

МИНОБРНАУКИ РОССИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Курсовой проект по дисциплине «Технологии программирования»

на тему “Веб-приложения для планирования дел”

Выполнили:

А.А.Переславцева, Д.А.Теремкова, В.И.Капусткин

Воронеж 2020

Оглавление

| | | |
|-------------|--|-----------|
| 1. | Введение | 3 |
| 2. | Постановка задачи | 4 |
| 2.1. | Цель | 4 |
| 2.2. | Сфера использования | 4 |
| 2.3. | Требования | 4 |
| 2.4. | Задачи | 4 |
| 3. | Анализ предметной области | 5 |
| 3.1. | Целевая аудитория | 5 |
| 3.2. | Результаты опроса | 5 |
| 3.3. | Описание предметной области | 6 |
| 3.4. | Пользователи системы | 6 |
| 3.5. | Границы функциональности | 7 |
| 3.6. | Анализ существующих решений | 8 |
| 3.7. | Воронки конверсии | 10 |
| 3.8. | Анализ задачи | 13 |
| 3.9. | Варианты использования приложения | 13 |
| 3.10. | Взаимодействие компонентов системы | 16 |
| 3.11. | Варианты состояния системы | 22 |
| 3.12. | Варианты действия в системе | 24 |
| 3.13. | Развертывание приложения | 26 |
| 3.14. | Диаграмма классов | 27 |
| 3.15. | Диаграмма объектов | 28 |
| 3.16. | Модель базы данных | 28 |
| 4. | Проектная часть | 30 |
| 5. | Заключение | 30 |
| 6. | Список используемых материалов | 30 |

1. Введение

Задумывались ли вы когда-нибудь над тем, почему одни люди успевают все, а другие ничего? Вовсе не потому, что первые могут делать несколько дел одновременно. Весь секрет здесь в четком планировании своего времени. Тот, кто знает ему настоящую цену, взял себе за правило пользоваться ежедневниками, занося туда важные дела по мере их появления. Иные же с вечера составляют список мероприятий на завтра и стараются придерживаться этого расписания. Дел бывает так много, что всех не упомнишь.

Некоторые пользуются записными книжками или блокнотами со специальной разметкой страниц, которые в изобилии продаются в магазинах и киосках «Информпечать» — это органайзеры в традиционной бумажной форме. Другие — более продвинутые — компьютерами или сотовыми телефонами — это органайзеры в электронной форме. И все таки проблема эта приобрела в наш век огромное значение.

В данном курсовом проекте разрабатывается простой и удобный органайзер, который поможет навести порядок в вашей жизни. Наглядный список задач поможет ничего не забыть и не упустить. С помощью календаря отмечайте важные даты и события.

2. Постановка задачи

2.1. Цель

Разработать веб-приложение для планирования дел, которое

- Поможет навести порядок в распределении дел пользователя с помощью возможности распределения дел по спискам
- Поможет уменьшить время, затрачиваемое на планирование
- Позволит отмечать на календаре важные события

2.2. Сфера использования

Повседневная жизнь.

2.3. Требования

- 2.3.1. Регистрация и авторизация пользователей
- 2.3.2. Возможность просмотра и добавления:
 - 2.3.2.1. Дел
 - 2.3.2.2. Списков дел
 - 2.3.2.3. Событий с помощью календаря
- 2.3.3. Редактирование профиля пользователя
- 2.3.4. Просматривание дел в отсортированном по дате виде

2.4. Задачи

- 2.4.1. Провести анализ рынка с целью выявления достоинств и недостатков схожих по функционалу систем
- 2.4.2. Спроектировать приложение с учетом информации, полученной ранее в ходе анализа
- 2.4.3. Реализовать прототип приложения, обладающий функционалом, описанным в требованиях
- 2.4.4. Описать процесс разработки и результат

3. Анализ предметной области

3.1. Целевая аудитория

Органайзер подойдет как для учеников/студентов, так и для работников любой сферы. Более 80% людей, по нашим опросам, хотят планировать свой день, неделю, месяц. Следовательно, это web-приложение ориентировано на большинство пользователей компьютера с доступом в интернет.

3.2. Результаты опроса

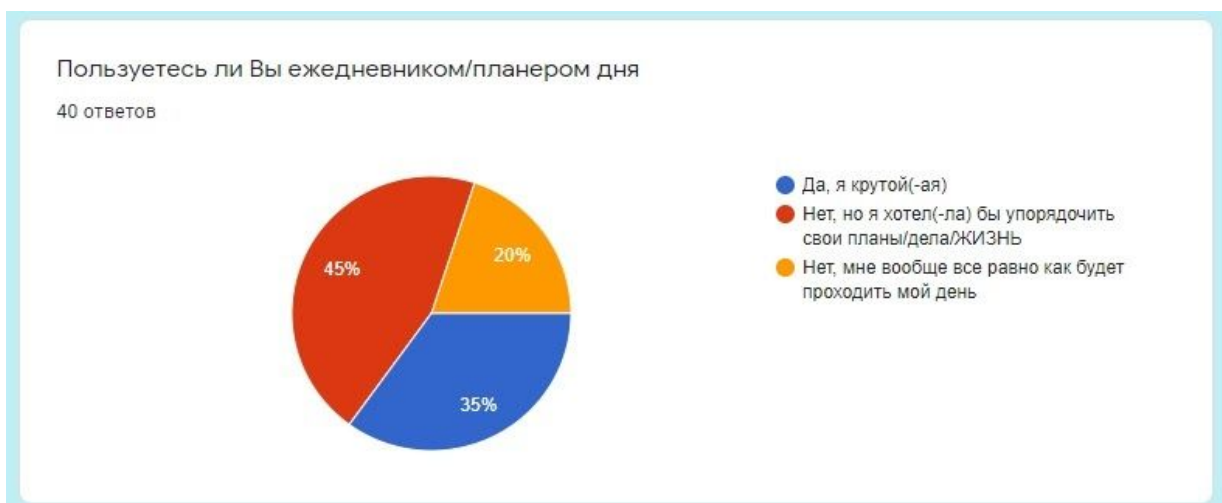


Рис.1. Опрос на целевую аудиторию

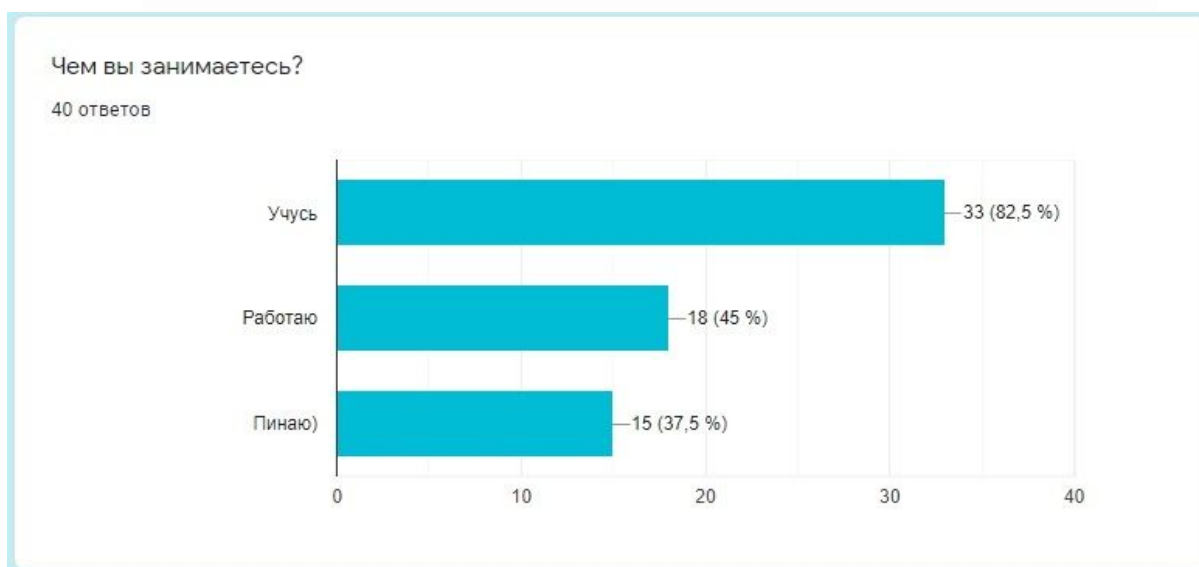


Рис.2. Опрос о занятости.

3.3. Описание предметной области

Органайзер изначально представляет собой небольшую книгу, содержащую календарь, адресную книгу и блокнот. Органайзер служит для организации информации о личных контактах и событиях. С развитием информационных технологий книга стала заменяться сначала электронными органами, затем карманными персональными компьютерами, компьютерными программами и онлайн-органами, которые могут обладать дополнительными функциями: напоминание о предстоящих событиях, защита и синхронизация информации.

Органайзер является средством управления временем. Предварительное планирование дел помогает повысить плодотворность любой деятельности, как личной, так и профессиональной.

В настоящее время под органом чаще понимается именно его программная версия для персонального компьютера или мобильного устройства. Их безусловными преимуществами являются большой объем вводимых данных, установка автоматических напоминаний.

В связи с этим, было принято решение создать онлайн-орган "Simple One".

3.4. Пользователи системы

Система предназначена для работы пользователя со списками, делами, событиями.

Пользователь обладает такими параметрами как:

- Имя пользователя
- Электронная почта
- Пароль
- Аватар

Использование системы происходит с помощью web-приложения “Simple One”, включает в себя работу с личным кабинетом, добавление и удаление дел, создание списков для разбиения дел по категориям и добавление событий в определенные дни.

3.5. Границы функциональности

Данная система реализовывает строго определенные возможности пользователя:

1. Создание личного кабинета
2. Изменение личных данных
3. Добавление дел
4. Просмотр всех дел
5. Добавление списков (категорий)
6. Просмотр дел в списках (категорий)
7. Добавление событий
8. Просмотр календаря

9. Изменение дел
10. Изменение списков
11. Просмотр подробной информации о деле
12. Отметить дело как выполненное/невыполненное
13. Удаление событий
14. Удаление списков

3.6. Анализ существующих решений

На данный момент существует несчетное множество органайзеров. Все органайзеры имеют разнообразные возможности. Из этого числа программ, лишь часть может поддерживать актуальность информации. В эту часть входят как онлайн-приложения, так и настольные и мобильные.

Некоторые онлайн-органайзеры, которые были найдены в результате исследования:

Thebigpic.org

The Big Picture: Signup

E-Mail:


First name:

Last name:

Password:

Repeat password:

Date style: ☒ 25/08/2009 ☐ 08/25/2009

Code above:  Incorrect code.

*By signing up you accept our [Terms of Service](#) and [Privacy Policy](#)

Достоинства данного приложения определить не получилось, так как не удалось пройти регистрацию. Предположительно у него имеются проблемы с проверкой кода безопасности.

Todolist.ru

Здесь могла быть ваша [реклама](#) :-)

demo ([Выход](#))

Создать список

Настройка

Списки задач

Другой список

0 не выполнено

Список

0 не выполнено

[Главная](#)
[Предложения](#)
[Блог](#)
[Пользовательское соглашение](#)
[Поддержка](#)

© Todolist.Ru, 2020

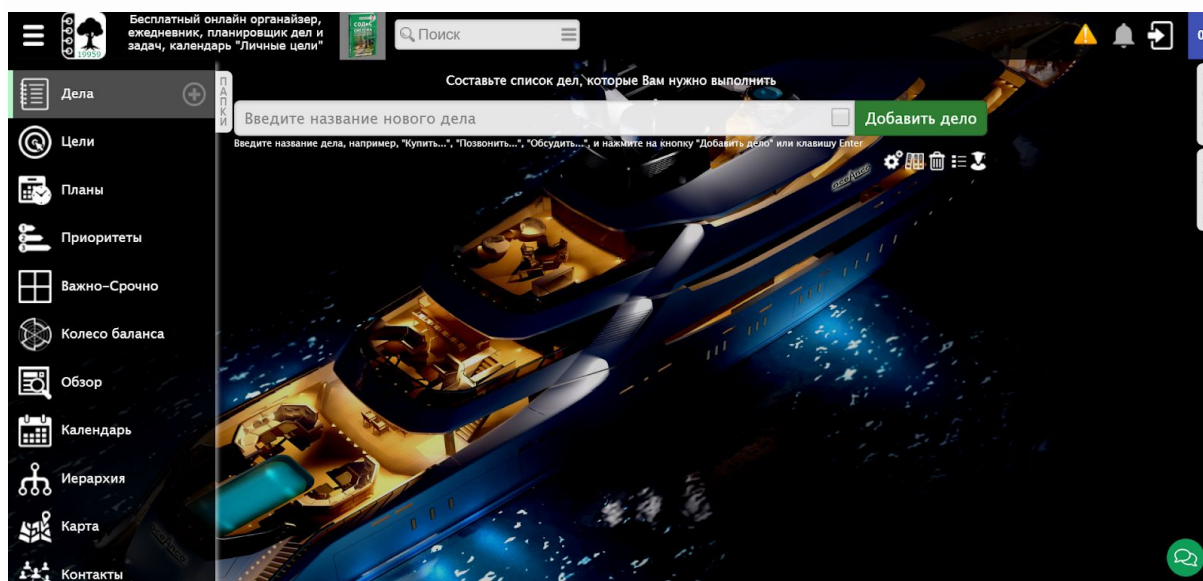
Достоинства:

- Возможность создания списков для распределения дел по категориям
- Возможность редактирования дел, в том числе определение сроков выполнения, добавление описания

Недостатки:

- Неудобная навигация по сайту
- Неудачное сочетание цветов для интерфейса

manprogress.com



Достоинства:

- Имеет множество полезных для планирования функций и методик
- Удобная навигация

Недостатки:

- Хотя функции и полезные, их слишком много

- Фоновое изображение значительно затрудняет восприятие информации

В итоге анализа существующих решений были сделаны следующие выводы:

- Для регистрации в системе должен запрашиваться минимальный необходимый список параметров (Имя пользователя, e-mail, пароль)
- Приложение должно иметь удобную систему навигации, с помощью которой можно без лишних действий попасть на любую необходимую страницу
- Система должна выполнять основные функции органайзера(добавление/изменение/удаление дел, списков; добавление/удаление событий), не быть перегруженной излишними возможностями
- Интерфейс приложения должен быть светлым; в качестве основных были выбраны три цвета: белый(равновесие, уверенность), голубой(успокоение), зеленый(жизнь)

3.7. Воронки конверсии

Яндекс.Метрика собирает информацию о взаимодействии пользователя с сайтом и фиксирует достижение целей. Затем, на основе собранных данных рассчитываются целевые метрики. Цель - это действие пользователя, в котором заинтересован владелец сайта.

Было принято решение создать три продуктовых воронки, основанных на целевых действиях пользователя Сервиса. Созданные цели

позволяют отслеживать события на сайте (нажатие кнопки, заполнение формы и пр.), при выполнении которых не меняется URL страницы. Информация о достижении такой цели передается в Яндекс.Метрику с помощью JavaScript, что позволяет отслеживать практически любые произвольные события.

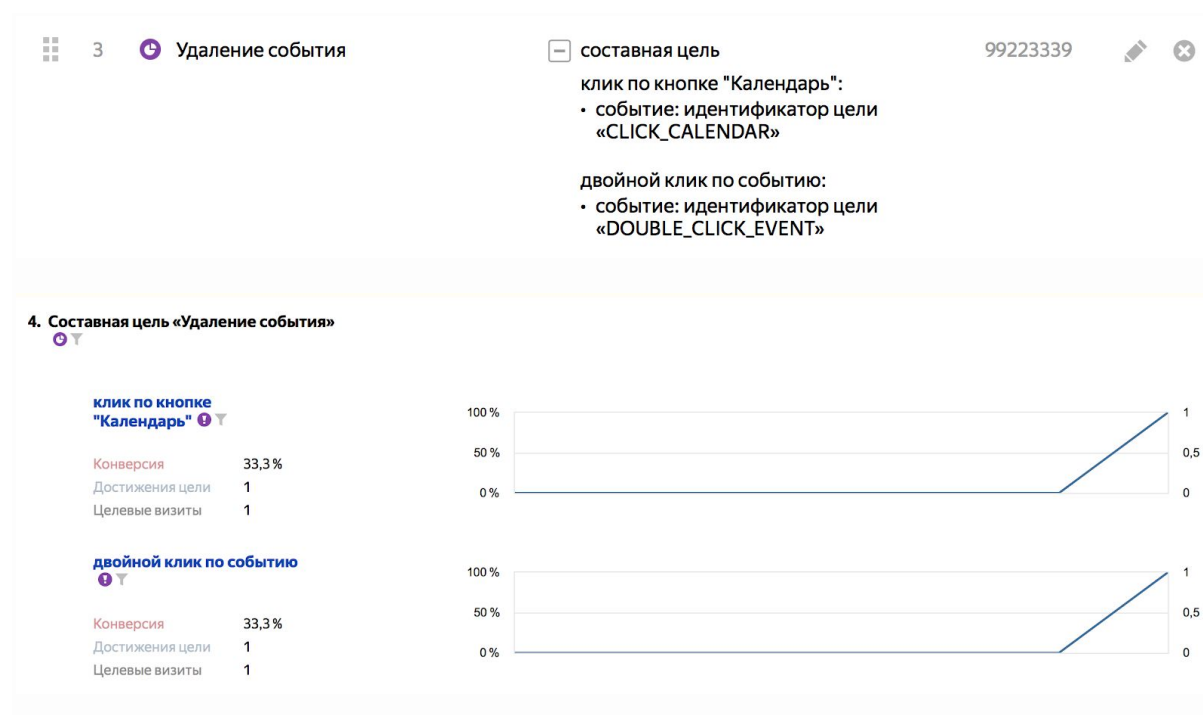
Достижение цели отслеживалось при выполнении условия, заданного в параметрах конкретной цели.

Yandex Metrica:

https://metrika.yandex.ru/stat/conversion_rate?period=week&id=62236273

Воронки:

3.7.1. Удаление события



3.7.2. Изменение дела

2 Изменение дела

— составная цель

99221143



клик по иконке "Изменить дело"
(ID: 99221146):

- событие: идентификатор цели «CLICK_CHANGE_THING»

ввод данных с формы (ID: 99221149):

- событие: идентификатор цели «CHANGE_THING_INFO»

клик по кнопке "Изменить дело"
(ID: 99221152):

- событие: идентификатор цели «CHANGE_THING»

3. Составная цель «Изменение дела»

клик по иконке "Изменить дело"

Конверсия 50 %
Достижения цели 4
Целевые визиты 1

ввод данных с формы

Конверсия 50 %
Достижения цели 14
Целевые визиты 1

клик по кнопке "Изменить дело"

Конверсия 50 %
Достижения цели 13
Целевые визиты 1



3.7.3. Создание списка

1 Создание списка

— составная цель

99220324



клик по кнопке "Добавить список"
(ID: 99220345):

- событие: идентификатор цели «CLICK_ADD_LIST»

ввод данных с формы (ID: 99220348):

- событие: идентификатор цели «ADD_INFO_LIST»

клик по кнопке "Создать"
(ID: 99220351):

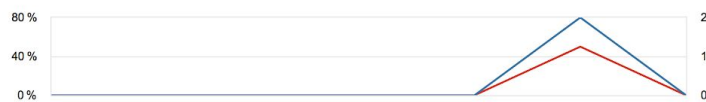
- событие: идентификатор цели «CREATE_LIST»

2. Составная цель «Создание списка»



клик по кнопке "Добавить список"

Конверсия 50 %
Достижения цели 2
Целевые визиты 1



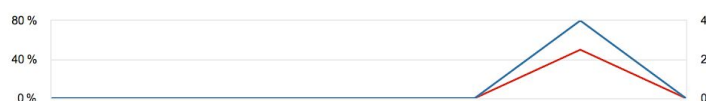
ввод данных с формы

Конверсия 50 %
Достижения цели 4
Целевые визиты 1



клик по кнопке "Создать"

Конверсия 50 %
Достижения цели 4
Целевые визиты 1



3.8. Анализ задачи

3.8.1. Варианты использования приложения

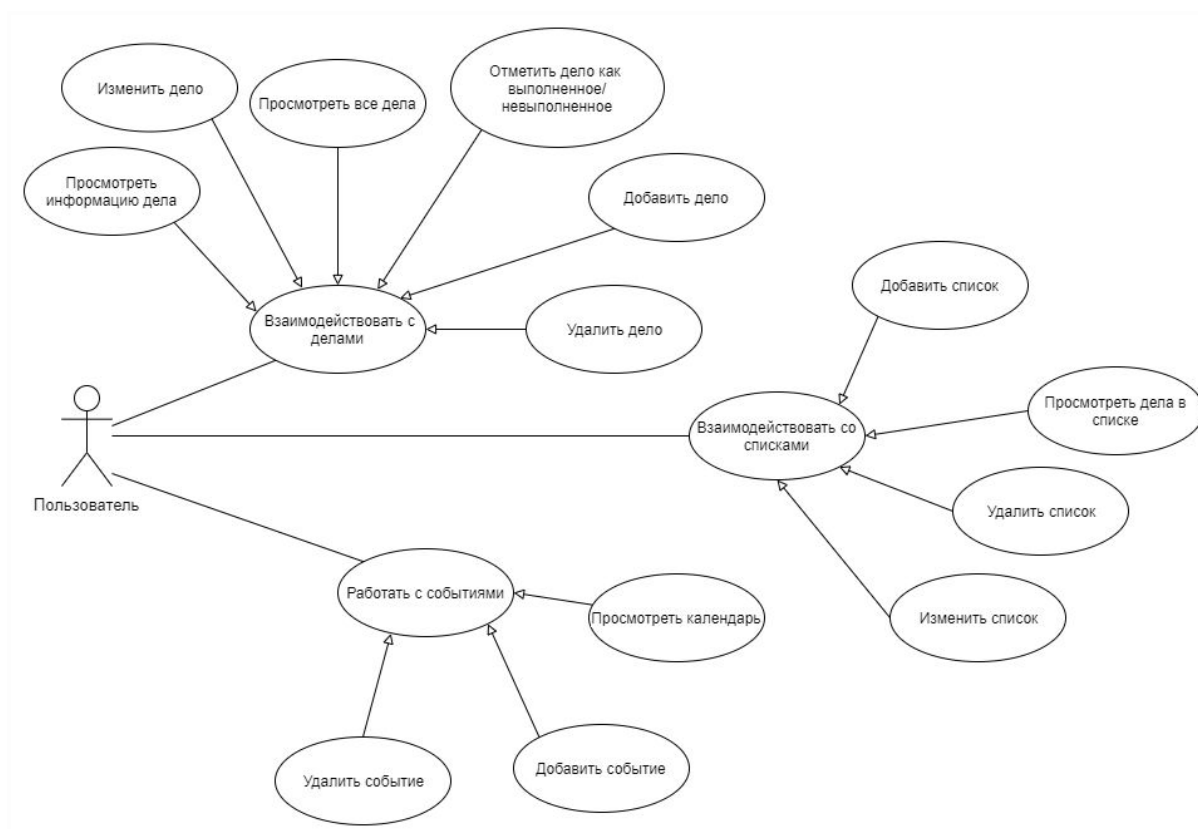


Рис.3. Диаграмма прецедентов.

При взаимодействии пользователя с приложением есть определенный список возможностей, который мы разбили на 3 обобщенных пункта, в которые входят дополнительные действия:

1) Работа с делами

- a) Просмотреть все дела
- b) Добавить дело
- c) Изменить дело
- d) Удалить дело
- e) Просмотреть информацию конкретного дела
- f) Отметить дело как выполненного/невыполненного

2) Работа с событиями

- a) Посмотреть календарь

b) Добавить событие

c) Удалить событие

3) Работа со списками

a) Добавить список

b) Просмотреть дела в списке

c) Изменить список

d) Удалить список

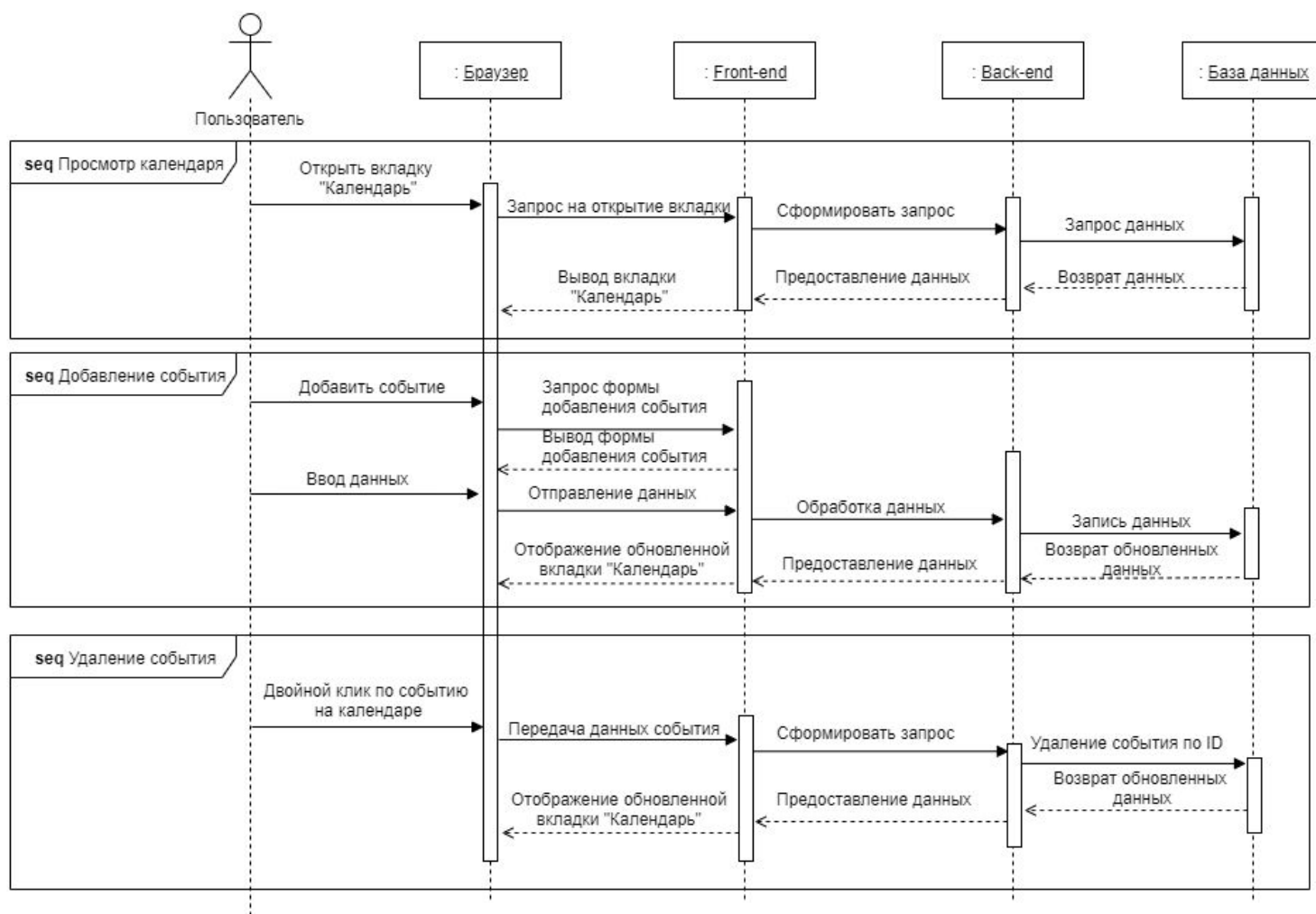


Рис.6. Диаграмма последовательностей для работы с событиями.

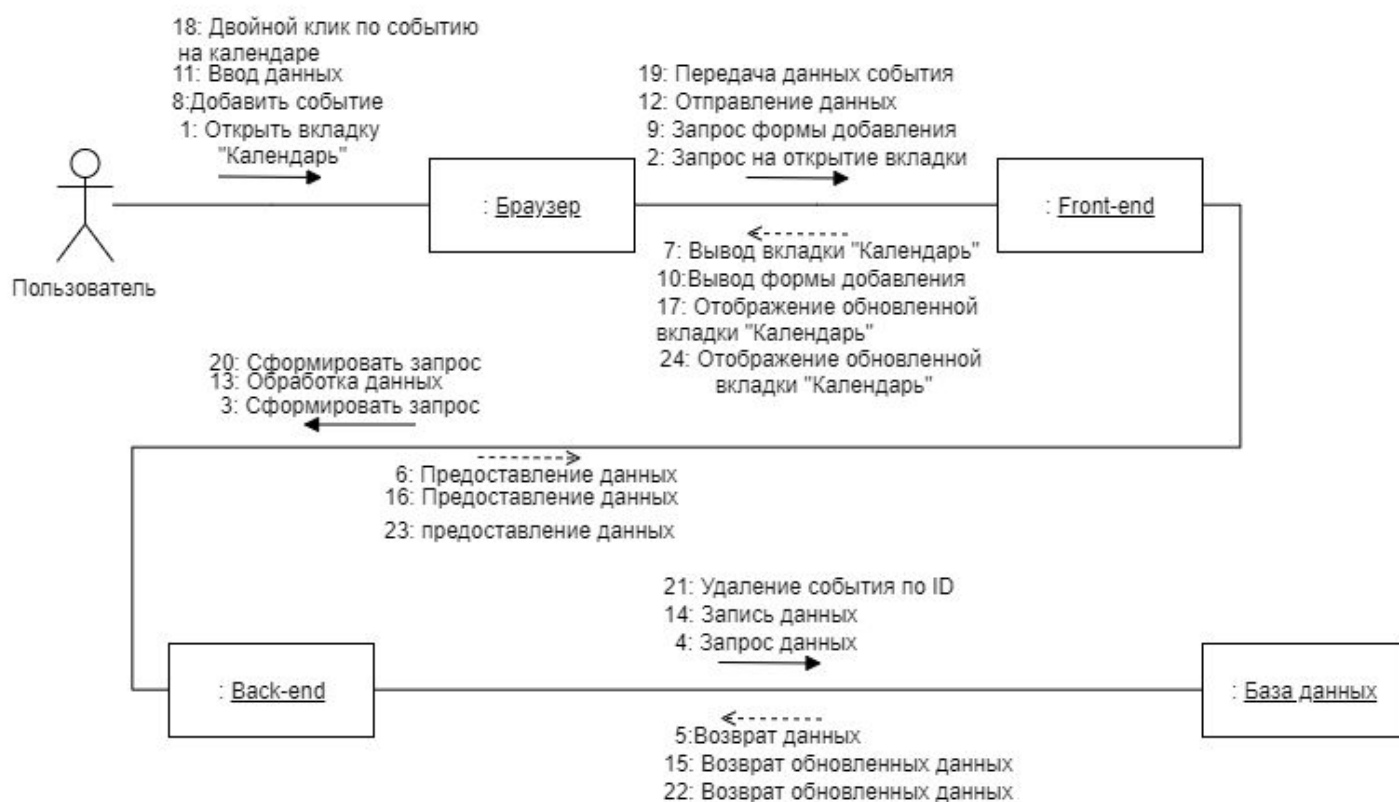


Рис.7. Диаграмма взаимодействия для работы с событиями.

На рисунке 7 показана диаграмма взаимодействия, на которой указываются отношения между объектами при работе с событиями в приложении, а на рисунке 6 изображена диаграмма последовательности, на которой изображены, упорядоченные во времени, взаимодействия объектов в данном блоке.

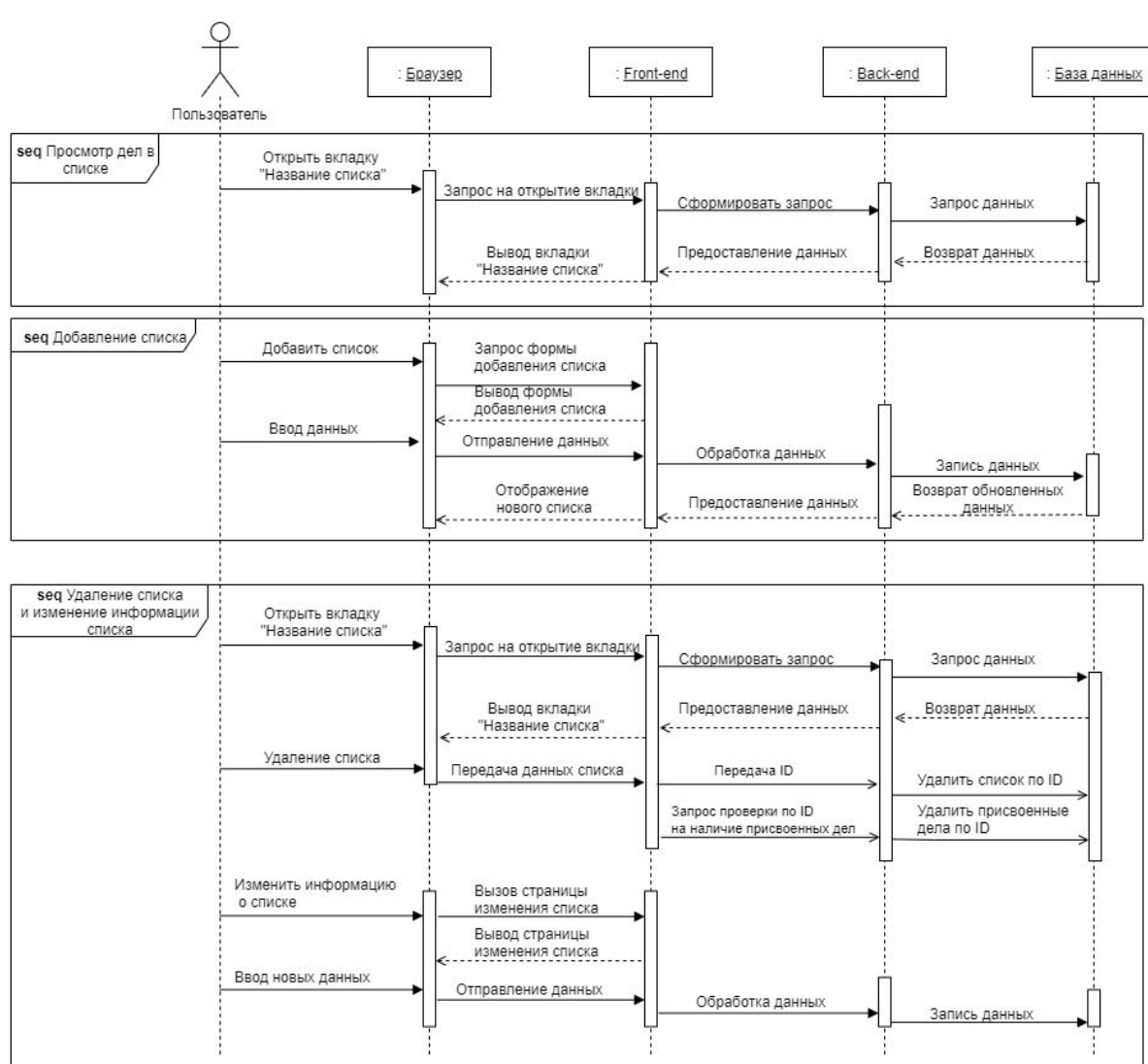


Рис.8. Диаграмма последовательности для работы со списками.

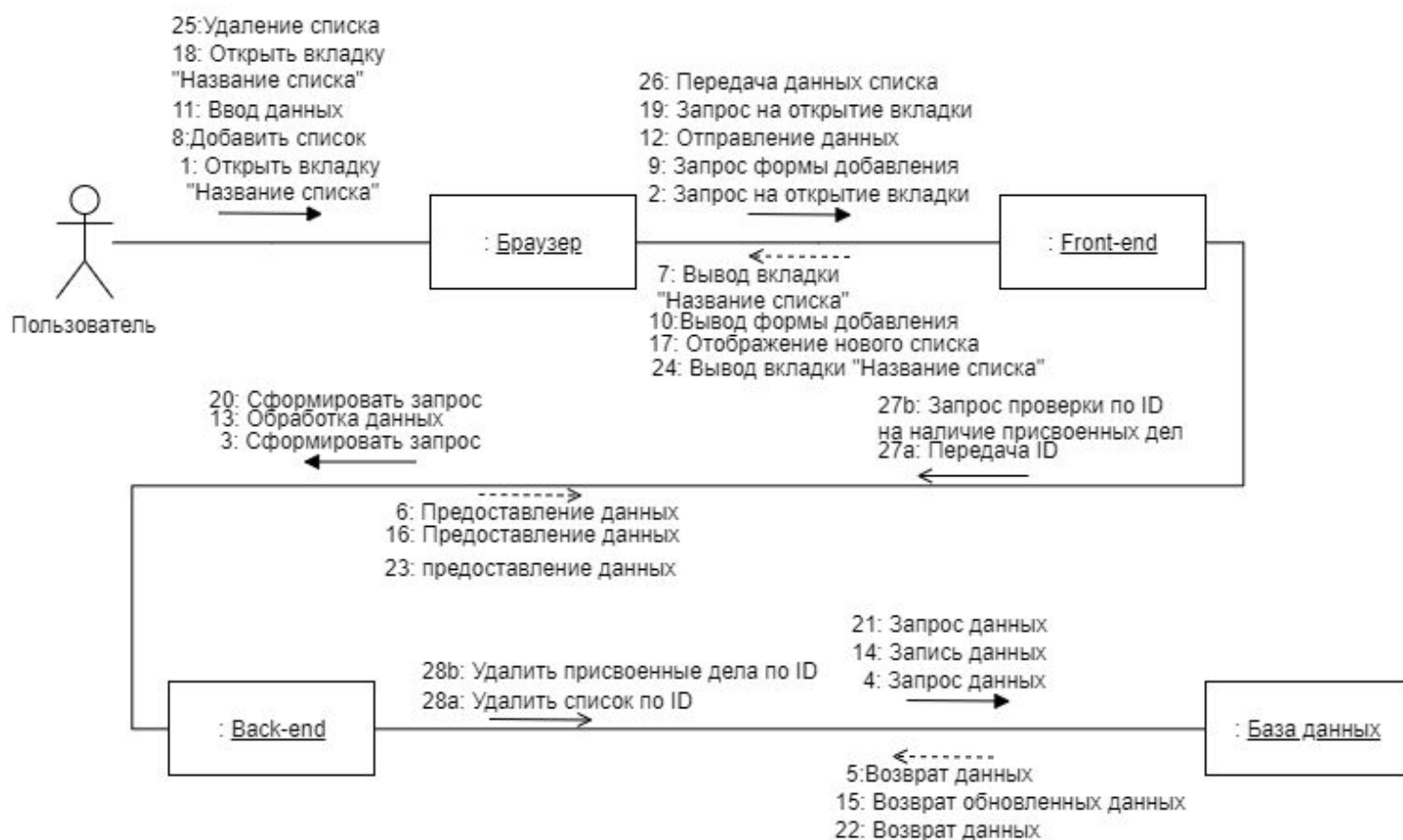


Рис.9. Диаграмма взаимодействия для работы со списками.

На рисунке 9 показана диаграмма взаимодействия, на которой указываются отношения между объектами при работе со списками в приложении, на рисунке 8 изображена диаграмма последовательности, на которой изображены, упорядоченные во времени, взаимодействия объектов в данном блоке.

3.8.3. Варианты состояния системы

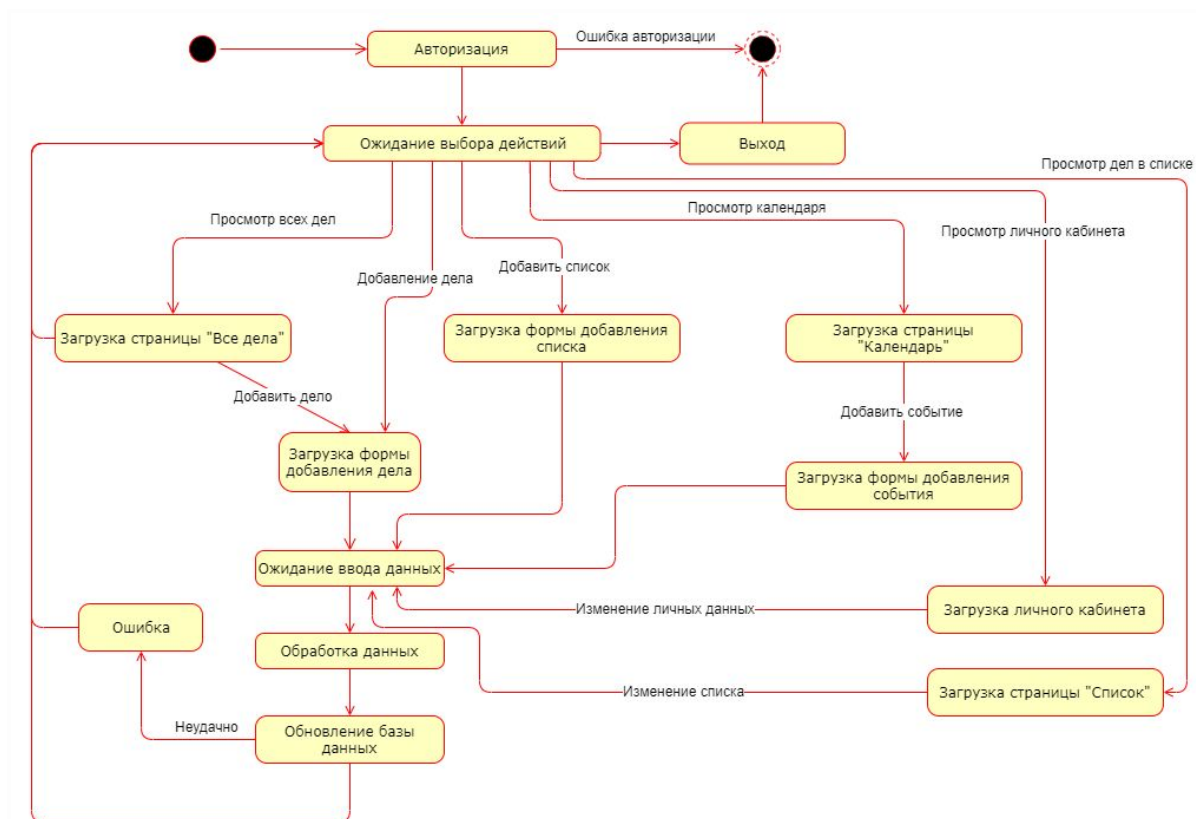


Рис.10. Диаграмма состояния.

Диаграмма состояний, изображенная на рисунке 10, отражает возможные состояния системы. При запуске приложение требует авторизоваться, после этого переходит в состояние ожидания выбора действия. В зависимости от выбора пользователя возможны 6 основных цепочек состояний:

- 1) Просмотр всех дел
- 2) Добавление дел
- 3) Добавление списка
- 4) Просмотр списка
- 5) Просмотр календаря
- 6) Просмотр личного кабинета

7) Выход из системы

Рассмотрим цепочку для добавления дел:

Если пользователь выбирает опцию добавление дела, система переходит в состояние загрузки формы добавления дела, после загрузки формы система переходит в состояние ожидания ввода данных, после ввода данных пользователем система переходит в состояние обработки данных, затем переходит в состояние обновление базы данных. Если все данные прошли проверки на валидность успешно, то система переходит в состояние ожидания выбора новой опции от пользователя. Если же данные не прошли проверки, то система переходит в состояние вывода ошибки после чего переходит в ожидание выбора новой опции от пользователя.

3.8.4. Варианты действия в системе

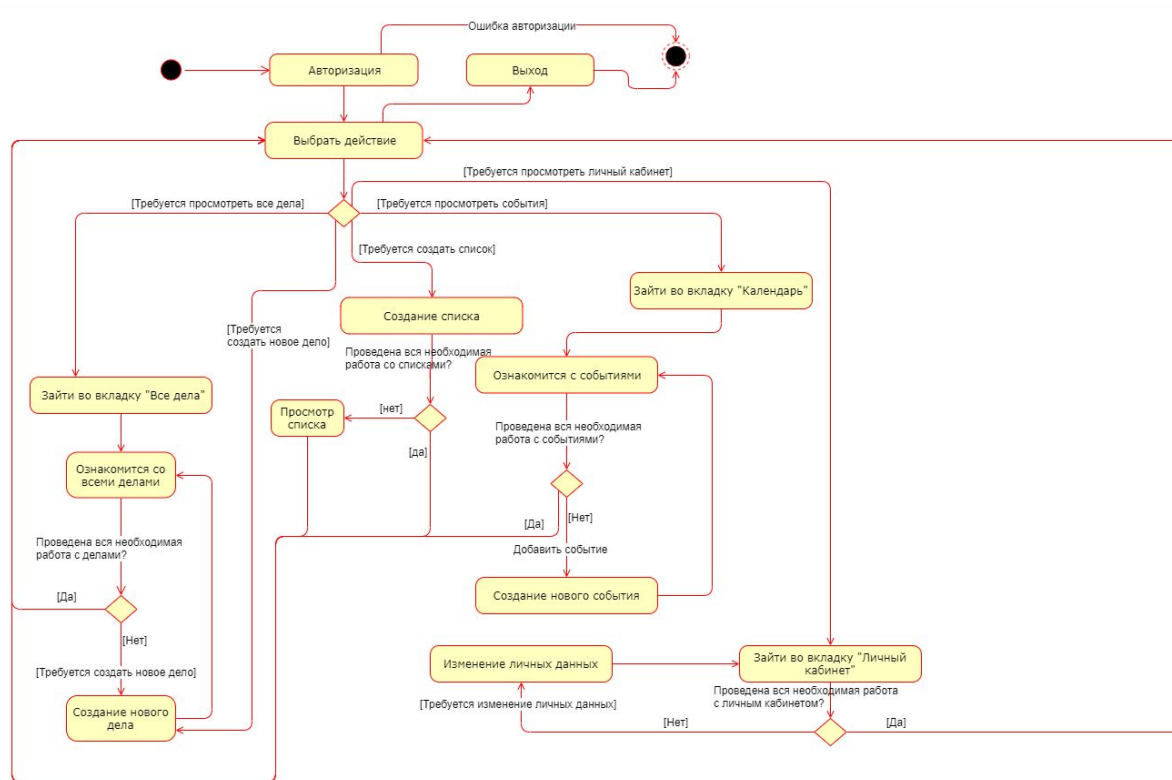
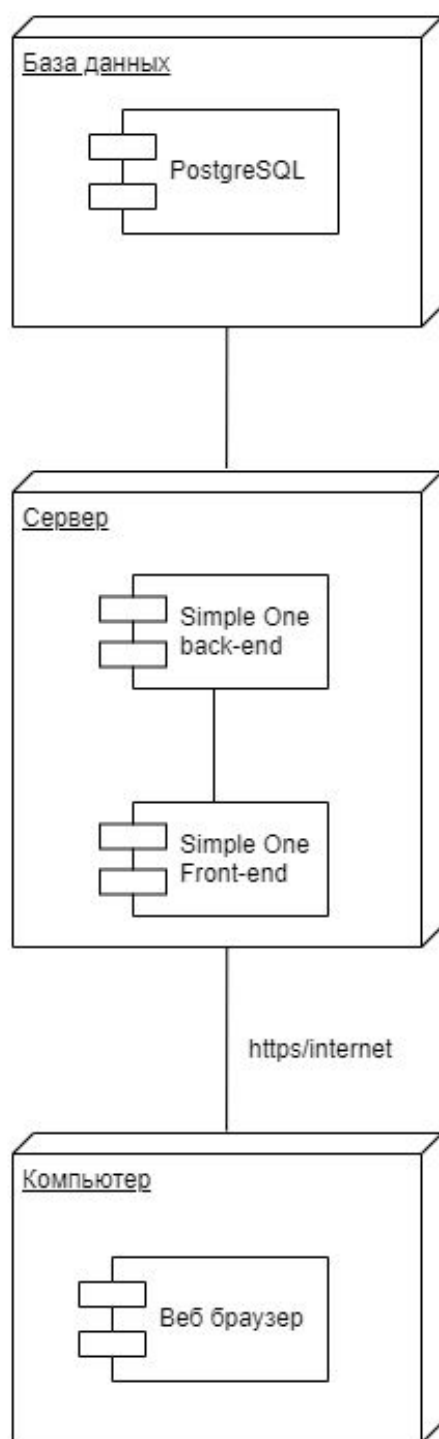


Рис.11. Диаграмма активности

Диаграмма активности, изображенная на Рисунке 11, отражает возможные действия, состояния которых описаны на диаграмме состояния (Рисунок 10). При запуске приложение требует авторизоваться, после этого переходит в состояние ожидания выбора действия. В зависимости от выбора пользователя возможны 6 основных цепочек состояний:

- 1) Просмотр всех дел
- 2) Добавление дел
- 3) Добавление списка
- 4) Просмотр календаря
- 5) Просмотр личного кабинета
- 6) Выход из системы

3.8.5. Развертывание приложения



На рисунке 12 представлена диаграмма развертывания, чтобы определить какие аппаратные компоненты («узлы») существуют, какие программные компоненты («артефакты») работают на каждом узле и как различные части этого комплекса соединяются друг с другом. Для разрабатываемого веб-приложения узлом устройства является компьютер, сервер и база данных, а в качестве узла среды выполнения выступает веб браузер. На серверной части развернуты front-end и back-end приложения и отдельно база данных.

Рис.12. Диаграмма развертывания

3.8.6. Диаграмма классов

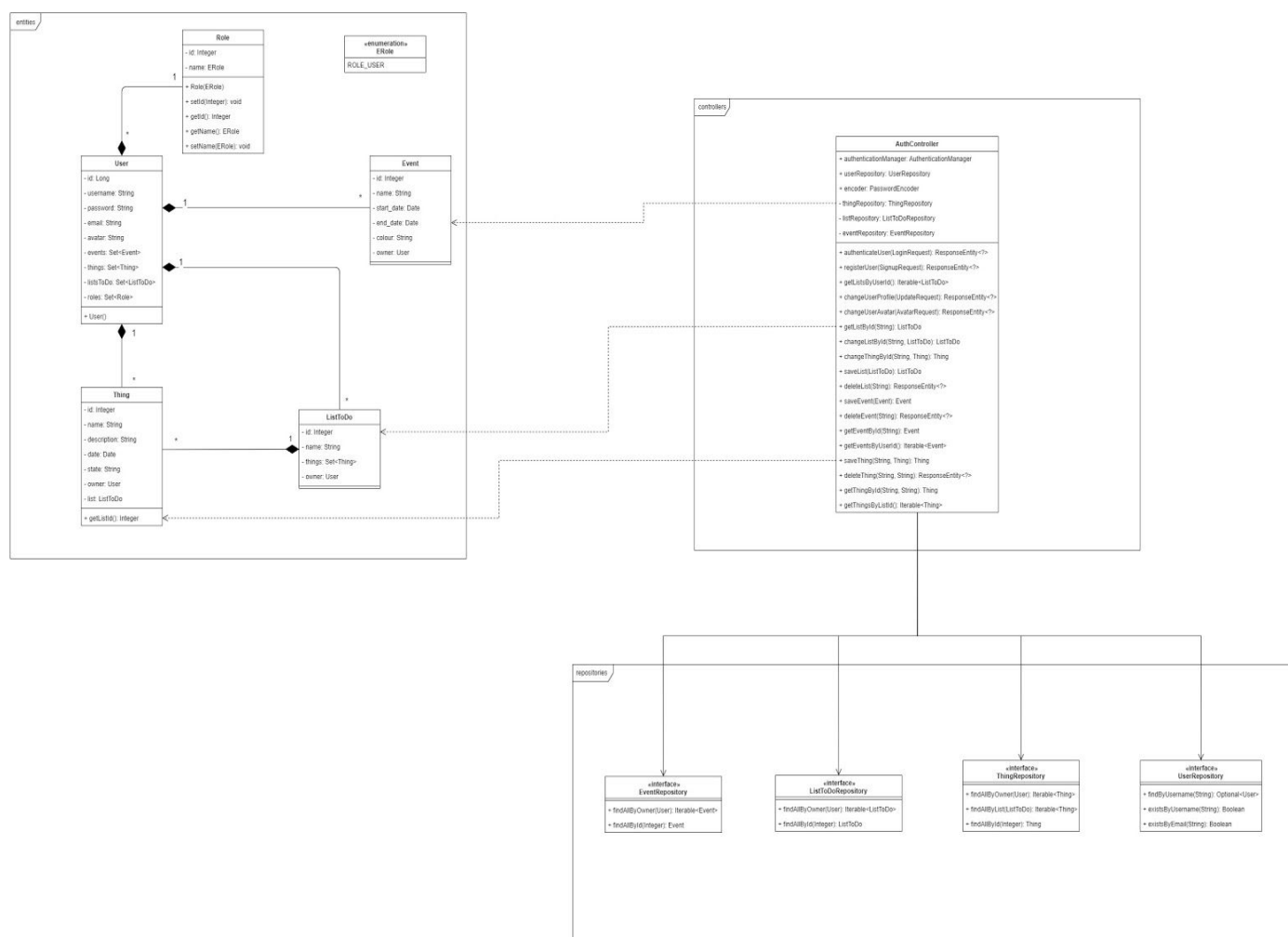


Рис.13. Диаграмма классов

На рисунке 13 представлена диаграмма классов, соответствующая архитектуре приложения.

В пакете `entities` размещены классы, описывающие модели сущностей базы данных. В пакете `controllers` описаны классы для взаимодействия между front-end, back-end и БД по получению, проверке, отправлению данных. В пакете `repositories` находятся интерфейсы,

наследуемые от JpaRepository - интерфейса фреймворка Spring Data, предоставляющего набор стандартных методов JPA для работы с БД.

3.8.7. Диаграмма объектов

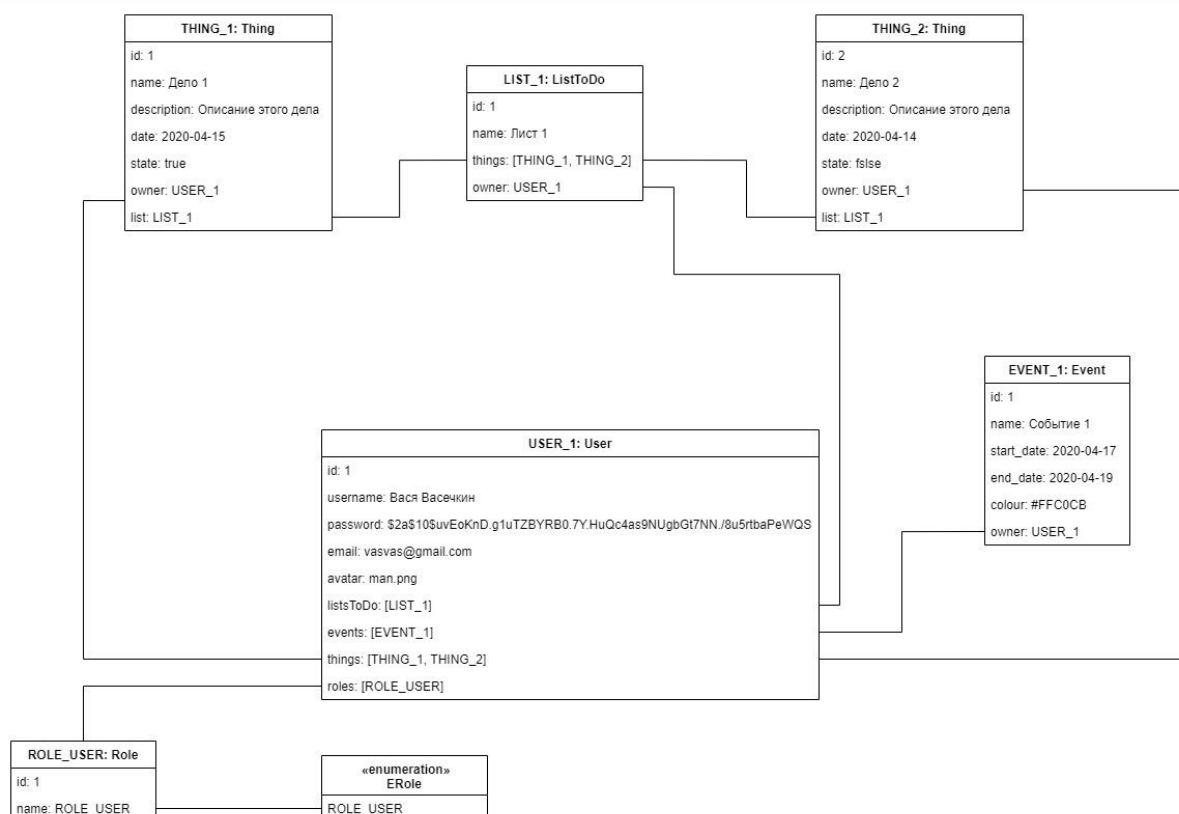


Рис.14. Диаграмма объектов

Для лучшего понимания диаграммы классов была составлена диаграмма объектов. На ней отображены объекты основных классов и их отношения в некоторый момент времени.

4. Архитектура приложения

В качестве средств реализации приложения были выбраны следующие технологии:

Технология Spring MVC, Spring Framework с Spring Boot и AngularTS - продуктивный и привлекательный стек для разработки небольших веб-приложений, в особенности таких, где требуется интенсивно работать с формами.

Фреймворк Spring MVC обеспечивает архитектуру паттерна (Модель — Отображение (далее — Вид) — Контроллер) при помощи слабо связанных готовых компонентов. Паттерн MVC разделяет логику ввода, бизнес-логику и логику UI, обеспечивая при этом свободную связь между ними.

Такая технология позволяет избегать написания многих аннотаций и конфигураций xml, и обеспечивает взаимодействие с системами **Spring JDBC**, **Spring Security** и др.

Spring Framework не только предлагает нам такие функции, как внедрение зависимостей или обработка транзакций, но также выступает в качестве основы для других фреймворков Spring. Лучшим примером для этого является Spring Boot. Spring Boot использует Spring Framework в качестве своей основы. Он упрощает зависимости Spring и запускает приложения прямо из командной строки. Он также не требует наличия внешнего контейнера приложений. Spring Boot помогает контролировать компоненты приложения и настраивает их извне. Благодаря таким функциям, как автоконфигурация, Spring Boot избавляет от написания лишнего кода и помогает избежать ненужной настройки.

AngularTS - это фреймворк, построенный на основе HTML и JavaScript, двух технологий, давно используемых в веб-разработке. Он позволяет использовать привычные редакторы и расширения для браузеров. Версия Angular **CLI** (Command Line Interface) стандартизирует структуру, позволяет создать сущности внутри приложения, а также автоматизировать его сборку. Node.js — это целые самостоятельные платформы, которые базируются на определенных языках, но имеют очень широкие возможности.

Заметим, что за последние годы отмечен бурный рост и растущая популярность Node.js, а также AngularTS.

На фреймворках разрабатываются довольно большие и сложные сайты с уникальным функционалом. Это значительно быстрее и дешевле, чем на чистом языке, но при этом такое решение позволяет разрабатывать действительно сложные вещи и оптимизировать все это под нагрузки. Кроме того, это почти всегда более безопасно, чем любая коробочная CMS.

Некоторые фреймворки:

1. PHP: Symfony, Laravel
2. Python: Django
3. Ruby: Ruby On Rails
4. Java: Spring
5. C#: .NET
6. JS: Node.js, AngularJS/AngularTS

Существует множество языков программирования, шаблонов и фреймворков. Но, благодаря многим функциям, упомянутым выше, Spring Framework и AngularTS является отличным выбором для нашего проекта.

В качестве СУБД была выбрана PostgreSQL в силу открытого доступа и высокой производительности, а также за счет поддержки данной СУБД schema-less данных, такие как JSON. PostgreSQL поддерживает интеграцию на различные платформы, взаимодействие с большинством языков программирования. PostgreSQL имеет преимущество перед другими DBMS, когда необходимо перенести базу данных из одной ОС в другую или в случае, когда реализация приложения производится на различных ОС.

Специфика клиентской стороны веб-приложения требует отладки кода во всех популярных браузерах. Наше приложение разрабатывалось под браузер Google Chrome. Этот браузер, основан на WebKit, обладает встроенным инструментом разработки Web Inspector, который очень хорошо развит и позволяет выполнять отладку JavaScript кода.

Общение между Front-end и Back-end происходит по средствам REST API, а обмен информации происходит с помощью передачи JSON файлов.

- 5. Проектная часть**
- 6. Заключение**
- 7. Список используемых материалов**