

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ» (ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук
Кафедра программирования и информационных технологий

Курсовой проект по курсу «Технологии программирования»
«Мобильное приложение Chirk»

Зав. кафедрой _____ *Махортов С.Д., д. ф.-м. н., профессор*

Обучающиеся _____ *Лукашев О.А., 3 курс, 6 группа*

_____ *Грибанова Д.А., 3 курс, 6 группа*

_____ *Мануковский И.С., 3 курс, 6 группа*

_____ *Пеканов Н.А., 3 курс, 6 группа*

_____ *Акмызрак М.Д., 3 курс, 6 группа*

_____ *Бабирье Д. К., 3 курс, 6 группа*

Руководитель _____ *Тарасов В.С., ст. преподаватель*

Воронеж 2023

Содержание

Содержание	2
Введение	4
1 Постановка задачи	5
1.1 Цели создания приложения	5
1.2 Задачи, решаемые при помощи приложения	5
1.3 Требования к приложению и программному обеспечению	5
1.4 Требования к оформлению и верстке страниц приложения	5
2 Анализ предметной области	7
2.1 Глоссарий	7
2.2 Существующие решения	7
2.2.1 Twitter	7
2.2.2 Mastodon	8
2.2.3 Diaspora	9
3 Реализация	11
3.1 Средства реализации	11
3.2 Архитектура серверной части приложения	13
3.3 Архитектура клиентской части приложения	14
3.4 Диаграммы, иллюстрирующие работу системы	14
3.4.1 Диаграмма вариантов использования	15
3.4.2 Диаграмма классов	16
3.4.3 Диаграмма активностей	18
3.4.4 Диаграмма последовательности	21
3.4.5 Диаграмма развертывания	23
3.4.6 Диаграмма сотрудничества	24
3.4.7 Диаграмма объектов	25
4 Экраны приложения	27
4.1 Авторизация	27
4.2 Регистрация	27
4.3 Лента «Чирков»	28
4.4 Создание «Чирка»	30
4.5 Профиль	32

4.6 Редактирование профиля.....	33
4.7 Понравившееся и не понравившееся	34
4.8 Онбординг.....	35
4.9 Лента созданных «Чирков».....	36
5 Тестирование	38
Заключение	39
Список использованных источников	40

Введение

Социальные сети, с момента их появления все больше проникают в нашу жизнь. Для многих они уже сейчас являются основным местом проведения времени в интернете. Это происходит потому, что всем нам хочется общения, но не все способны общаться в реальной жизни, так как существует множество психологических барьеров. Социальные сети рушат эти барьеры, позволяя совершенно незнакомым людям найти общий язык, не только посредством общения, но и посредством открытой информации, которую оставляют пользователи.

Однако, несмотря на популярность существующих социальных сетей, в каждой из них есть свои недостатки. Причем недостатки имеются как в функциональности, так и в удобстве взаимодействия пользователя с системой.

Целью данного проекта является разработка приложения, позволяющего пользователям делиться своими мыслями и реагировать на мысли других.

Также стоит отметить, что приложение разрабатывается не только для обычных пользователей, но и для модераторов, которые имеют возможность удалять нежелательные, оскорбительные или выходящие за рамки приличия публикации.

Актуальность темы данного курсового проекта обусловлена тем, что часть популярных социальных сетей на данный момент недоступна на территории Российской Федерации, а также тем, что у всех социальных сетей есть свои недостатки, например, отсутствие отрицательной реакции на публикацию или отсутствие возможности скрыть публикацию от других пользователей.

1 Постановка задачи

1.1 Цели создания приложения

Приложение предназначено для:

- Предоставления возможности делиться своими мыслями с другими пользователями;
- Распространения новостей;
- Предоставления пользователям альтернативного способа обмена информацией.

1.2 Задачи, решаемые при помощи приложения

Приложение создается для обеспечения возможности:

- Пользователям добавлять «Чирк»;
- Смотреть и реагировать на «Чирки» других пользователей;
- Модераторам удалять неприемлемые «Чирки»;
- Пользователям удалять свои «Чирки»;
- Пользователям скрывать свои «Чирки»;
- Пользователям создавать временные «Чирки».

1.3 Требования к приложению и программному обеспечению

К разрабатываемому приложению выдвинуты следующие требования:

- Приложение должно поддерживаться Android-устройствами с версией 8.0 и новее;
- Приложение должно иметь интерфейс с ясной и логичной навигацией.

1.4 Требования к оформлению и верстке страниц приложения

Выделены следующие требования к оформлению и верстке:

- Приложение должно иметь понятную навигацию. Пользователи должны иметь возможность перемещаться по приложению и находить то, что им нужно;
- Цветовая гамма должна быть визуально привлекательна;
- Экраны приложения должны быть минималистичны, чтобы пользователя ничего не отвлекало от использования приложения.

2 Анализ предметной области

2.1 Глоссарий

Проект – разрабатываемое приложение.

Мобильное приложение – специально разработанное приложение под конкретную мобильную платформу.

База данных (БД) — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе.

GitHub – веб-сервис для хостинга IT-проектов и их совместной разработки.

Java – строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems.

JDK – комплект разработчика приложений на языке Java, включающий в себя компилятор Java, стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java.

Backend – логика работы сайта, внутренняя часть продукта, которая находится на сервере и скрыта от пользователей.

Frontend – презентационная часть информационной или программной системы, ее пользовательский интерфейс и связанные с ними компоненты.

«Чирк» — текстовая публикация.

Онбординг— процесс адаптации клиента к новому продукту.

2.2 Существующие решения

2.2.1 Twitter

«Twitter» — американский сервис микроблогов и социальная сеть, в которой пользователи публикуют сообщения, известные как «твиты», и взаимодействуют с ними.

Из плюсов можно отметить, что сервис прост в использовании и имеет понятный интерфейс. Также сервис предоставляет возможность быстрого обмена информацией. К тому же, в сервисе есть возможность оставлять положительные реакции на публикации.

Из минусов можно отметить, что на данный момент сервис недоступен на территории Российской Федерации, а также в сервисе не предусмотрена возможность отрицательной реакции на публикацию и нет возможности временной публикации. Еще стоит отметить, что из-за большой аудитории многие публикации теряются и остаются незамеченными.



Рисунок 1 - Twitter

2.2.2 Mastodon

Mastodon — это некоммерческая социальная сеть, которая входит в состав объединенных децентрализованных интернет-серверов Fediverse. В Mastodon нет единого управления, соцсеть состоит из множества независимых

серверов, в каждом из которых действуют свои правила поведения и политика конфиденциальности.

Из плюсов можно отметить, что приложение бесплатно и есть возможность скрывать публикации из общей ленты.

Из минусов можно отметить, что в сервисе не предусмотрена возможность временной публикации, нет возможности ставить положительные или отрицательные реакции, а также интерфейс Mastodon может показаться неудобным для пользователей, которые привыкли работать с централизованными социальными сетями.

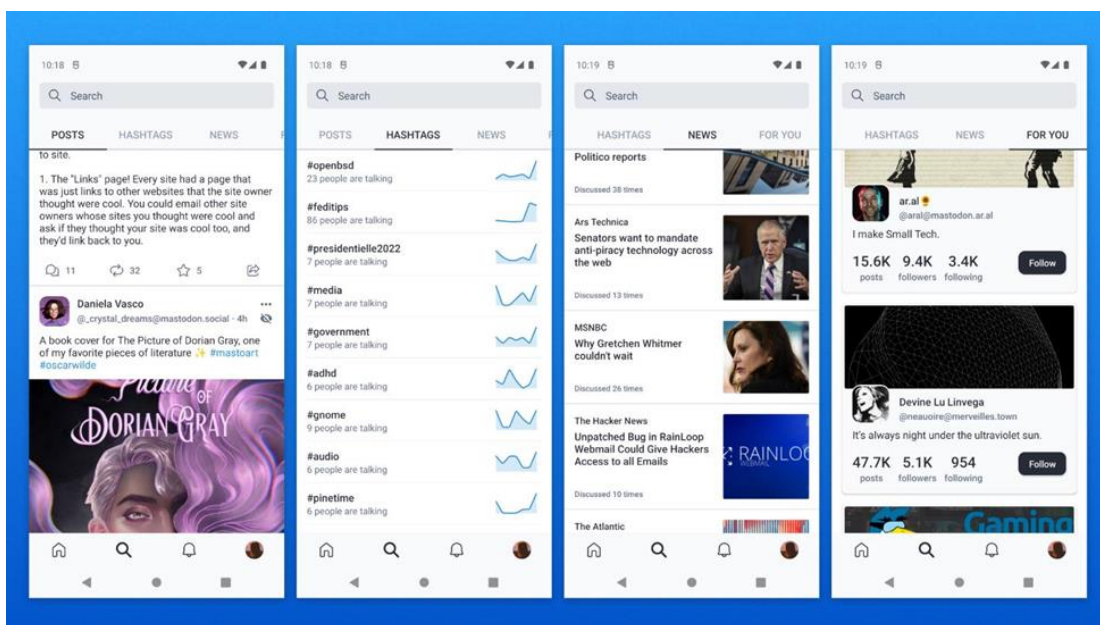


Рисунок 2 - Mastodon

2.2.3 Diaspora

Diaspora — некоммерческая распределенная децентрализованная социальная сеть, построенная на основе одноименного свободного программного обеспечения с открытым исходным кодом.

Из плюсов можно отметить, что приложение бесплатно, есть возможность скрывать публикации и ставить положительные реакции.

Из минусов стоит отметить, что пользовательский интерфейс Diaspora может показаться неудобным, особенно для тех, кто использует

централизованные социальные сети, также нет гарантии удаления публикации и нет возможности отрицательной реакции.

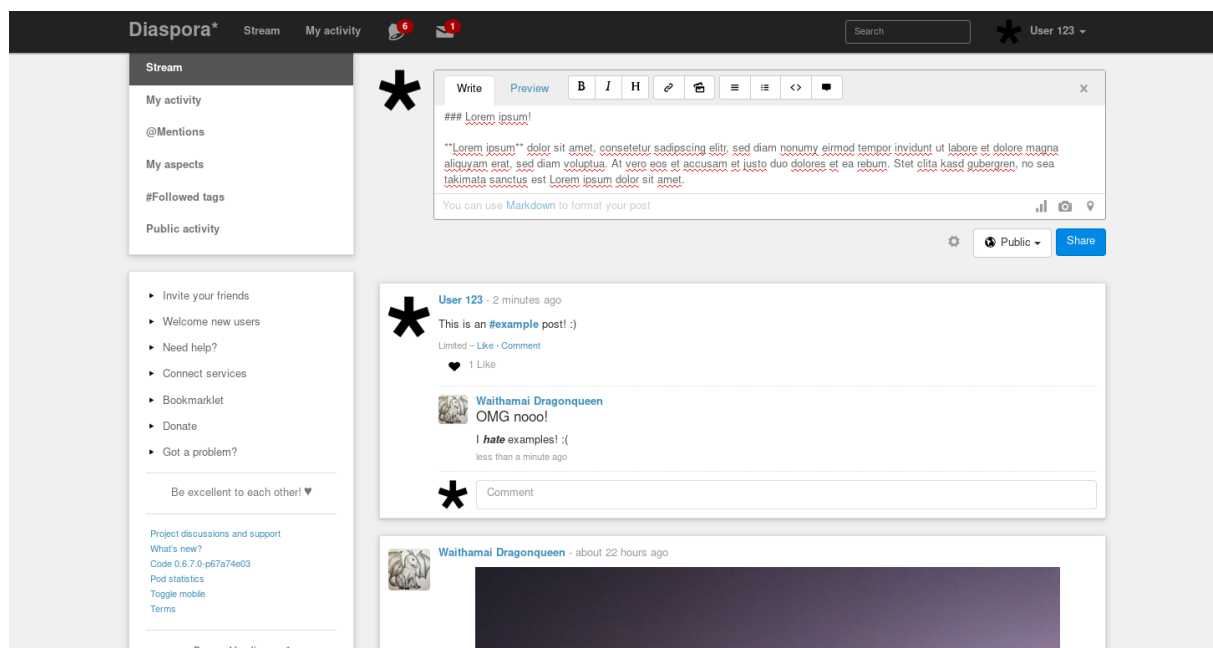


Рисунок 3 - Diaspora

3 Реализация

3.1 Средства реализации

Для реализации приложения выбраны технологии:

- Язык программирования Java 17 версии;
- Фреймворк Spring Boot;
- СУБД PostgreSQL 13;
- Инструмент для создания документации API Swagger;
- Flutter;
- Dart;
- Docker;
- TimeWeb.

Данные технологии были выбраны исходя возможностей, которые они дают для решения задач.

Преимущества Java и фреймворка SpringBoot:

- Большое количество доступной документации: SpringBoot имеет огромное сообщество разработчиков, которые создают документацию, учебники, видеоуроки и т.д., что делает его очень популярным и удобным для изучения и использования;
- Готовые решения для реализации RestFul архитектуры: SpringBoot предоставляет множество инструментов и библиотек для создания RestFul API, что упрощает и ускоряет процесс разработки;
- Удобные инструменты для работы с PostgreSQL БД: SpringBoot обеспечивает хорошую интеграцию с PostgreSQL БД, что позволяет разработчикам работать с ней легко и удобно;
- Готовые встроенные серверы (Tomcat), обеспечивающие ускоренное и более продуктивное развертывание приложений: SpringBoot может быть запущен на встроенном сервере, таком как Tomcat, что делает процесс развертывания приложений быстрым и легким;

- Автоматическое внедрение зависимостей: с помощью автоматического внедрения зависимостей, SpringBoot может упростить процесс настройки приложения и уменьшить количество кода;
- Автоконфигурирование огромного количества компонентов: SpringBoot позволяет автоматически сконфигурировать множество компонентов, таких как базы данных, кэши, безопасность и т.д., что увеличивает производительность и уменьшает количество написанного кода. [1].

Преимущества PostgreSQL БД:

- Большой бесплатный функционал: PostgreSQL является открытым исходным кодом, поэтому предоставляет пользователю широкий набор бесплатных функций для использования в различных проектах. Это может включать в себя инструменты администрирования, оптимизации производительности, безопасности и многое другое;
- Надежность и высокая производительность: PostgreSQL известен своей стабильной работой и хорошей производительностью. Система предоставляет ряд механизмов обеспечения целостности данных, что делает ее надежной и защищенной от ошибок. Она также имеет множество встроенных возможностей [2].

Преимущества Flutter:

- Мультиплатформенность: Flutter — это фреймворк для разработки мобильных приложений, который позволяет создавать приложения для Android и iOS из одной кодовой базы. Это объединяет процесс разработки и снижает время и затраты на разработку, так как необходимо писать только один набор кода, что значительно экономит время и уменьшает количество ошибок;

- Понятная и полная документация: В Flutter имеется всесторонняя документация, которая охватывает большое количество тем, начиная от основ и заканчивая продвинутыми технологиями;
- Возможность быстро проектировать мобильные приложения: Фреймворк предоставляет широкий выбор готовых компонентов и виджетов, которые можно использовать для создания пользовательского интерфейса, а также библиотеку Material Design, которая обеспечивает единообразный дизайн приложений [3].

3.2 Архитектура серверной части приложения

Серверная часть приложения реализована соответственно трехслойной архитектуре.

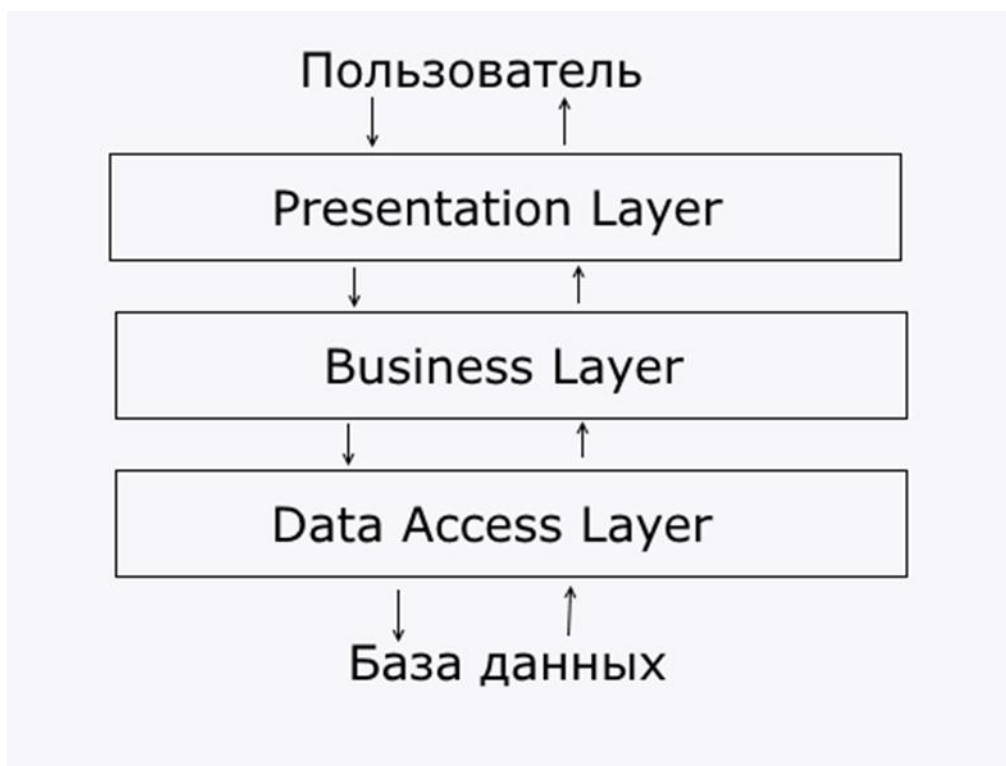


Рисунок 4 - Архитектура серверной части приложения

Presentation Layer - этот слой включает в себя контроллеры REST API, которые обрабатывают HTTP-запросы и возвращают данные клиенту в формате JSON. Контроллеры связаны с бизнес-логикой через слой сервисов.

Business Logic Layer - этот слой содержит сервисы, которые содержат бизнес-логику приложения. Сервисы служат прослойкой между контроллерами и слоем репозитория, обрабатывая запросы от контроллеров, выполняя необходимые операции и передавая данные назад в контроллеры.

Data Access Layer - этот слой содержит интерфейсы репозитория, которые обеспечивают доступ к базе данных. Репозитории позволяют получать и сохранять данные в хранилище данных [4].

3.3 Архитектура клиентской части приложения

Elementary — библиотека, которая предоставляет механизмы для написания приложения по правилам Clean Architecture с разделением модулей на блоки формата MVVM. Это архитектурный пакет для Flutter, который позволяет четко разделить слои согласно ответственностям, сделать эти ответственности прозрачнее, а код проще для восприятия и тестирования.

Библиотека включает несколько основных модулей с четкой структурой.

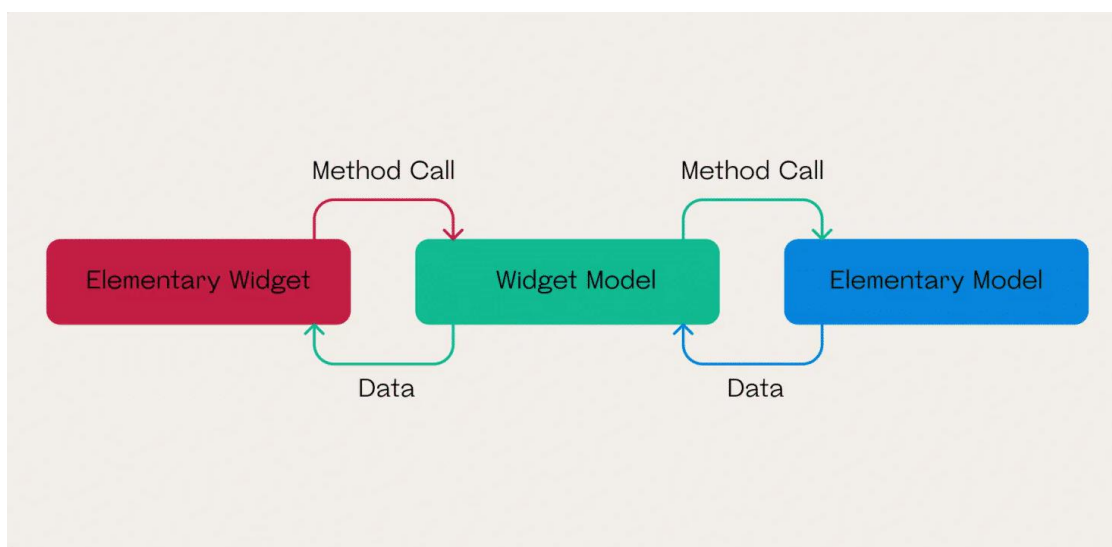


Рисунок 5 - Визуальное представление взаимодействия компонентов Elementary

ElementaryWidget — представление, в котором формируется структура экрана.

WidgetModel (WM) содержит презентационную логику.

ElementaryModel содержит бизнес-логику и логику компонента [5].

3.4 Диаграммы, иллюстрирующие работу системы

3.4.1 Диаграмма вариантов использования

На диаграммах вариантов использования отображается взаимодействие между вариантами использования, представляющими функции системы, и действующими лицами, представляющими людей или системы, получающие или передающие информацию в данную систему. Из диаграмм вариантов использования можно получить довольно много информации о системе. Этот тип диаграмм описывает общую функциональность системы [6].



Рисунок 6 - Диаграмма вариантов использования

На данной диаграмме присутствуют четыре актера: неавторизованный пользователь, авторизованный пользователь, модератор и администратор.

Неавторизованный пользователь – пользователь, не прошедший авторизацию или не зарегистрированный в системе. Данный пользователь может:

- Просматривать общую ленту;
- Авторизоваться;
- Зарегистрироваться.

Авторизованный пользователь – пользователь, прошедший авторизацию в системе. Авторизованный пользователь может:

- Просматривать общую ленту;
- Публиковать текстовые записи;
- Удалять записи;
- Скрывать записи из общей ленты;
- Публиковать временные записи;
- Ставить положительные реакции на публикации;
- Ставить отрицательные реакции на публикации;
- Просматривать свой профиль;
- Просматривать понравившиеся публикации;
- Просматривать не понравившиеся публикации;
- Редактировать профиль;
- Выйти из профиля.

Помимо функций, доступных авторизованному пользователю, модератор может:

- Удалять любые публикации из ленты.

Помимо функций, доступных авторизованному пользователю и модератору, администратор может:

- Назначить модератора;
- Убрать модератора.

3.4.2 Диаграмма классов

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования.

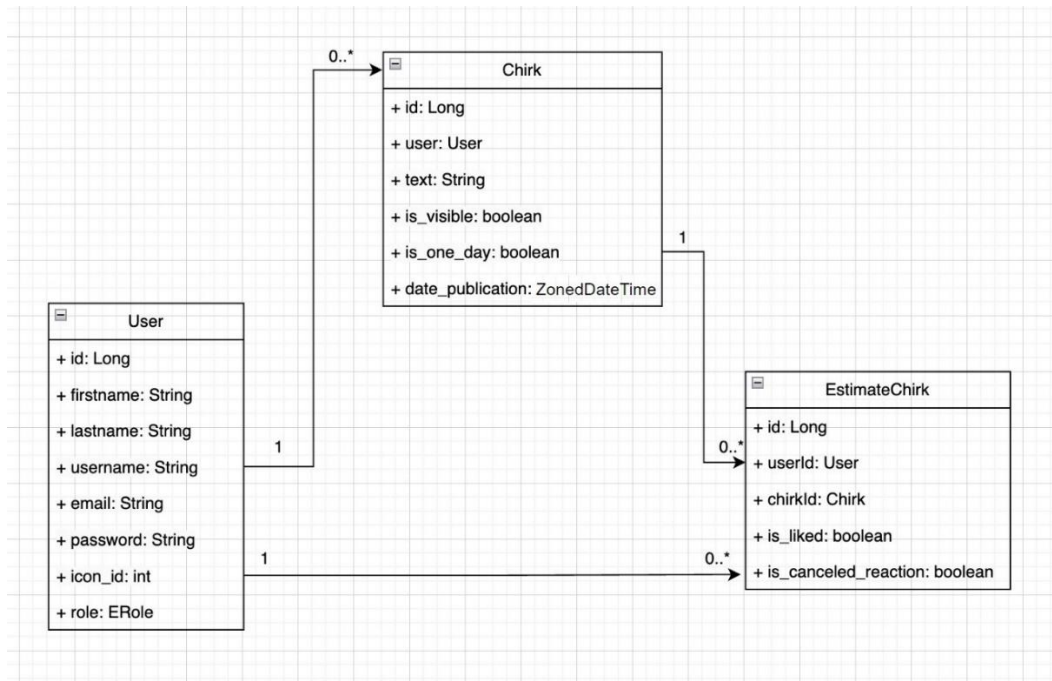


Рисунок 7 - Диаграмма классов

У класса User:

- id – уникальный идентификатор пользователя;
- firstname – имя пользователя;
- lastname – фамилия пользователя;
- username – уникальное имя пользователя;
- email – почта;
- password – пароль;
- icon_id – идентификатор;
- role – роль пользователя.

У класса Chirk:

- id – уникальный идентификатор публикации;
- user – пользователь, создавший публикацию;
- text – текст публикации;
- is_visible – состояние видимости публикации;

- `is_one_day` – переменная, показывающая, временная ли публикация;
- `date_publication` – дата публикации.

У класса `EstimateChirk`:

- `id` – уникальный идентификатор реакции;
- `userID` – пользователь, поставивший реакцию;
- `chirkID` – публикация, на которую была поставлена реакция;
- `is_liked` – реакция;
- `is_canceled_reaction` – переменная, показывающая, отменена ли реакция.

3.4.3 Диаграмма активностей

Диаграммы активностей позволяют не только представить процесс изменения состояний анализируемой системы, но и детализировать особенности алгоритмической и логической реализации выполняемых системой операций.

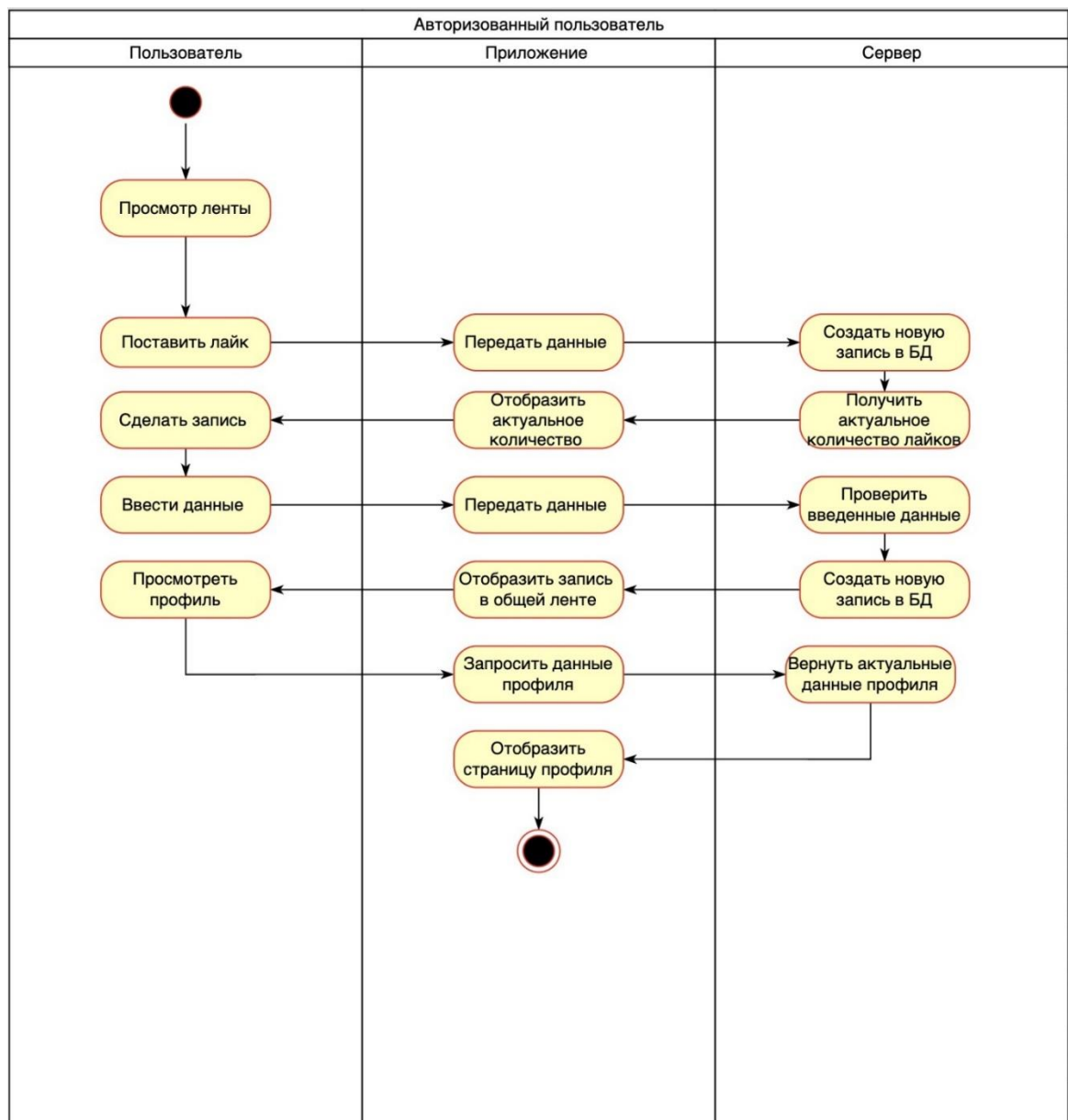


Рисунок 8 - Диаграмма активностей авторизованного пользователя

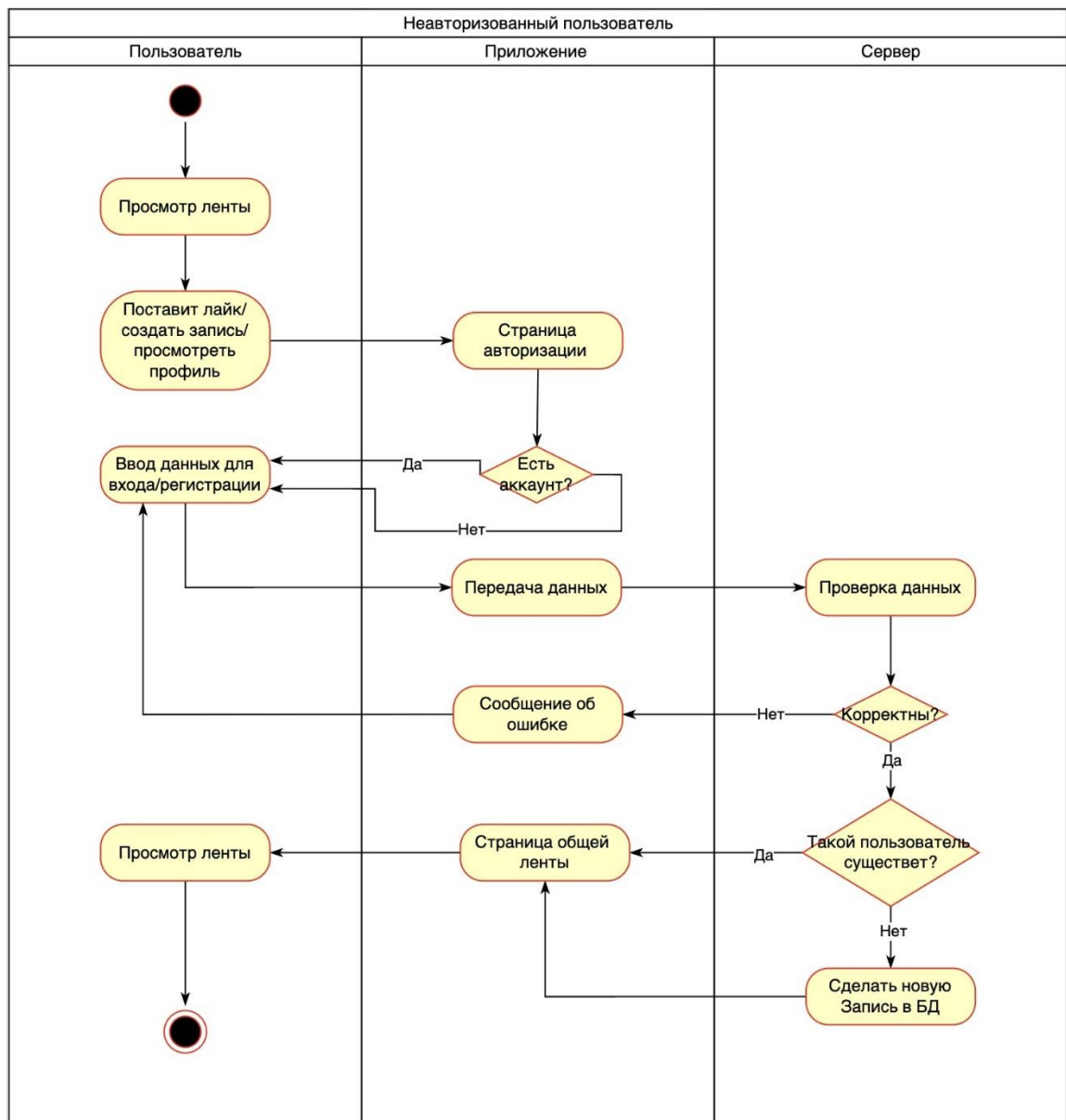


Рисунок 9 - Диаграмма активностей неавторизованного пользователя

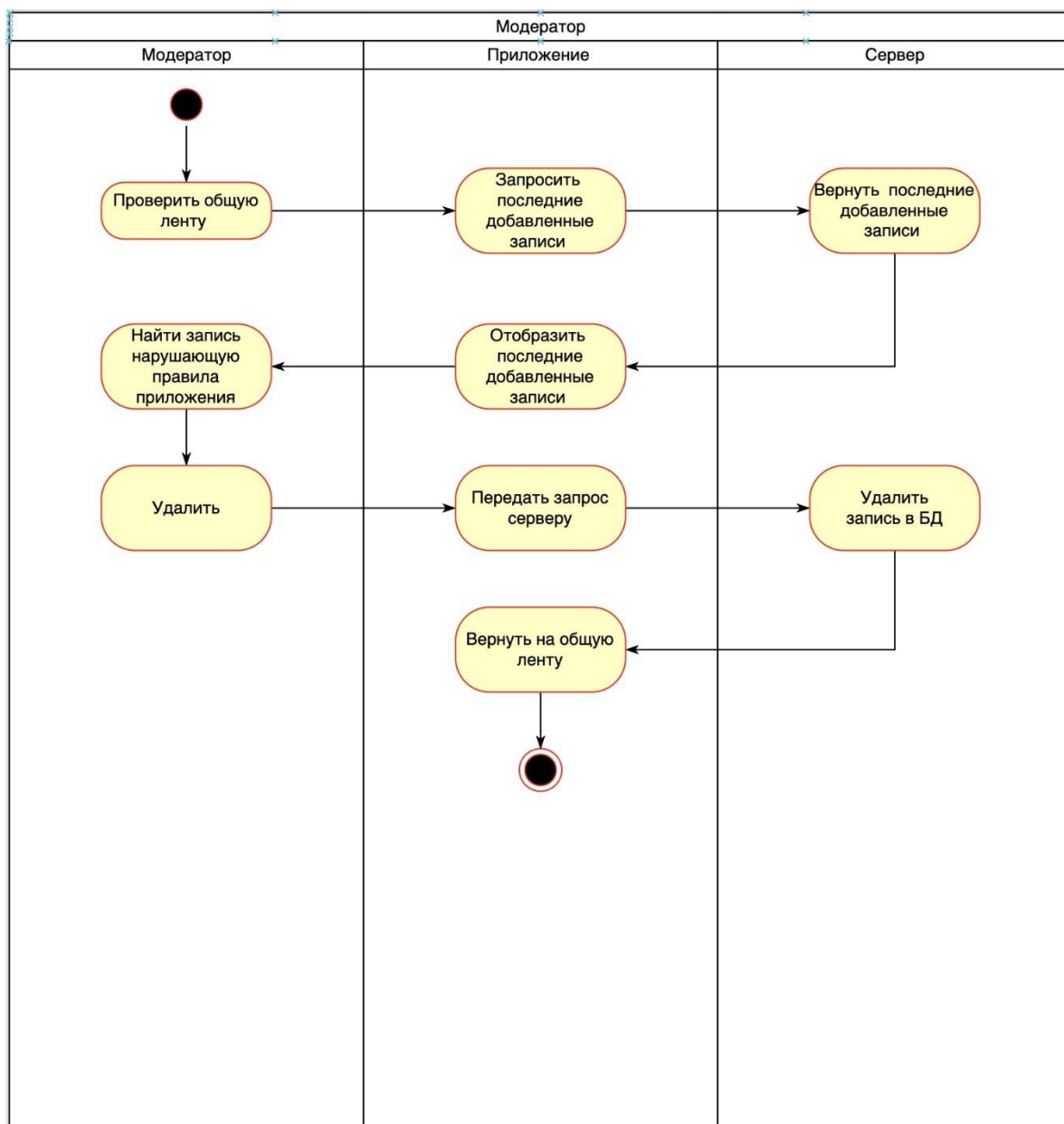


Рисунок 10 - Диаграмма активностей модератора

3.4.4 Диаграмма последовательности

Диаграммы последовательности отражают поток событий, происходящих в рамках варианта использования. На этих диаграммах изображаются только те объекты, которые непосредственно участвуют во взаимодействии т.к. ключевым моментом является именно динамика взаимодействия объектов во времени и не используются возможные статические ассоциации с другими объектами.

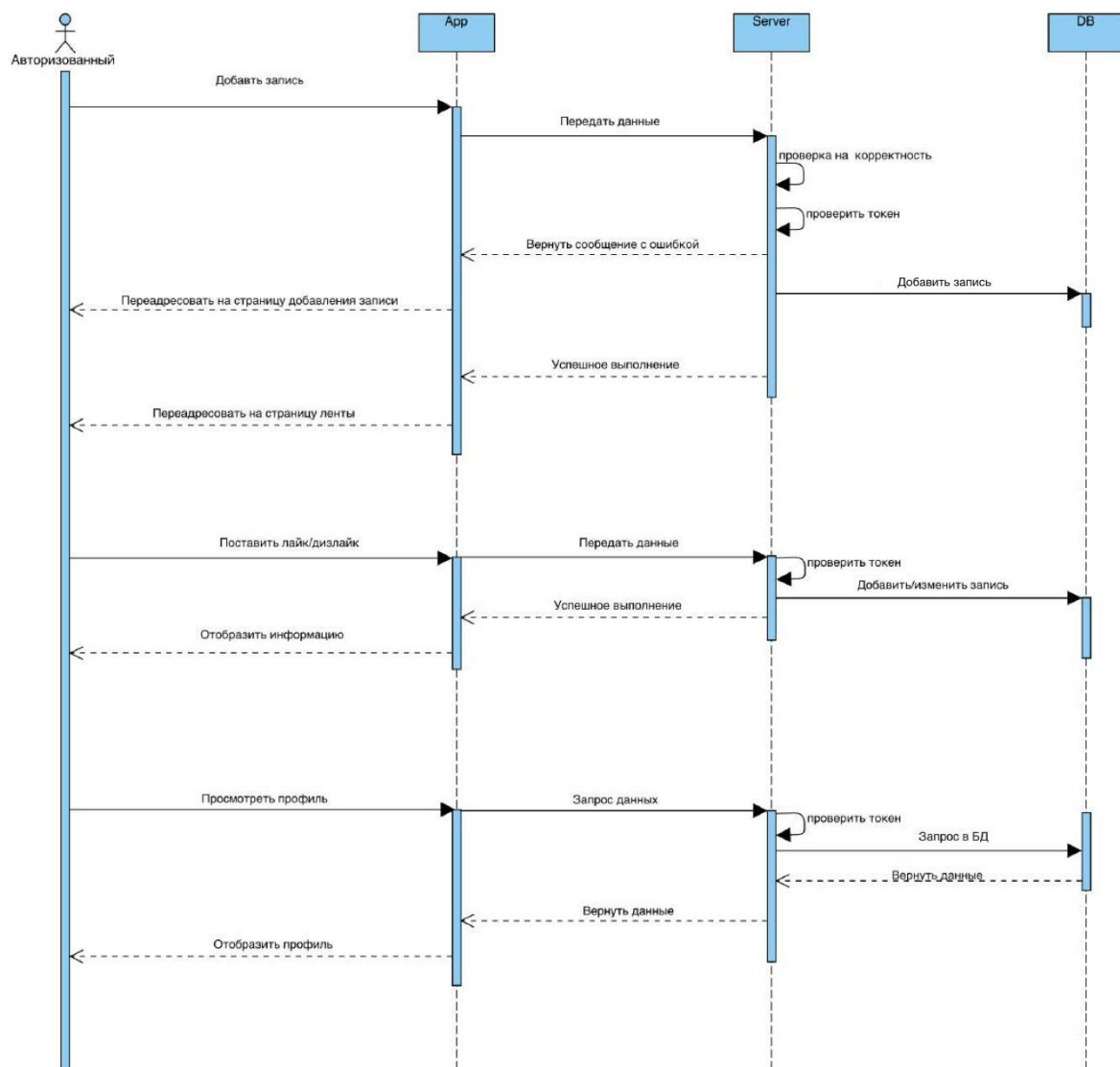


Рисунок 11 - Диаграмма последовательности авторизованного пользователя

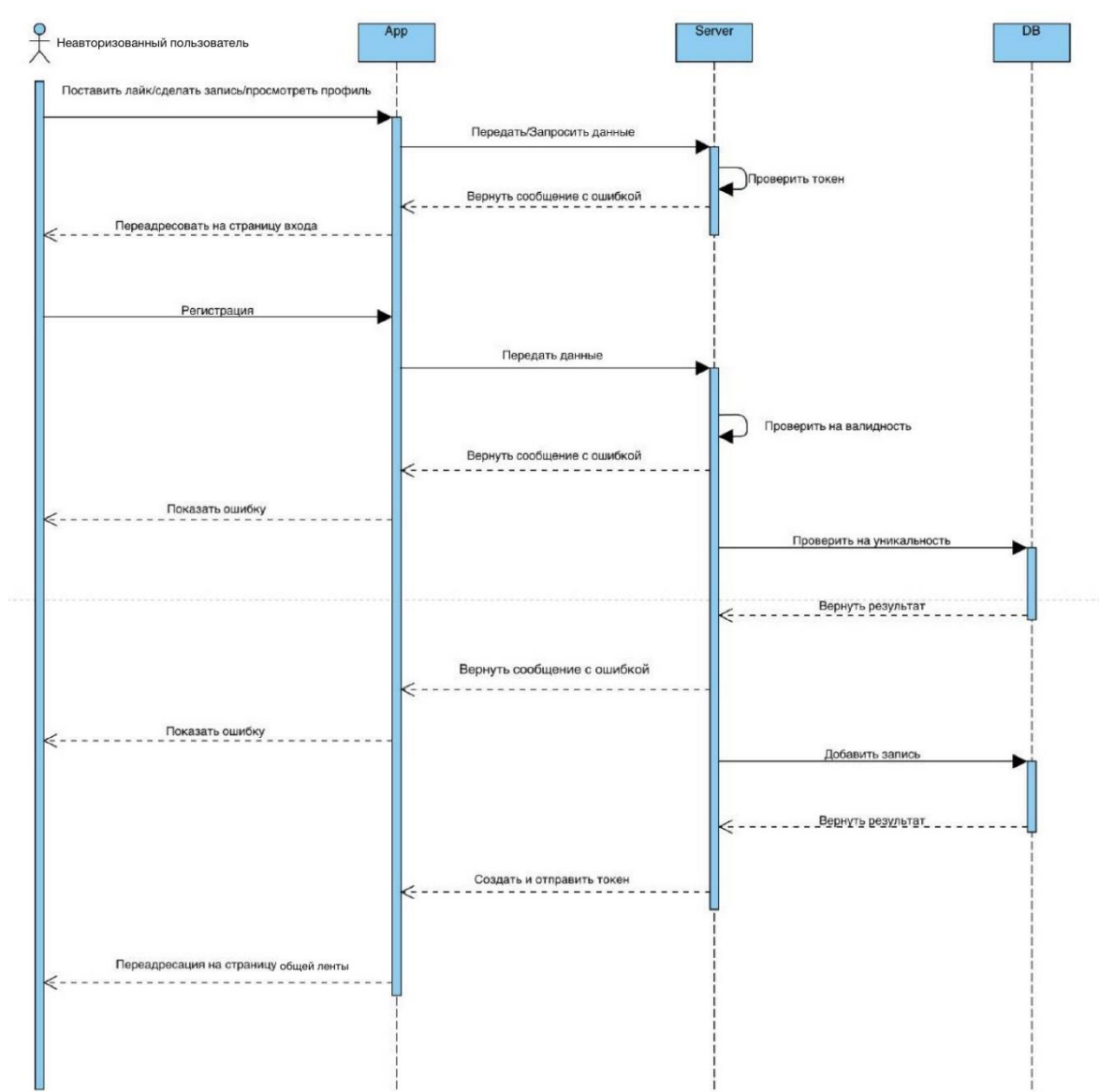


Рисунок 12 - Диаграмма последовательности неавторизованного пользователя

3.4.5 Диаграмма развертывания

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения. При этом представляются только компоненты-экземпляры программы, являющиеся исполнимыми файлами или динамическими библиотеками.

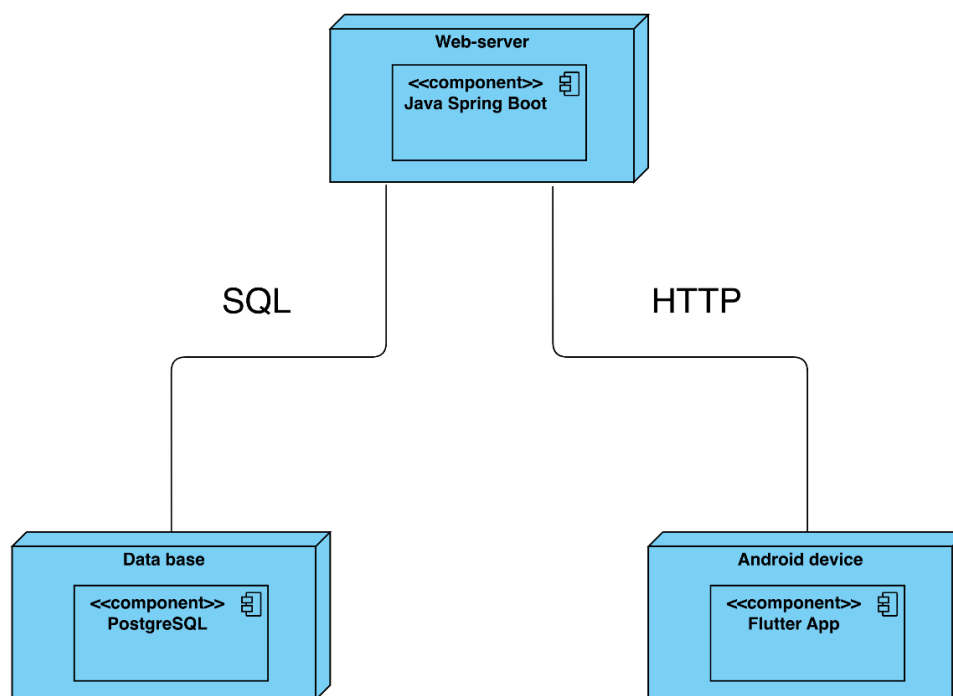


Рисунок 13 - Диаграмма развертывания

3.4.6 Диаграмма сотрудничества

Диаграммы сотрудничества отображают поток событий в конкретном сценарии варианта использования. Главная особенность диаграммы сотрудничества заключается в возможности графически представить не только последовательность взаимодействия, но и все структурные отношения между объектами, участвующими в этом взаимодействии.



Рисунок 14 - Диаграмма сотрудничества

3.4.7 Диаграмма объектов

Диаграмма объектов описывает особенности физического представления системы. Диаграмма объектов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Основными графическими элементами диаграммы объектов являются компоненты, интерфейсы и зависимости между ними.

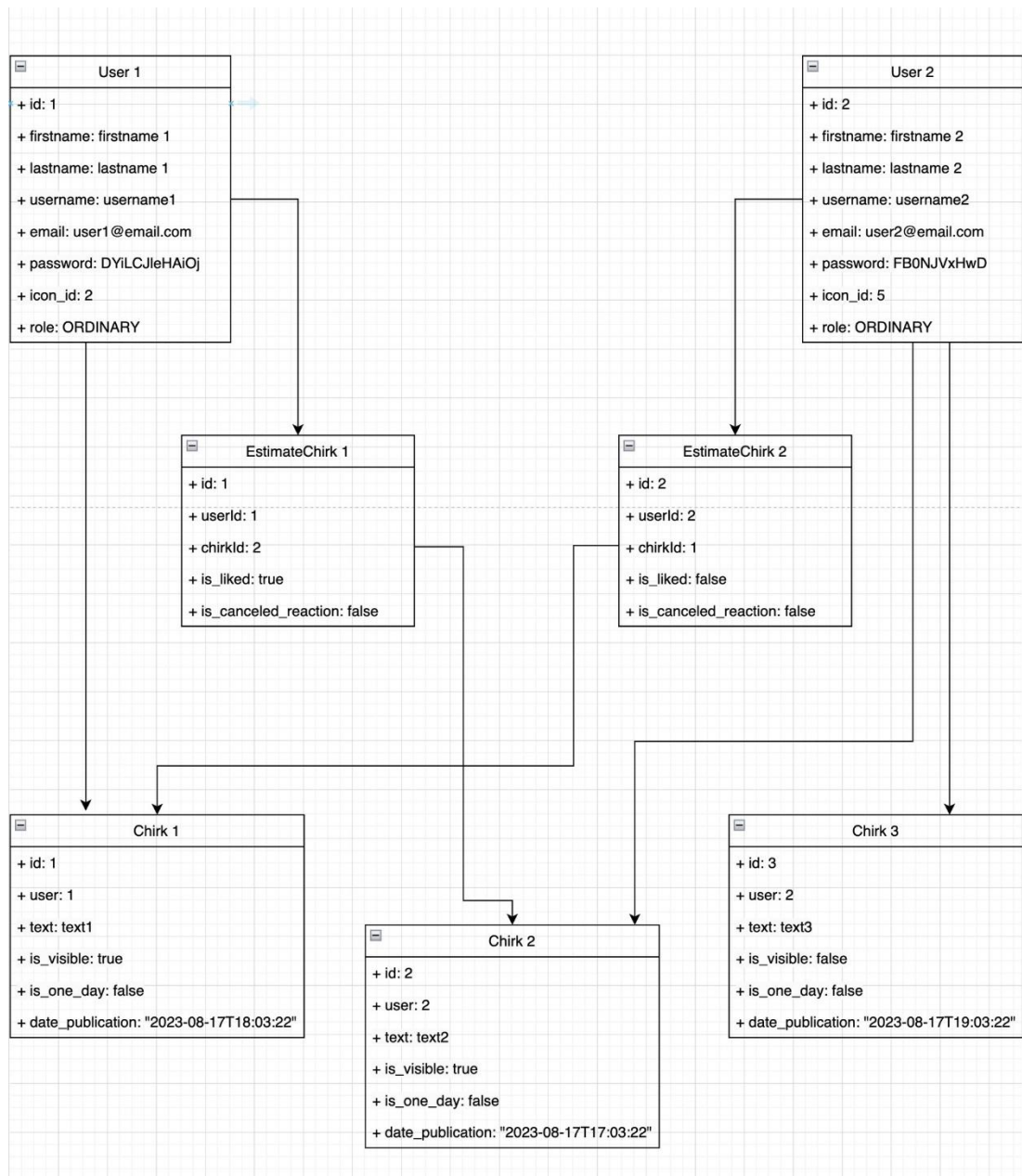


Рисунок 15 - Диаграмма объектов

4 Экраны приложения

4.1 Авторизация

Страница авторизации содержит поля, необходимые для аутентификации пользователя: email и пароль. После успешного входа, пользователь переходит на общую ленту публикаций. Возможен вариант перехода на страницу регистрации.

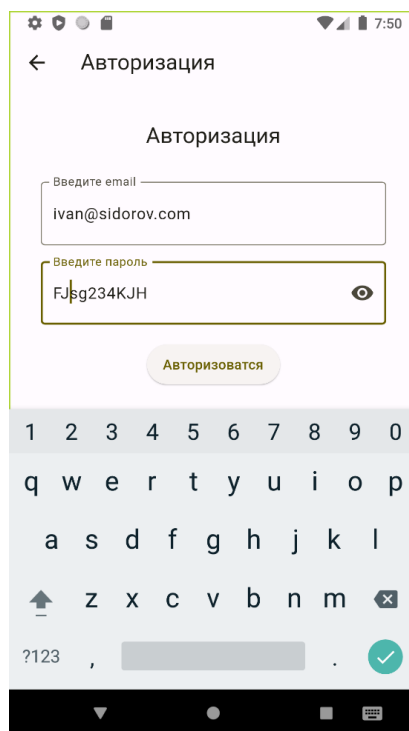


Рисунок 16 - Страница авторизации

4.2 Регистрация

Страница регистрации содержит поля, необходимые для инициализации пользователя: email, пароль, который для предотвращения ошибки вводится два раза, имя и фамилия. После успешной регистрации, пользователь переходит на общую ленту публикаций. Возможен вариант перехода на страницу авторизации.

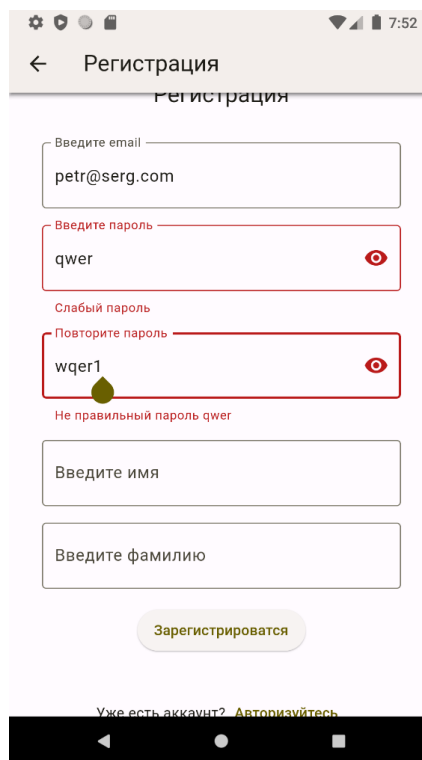


Рисунок 17 - Страница регистрации

4.3 Лента «Чирков»

Основную часть экрана занимают «Чирки», которые можно листать. В каждом «Чирке» имеется 2 кнопки реакции «Лайк» и «Дизлайк», которые не могут быть одновременно нажаты. В нижней части экрана расположен навигационный бар, который включает в себя 3 кнопки «Лента чирков», «Создать чиркнуть», «Профиль», которые перебрасывают на экраны «Лента «Чирков»», «Создание «Чирка»», «Профиль», соответственно. Данный навигационный бар также присутствует на экранах создания записи и профиля. При попытке неавторизованного пользователя поставить реакцию, появится окно с просьбой авторизоваться. У модератора дополнительно есть кнопка удаления на каждой публикации.

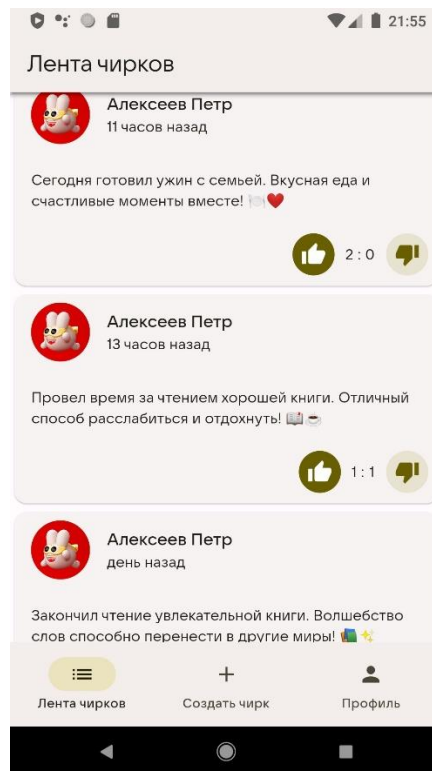


Рисунок 18 - Лента «Чирков»

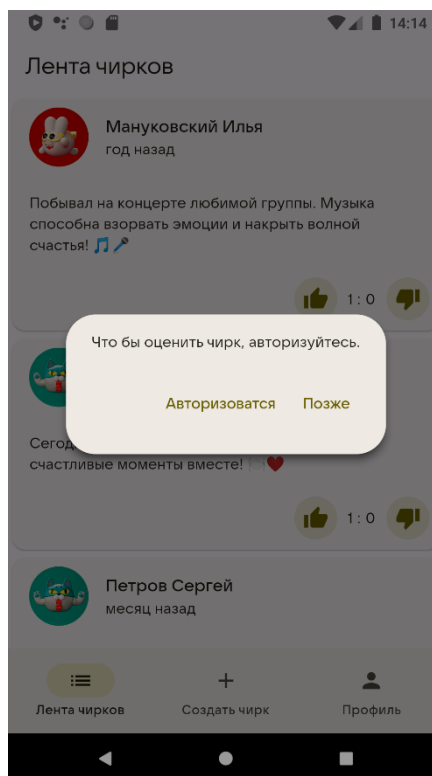


Рисунок 19 - Попытка неавторизованного пользователя поставить реакцию

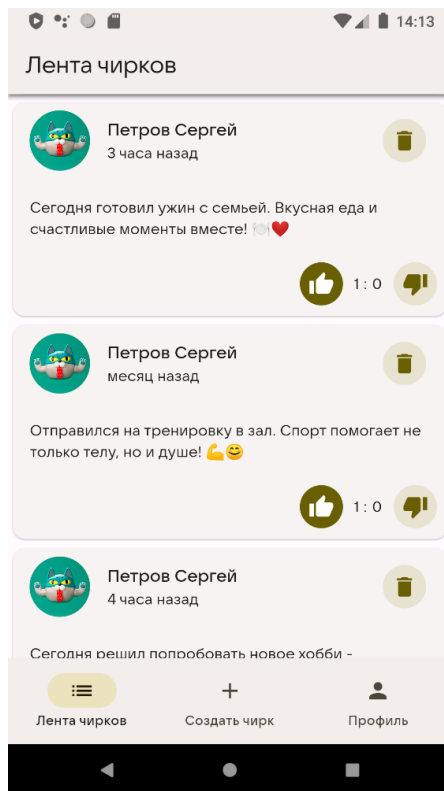


Рисунок 20 - Лента «Чирков» для модератора

4.4 Создание «Чирка»

На данном экране 1 поле для ввода, ниже флаг «Скрыть сообщение через 24 часа». Под ним расположена кнопка публикации «Опубликовать чирк». Неавторизованный пользователь видит только кнопки «Авторизуйтесь» и «Зарегистрируйтесь».

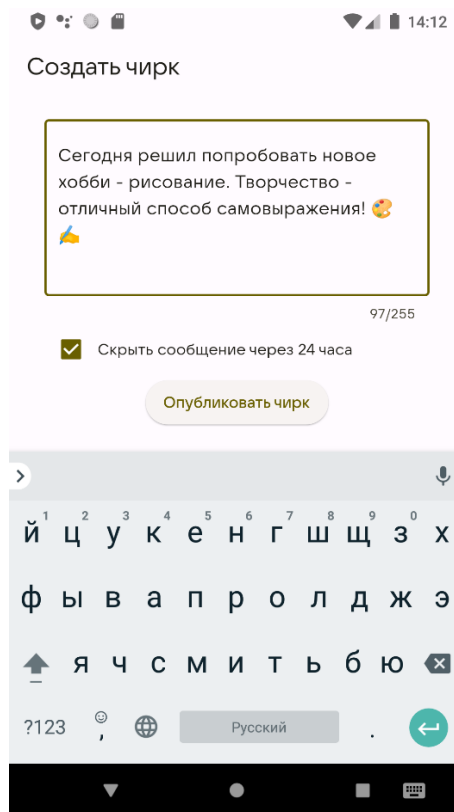


Рисунок 21 - Создание «Чирка»

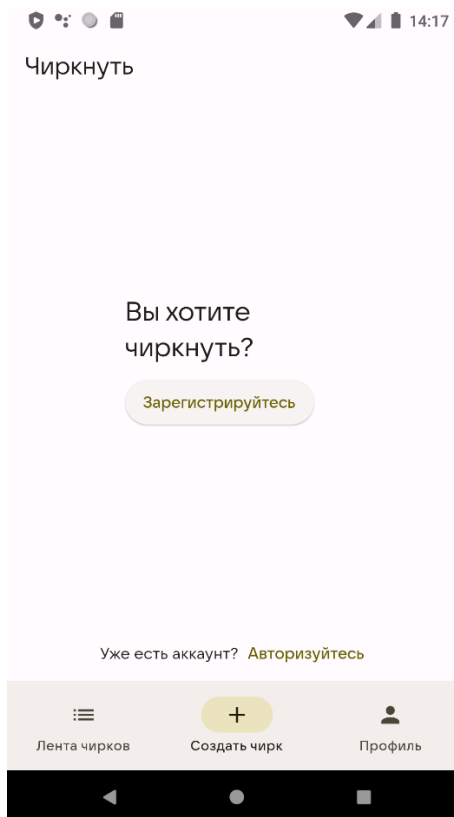


Рисунок 22 - Создание «Чирка» для неавторизованного пользователя

4.5 Профиль

На данном экране есть кнопка «Изменение профиля» справа от фамилии. Ниже находятся персональные данные пользователя. Еще ниже находятся 3 ссылки, которые перебрасывают на экран со своими «Чирками», с «Понравившимися» и с «Не понравившимися». Неавторизованный пользователь увидит только кнопки «Авторизации» и «Регистрации». В верхнем правом углу находится кнопка выхода из профиля.

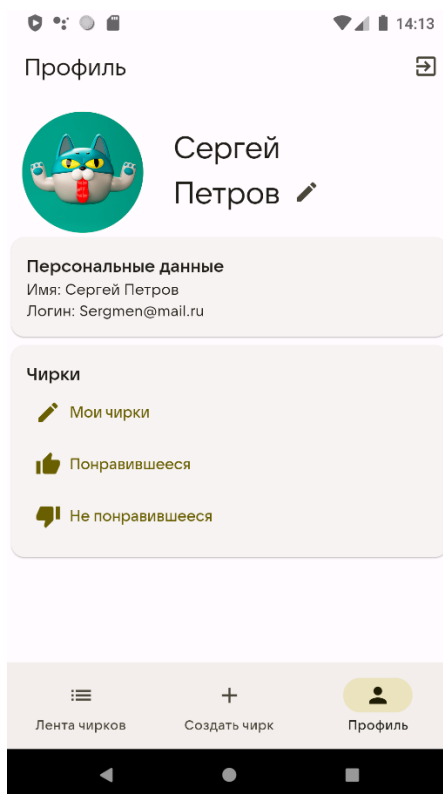


Рисунок 23 - Профиль

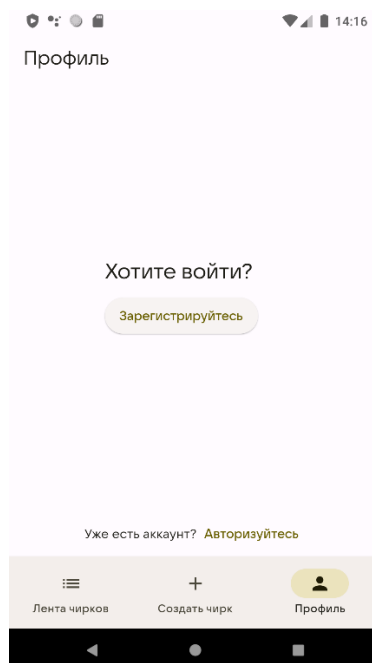


Рисунок 24 - Профиль для неавторизованного пользователя

4.6 Редактирование профиля

На данном экране есть кнопка «Сменить личные данные», нажав на которую, можно изменить свои имя и фамилию. Также присутствуют кнопки «Сменить пароль» для изменения пароля, «Автоматическая смена темы» и «Светлая/темная тема» для изменения темы интерфейса.

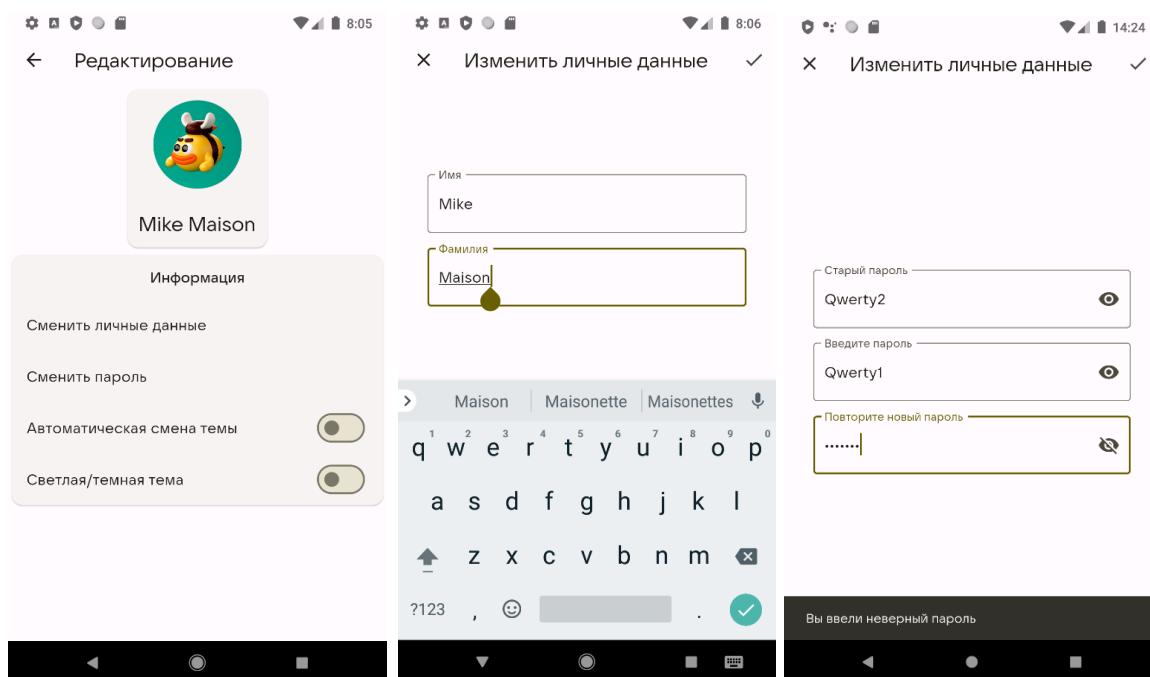


Рисунок 25 - Редактирование профиля

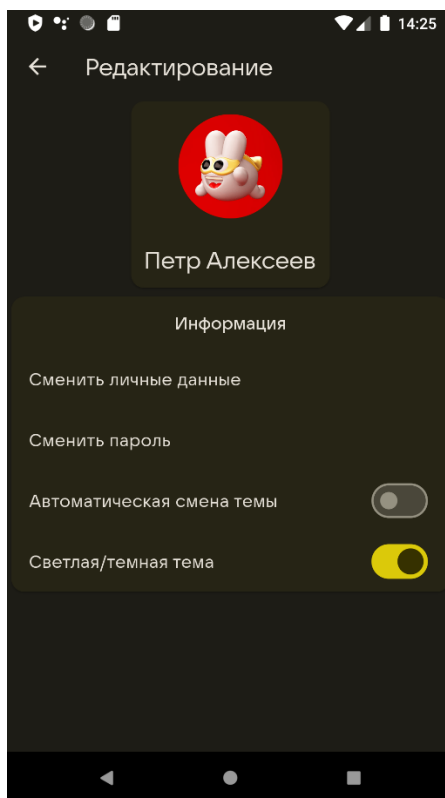


Рисунок 26 - Темная тема

4.7 Понравившееся и не понравившееся

На данных экранах пользователь может на каждом из «Чирков» убрать реакцию или поменять на противоположную соответствующими кнопками.

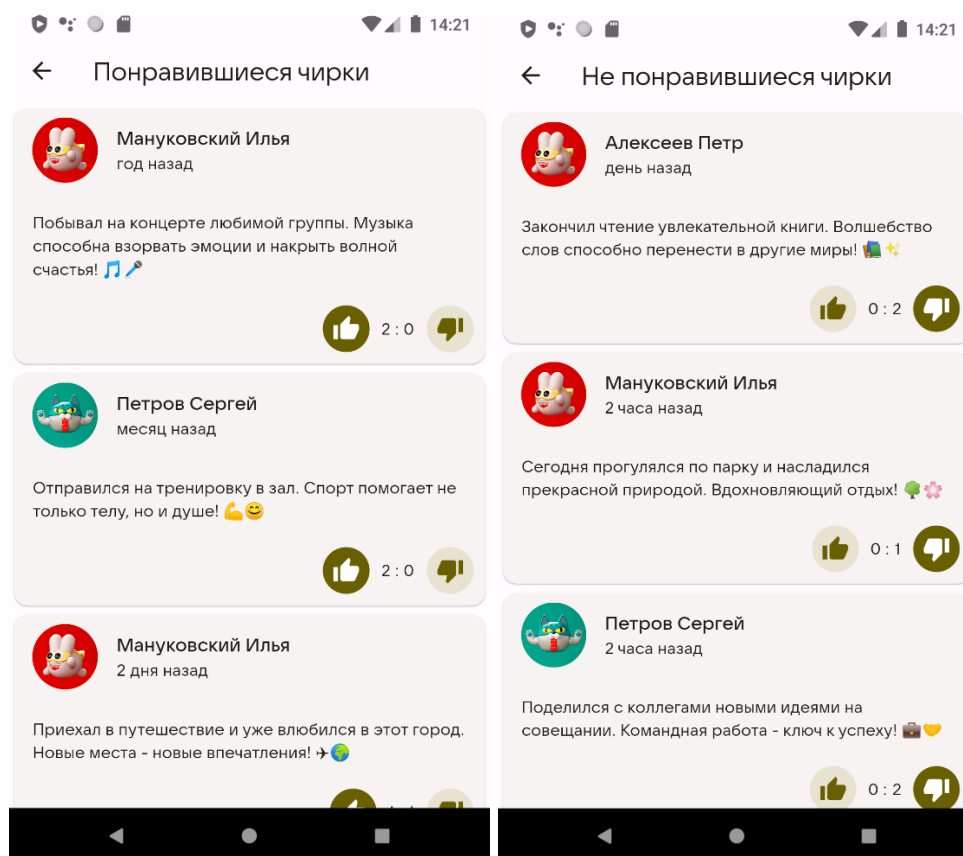


Рисунок 27 - Экраны понравившихся и не понравившихся «Чирков»

4.8 Онбординг

Данный экран содержит в себе 3 слайда, которые можно пропустить, нажав кнопку «Пропустить» в левом нижнем углу. Пользователь может листать слайды, предоставляющие краткую информацию о приложении, по кнопке «Далее». На последнем слайде есть кнопка «Зарегистрируйтесь» и «Начать», для перехода на общую ленту.

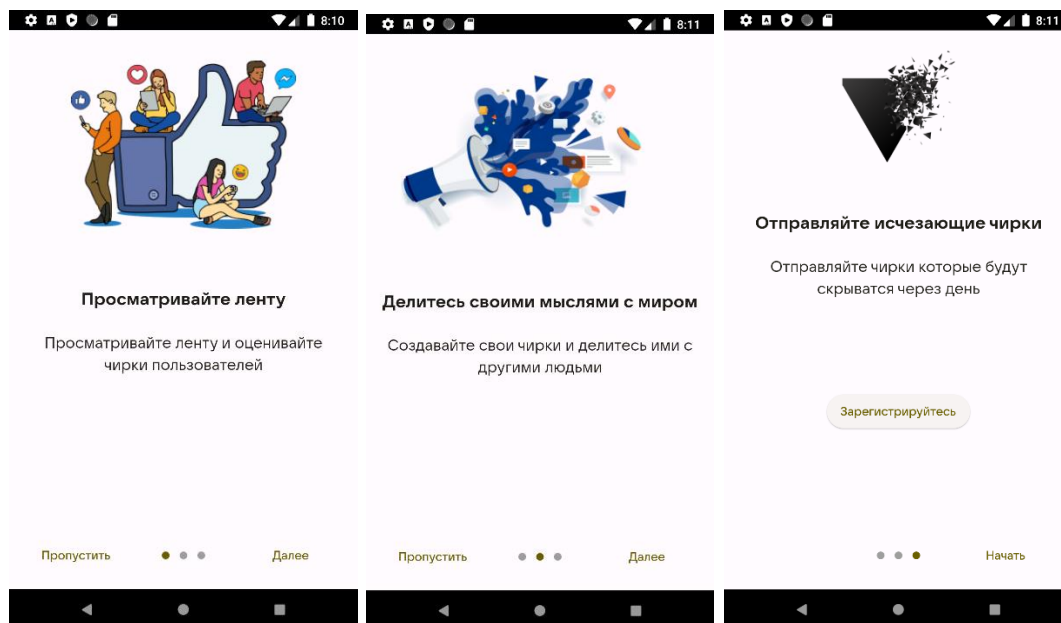


Рисунок 28 - Слайды с информацией о приложении

4.9 Лента созданных «Чирков»

На данном экране находятся «Чирки» пользователя. На каждой записи есть кнопки «Удалить» и «Скрыть/Отобразить». Скрытые из общей ленты записи отображаются более прозрачно.

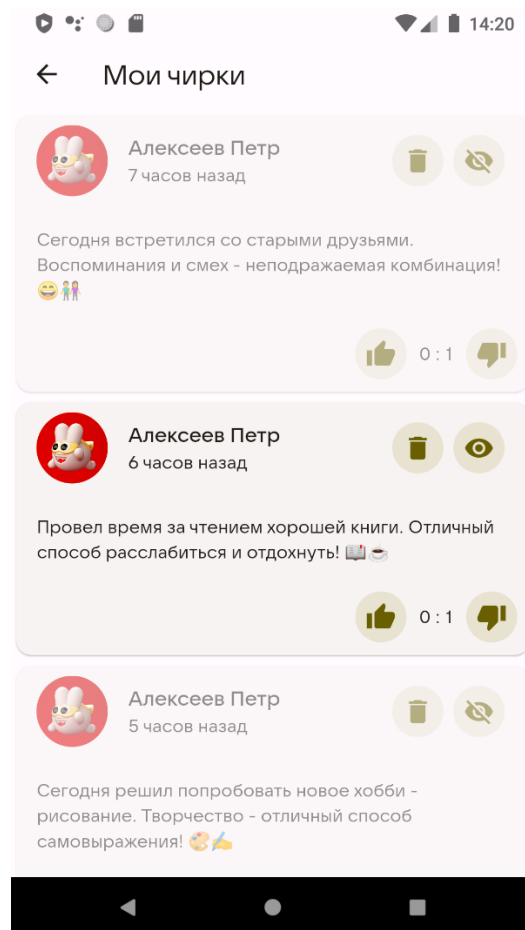


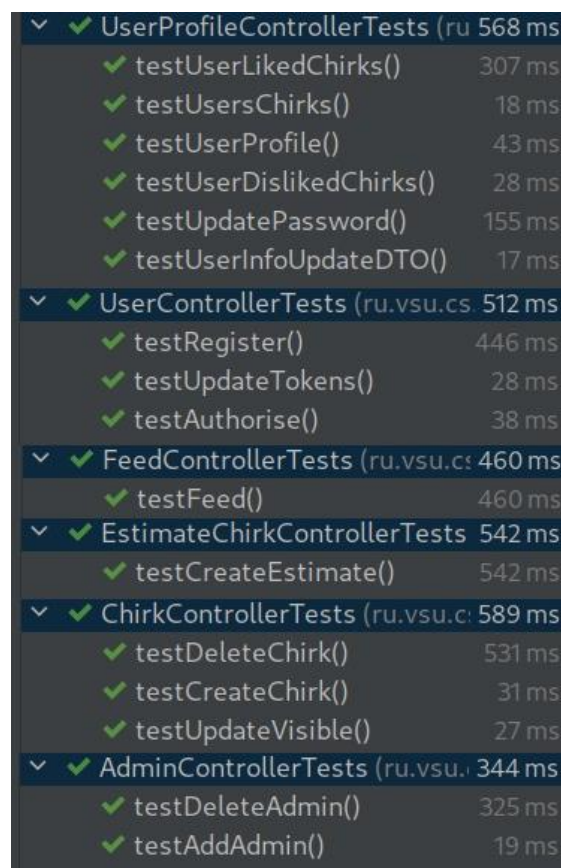
Рисунок 29 - Экран созданных «Чирков»

5 Тестирование

Unit-тесты являются неотъемлемой частью процесса разработки программного обеспечения и применяются для проверки работы отдельных модулей или компонентов программы. Unit-тесты разработаны для проверки функциональности отдельных элементов кода, таких как классы, методы или функции.

Unit-тесты направлены на то, чтобы убедиться, что каждая отдельная единица программного кода работает верно в отрыве от других компонентов системы. Unit-тесты проверяют, что методы возвращают ожидаемые результаты при заданных входных данных, а также правильно обрабатывают ошибочные ситуации.

Результаты тестирования серверной части приложения представлены на рисунке 30. В данном случае проверялась работа контроллеров, а unit-тесты были написаны в классе с аннотацией `@WebMvcTest`.



✓	UserProfileControllerTests (ru	568 ms
✓	testUserLikedChirks()	307 ms
✓	testUsersChirks()	18 ms
✓	testUserProfile()	43 ms
✓	testUserDislikedChirks()	28 ms
✓	testUpdatePassword()	155 ms
✓	testUserInfoUpdateDTO()	17 ms
✓	UserControllerTests (ru.vsu.cs	512 ms
✓	testRegister()	446 ms
✓	testUpdateTokens()	28 ms
✓	testAuthorise()	38 ms
✓	FeedControllerTests (ru.vsu.cs	460 ms
✓	testFeed()	460 ms
✓	EstimateChirkControllerTests	542 ms
✓	testCreateEstimate()	542 ms
✓	ChirkControllerTests (ru.vsu.c	589 ms
✓	testDeleteChirk()	531 ms
✓	testCreateChirk()	31 ms
✓	testUpdateVisible()	27 ms
✓	AdminControllerTests (ru.vsu.	344 ms
✓	testDeleteAdmin()	325 ms
✓	testAddAdmin()	19 ms

Рисунок 30 - Результаты unit-тестов

Заключение

Результатом работы является клиент-серверное мобильное приложение, поддерживаемое операционной системой android.

Система выполняет все поставленные задачи и обеспечивает возможность:

- Добавлять «Чирк» пользователем;
- Смотреть и реагировать на «Чирки» других пользователей;
- Удалять неприемлемые «Чирки» модераторами;
- Удалять свои «Чирки» пользователями;
- Скрывать свои «Чирки» пользователями;
- Создавать временные «Чирки» пользователями.

Список использованных источников

1. Документация SpringBoot [Электронный ресурс]. – Режим доступа: URL: <https://docs.spring.io/spring-boot/docs/current/reference/html/>– Заглавие с экрана. — (Дата обращения 21.03.2023).
2. Система управления объектно-реляционными базами данных PostgreSQL [Электронный ресурс]. — Режим доступа: <https://webcreator.ru/technologies/webdev/postgresql/> — Заглавие с экрана. — (Дата обращения: 3.04.2023).
3. Документация Flutter [Электронный ресурс]. – Режим доступа: URL: <https://docs.flutter.dev/>– Заглавие с экрана. — (Дата обращения 02.04.2023).
4. Шаблоны корпоративных приложений/ Мартин Фаулер. — М.: Диалектика, 2018. — 544 с.
5. Библиотека Elementary [Электронный ресурс]. – Режим доступа: URL: <https://surf.ru/cases/biblioteka-elementary/>– Заглавие с экрана. — (Дата обращения 02.04.2023).
6. Введение в UML от создателей языка/ Гради Буч, Джеймс Рамбо, Ивар Якобсон. — М.: ДМК Пресс, 2015. — 496 с.