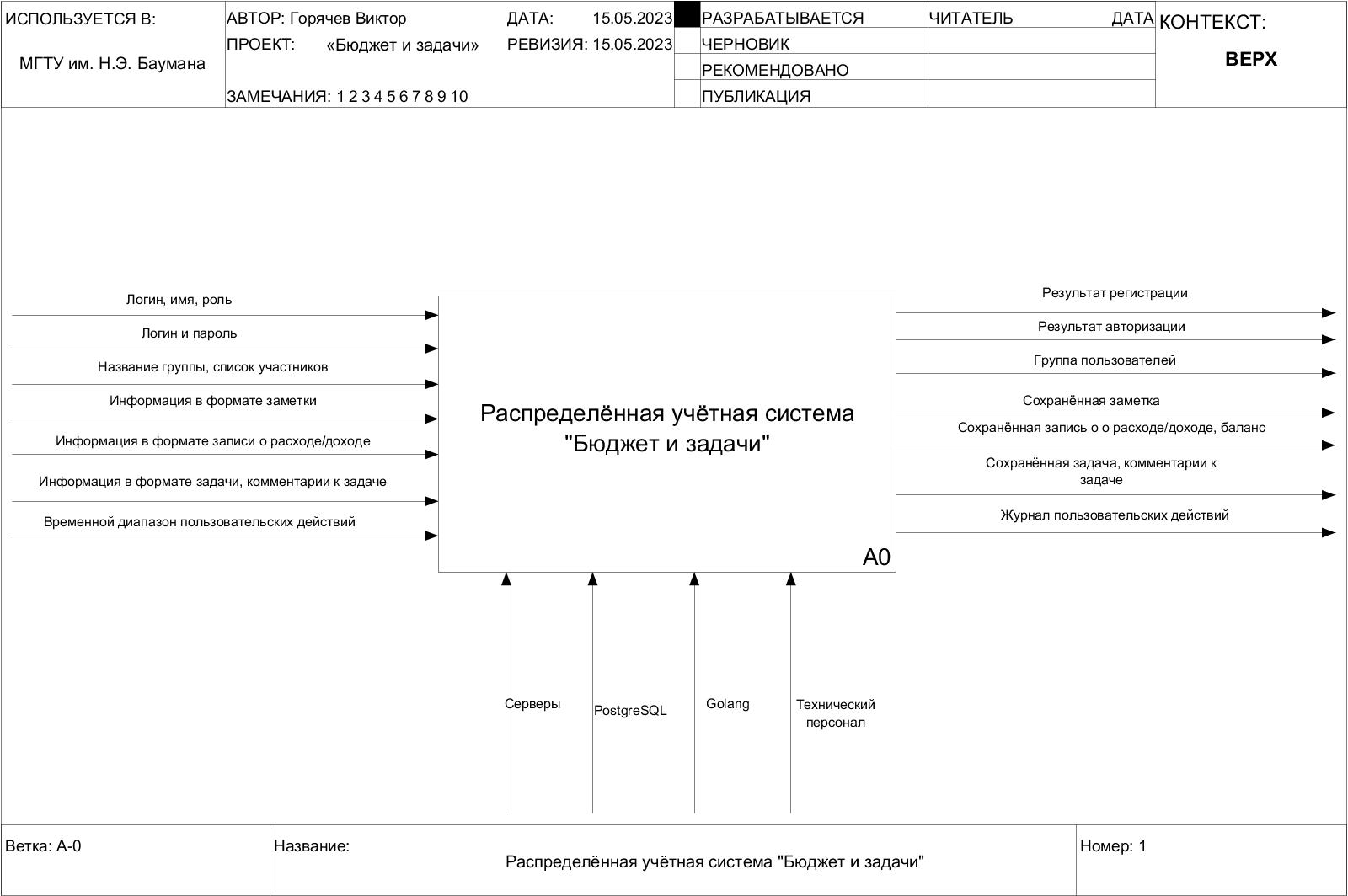


Рисунок 4 – Концептуальная модель системы в нотации IDEF0

На рис. 5 представлена детализированная диаграмма, которая определяет последовательность выполнения операций в системе при создании задачи.

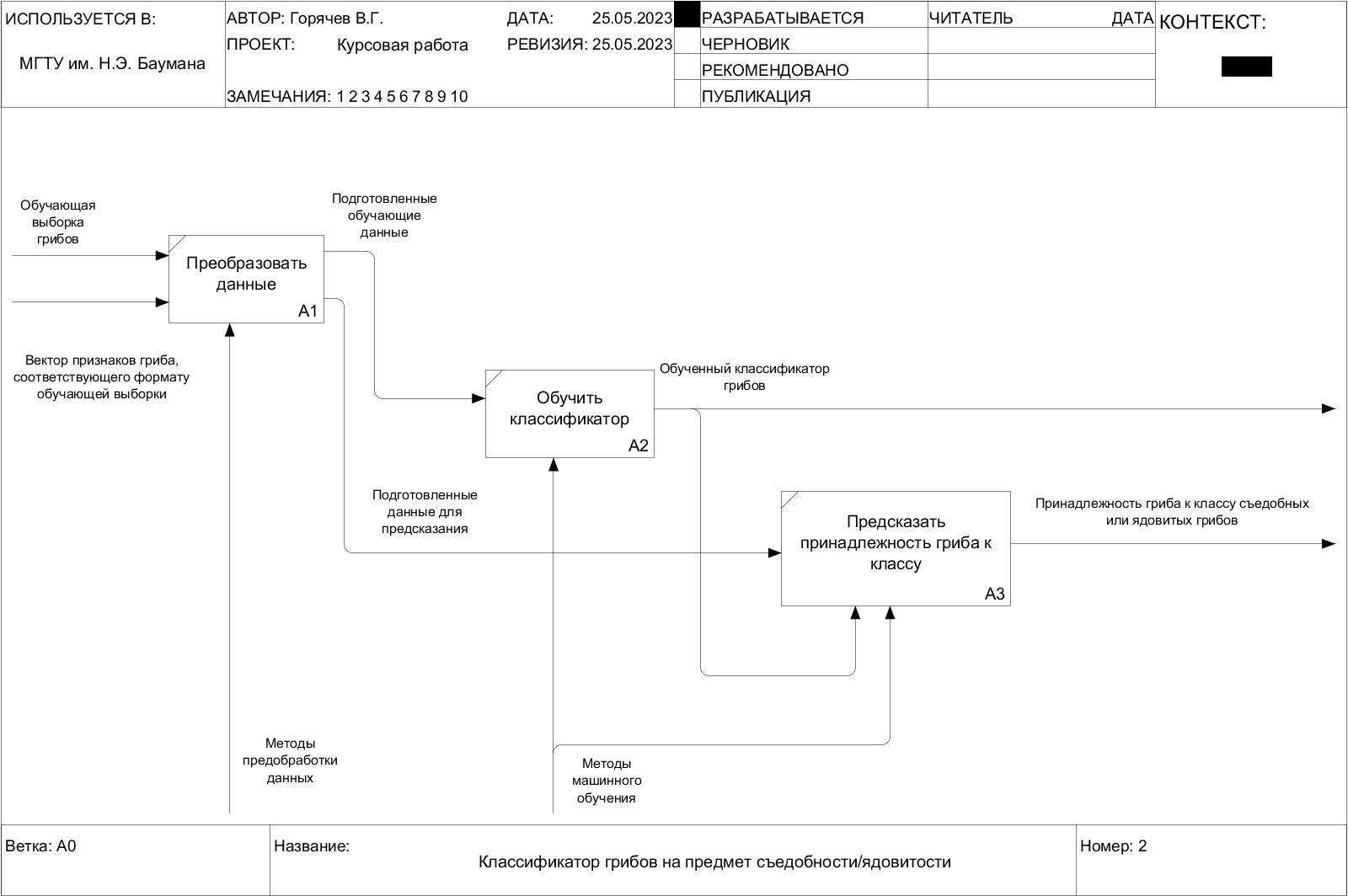


Рисунок 5 – Детализированная концептуальная модель системы в нотации IDEF0 при создании задачи

## 2.2 Архитектура системы

Общий вид, структура Системы представлены на рис. 3 в соответствии с СОА. Географически данная структура может быть разнесена по разным серверам (каждый отмеченный на схеме сервис, каждая БД), но может функционировать и на одном сервере. В таком случае допустимо все *логические* БД сервисов хранить под управлением одной СУБД, в одном сервисе.

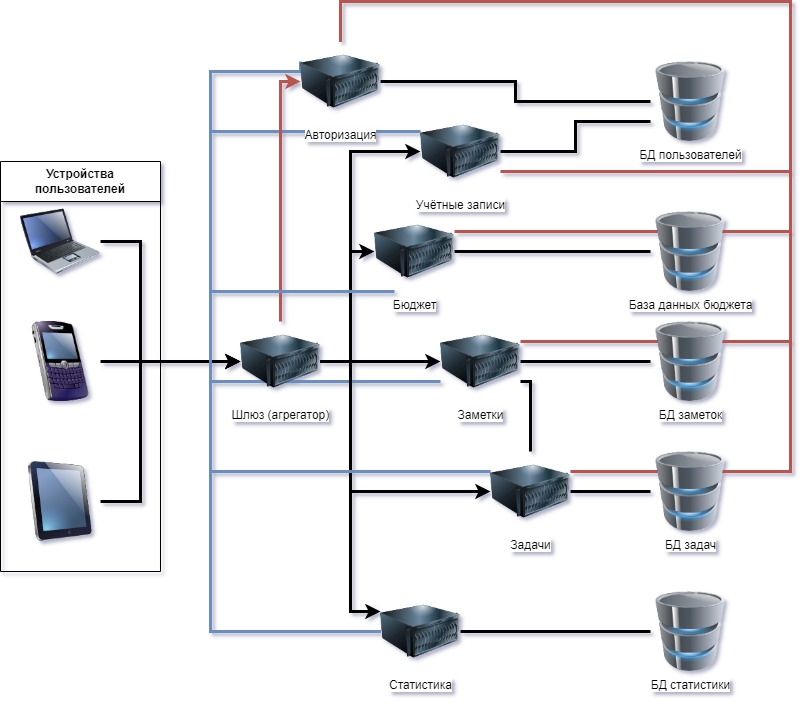


Рисунок 6 — Структура системы. Красными линиями обозначены связи, используемые для проверки прав доступа, синими – взаимодействие в рамках сбора статистики, чёрными – выполнение сервисами основных функций. Стрелки обозначают только «заимствование» функционала сервисом-агрегатором

Каждый сервис сформирован в соответствии с логическим делением функциональности, заданной требованиями. Однако про реализации сервисы авторизации и учётных записей могут быть объединены.

Пользователи взаимодействуют, используя браузер своего устройства, с сервисом-агрегатором, который позволяет им авторизоваться и использовать функционал Системы. В соответствии с требованиями, Шлюз отправляет пользователю при первом запросе фронтенд часть, после чего играет роль обработчика запросов от фронтенда, который пересылает запросы на другие сервера. Поскольку не подразумевается использование локальных баз данных, а данные достаточно независимы друг от друга, проблема синхронизации и транзакций будет стоять не так остро, как могла бы.

## 2.3 Структура базы данных

На основе функциональных требований к выделенным подсистемам, а также объектов, о которых необходимо хранить данные в системе, была разработана схема данных приложения. Результат ее проектирования отображен на условной схеме базы данных, представленной на рис. 7.

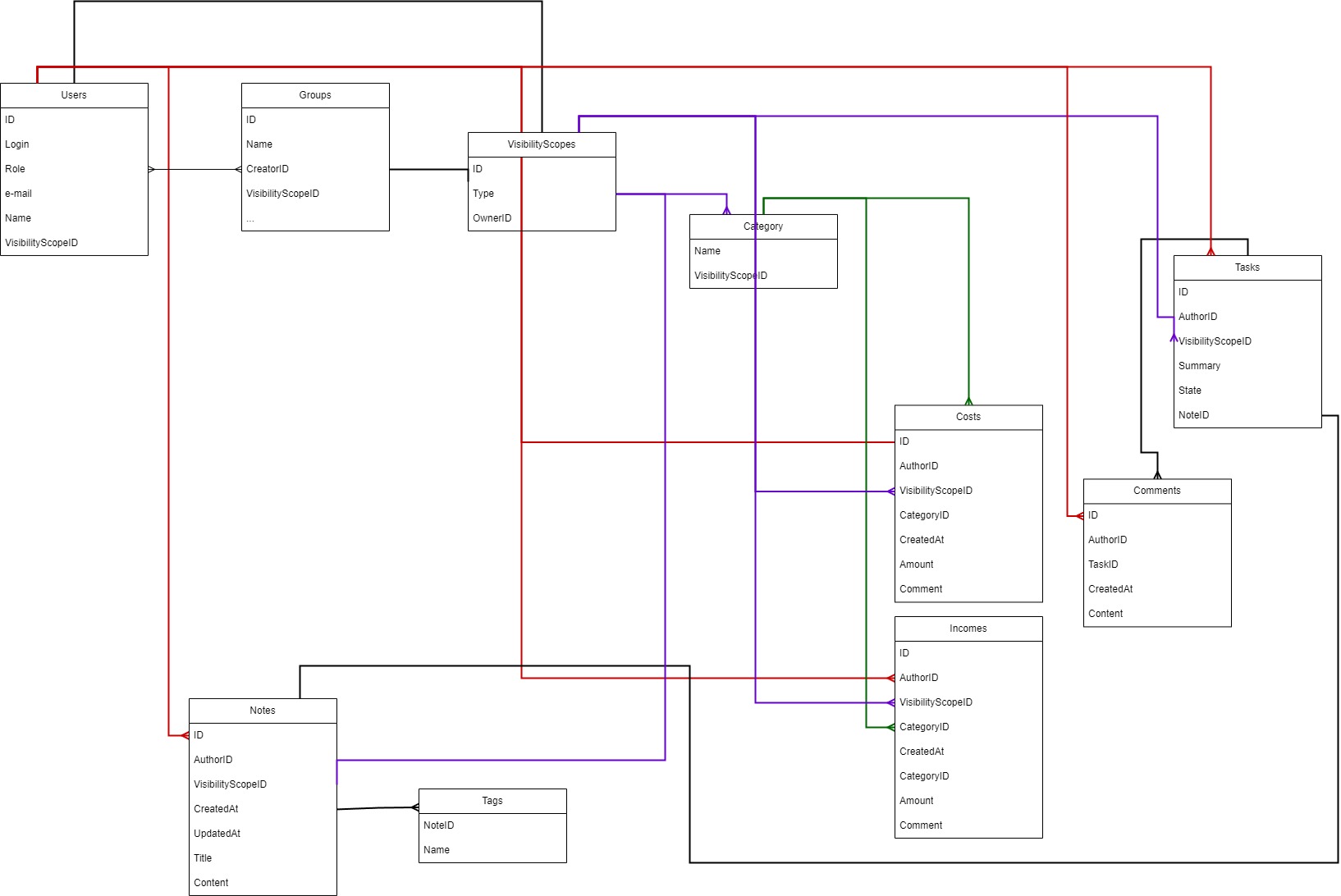


Рисунок 7 — Общая (неразделённая на микросервисы) схема базы данных

Также каждая таблица описана соответствующей спецификацией.

**Спецификация таблицы Users**

Данные о профиле пользователя

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | guid | Идентификатор пользователя |
| Login | string |  |
| Name | string | Имя пользователя |
| Role | int | Права пользователя |
| Email | string | Электронная почта |
| VisibilityScopeID | int | Идентификатор личной ОВ пользователя |

**Спецификация таблицы GroupMembers**

Данные о связи групп и пользователей

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| UserID | guid | Идентификатор пользователя |
| GroupID | guid | Идентификатор группы |

**Спецификация таблицы Groups**

Данные о группе

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | guid | Идентификатор группы |
| Name | string | Название группы |
| CreatorID | guid | Идентификатор создателя группы |
| VisibilityScopeID | int | Идентификатор ОВ группы |

**Спецификация таблицы VisibilityScopes**

Данные об областях видимости

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | int | Идентификатор ОВ |
| Type | int | Тип ОВ, групповая или личная |
| OwnerID | guid | Идентификатор владельца ОВ |

**Спецификация таблицы Category**

Данные о категориях доходов/расходов

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | int | Идентификатор |
| Name | string | Название категории |
| VisibilityScopeID | int | Идентификатор ОВ категории |

**Спецификация таблицы Costs**

Данные о расходах

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | int | Идентификатор записи |
| AuthorID | guid | Идентификатор автора |
| VisibilityScopeID | int | Идентификатор ОВ |
| CategoryID | int | Идентификатор категории |
| CreatedAt | timestamp | Время создания |
| Amount | int | Сумма расхода |
| Comment | string | Комментарий |

**Спецификация таблицы Incomes**

Данные о доходах

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | int | Идентификатор записи |
| AuthorID | guid | Идентификатор автора |
| VisibilityScopeID | int | Идентификатор ОВ |
| CategoryID | int | Идентификатор категории |
| CreatedAt | timestamp | Время создания |
| Amount | int | Сумма дохода |
| Comment | string | Комментарий |

**Спецификация таблицы Notes**

Данные о заметках

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | int | Идентификатор записи |
| AuthorID | guid | Идентификатор автора |
| VisibilityScopeID | int | Идентификатор ОВ |
| UpdatedAt | timestamp | Время обновления |
| CreatedAt | timestamp | Время создания |
| Title | string | Заголовок |
| Content | string | Содержание |

**Спецификация таблицы Tags**

Данные о метках (тегах)

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| NoteID | int | Идентификатор родительской заметки |
| Name | string | Название-содержание |

**Спецификация таблицы Tasks**

Данные о профиле пользователя

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | int | Идентификатор записи |
| AuthorID | guid | Идентификатор автора |
| VisibilityScopeID | int | Идентификатор ОВ |
| UpdatedAt | timestamp | Время обновления |
| CreatedAt | timestamp | Время создания |
| Summary | string | Краткое содержание |
| State | string | Состояние |
| NoteID | int | Идентификатор поясняющей заметки |

**Спецификация таблицы Comments**

Данные о профиле пользователя

|  |  |  |
| --- | --- | --- |
| **Имя атрибута** | **Тип атрибута** | **Описание атрибута** |
| ID | guid | Идентификатор пользователя |
| AuthorID | guid | Идентификатор автора |
| TaskID | int | Идентификатор задачи |
| CreatedAt | timestamp | Время создания |
| Content | string | Текст |

## 2.4 Входные данные

На входные данные системы накладываются следующие ограничения.

При регистрации, работе со своей учётной записью и группами пользователь должен будет передавать следующую информацию:

* логин (строка длиной до 20 символов, без пробелов и специальных символов, кроме «\_» и «-», латиница, не может быть пустой);
* пароль (строка длиной до 20 символов, без пробелов, латиница и спец. символы, не может быть пустой);
* имя (строка длиной до 80 символов, латиница, кириллица, пробелы, апострофы, дефисы, без остальных спец. символов, не может быть пустым);
* электронная почта (строка длиной до 40 символов, должна соответствовать формату электронной почты: \*@\*.\*);
* название группы (строка длиной до 80 символов, не может быть пустой);
* логин пользователя, которого необходимо добавить в группу.

При работе со списками расходов и доходов:

* для категорий:
  + название (строка длиной до 30 символов, не может быть пустой);
  + выбранная принадлежность (пользователю должен быть предоставлен список, включающий доступные группы и принадлежность «Личная»);
* для записей о расходах или доходах:
  + выбранная ОВ;
  + категория, выбранная из списка доступных категорий;
  + комментарий (до 400 символов);
  + сумма в рублях (вещественное число, не может быть отрицательным).

При работе с заметками:

* заголовок (строка длиной до 50 символов);
* содержание (текст длиной до 2000 символов);
* метка (строка длиной до 20 символов).

При работе с задачами:

* «встроенная» заметка;
* выбранный статус задачи (строка из заданного перечня: «В очереди», «В работе» (допустим вариант «Выполняется»), «Выполнена», «Отменена», «Ожидание» (выполнение приостановлено по независящим от исполнителя причин, например, в ожидании проверки));
* краткое описание (до 150 символов);
* текст комментария (до 300 символов);

## 2.5 Выходные данные

Выходными параметрами системы являются веб-страницы. Они должны содержать следующую информацию:

* введённые ранее пользователем данные (задачи, заметки, расходы и т.д.) в структурированном виде;
* баланс (вычисленный по расходам и доходам);
* записи журнала пользовательских действий: адрес запроса, время, продолжительность, код ответа.

## 2.6 Проектирование взаимодействия

В качестве примера взаимодействия сервисов можно рассмотреть операцию создания задачи. Её диаграмма последовательности действий при успешном выполнении представлена на рис. 8. Эта схема является примером тех случаев, когда система способна корректно отработать даже в условиях деградации функциональности, когда, например, сервис заметок недоступен (хоть заметка и важна для задачи, но она не является критический необходимой), а остальные сервисы продолжают функционировать.

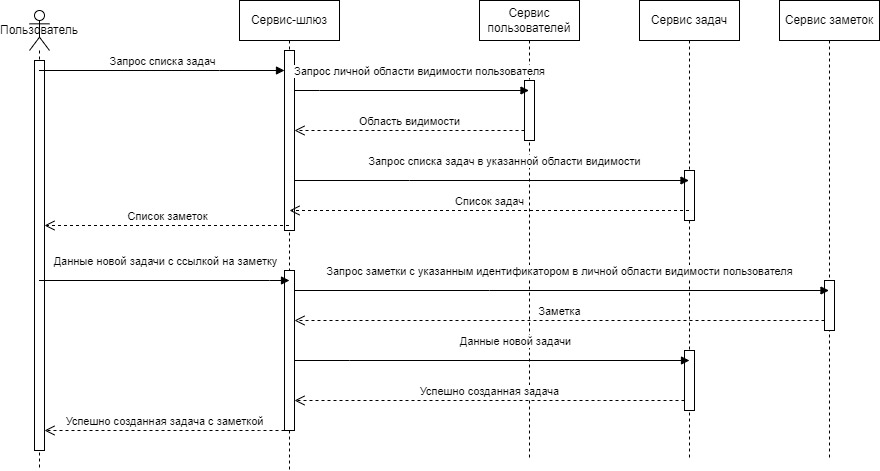


Рисунок 8 — Диаграмма последовательности создания задачи с ссылкой (идентификатором) существующей заметки в личной области видимости пользователя. Стандартный ход выполнения без ошибок

# 3. Технологический раздел

## 3.1 Выбор языка и среды программирования

В качестве языка серверной части выбран язык программирования Go [1], как один из ориентированных на веб-разработку и предоставляющий удобный инструментарий для этого. Клиентская часть написана с использованием фреймворка React, использующего язык Typescript [3].

## 3.2 Описание реализации серверной части

В качестве СУБД используется PostgreSQL версии 15 [4]. Для потока статистических сообщений используется Kafka [5].

Код серверных приложений написан без использования фреймворков, с использованием только библиотек, что обеспечивает определённую взаимозаменяемость различных модулей программы. Контроль зависимостей (отслеживание версий, косвенные зависимости, загрузка) осуществляется штатным инструментарием языка.

Все сервисы написаны для запуска в виде Docker-контейнеров (в качестве примера на листинге 1 представлен один из сборочных файлов) с использованием двухшаговой сборки, ускоряющей сборку итоговых контейнеров и уменьшающей их же размер. Управлением всей системой во время разработки осуществляется с помощью Docker compose, файл с настройками для которого приведён в приложении А.

Листинг 1 — Dockerfile сервиса заметок

|  |
| --- |
| 1. FROM golang:1.20 2. WORKDIR /pain/ 3. COPY ./go.mod . 4. COPY ./go.sum . 5. COPY ./vendor\* . 6. RUN go mod tidy && go mod vendor 7. COPY . . 8. RUN CGO\_ENABLED=0 go build -a -installsuffix cgo -o app ./cmd/notes 9. FROM alpine:3.18.0 10. RUN apk --no-cache add ca-certificates 11. WORKDIR /root/ 12. COPY --from=0 /pain/app ./ 13. EXPOSE 8050 14. CMD ["./app"] |

## 3.3 Развёртывание системы

Развёртывание системы может осуществляться двумя способами: с помощью kubernetes и helm charts на сервисе хостинга Okteto [6] или на отдельном компьютере, играющем роль сервера. Система запускается с помощью Docker compose, после чего точка входа системы становится доступной внешней сети с помощью сервиса Ngrok [7].

## 3.4 Тестирование

Интеграционное тестирование осуществлялось с помощью Postman [8], результаты которого представлен на рис. 8. Также язык Go предоставляет встроенный инструментарий для написания Unit-тестов.

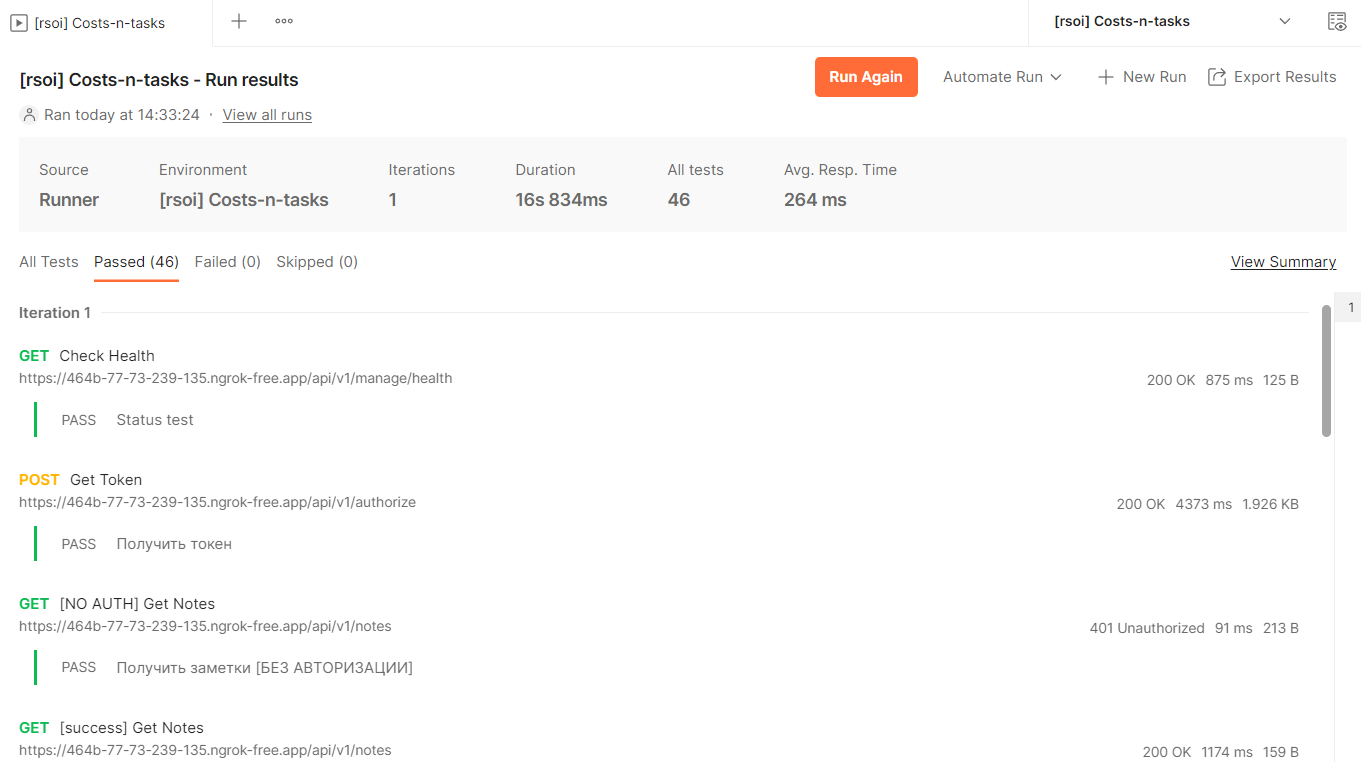


Рисунок 8 — Результаты интеграционного тестирования

## 3.5 Программный интерфейс системы (API)

Программный интерфейс был разработан по методологии REST API. Его спецификация в формате Swagger представлена на рис. 9-11, соответственно, каждый доступный вызов имеет описание и, в зависимости от типа вызова, параметры, тело и возвращаемые ответы и коды. В качестве средства проверки доступа в интерфейсе Swagger используется JWT-токен [9], получаемый в результате авторизации.

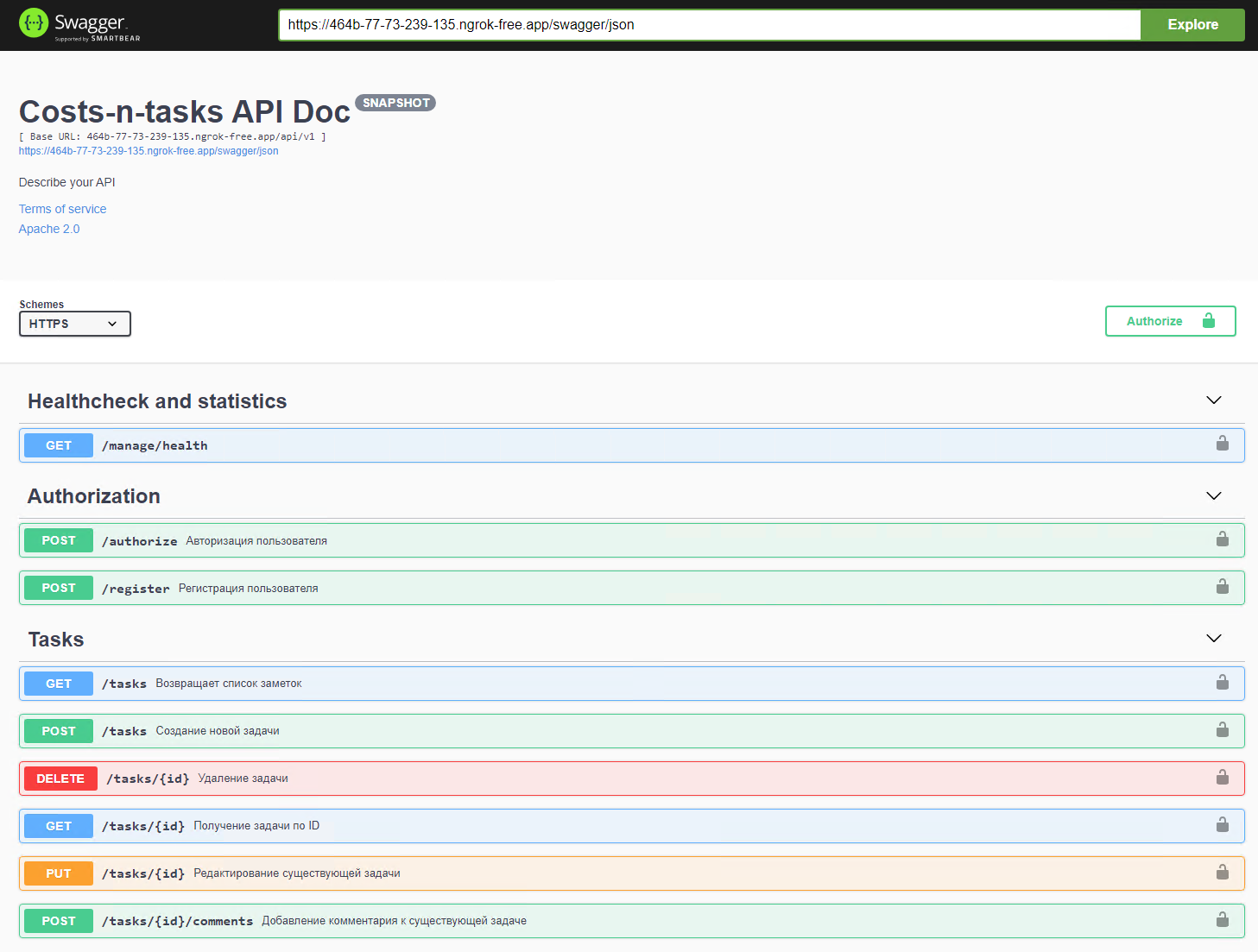


Рисунок 9 — Проверка состояния сервера, авторизация и задачи

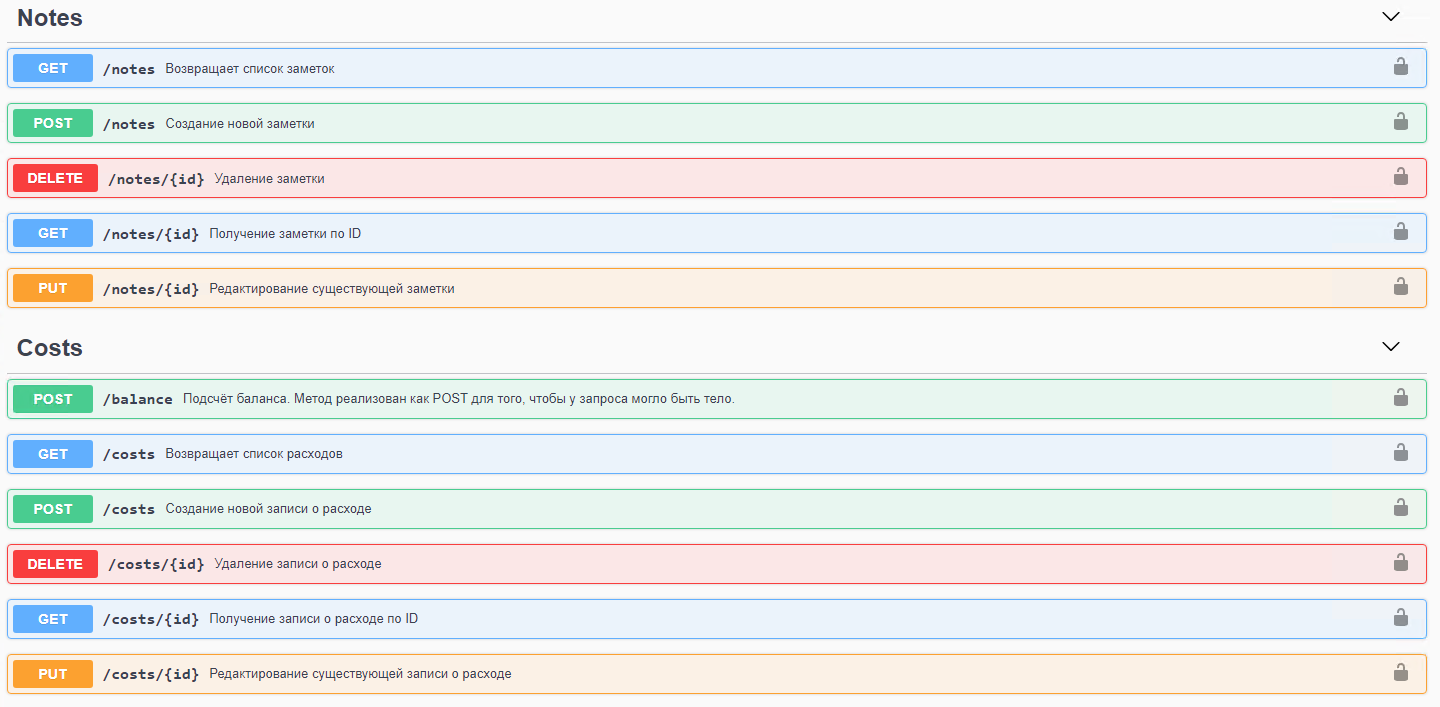


Рисунок 10 — Заметки и записи о расходах



Рисунок 11 — Записи о доходах, подсчёт баланса

# Заключение

В результате проделанной работы была достигнута поставленная цель: была разработана система, позволяющая вести заметки, задачи и записи о расходах и доходах с разделением по группам.

Были решены все поставленные задачи:

а) формализовано задание и определён необходимый функционал;

б) проведён анализ существующих решений;

в) описана архитектуру системы;

г) описана структура базы данных;

д) реализован прототип системы «Бюджет и задачи».

Существуют перспективы развития работы:

а) добавление продвинутой фильтрации и поиска;

б) добавление возможности прикрепления к записям изображений и других файлов;

в) расширение доступных возможностей по вычислениям баланса, средних расходов и т.д.;

г) кэширование данных для повышения эффективности работы.

# Список использованных источников

1. Главная страница // Официальный сайт языка Go URL: https://go.dev/ (дата обращения: 10.06.2023).
2. Главная страница // Официальный сайт Typescript URL: https://www.typescriptlang.org/ (дата обращения: 10.06.2023).
3. Главная страница // Официальный сайт PostgreSQL URL: https://www.postgresql.org/ (дата обращения: 10.06.2023).
4. Главная страница // Официальный сайт Kafka URL: https://kafka.apache.org/ (дата обращения: 10.06.2023).
5. Главная страница // Официальный сайт Okteto URL: https://www.okteto.com/ (дата обращения: 10.06.2023).
6. Главная страница // Официальный сайт Ngrok URL: https://ngrok.com/ (дата обращения: 10.06.2023).
7. Главная страница // Официальный сайт Postman URL: https://www.postman.com/ (дата обращения: 10.06.2023).
8. Введение в jwt-токены // Официальный сайт jwt URL: https://jwt.io/introduction (дата обращения: 10.06.2023).

# Приложение А

**docker-compose.yml**

Листинг A1. Список зависимостей

|  |
| --- |
| 1. version: "3" 2. volumes: 3. db-data: 5. networks: 6. default: 7. driver: bridge 8. ipam: 9. driver: default 10. config: 11. - subnet: 172.28.0.0/16 12. services: 13. postgres: 14. image: library/postgres:15 15. container\_name: postgres 16. restart: always 17. environment: 18. POSTGRES\_USER: postgres 19. POSTGRES\_PASSWORD: "postgres" 20. POSTGRES\_DB: postgres 21. volumes: 22. - db-data:/var/lib/postgresql/data 23. - ./postgres/:/docker-entrypoint-initdb.d/ 24. ports: 25. - "5432:5432" 26. users-service: 27. build: 28. context: src/users 29. dockerfile: Dockerfile 30. image: drstarland/testus-users:1.0 31. container\_name: testus-users 32. environment: 33. PORT: "8040" 34. ports: 35. - "8040:8040" 36. restart: always 37. notes-service: 38. build: 39. context: src/notes 40. dockerfile: Dockerfile 41. image: drstarland/testus-notes:1.0 42. container\_name: testus-notes 43. environment: 44. PORT: "8050" 45. ports: 46. - "8050:8050" 47. depends\_on: 48. - users-service 49. restart: always 50. tasks-service: 51. build: 52. context: src/tasks 53. dockerfile: Dockerfile 54. image: drstarland/testus-tasks:1.0 55. container\_name: testus-tasks 56. environment: 57. PORT: "8060" 58. ports: 59. - "8060:8060" 60. restart: always 61. costs-service: 62. build: 63. context: src/costs 64. dockerfile: Dockerfile 65. image: drstarland/testus-costs:1.0 66. container\_name: testus-costs 67. environment: 68. PORT: "8070" 69. ports: 70. - "8070:8070" 71. statistics-service: 72. build: 73. context: src/statistics 74. dockerfile: Dockerfile 75. image: drstarland/testus-statistics:1.0 76. container\_name: testus-statistics 77. environment: 78. PORT: "8030" 79. ports: 80. - "8030:8030" 81. depends\_on: 82. - kafka-service 83. - postgres 84. links: 85. - postgres 86. zookeeper-service: 87. image: confluentinc/cp-zookeeper:7.3.2 88. container\_name: zookeeper 89. ports: 90. - '31000:31000' 91. environment: 92. ZOOKEEPER\_CLIENT\_PORT: 2181 93. ZOOKEEPER\_TICK\_TIME: 2000 94. KAFKA\_JMX\_HOSTNAME: "localhost" 95. KAFKA\_JMX\_PORT: 31000 96. kafka-service: 97. image: confluentinc/cp-kafka:7.3.2 98. container\_name: testus-kafka 99. ports: 100. - '9092:9092' 101. - '31001:31001' 102. depends\_on: 103. - zookeeper-service 104. environment: 105. KAFKA\_BROKER\_ID: 1 106. KAFKA\_ZOOKEEPER\_CONNECT: zookeeper:2181 107. KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT\_HOST:PLAINTEXT 108. KAFKA\_INTER\_BROKER\_LISTENER\_NAME: PLAINTEXT 109. KAFKA\_ADVERTISED\_LISTENERS: PLAINTEXT://testus-kafka:29092,PLAINTEXT\_HOST://localhost:9092 110. KAFKA\_AUTO\_CREATE\_TOPICS\_ENABLE: "true" 111. KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR: 1 112. KAFKA\_GROUP\_INITIAL\_REBALANCE\_DELAY\_MS: 100 113. KAFKA\_JMX\_HOSTNAME: "localhost" 114. KAFKA\_JMX\_PORT: 31001 115. gateway-service: 116. build: 117. context: src/gateway 118. dockerfile: Dockerfile 119. image: drstarland/testus-gateway:1.0 120. container\_name: testus-gateway 121. environment: 122. PORT: "8080" 123. ports: 124. - "8080:8080" 125. depends\_on: 126. - costs-service 127. - notes-service 128. - tasks-service 129. - users-service 130. - kafka-service |