|  |  |
| --- | --- |
|  | **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 ИУ5 «Системы обработки информации и управления»**

Дисциплина «Разработка интернет-приложений»

Расчетно-пояснительная записка

Тема: «Система удаленного вычисления булевых операций»

Студент: Даниелян А.А.

Группа ИУ5-54Б

Преподаватель: Канев А.И.

2023г.

# **ВВЕДЕНИЕ**

В последнее десятилетие мы видим активное развитие сферы облачных вычислений. Так за 2021 год глобальный рынок облачных вычислений достигал уже порядка 305 млрд долларов, что оказалось на 40 млрд долларов больше, чем в 2020 году и на 90 млрд долларов больше, чем в 2019 году. Эксперты предполагают, что рынок облачных вычислений будет устойчиво развиваться с приростом порядка 17% в год вплоть до 2025 года. Поэтому возникает необходимость обеспечения безопасности и конфиденциальности данных, передаваемых и обрабатываемых в облаке, а также удобный инструмент для выполнения вычислений.

Целью работы является реализация системы для удаленного вычисления булевых операций, включающую в себя веб-сервис, веб-приложение, мобильное приложение и выделенный сервис для подсчета результатов вычислений.

Система предназначена для инициаторов вычислений и модераторов запросов на вычисления. Предусмотрен доступ к списку булевых операций с подробной информацией. Инициаторы вычислений могут создавать заявку на вычисления, выбирая булевы операции и указывая значения операндов. Система предоставляет автоматизированный способ создания и выполнения вычислений, обеспечивая эффективное управление созданием и обработкой заявок на вычисления. Модераторы имеют возможность подтверждать или отклонять заявки на вычисление, а также редактировать существующие операции или создавать новые булевы операции.

Нефункциональные требования к разрабатываемой системе:

1. Должна поддерживаться кроссплатформенность.
2. Интерфейс системы и текст ошибок должны быть русифицированы.

В ходе работы необходимо выполнить следующие задачи:

* 1. Разработать дизайн приложения.
  2. Создать базу данных в PostgreSQL.
  3. Создать веб-сервис на языке Python с использованием Django Rest Framework .
  4. Реализовать интерфейс гостя на технологии React.
  5. Развернуть веб-приложение React на Github Pages.
  6. Добавить авторизацию и аутентификацию в веб-сервис.
  7. Реализовать интерфейс инициатора вычислений в React.
  8. Реализовать интерфейс модератора React.
  9. Создать мобильное приложение на React-Native.
  10. Создать асинхронный сервис для выполнения расчетов на языке GO.
  11. Подготовить набор документации, включающий РПЗ, ТЗ и набор диаграмм.

# **БИЗНЕС-ПРОЦЕСС**

Облачные вычисления[1] использует бизнес самых различных направлений: IT, медицина, финансы, продажи, реклама. В каждой из этих сфер может возникнуть необходимость принятия решений или составления рекомендаций. И то и другое выполняется на основе некоторых логических условий и требует вычисления булевых операций [2]. Заявки на вычисления булевых операций могут составляться младшим персоналом, например, младшими аналитиками. Каждая заявка на вычисления вначале создается как черновик, в который можно добавлять и из которого можно удалять операции, также в черновике можно прописать аргументы, над которыми необходимо выполнить операцию. В одной заявке на вычисления можно указать множество операций с различными аргументами, в таком случае будет посчитан результат для каждой комбинации операндов и операций.

Когда инициатор вычислений уверится, что выбраны все необходимые операции и указаны все параметры, он формирует заявку на вычисление, которая отправляется на рассмотрение модераторам. После этого заявку редактировать нельзя. У инициатора вычислений есть также возможность просмотра истории своих заявок. Крайне важно указывать разумные аргументы и составлять заявку таким образом, чтобы вычисления имели смысл. Для того, чтобы уберечь вычислительные ресурсы от чрезмерной и бесполезной нагрузки, модераторы в лице руководителей команд аналитиков отклоняют или подтверждают заявки так, чтобы на вычисление отправлялись только корректные заявки, имеющие смысл.

Булева операция — это одна из множества определенных на данный момент булевых операций. В сервисе предоставляется краткое описание каждой операции, название булевой операции, для визуальной ассоциации каждая операция имеет изображение со своим условным графическим изображением. В информации об операции могут содержаться ошибки, в этом случае модератор может удалить операцию или же отредактировать ее таким образом, чтобы все данные были корректными и актуальными. Также модератор может добавить новую операцию в список доступных операций. Функции пользователей с разными ролями описаны на диаграмме прецедентов (рис.1).

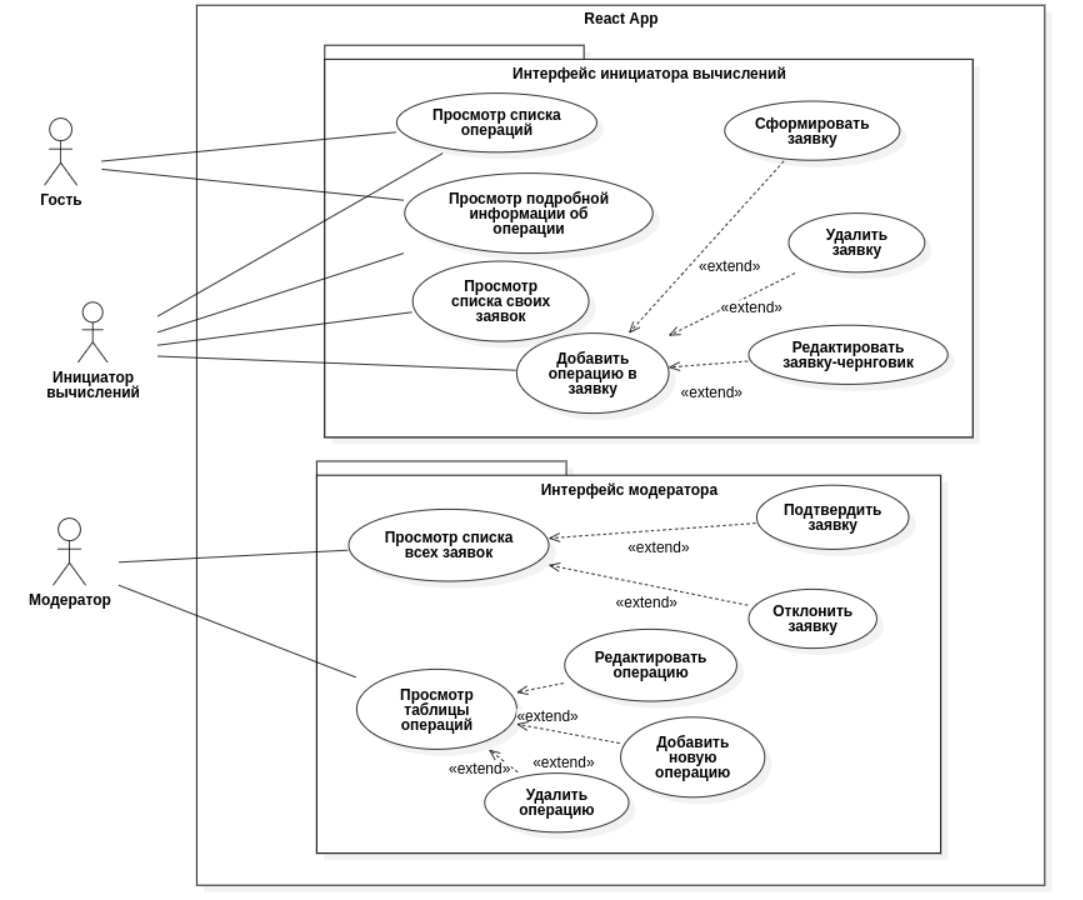


Рисунок 1 - Диаграмма прецедентов

Гостям доступен просмотр операций. Зарегистрированные гости считаются пользователями сервиса, то есть инициаторами вычислений. Они могут добавлять операции в заявку на вычисление, просматривать список своих заявок и формировать текущую заявку. Заявки обрабатываются модераторами. В результате обработки заявки, она оказывается либо подтверждена, либо отклонена. Также модератору доступны такие функции, как просмотр всех операций, создание, удаление и редактирование операций, а также просмотр всех операций в табличном виде. Процесс формирования и обработки заявки изображен на диаграмме деятельности (рис.2).

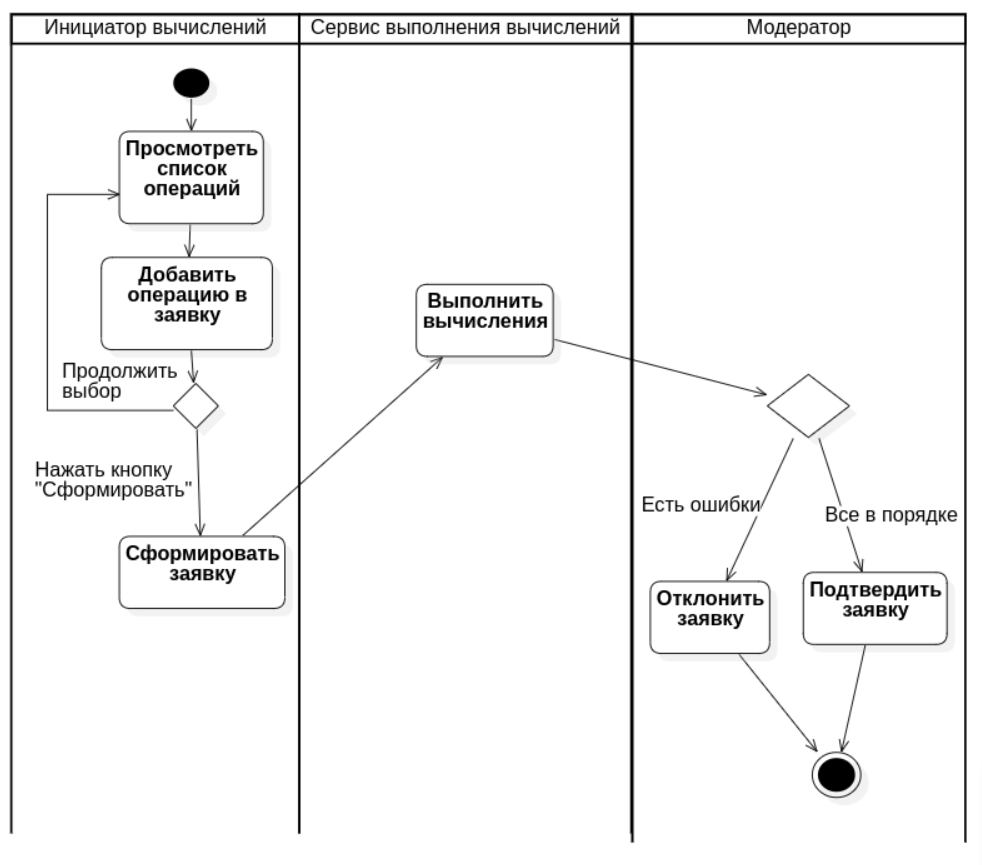


Рисунок 2 - Диаграмма деятельности

Инициатор вычислений выбирает операции, затем формирует на основе выбранных операций заявку на вычисление. Эта заявка обрабатывается асинхронным сервисом вычисления значений, а затем модератором. Если заявка одобрена модератором, то вычисляется ее значение в асинхронном сервисе. Состояния, в которых может находиться заявка отображены на диаграмме состояний (рис.3).

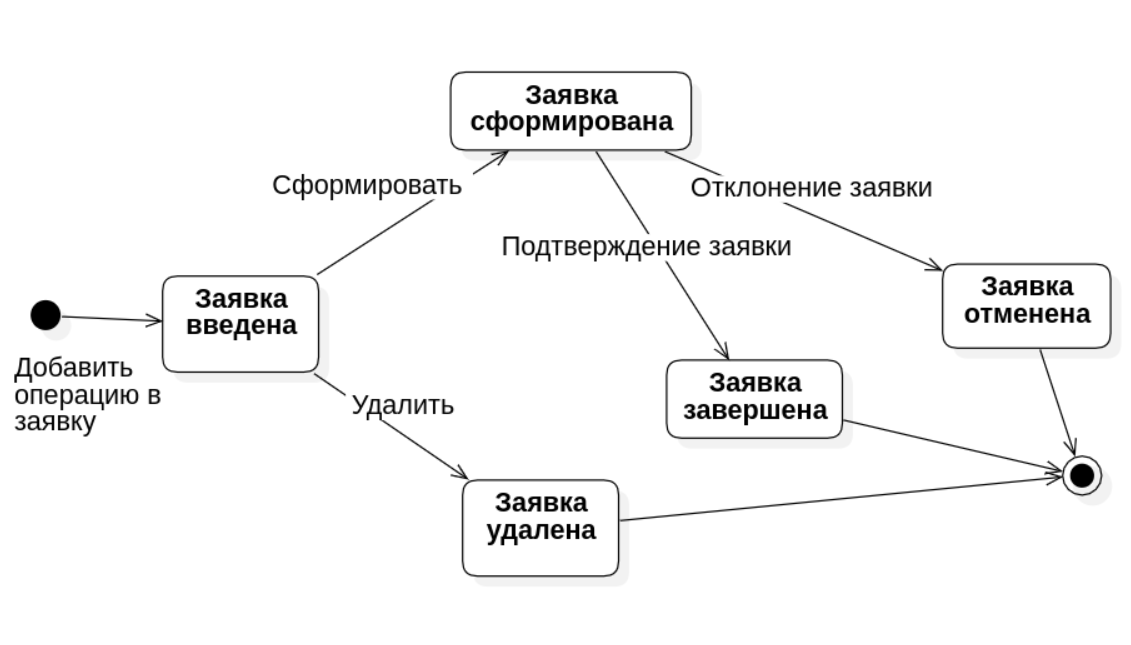


Рисунок 3 - Диаграмма состояний заявки

При выборе первой операции создается черновик заявки. Далее каждая выбранная операция попадает в этот черновик. Инициатор вычислений после этого формирует заявку, удаляет ее или выходит из приложения. Сформированная заявка после может быть одобрена или отклонена модератором.

# **АРХИТЕКТУРА**

Архитектура системы отображена на диаграмме развертывания (рис. 4). Она предполагает использование нескольких технологий для обеспечения функционирования различных компонентов системы. Фронтенд реализован на React Native [3] и React [4], обеспечивая пользовательский интерфейс для нативного и браузерного приложений соответственно. React используется для построения эффективных и масштабируемых интерфейсов. Веб-сервис, обрабатывающий логику бизнес-процессов, реализован на Django [5], предоставляя API для взаимодействия с фронтендом. Django был выбран из-за своей гибкости, простоты в использовании и мощных инструментов разработки.

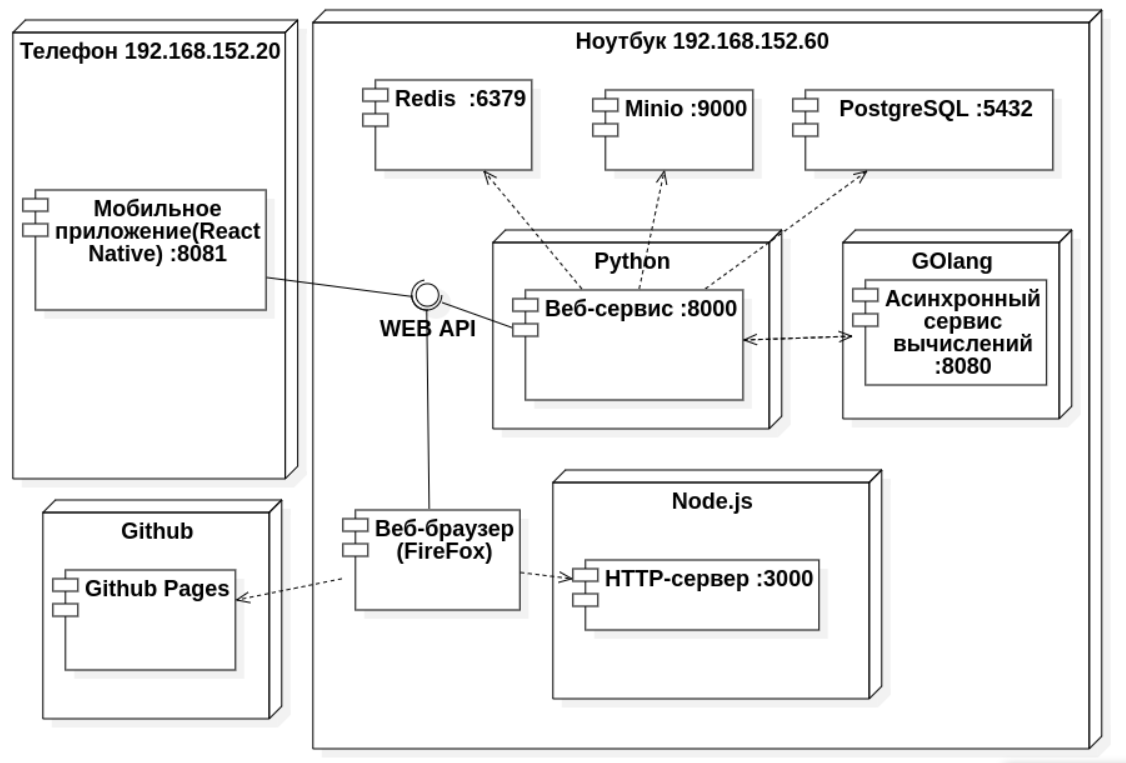


Рис. 4 - Диаграмма развертывания

Для обработки файлов и объектов используется Minio [6], облачное хранилище данных, обеспечивающее эффективное хранение и управление большими объемами информации. Для асинхронной обработки задач и управления фоновыми процессами используется асинхронный сервис на языке программирования Go [7]. Этот сервис позволяет выполнять задачи в фоновом режиме, обеспечивая более эффективную работу системы и обработку операций в реальном времени. Кроме того, для управления кэшированием и повышения производительности системы используется Redis [8], который обеспечивает быстрый доступ к данным и временное хранение информации для оптимизации процессов.

Данные хранятся в СУБД PostgreSQL [9], их структура отражена на ER диаграмме (рис. 5). СУБД PostgreSQL является одним из стандартов индустрии, поэтому было решено использовать её. Структура данных довольна проста. Помимо базовых полей, заявка также имеет поля operand1, operand2, в которых хранятся числа, над которыми выполняется булева операция, а также поле result, в котором хранятся результаты выполнения операций, которые получаются от асинхронного сервиса.

Модель «Operations» является набором полей, которые нужны для решения задач бизнес-логики. Для того, чтобы в одной заявке можно было хранить несколько операций, используется промежуточная таблица «OperationRequest», которая реализовывает связь М-М.

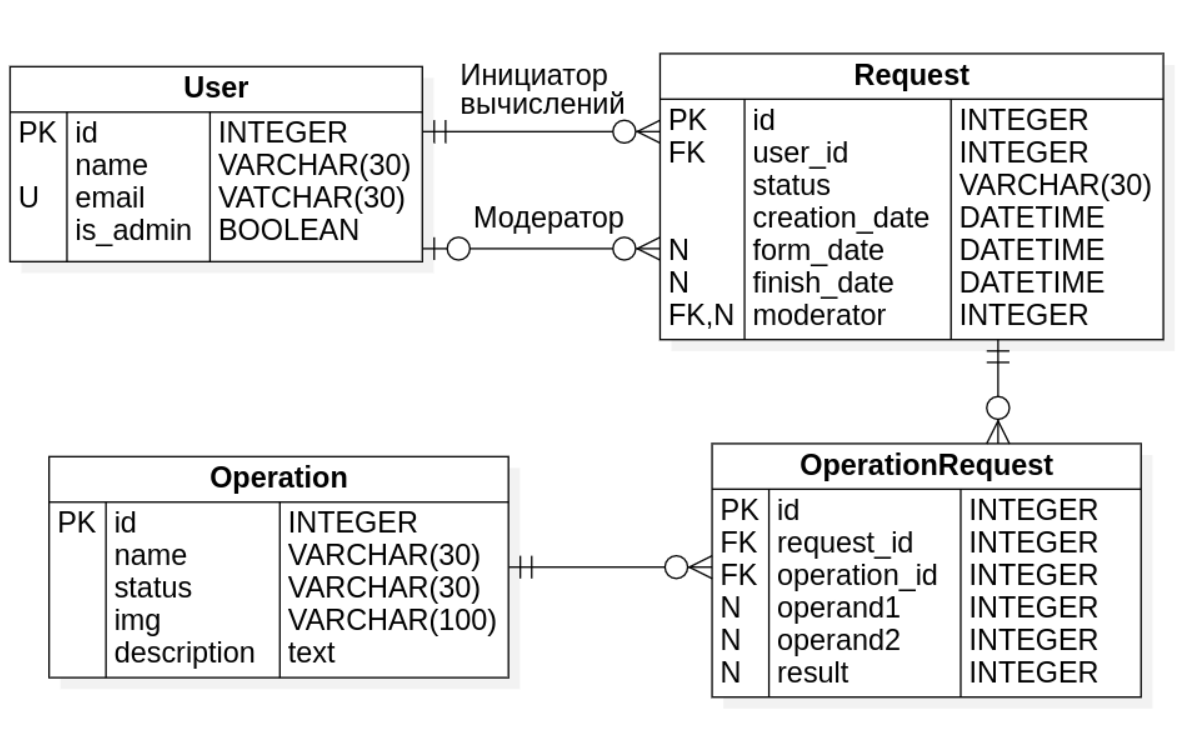


Рисунок 5 - ER диаграмма

Устройство бэкенда приложения изображено на диаграмме классов бэкенда (рис 6.). Модели имеют связи с таблицами в базе данных. Также модель заявки имеет связь с асинхронным сервисом. Операции имеют связь с сервером Minio, т.к. в операциях хранится ссылка на их изображение, хранимое на сервере статических файлов.

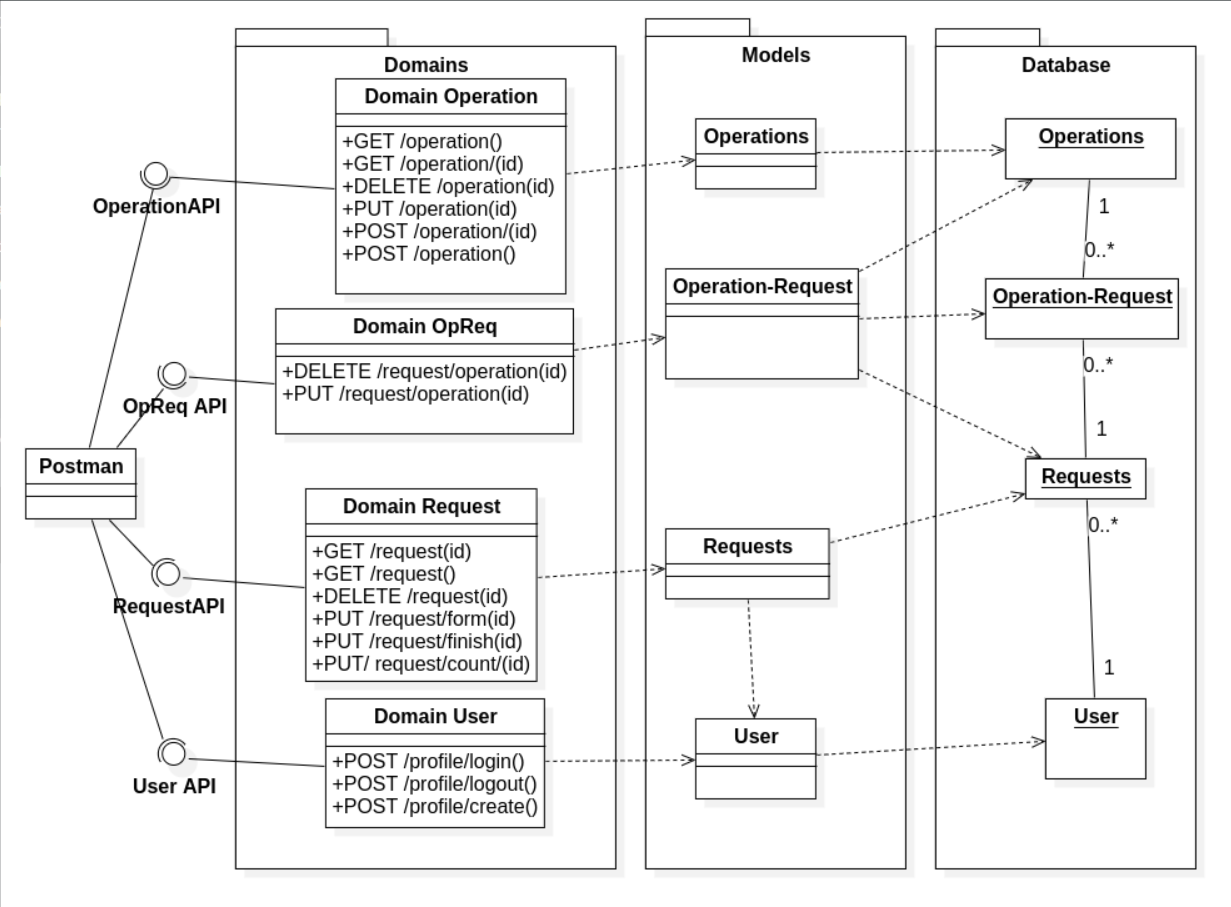


Рисунок *6* *-* Диаграмма классов бэкенда

Связь фронтенда и бэкенда отражена на диаграмме классов фронтенда (рис. 7). Ключевые страницы имеют связь с API аутентификации, т.к. доступ к ним осуществляется только для авторизированных пользователей с определенными правами (ролями).

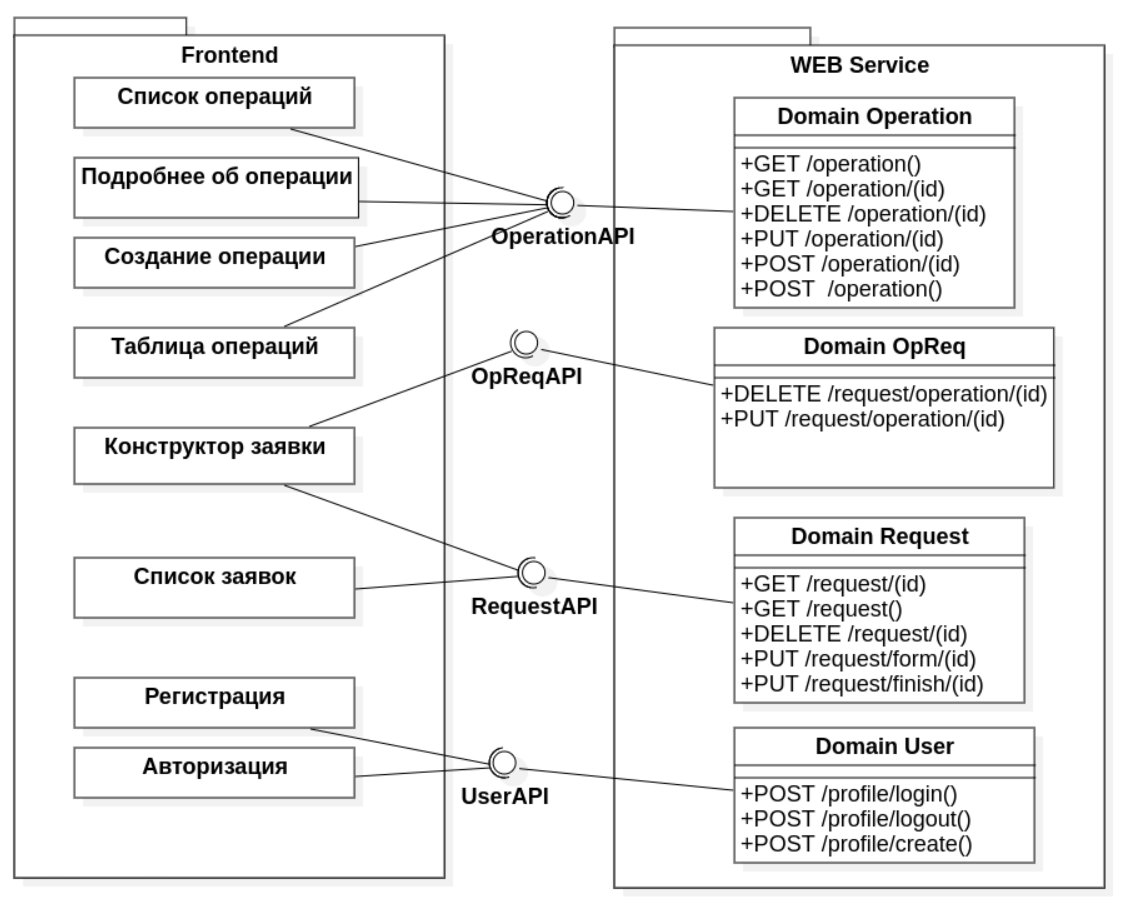


Рисунок 7 *-* Диаграмма классов фронтенда

# **АЛГОРИТМЫ**

Алгоритм работы системы отображен на диаграмме последовательности (рис. 8). В основе системы лежит веб-сервис, реализующий внутри себя всю бизнес-логику. Он предоставляет доступ к методам из следующих доменов: Operatiions, Requestss, User. Методы соответствуют правилам REST API.

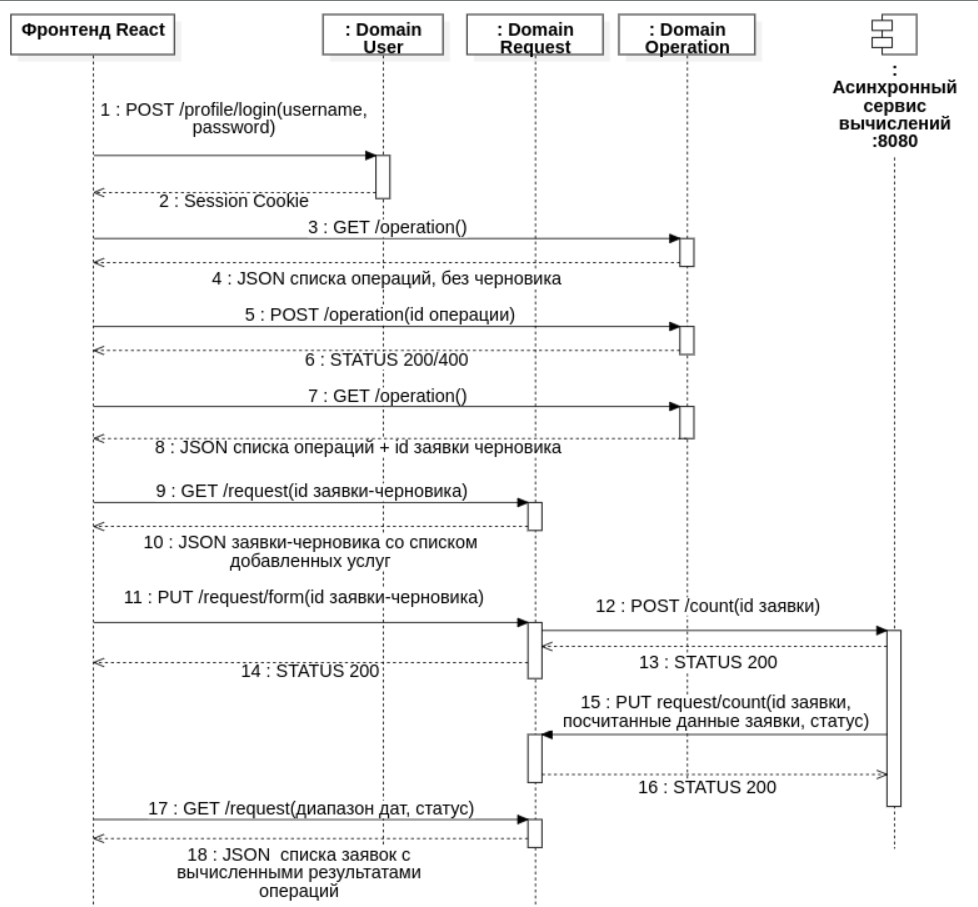


Рисунок 8 - Диаграмма последовательности

В начале бизнес-процесса происходит аутентификация пользователя. Для этого он отправляет через графический интерфейс запрос, передавая в нем логин и пароль. Если аккаунт с такими данными существует, то пользователь получает сессионную Cookie в ответном запросе. Если же такого аккаунта не существует, или пароль введен неверно, пользователь получит ошибку. В таком случае ему надо либо пройти регистрацию, либо ввести пароль верно. Затем графический интерфейс пользователя запрашивает у веб-сервиса список операций, которые возвращаются в JSON формате. Инициатор вычислений выбирает операцию, которую хочет выполнить, и, нажимая кнопку «Добавить» в графическом интерфейсе, отправляет запрос на добавление операции в заявку-черновик. Этот процесс может повторяться.

Когда инициатор вычислений определился с выбором, он нажимает на кнопку «Сформировать» в графическом интерфейсе. После этого приложение запрашивает id черновой заявки инициатора вычислений и затем отправляет запрос на формирование этой заявки. В этот момент основной веб-сервис выполняет асинхронный запрос к другому сервису, чтобы он высчитал значения операций. Через некоторое количество времени инициатор вычисления может увидеть, что статус его заявки изменился.

Процесс рассмотрения заявок происходит также через графический интерфейс. Модераторы могут просматривать списки всех заявок и, нажимая на соответствующие кнопки, отправлять запросы на подтверждение или отклонение заявок в основной веб-сервис. В эти запросы также можно включить фильтры по имени инициатора вычислений заявки, одному из статусов заявки и диапазону дат формирования. Также модераторы могут управлять операциями через графический интерфейс. Им доступны такие функции, как создание и редактирование операций, просмотр списка операций в виде таблицы и удаление их. Для каждой из этих функций присутствует свой метод, отправляемый на основной веб-сервис.

# **ОПИСАНИЕ ИНТЕРФЕЙСА**

Главное меню приложения включает пункты, которые доступны в зависимости от роли пользователя (рис. 9, 10).

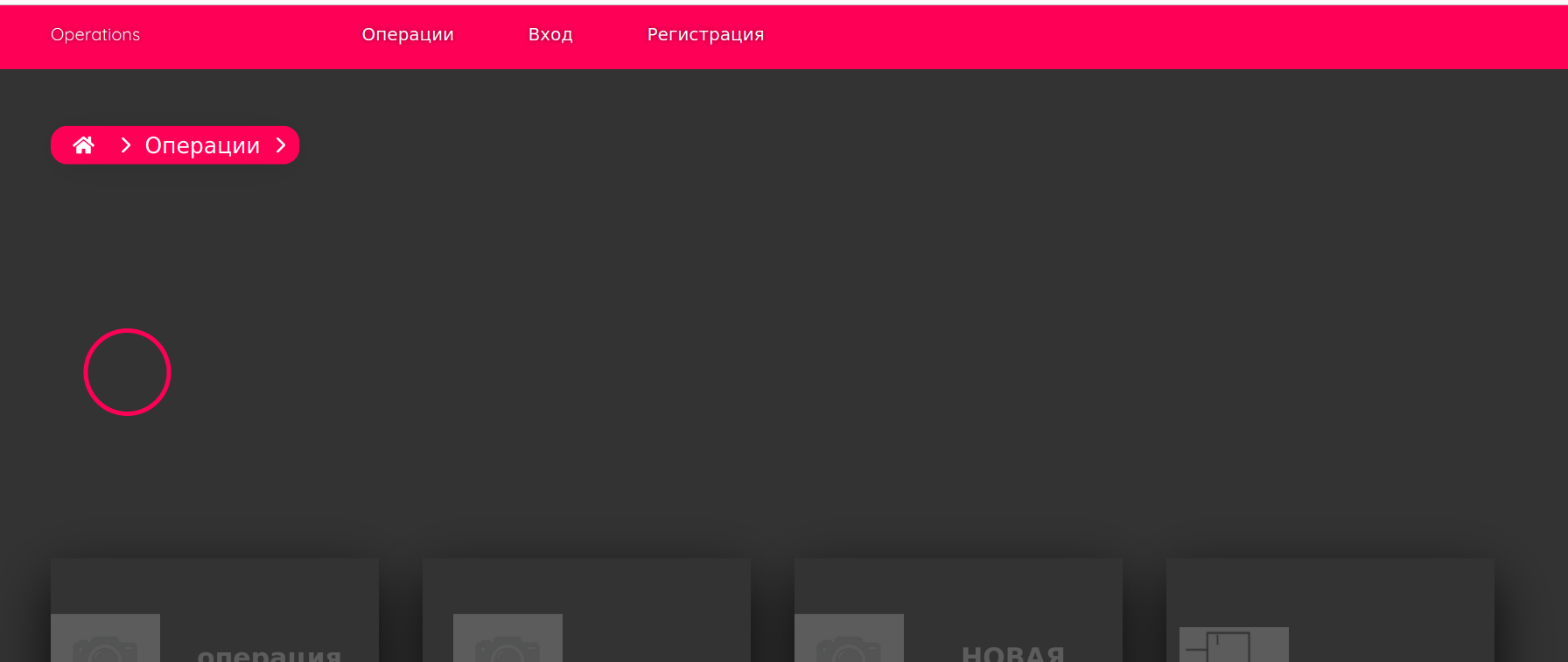
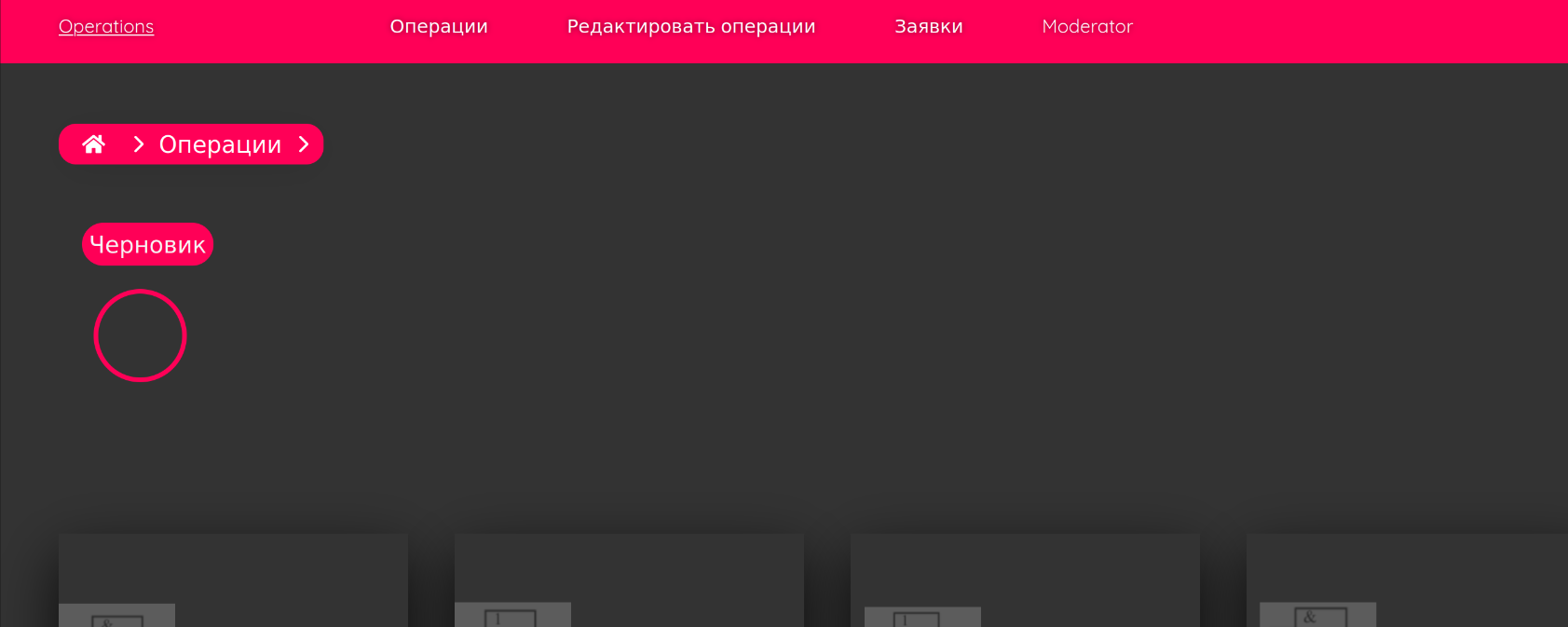
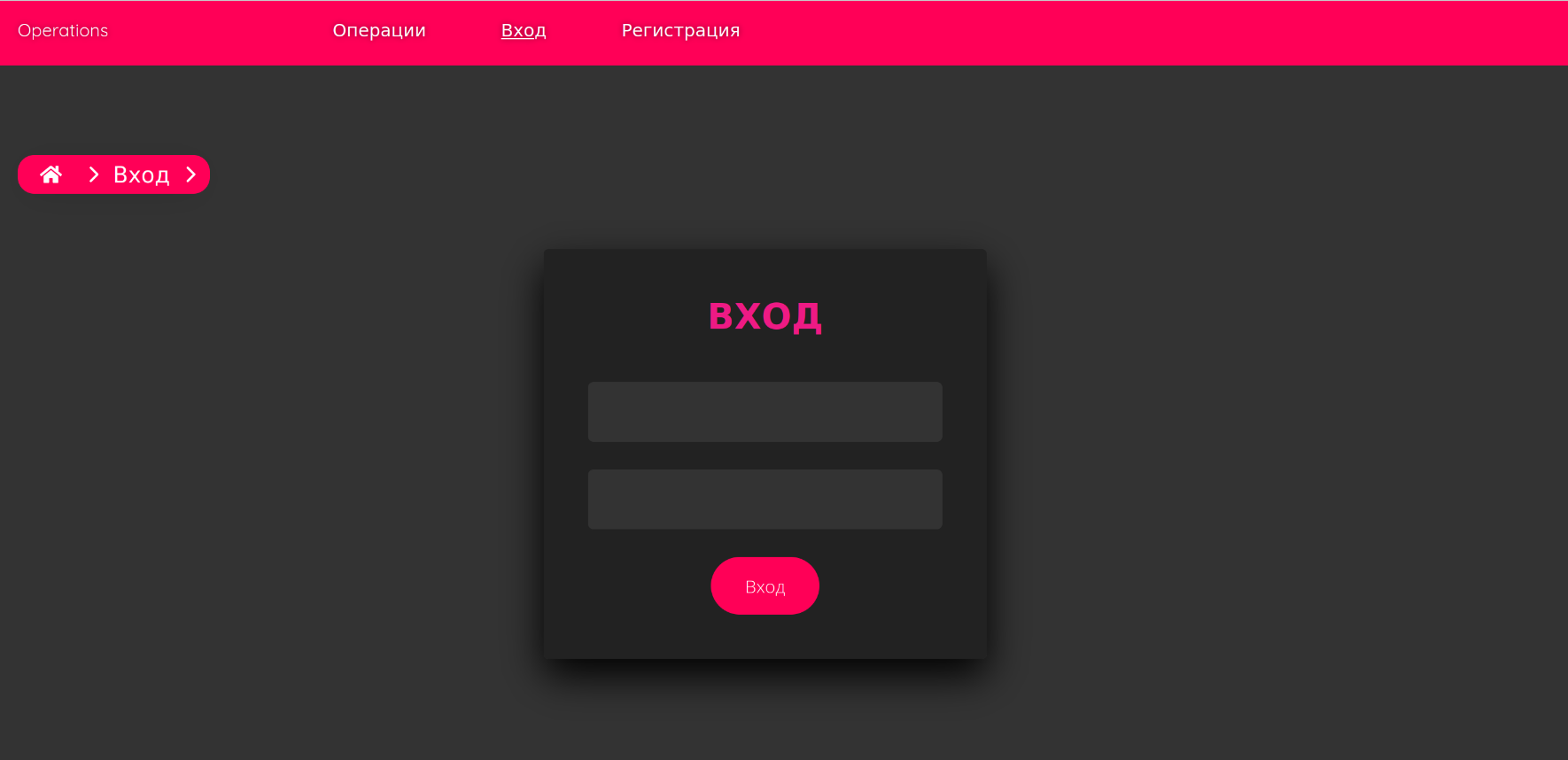


Рисунок 9 - Главное меню (неавторизированный пользователь)

Рисунок 10 - Главное меню (модератор)

Первоначальная страница для всех пользователей и гостей. В зависимости от типа пользователя её содержимое меняется. Гости, например, могут посмотреть список операций и подробную информацию о них, а также нажать на кнопку «Войти», либо «Регистрация». А у инициатора вычислений уже есть кнопки «Заявки», «Черновик» и «Выход».

На странице с формой авторизации (рис. 11) отображается форма, через которую гость входит в свой аккаунт. При успешном вводе логина и пароля гость получает сессионную cookie, которая сохраняется в localstorage и используется при отправлении запросов.

Рисунок 11 - Страница авторизации

На странице с формой регистрации (рис. 12) гости могут завести аккаунт. Для этого нужно указать логин,пароль и адрес электронной почты. Если введенный логин уже занят система попросит пользователя сменить его.

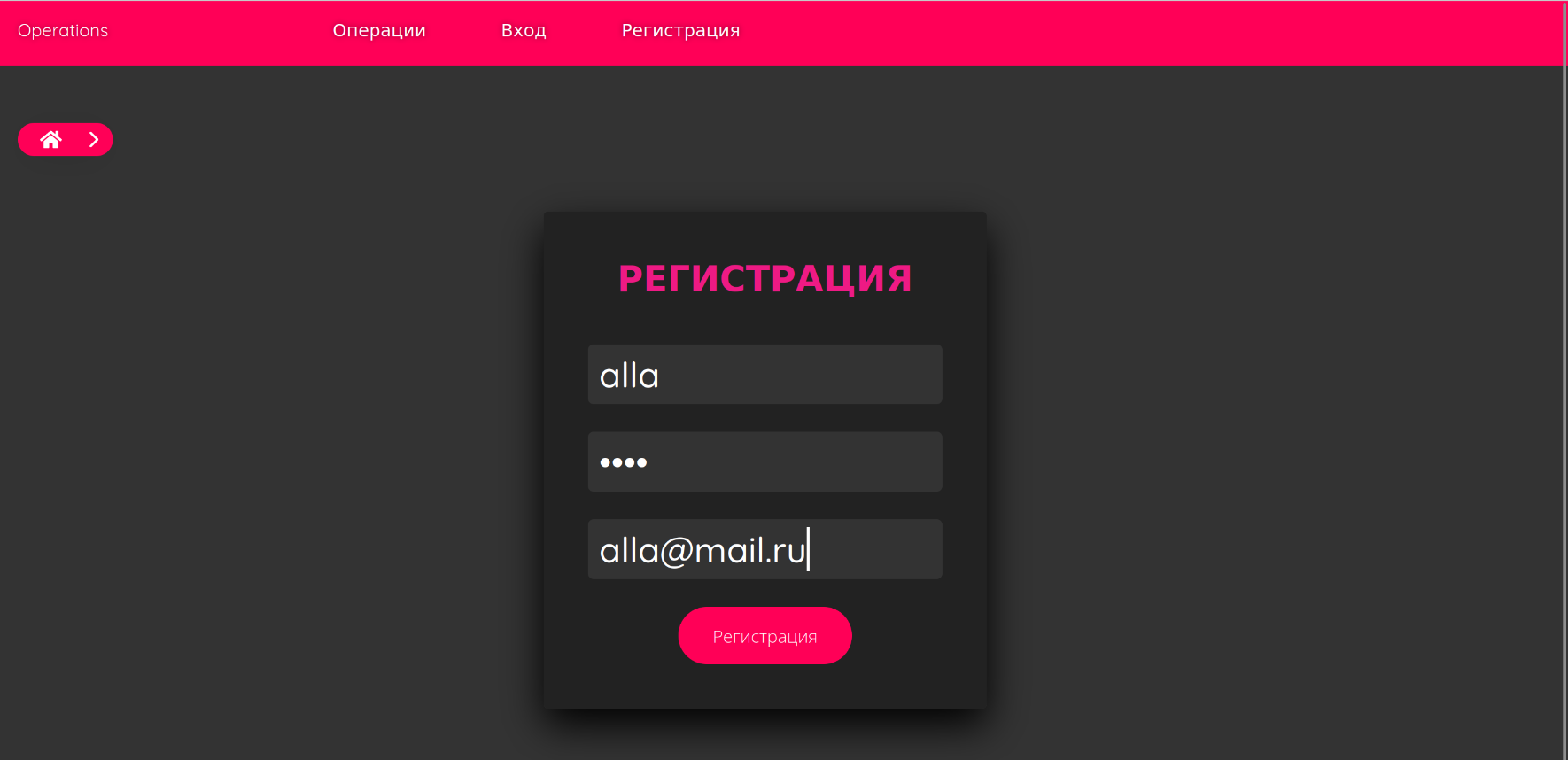
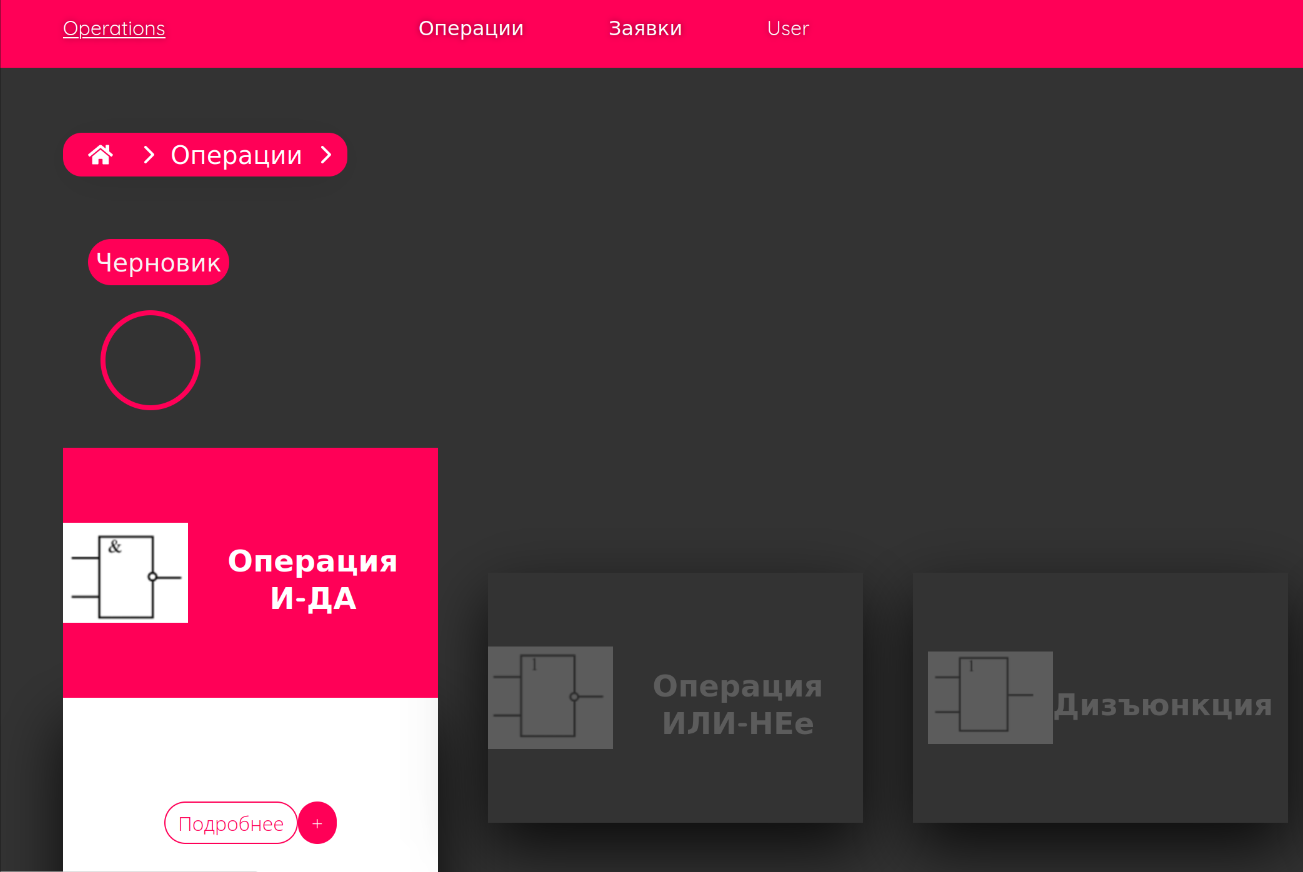
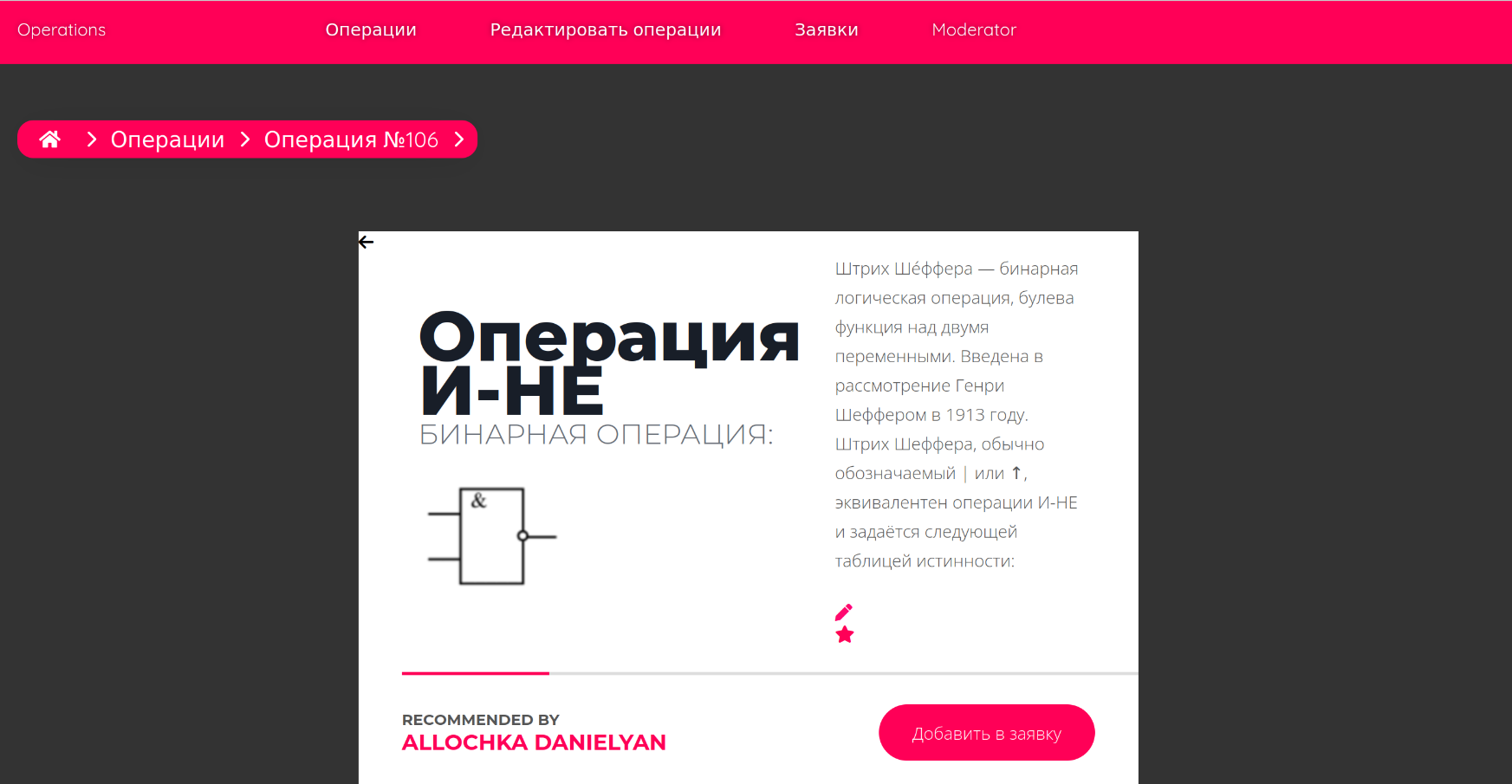


Рисунок 12 - Страница регистрации

На странице со списком операций (рис. 13) отображается список операций в виде карточек. У каждой карточки есть кнопка «Подробнее», переносящая пользователя на страницу с подробной информацией об операции, и кнопка «+» для добавления операции в черновую заявку. Сверху находится фильтр операций, который совершает поиск операции по названию.

Рисунок 13 - Страница со списком операций

На странице с подробным описанием операции (рис. 14) отображается описание операции. На этой странице также присутствует кнопка «Добавить в заявку» при нажатии которой операция добавляется в заявку черновик.

Рисунок 14 - Страница с подробным описанием операции

На странице со списком заявок (рис. 15, 16) отображается список заявок. В зависимости от типа пользователя этот список будет функционально отличаться. Так, для инициатора вычислений отображается список созданных им заявок: «Подробнее» для открытия заявки. Также можно отфильтровать заявки по статусу, либо по дате формирования заявки, выбрав либо одну дату, либо промежуток.

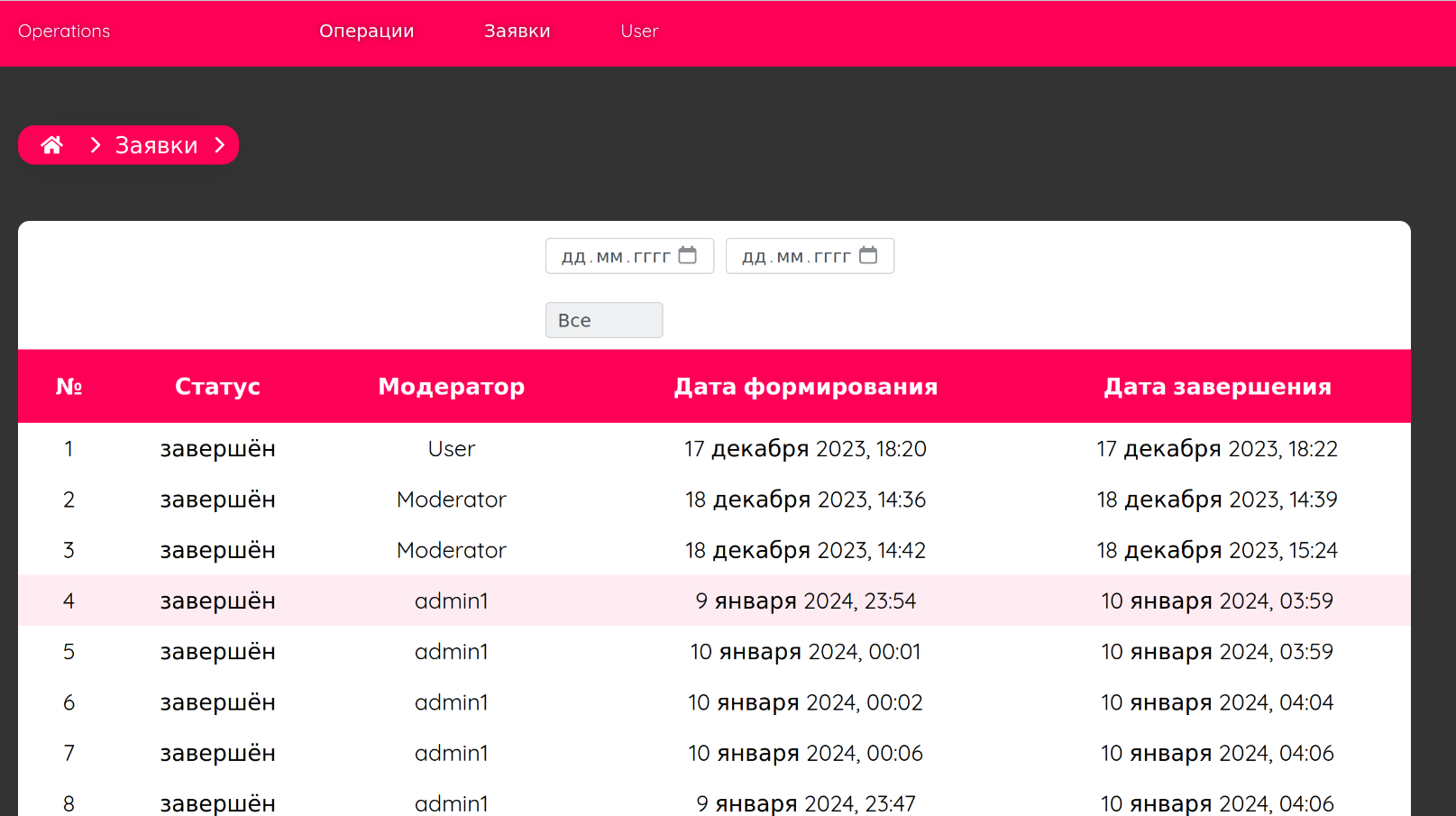


Рисунок 15 - Страница со списком заявок (инициатор вычислений)

Для модератора функция этой страницы шире. Для них отображается список всех заявок всех инициаторов вычислений, кроме всего прочего им доступны две дополнительные кнопки:«Подтвердить» и «Отклонить» для подтверждения или отмены заявки. Также присутствует фильтр по инициатору вычисления.

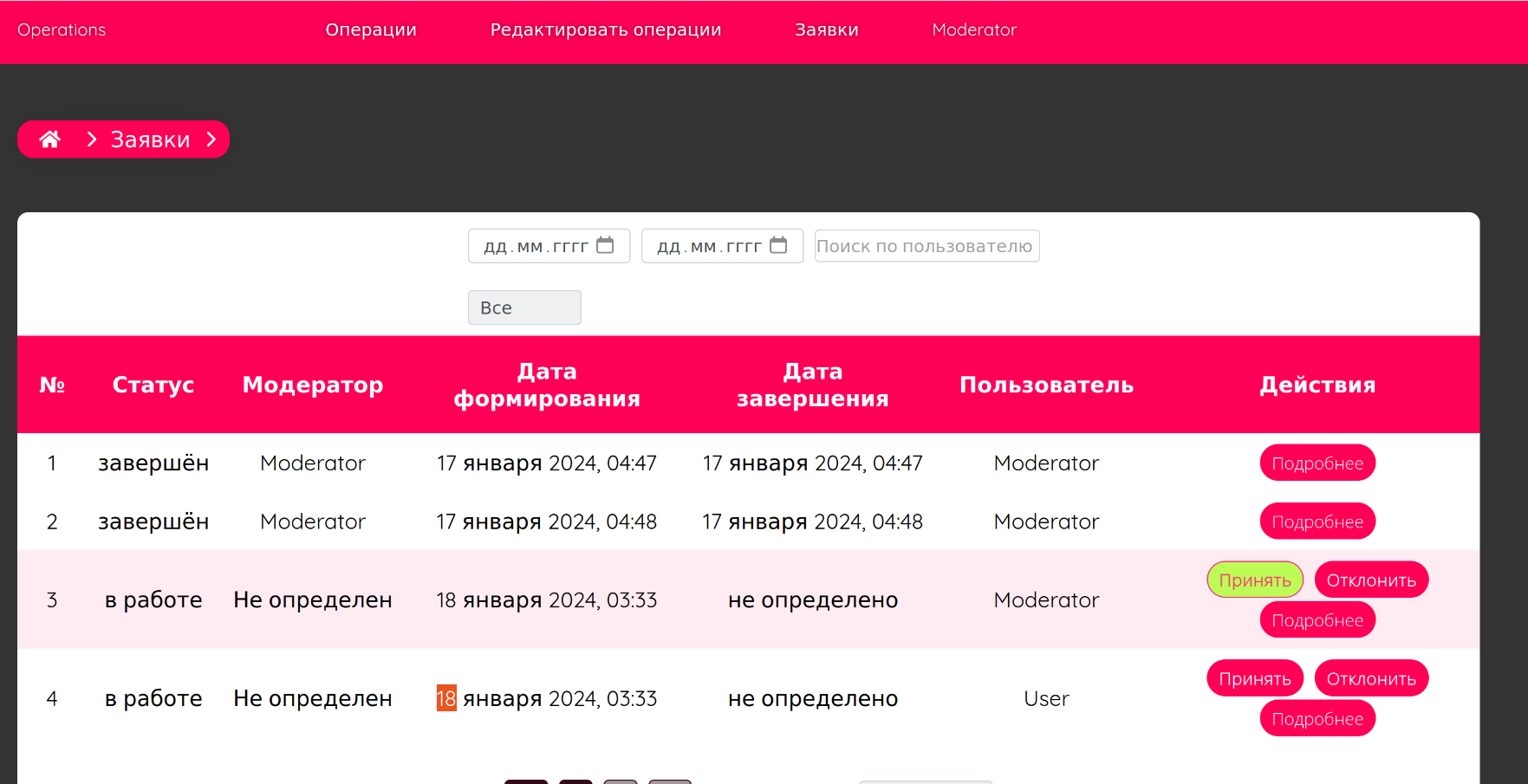


Рисунок 16 - Страница со списком заявок(модератор)

На странице с подробным описанием заявки (рис. 17) отображается подробная информация о заявке. Список выбранных операций в виде карточек, данные об операндах и результатах выполнения операций.

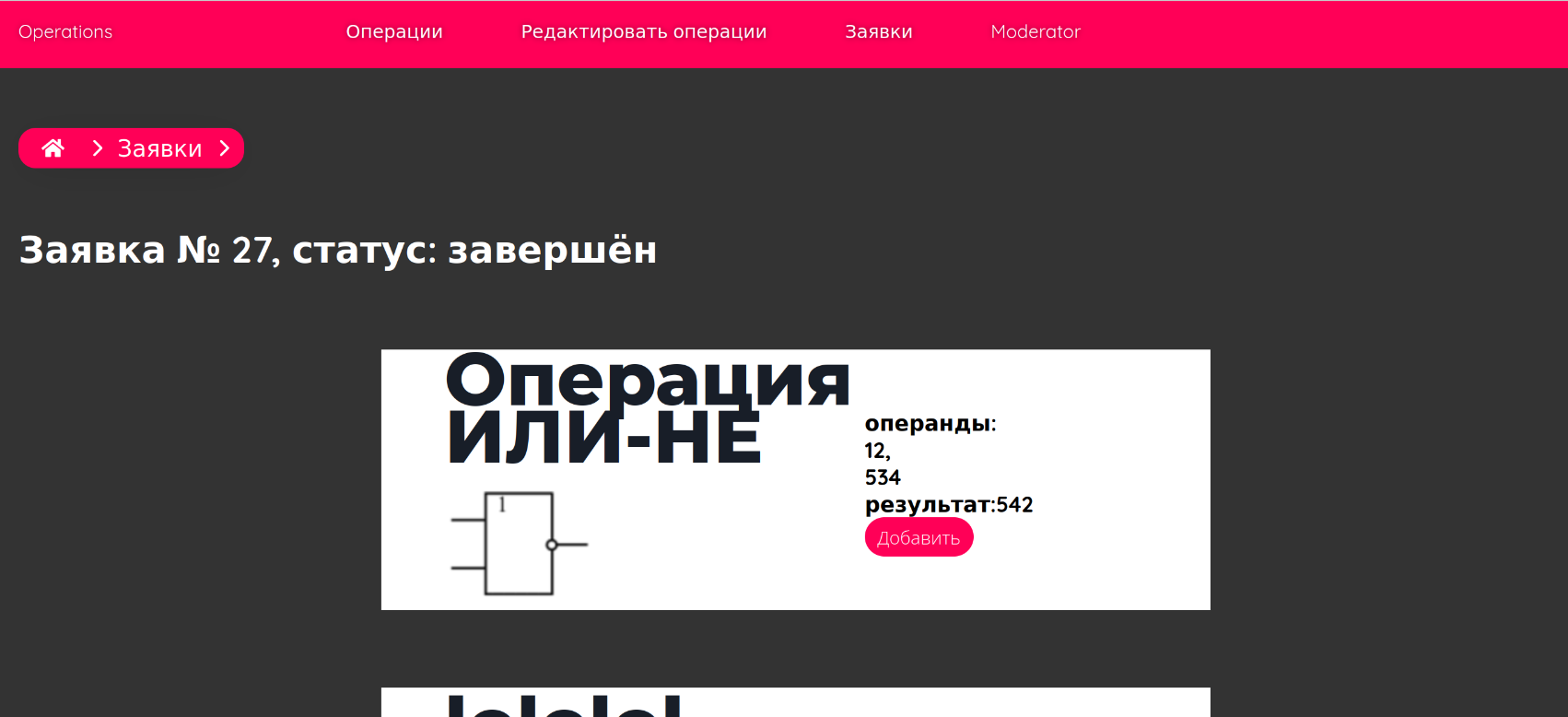


Рисунок 17 - Страница с подробным описанием заявки

На странице с таблицей операций (рис. 18) модератор может в компактном и удобном формате просмотреть список всех операций, существующих в системе. Отображаются следующие поля: изображение операции, название операции. Также можно перейти к добавлению новой операции или редактированию уже существующей (рис. 19). Присутствует фильтрация по названию операции.

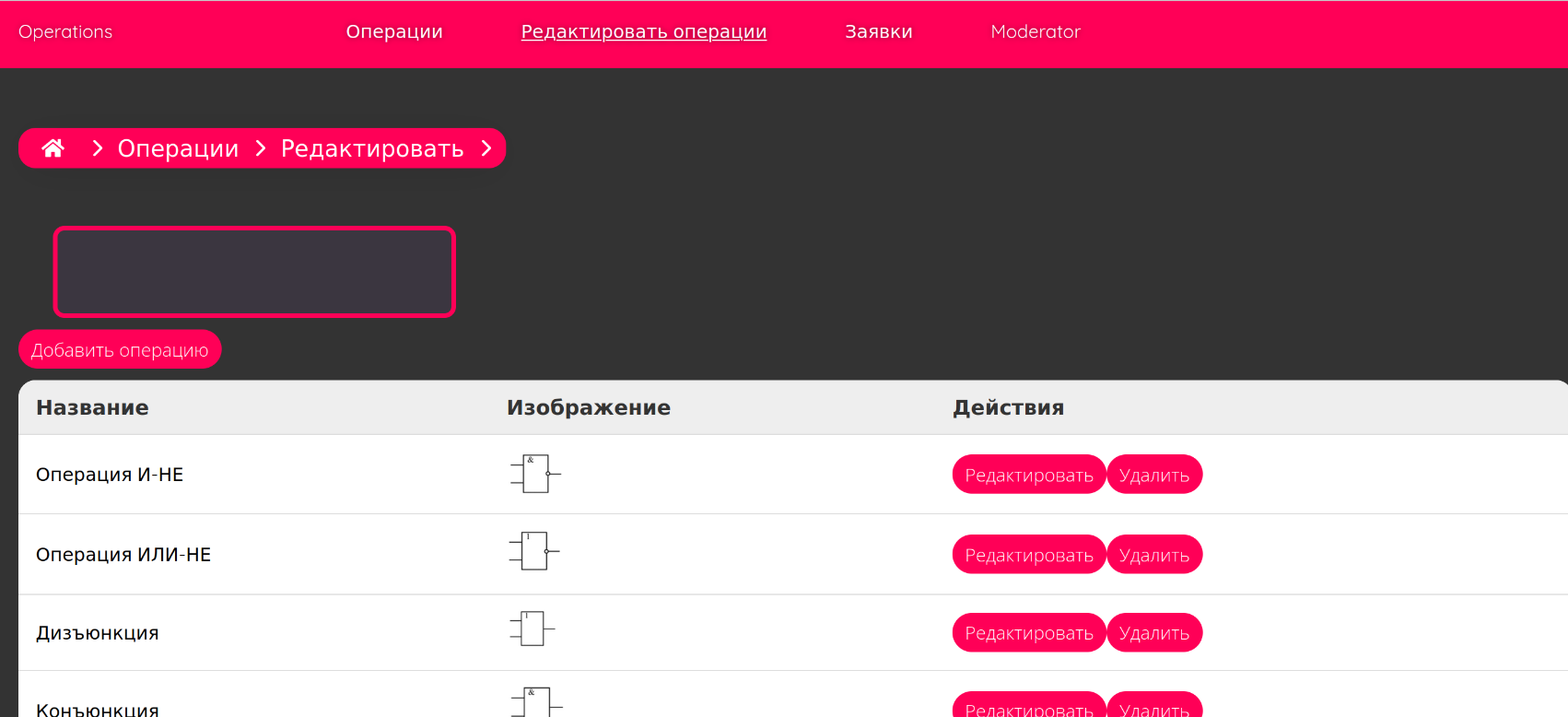


Рисунок 18 - Страница с таблицей операций

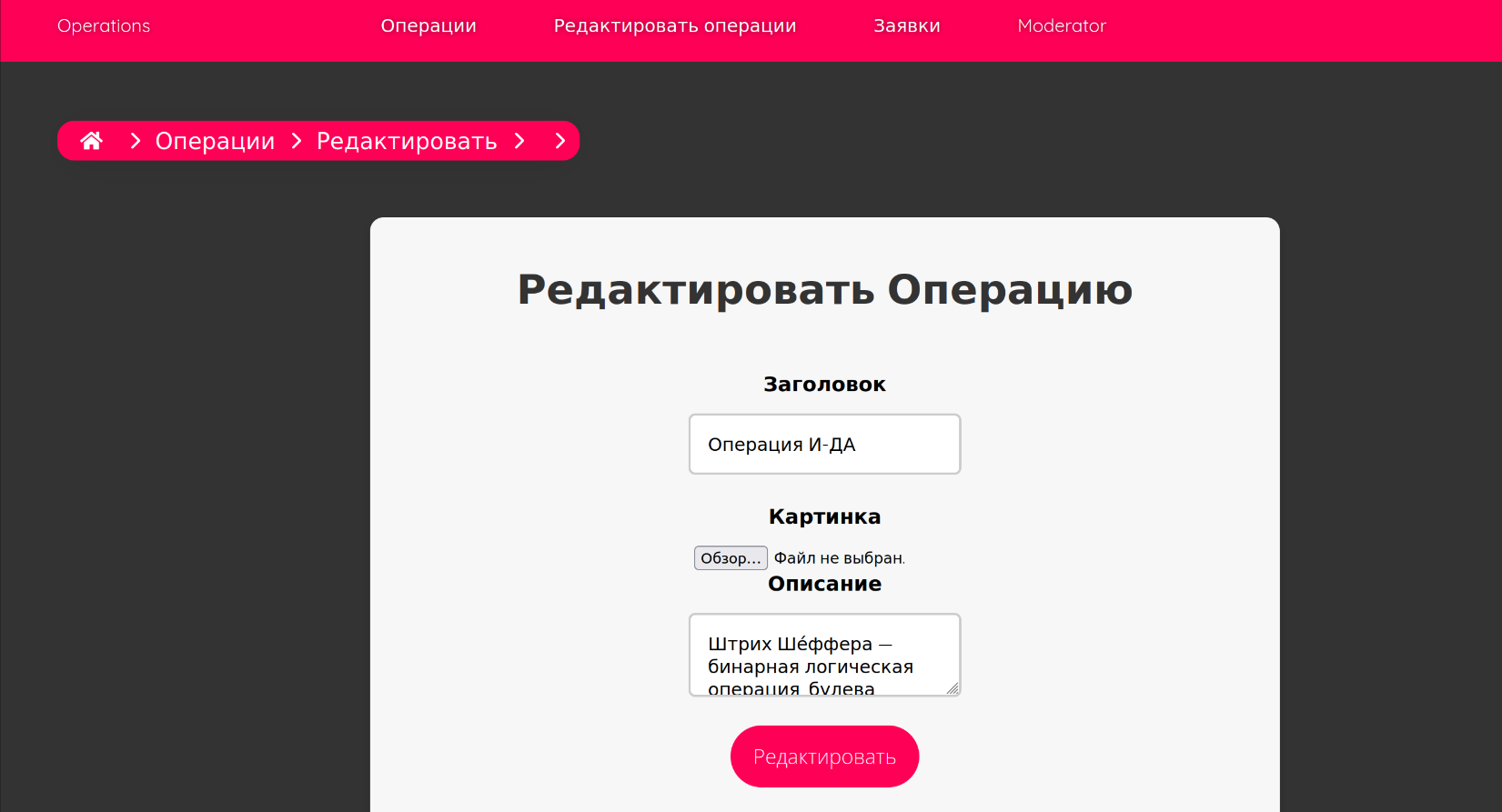


Рисунок 19 - Страница редактирования операции

На странице редактирования операции (рис. 19) модератор может отредактировать существующую операцию или создать новую. Для редактирования доступны все поля: описание, изображение, название.

# **ЗАКЛЮЧЕНИЕ**

В ходе работы были достигнуты следующие результаты:

1. Был разработан дизайн приложения с помощью набора стилей CSS и HTML тегов.
2. Была создана база данных.
3. Был создан веб-сервис на Python с применение Django Rest Framework
4. Разработан интерфейс гостя с использованием технологии React Framework и подключен к веб-сервису.
5. Приложение интерфейса было развернуто на сервисе Github Pages по ссылке https://alladan04.github.io/gp\_frontend/operation.
6. В веб-сервис добавлена авторизация через сессионные cookie, а методы задокументированы через Swagger.
7. Реализован интерфейс инициатора вычислений. Доступ к нему имеют только авторизированные пользователи.
8. Выделенный сервис был разработан с использованием языка Go.
9. Реализован интерфейс модератора для подтверждения новых заявок и редактирования/добавления операций.
10. Было реализовано мобильное приложение на React Native, повторяющее интерфейс веб-приложения на React.
11. Подготовлен набор документации, включающий РПЗ, ТЗ и набор UML диаграмм.
12. Исходный код проекта доступен в GitHub https://github.com/Alladan04/rip

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

* 1. Булева алгебра [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/%D0%91%D1%83%D0%BB%D0%B5%D0%B2%D0%B0\_%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0
  2. Облачные вычисления Cloud Computing [Электронный ресурс] // URL: <https://www.tadviser.ru/index.php/Cloud_Computing>
  3. Руководство по React Native [Электронный ресурс] // URL: <https://reactnative.dev/docs/getting-started>
  4. Руководство по React [Электронный ресурс] // URL: https://metanit.com/web/react/ (дата обращения: 11.11.2023)..
  5. Документация Django Rest Framework // URL: https://www.django-rest-framework.org/
  6. Документация Minio [Электронный ресурс] // . URL:https://min.io/docs/minio/linux/index.html
  7. Документация Go [Электронный ресурс] // URL: https://go.dev/doc/
  8. Quick Start – Redis [Электронный ресурс] // URL: https://redis.io/docs/get-started/
  9. Руководство по PostgreSQL [Электронный ресурс] // URL: https://metanit.com/sql/postgresql/

**Приложение. Техническое задание**

|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 ИУ5 «Системы обработки информации и управления»**

Дисциплина «Разработка интернет-приложений»

Техническое задание

Тема: «Уведомления электронных услуг»

Студент: Даниелян А.А.

Группа ИУ5-54Б

Преподаватель: Канев А.И.

Москва, 2023

# **Цель**

Разработка системы удаленных вычислений булевых операций, состоящей из веб-сервиса, веб-приложения, мобильного приложения и асинхронного сервиса для обеспечения эффективной работы, и координации системы.

# **Назначение**

Система предназначена для инициаторов вычислений и модераторов. Система предоставляет инициатору вычислений интерфейс для просмотра и вычисления булевых операций. Модераторы могут взаимодействовать с базой данных и выполнять управление заявками инициаторов вычислений и списком доступных операций. Главная задача системы – предоставить удобный интерфейс для выполнения вычислений на удаленных серверах.

# **Задачи:**

* 1. Разработать дизайн приложения.
  2. Создать базу данных в PostgreSQL.
  3. Создать веб-сервис на языке Python с использованием Django Rest Framework
  4. Реализовать интерфейс гостя на технологии React.
  5. Развернуть веб-приложение React на Github Pages.
  6. Добавить авторизацию и аутентификацию в веб-сервис.
  7. Реализовать интерфейс инициатора вычсилений в React.
  8. Реализовать интерфейс модератора в React.
  9. Создать мобильное приложение на React Native.
  10. Создать асинхронный сервис для отслеживания отправки уведомления на Go.
  11. Подготовить набор документации, включающий РПЗ, ТЗ и набор диаграмм.

# **Методы веб-сервиса**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | **Описание** | **Метод** | **Path** | **Вход** | **Выход** |
| 4.1 Методы операций | | | | | |
| 4.1.1 | Возвращает список операций. | GET | /operation/ | "text": string | {  "data": [  {  "name": VARCHAR,  "status":VARCHAR,  "type": VARCHAR, "description": TEXT,  "pk": INTEGER,  "img": VARCHAR,  "image": VARCHAR  },  …  ],  "request\_id":INTEGER  } |
| 4.1.2 | Добавляет новую операцию. Доступно только авторизованным модераторам. | POST | /operation/ | {  "pk":INTEGER,  "name":VARCHAR,  "description":"VARCHAR,  "status":VARCHAR,  "image":VARCHAR,  "img":VARCHAR  } | {  "data": [  {  "name": VARCHAR,  "status":VARCHAR,  "type": VARCHAR, "description": TEXT,  "pk": INTEGER,  "img": VARCHAR,  "image": VARCHAR  },  …  ],  "request\_id":INTEGER  } |
| 4.1.3 | Возвращает одну операцию. | GET | /operation/<int:id> | Отсутствует | {  "data":  {  "name": VARCHAR,  "status":VARCHAR,  "type": VARCHAR, "description": TEXT,  "pk": INTEGER,  "img": VARCHAR  },  "image": VARCHAR  }, |
| 4.1.4 | Обновляет информацию об операции. Доступно только авторизованным модераторам. | PUT | /operation/<int:pk> | {  "pk":INTEGER,  "name":VARCHAR,  "description":"VARCHAR,  "status":VARCHAR,  "image":VARCHAR,  "img":VARCHAR  } | {  "data":  {  "name": VARCHAR,  "status":VARCHAR,  "type": VARCHAR, "description": TEXT,  "pk": INTEGER,  "img": VARCHAR  },  "image": VARCHAR  }, |
| 4.1.5 | Удаляет операцию. Доступно только авторизованным модераторам. | DELETE | /operation/<int:pk> | Отсутствует | {  "data": [  {  "name": VARCHAR,  "status":VARCHAR,  "type": VARCHAR, "description": TEXT,  "pk": INTEGER,  "img": VARCHAR,  "image": VARCHAR  },  …  ],  "request\_id":INTEGER  } |
| 4.1.6 | Добавляет операцию в заявку. Доступно только авторизованным пользователям. | POST | /operation/<int:pk> | Отсутствует | {  "data": {  "request": {  "id":INTEGER,  "status": VARCHAR,  "creation\_date": DATETIME,  "form\_date": DATETIME,  "finish\_date": DATETIME,  "user": INTEGER,  "admin": INTEGER  },  "items": [  {  "id": INTEGER,  "operand1": INTEGER,  "operand2": INTEGER,  "result": INTEGER,  "operation": {  "name": VARCHAR,  "status": VARCHAR,  "type": VARCHAR,  "description": VARCHAR,  "pk": INTEGER,  "img": VARCHAR,  },  "request": INTEGER  },  …  ]  } |
| 4.2. Методы заявок | | | | | |
| 4.2.1 | Возвращает список заявок. Доступно только авторизованным пользователям | GET | /request/ | {  status\_list: «отменён» || «завершён» || «в работе»;  update: DATETIME;  doundate: DTETIME  } | {  "data": [  "request": {  "id":INTEGER,  "status": VARCHAR,  "creation\_date": DATETIME,  "form\_date": DATETIME,  "finish\_date": DATETIME,  "user": VARCHAR,  "admin": INTEGER  },  …  ]  } |
| 4.2.2 | Формирует заявку. Доступно только авторизованным пользователям. | PUT | /request/form/ | Отсутствует | {  "data": {  "id":INTEGER,  "status": VARCHAR,  "creation\_date": DATETIME,  "form\_date": DATETIME,  "finish\_date": DATETIME,  "user": VARCHAR,  "admin": INTEGER  }  } |
| 4.2.3 | Удаляет заявку черновик. Доступно только авторизованным пользователям. | DELETE | /request/ | Отсутствует | Отсутствует |
| 4.2.4 | Возвращает одну заявку. Доступно только авторизованным пользователям. | GET | /request/<int:id>/ | Отсутствует | {  "data": {  "request": {  "id":INTEGER,  "status": VARCHAR,  "creation\_date": DATETIME,  "form\_date": DATETIME,  "finish\_date": DATETIME,  "user": INTEGER,  "admin": INTEGER  },  "items": [  {  "id": INTEGER,  "operand1": INTEGER,  "operand2": INTEGER,  "result": INTEGER,  "operation": {  "name": VARCHAR,  "status": VARCHAR,  "type": VARCHAR,  "description": VARCHAR,  "pk": INTEGER,  "img": VARCHAR,  },  "request": INTEGER  },  …  ]  } |
| *4.2.5* | Принимает или отклоняет заявку. Доступно только авторизованным модераторам. | *PUT* | /request/finish/<int:id>/ | {  "status":VARCHAR,  } | {  "data": {  "id":INTEGER,  "status": VARCHAR,  "creation\_date": DATETIME,  "form\_date": DATETIME,  "finish\_date": DATETIME,  "user": INTEGER,  "admin": INTEGER  *}*  *}* |
| 4.3. Методы изменения М-М | | | | | |
| 4.3.1 | Удаление операции из заявки . Доступно только авторизованным пользователям. | DELETE | request/operation/<int:id> | Отсутствует | Отсутствует |
| 4.3.2 | Изменение значения операндов. Доступно только авторизованным пользователям | PUT | request/operation/<int:id> | {  "operand1":INTEGER,  "operand2":INTEGER  } | {  "data": {  "id": INTEGER,  "operand1": INTEGER,  "operand2": INTEGER,  "result": **INTEGER**,  "operation": INTEGER,  "request": INTEGER  }  } |
| 4.4. Методы авторизации и аутентификации | | | | | |
| 4.4.1 | Авторизация | POST | profile/login | {  "login": CHARACTER VARYING,  "password": CHARACTER VARYING,  } | Session cookie |
| 4.4.2 | Создать новый аккаунт | POST | profile/create | {  "login": CHARACTER VARYING,  "password": CHARACTER VARYING, "is\_moderator": BOOLEAN,  "is\_staff": BOOLEAN,  “email”: CHARACTER VARYING  } | {  "login": CHARACTER VARYING,  "password": CHARACTER VARYING, "is\_moderator": BOOLEAN,  "is\_staff": BOOLEAN,  “email”: CHARACTER VARYING  } |
| 4.4.3 | Выход из cистемы. Доступно только авторизованным пользователям. | POST | profile/logout | Session cookie |  |
| 4.5. Методы внешнего сервиса | | | | | |
| 4.5.1 | Метод выполнения вычислений | POST | /count | Id заявки-черновика |  |

1. **Функциональные требования:**
   1. Главное меню. Здесь находятся кнопки для регистрации и входа или информация о текущем пользователе, и меню со списком доступных пунктов.
      1. Доступна всем пользователям.
      2. Действия
         1. Войти – переход на страницу 5.2. Только для гостей.
         2. Зарегистрироваться – переход на страницу 5.3. Только для гостей.
         3. Операции – переход на страницу 5.4. Для всех пользователей.
         4. Заявки – переход на страницу 5.6. Только для авторизованных пользователей.
         5. Редактировать операции – переход на страницу 5.9. Только для модераторов.
   2. Страница с формой авторизации. На этой странице находится форма авторизации.
      1. Доступна гостям
      2. Действия
         1. Войти – производит запрос (метод 4.4.1).
   3. Страница с формой регистрации. На этой странице находится форма регистрации.
      1. Доступна гостям
      2. Действия
         1. Зарегистрироваться – производит запрос (метод 4.4.2.), в котором передаются введенные данные формы.
   4. Страница со списком операций. На этой странице располагается список всех операций и фильтр по названию.
      1. Доступна всем пользователям.
      2. Выводится информация об операциях в виде карточек (метод 4.1.1.).
         1. Изображение операции.
         2. Название операции.
      3. Действия
         1. Отфильтровать операции по названию(метод 4.1.1.).
         2. «Черновик» - перенаправляет на страницу заявки-черновика 5.7. Только для авторизованных пользователей.
         3. Добавить операцию в заявку (метод 4.1.6.). Только для авторизованных пользователей.
         4. Подробнее – перенаправляет на страницу 5.5.
   5. Страница с подробным описанием операции
      1. Доступна всем пользователям.
      2. Выводится подробная информация об операции.
         1. Изображение операции.
         2. Название операци.
         3. Описание операции.
   6. Страница со списком заявок
      1. Доступна только авторизированным пользователям.
      2. Выводится информация о заявках в табличном формате (метод 4.2.1).
         1. Пользователь. Только для модераторов.
         2. Статус.
         3. Модератор.
         4. Дата формирования.
         5. Дата завершения.
      3. Действия
         1. Открыть подробную информацию о заявке – перенаправляет на страницу 5.7.
         2. Установить заявке статус «отклонено» или «завершено» (метод 4.2.5.). Только для модераторов.
         3. Отфильтровать список заявок по диапазону дат формирования и/или статусу (метод 4.2.1) и/или пользователю, создавшему заявку (только для модератора).
   7. Страница с подробным описанием заявки. Пользователь может изменить набор операций в заявку. Модератор может просматривать заявки всех пользователей.
      1. Доступна только авторизированным пользователям.
      2. Выводится информация о заявке. Список операций, в виде карточек (метод 4.2.4.).
         1. Изображение операции.
         2. Название операции
         3. Первый операнд. Если статус отличен от статуса черновика.
         4. Второй операнд. Если статус отличен от статуса черновика.
         5. Результат. Если статус отличен от статуса черновика.
      3. Действия
         1. Удалить операцию из заявки(метод 4.3.1). Если заявка находится в статусе черновика. Только для создателя заявки.
         2. Сформировать заявку (метод 4.2.2.). Если заявка находится в статусе черновика. Только для создателя заявки.
         3. Удалить заявку (метод 4.2.3.). Если заявка находится в статусе черновика. Только для создателя заявки.
         4. Сохранить операнды (метод 4.3.2). Если заявка находится в статусе черновика. Только для создателя заявки.
         5. Просмотреть операцию – перенаправляет на страницу 5.5.
   8. Страница редактирования/создания операции. Модератор может изменить существующую или создать новую операцию.
      1. Доступна только модераторам.
      2. Выводится информация об операции.
         1. Изображение операции.
         2. Название операции.
         3. Описание операции.
      3. Изменение полей операции. Можно изменять все вышеперечисленные поля операции (методы 4.1.2. и 4.1.4.).
   9. Страница с таблицей операций. Предоставляет модератору удобный способ отображения всех операций.
      1. Доступна только модераторам.
      2. Выводит информацию об операциях в табличном виде (метод 4.1.1).
         1. Изображение операции.
         2. Название операции.
      3. Действия
         1. Удалить операцию (метод 4.1.5)
         2. Редактировать операцию – перенаправляет на страницу 5.8
         3. Добавить операцию – перенаправляет на страницу 5.8
2. **Требования к программному обеспечению:**
   1. Серверная часть
      1. ОС: Linux 22
      2. Python 3.10.12
      3. PostgreSQL 14.10
      4. Node.js 21.2.0
      5. Redis 6.0.16
      6. Minio
   2. Клиентская часть
      1. ОС: Windows/MacOS/Linux
      2. Веб-браузер: Safari 11.1+/Chrome 40+/Opera 27+/Firefox 44+/Yandex 20+
3. **Требования к аппаратному обеспечению:**
   1. Серверная часть
      1. Процессор минимум 2-ядерный с частотой от 2 ГГц.
      2. Оперативная память от 4 Гб.
      3. Место на жестком диске от 2 Гб.
   2. Клиентская часть
      1. Процессор с частотой от 1ГГц.
      2. Оперативная память от 512 Мб.