**Автор: Паршин Даниил Денисович**

Название доклада: **Создание Android-приложения с использованием языка программирования Java**

**и облачной базой данных Firebase Database.**

Учреждение: МАОУ «Лицей 38»

Город: Нижний Новгород, Россия

Класс : 10

Секция: **Информационные технологии**

Оглавление

[Введение 3](#_Toc34854983)

[1. Анализ и аналитика будущего приложения. 5](#_Toc34854984)

[2. Программы и инструменты для работы с графикой 8](#_Toc34854985)

[3. Производительность …………………………………………………...9](#_Toc34854986)

[4. Оценки и пожелания пользователей 11](#_Toc34854987)

[5. Заключение. 12](#_Toc34854989)

[Список литературы. 13](#_Toc34854990)

Скриншоты окон приложения………………………………………………..14

# 

# Введение

Знание языков программирования сегодня востребовано как никогда. Многие из нас мечтают связать свою дальнейшую жизнь с IT технологиями и погрузиться с головой в данную индустрию. Мы можем перенести свои идеи и фантазии в код, написав программу, способную принести пользу. Разве это не интересно? Сначала будет нелегко: возникнет много вопросов и сомнений, ошибок и уязвимостей в коде. Но после усердной работы и терпения приятно получить результат – продукт, который работает по твоему собственному алгоритму.

В качестве проекта я выбрал разработку мессенджера, в основу которого была заложена концепция CRUD1[[1]](#footnote-2). Это даст мне бесценный опыт разработки и поддержания масштабного проекта. Конечный продукт позволит пользователям обмениваться информацией (сообщения и графические ресурсы) между собой при наличии wifi-соединения. Несмотря на то, что на сегодняшний день существует множество приложений, оснащенные тем же функционалом, что и мое приложение, у моего приложения есть преимущества: отсутствие рекламы и спама, приватность персональных данных, интуитивно понятный интерфейс.

**Объекты исследования:**

Объектами исследования являются: графический редактор AdobeXD, язык программирования Java, среда разработки Android Studio, XML 2[[2]](#footnote-3), Firebase3[[3]](#footnote-4).

**Задачи и цели:**

* Получить, прежде всего, опыт от работы в среде Android Studio и AdobeXD, написания Android-приложения, проявив свои умения как в back-end4[[4]](#footnote-5), так и в front-end5[[5]](#footnote-6) разработке.
* Проанализировать пожелания пользователя и максимально реализовать их в данном проекте.
* Постараться сделать интерфейс приложения минималистичным и понятным пользователю, добавить динамики элементам и переходам между активностями.
* Добиться максимальной производительности, реализовав автоматическую очистку кэша приложения и возможность изменения разрешения качества графических ресурсов (фотографии) для уменьшения их объема.
* Опробовать принципы ООП6[[6]](#footnote-7) на практике, сравнив их эффективность как в малых, так и в более масштабных проектах.
* Улучшить свои знания в таких языках, как: Java, XML. Получить опыт от проектирования и дизайнинга элементов интерфейса в AdobeXD, верстки анимаций, активностей и элементов при помощи XML.
* Научиться работать с официальной документацией, искать нужную информацию в разных источниках и на разных языках.

**При проведении исследования применялись следующие методы:**

* Изучение и сопоставление;
* Моделирование и измерение;
* Анализ и синтез.

**Значимость работы:**

Конечно же, как бы это не звучало банально, предоставить возможность общаться с другими пользователями приложения, отправлять фотографии и текстовые сообщения, редактировать профиль и восстанавливать пароль посредством отправки письма на почтовый ящик. Открытый код приложения позволит пользователю модифицировать код в соответствии со своими потребностями, так же может использоваться в качестве учебного пособия.

**В процессе разработки:**

* Спроектирован и создан мессенджер на базе Android.
* Получен опыт разработки приложений на языке Java и XML с помощью Android Studio и Firebase.
* Приобретен опыт использования редактора AdobeXD и создания интерфейсов с их дальнейшим внедрением в проект.

В результате получился легко масштабируемый проект — приложение, удовлетворяющее базовые потребностям пользователя, которое в дальнейшем будет расширять свой функционал, наращивать производительность и быстродействие. Все потому, что ООП дает возможность создавать расширяемые системы. Это одно из самых значительных достоинств ООП и именно оно отличает данный подход от традиционных методов программирования. Расширяемость означает, что существующую систему можно заставить работать с новыми компонентами, причем без внесения в нее каких-либо изменений. Компоненты могут быть добавлены на этапе выполнения.

# 1. Анализ и аналитика будущего приложения.

Передо мной стояла задача написать максимально функциональное приложение (насколько хватит времени на разработку и изучение), код которого был бы легко понятен для его дальнейшего поддержания. Сначала был разработан макет будущего приложения, блок-схема, которая отображала все активности и их взаимосвязи между друг другом.

В процессе работы над проектом выстроилась следующая логика приложения:

при запуске на экране появляется окно приветствия, то есть Splash Screen. На нем отображаются логотип приложения и некий слоган (или название), сопровождаемые анимацией. Далее мы попадаем на окно навигации, где пользователь выбирает дальнейшие действия для входа в систему — регистрация или аутентификация. Помимо этого, в приложение предусматривается возможность восстановления / сброса пароля, реализованной в отдельной активности — Password Reset. После успешного входа в систему пользователь переходит в основную активность — Main Activity, представленной в виде контейнера фрагментов, который заполняется тремя основными фрагментами: чаты, пользователи и профиль самого пользователя. От этого контейнера идет ответвление в настройки — Settings, где вносятся некие корректировки, а именно: язык системы, тема, пользовательские данные и т.д. Ну и последняя активность — Chat Activity, отвечающая за создание канала общения между двумя пользователями. Подробное описание схемы приведено на рис.1.

|  |
| --- |
|  |
| *Рис 1.* |

Все переходы должны сопровождаться анимациями, а цветовая палитра соответствовать правилам цветового круга — подобраны соседствующие оттенки (см. рис. 2). Было решено придерживаться васильковых (светло-синих) оттенков цвета, так как он умиротворяет и успокаивает, расслабляет и снимает напряжение. Именно по этой причине кресла в самолетах синего цвета. Несколько примеров приведено ниже (см. рис.3).

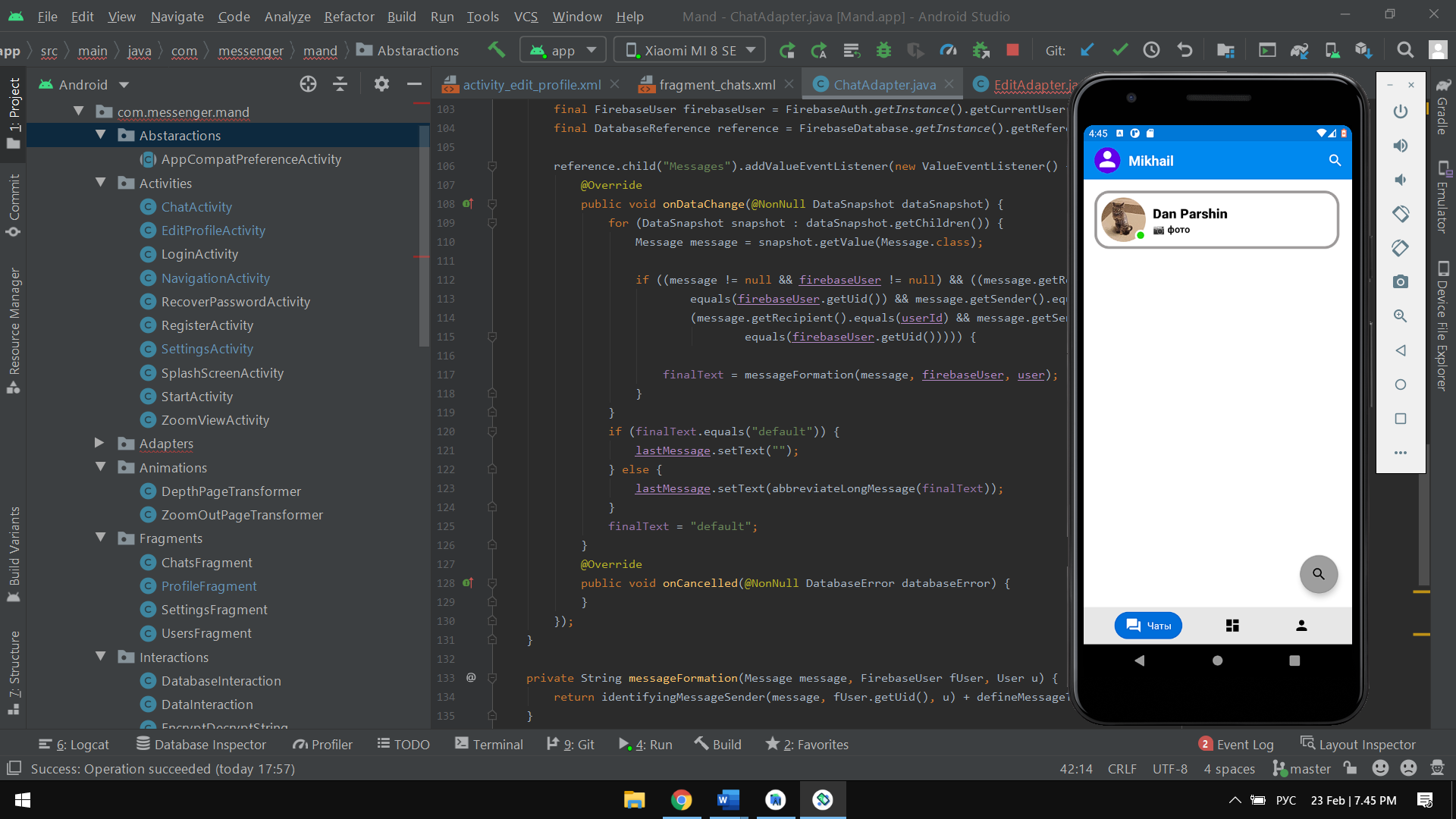
Был сделан акцент на (сдержанность) дизайна и интуитивность интерфейса. Использование в дизайне не острых углов у графических объектов, а скруглённых, позволило облегчить восприятие графики и обработку информации мозгом. Никаких сомнений, что они выглядят привлекательно, но эти дополнительные свойства делают их ещё более предпочтительными для использования в дизайне (пример приведен на рис.3).

|  |
| --- |
|  |
| *Рис 2.* |

|  |
| --- |
|  |
| *Рис 3.* |

# 2. Инструменты для разработки.

Основным инструментом являлась среда разработки — Android Studio. Ее преимущество заключается в богатом инструментарии: возможность взаимодействия с БД непосредственно из панели, интеграция Firebase, Git и GitHub, Profiler, тестирование на различных моделях устройств (эмуляторы) и многое другое. Большая часть работы с написанием дизайна активностей выполнялась здесь же. Дополнительным инструментом выступал Adobe XD — программа для разработки интерфейсов, поддерживающая векторную графику, веб-верстку, создание небольших активных прототипов.



Тестирование проводилось на устройствах с версией Android 9 и выше: виртуальное и физическое устройства. Среда разработки может похвастаться богатым функционалом и возможностями для различного спектра задач.

# 3. Производительность приложения и требуемые ресурсы.

Использование большого количества графических ресурсов высокого разрешения и данных (текст) влечет за собой нагрузку на устройство, а именно: увеличивается объем используемой памяти, замедляется скорость работы приложения, увеличивается энергопотребление. Это, конечно же, влияет на выбор, который пользователь делает в пользу того или иного продукта, снижается конкурентоспособность созданного приложения на рынке. Следовательно, каждому разработчику стоит обращать внимание на производительность, энергопотребление. Поэтому, передо мной стояла задача максимально освободить неиспользуемые данные приложения. Уменьшить размер приложения поможет автоматическая очистка кэша приложения и компрессия изображений, уменьшение их разрешения. Ниже демонстрируется работа приложения в течение 3 минут (рис. 6):

|  |
| --- |
|  |
| *Рис. 6* |

Мы видим, что контейнер не переполняется выше 250МБ, то есть не происходит утечка памяти. Метод очистки кэша успевает очищать ненужные данные. Инструмент Profiler позволяет с легкостью контролировать каждое действие, выполненное программой. Память устройства и способы ее грамотного использования являются важными этапами в разработке, особенно в ресурсоемких приложениях. С целью экономии памяти и увеличения быстродействия, было принято решение уменьшить разрешение изображений и перенести большую часть данных в базу данных.

Но, несмотря на это, у данной реализации есть свои недостатки. Так как данные пользователя (сообщения и фотографии) хранятся непосредственно в базе данных, то с увеличением их объема будет увеличиваться и нагрузка сервера. Это не есть хорошо, так как если речь пойдет не 100 пользователях, а о 1000 (или более), где у каждого из них сотни строк сообщений и фотографий, то сервер просто ляжет. Самым банальным выходом является грамотное проектирование базы данных и хранение в ней информации (желательно как можно меньше). Ещё хорошей практикой на сегодняшний день является хранение всех пользовательских данных непосредственно на самом устройстве (внутренняя / защищённая память). Это позволит обезопасить личные данные и ключи шифрования от злоумышленников. Подобная практика применяется в таком мессенджере, как Viber. Но и у нее есть свои недостатки: память устройства будет увеличиваться с повышением объема данных (фотографий, сообщений и т. д.), доходя до 1 Гб и более. Дополнительно можно будет реализовать получение / передачу публичного ключа через сервер по HTTP другому собеседнику(ам). Но так как я был ограничен как в ресурсах, так и во времени, то данная реализация отложена на будущее.

# 

# 4. Оценки и пожелания пользователей

Для изучения мнения пользователей о предоставленном приложении, был проведен опрос. Количество респондентов, опрошенных по теме составило 35 человек (наша фокус-группа). Опросник состоял из двух частей: нас интересовало отношение пользователя к технической составляющей мессенджера, дизайн приложения, комфорт взаимодействия. Этот опрос необходим для того, чтобы выявить сильные и слабые стороны, проанализировать ошибки и постараться их устранить. Результаты опроса представлены диаграммами (рис.8,9)

|  |
| --- |
|  |
| *Рис.8* |

Из данных опроса видно, что приложение, в целом, понравилась, мои друзья и знакомые с удовольствием переписывались между собой. Но многие попросили реализовать групповой чат, дополнительные способы регистрации, новостную ленту. Все пожелания и просьбы я обязательно учту и реализую в следующем обновлении приложения.

В дальнейшем я буду улучшать приложение, добавляя новые функции, более продуманный способ взаимодействия между пользователем и данными, дополнять бизнес-логику новыми идеями и предложениями, поступающих от пользователей.

# 5. Заключение.

В результате работы над проектом, было создано приложение мессенджер, которое имеет интересный дизайн, соответствует большей части пользовательских ожиданий. Также были решены задачи, поставленные перед выполнением проекта:

1. Изучены основы разработки Android приложения, освоена среда разработки Android Studio.

2. Получены навыки применения языков программирования Java и XML на практике.

3. Проанализированы пожелания пользователей и реализованы важные из них.

4. Освоены принципы ООП и их применение на практике.

5. Получен опыт верстки дизайна и базовых анимаций.

6. Изучена архитектура Android приложений и Firebase.

В дальнейшем планируется расширить набор функций, предоставляемых приложением по мере их разработки, например, реализовать возможность создания групповых чатов, отправки сообщений по таймингу, импортировать проект в web, повысить безопасность пользователей, оптимизировать процесс сжатия ресурсов, внедрить многопоточность (позволит разгрузить основной поток, ускорить загрузку данных пользователя и т.п.).

# Список литературы.

1. Официальная документация по Android разработке: [https://developer.android.com](https://developer.android.com/)
2. Официальная документация по Material design (стиль графического дизайна интерфейсов программного обеспечения и приложений): <https://material.io/design>
3. Система вопросов и ответов о программировании по любой теме:

[https://stackoverflow.com](https://stackoverflow.com/)

1. Официальная страница Firebase с полной документацией:

<https://firebase.google.com/docs>

1. Библиотека анимаций для различных платформ разработки:

<https://airbnb.design/lottie>

1. Русскоязычный веб-сайт в формате системы тематических коллективных блогов с элементами новостного сайта:

[https://habr.com/ru/flows/develop]( https://habr.com/ru/flows/develop)

1. Канал зарубежного разработчика:

<https://www.youtube.com/c/TVACStudio/videos>

1. Учебник по Android для начинающих и продвинутых:

<https://startandroid.ru/ru>

**Скриншоты окон приложения.**

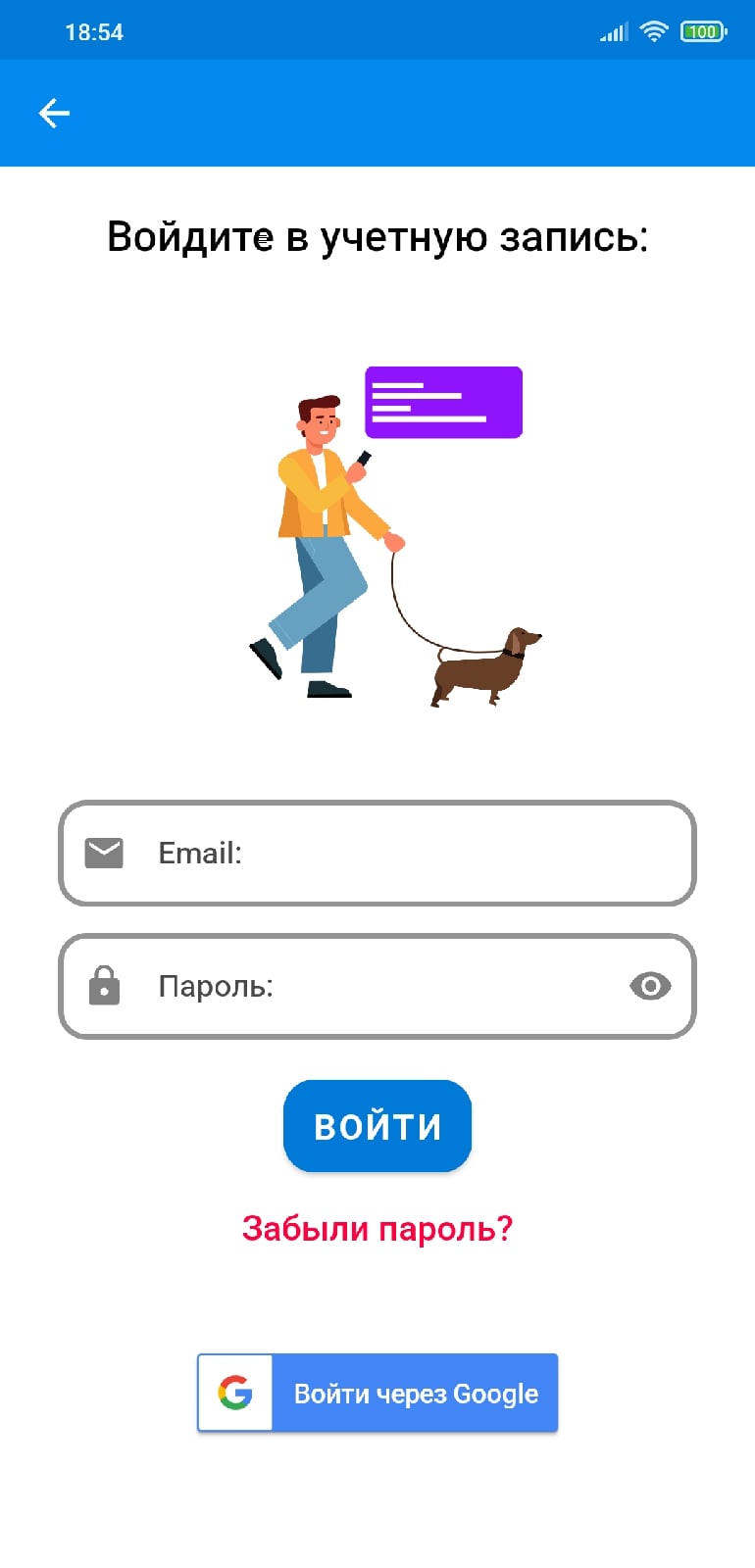


Рис.11

Рисунок 1: окно входа

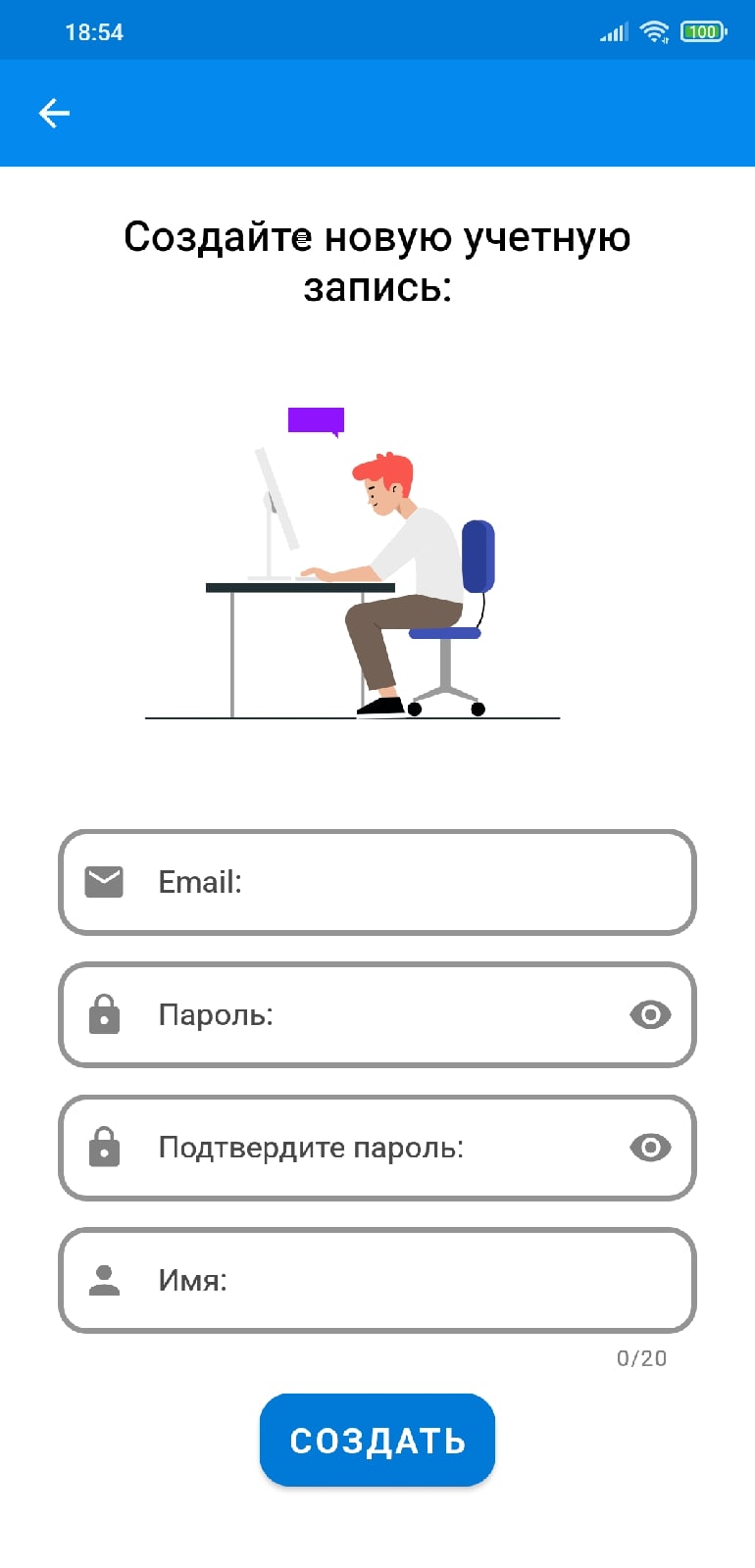


Рис.10

Рисунок 1: окно входа

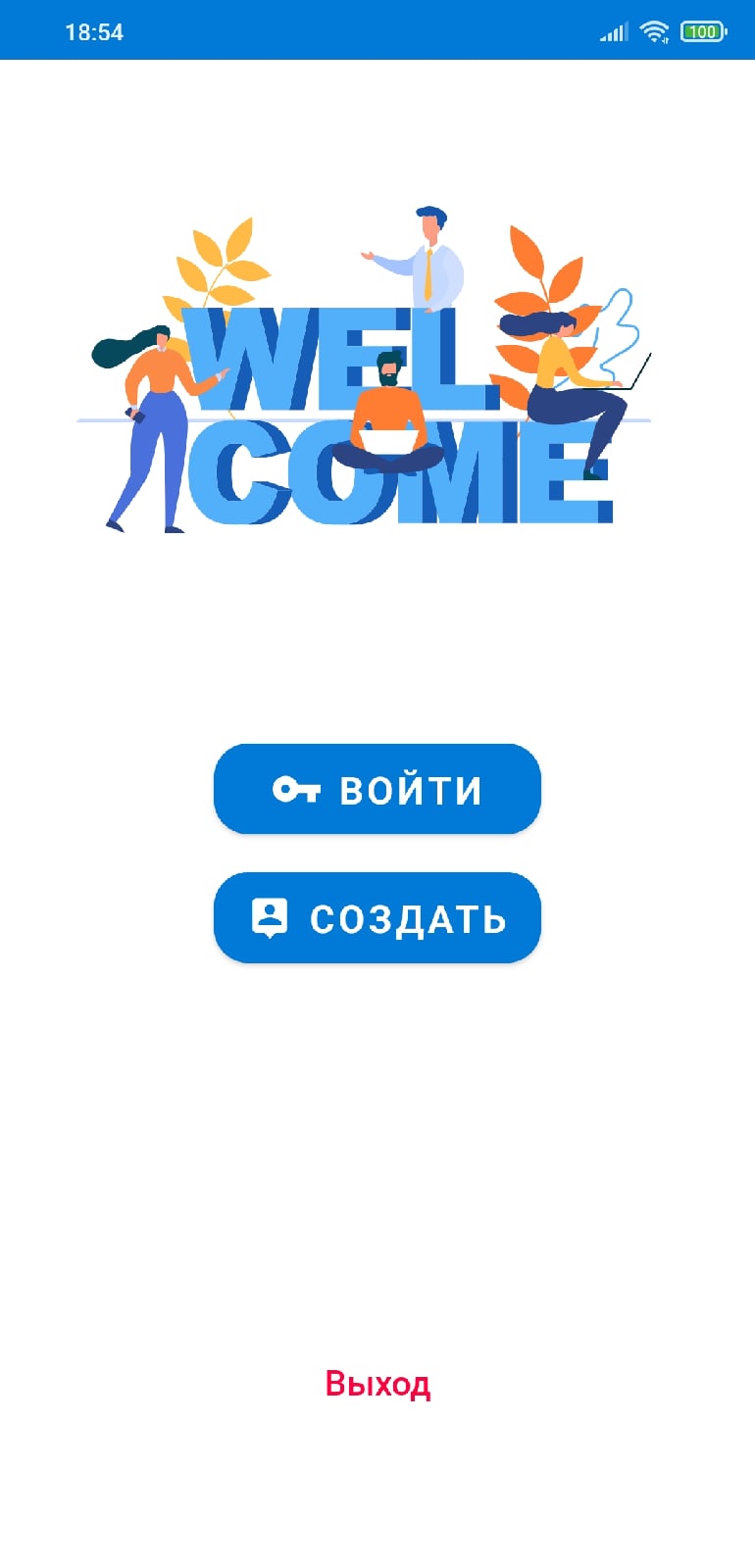


Рис.9

Рисунок 1: окно входа

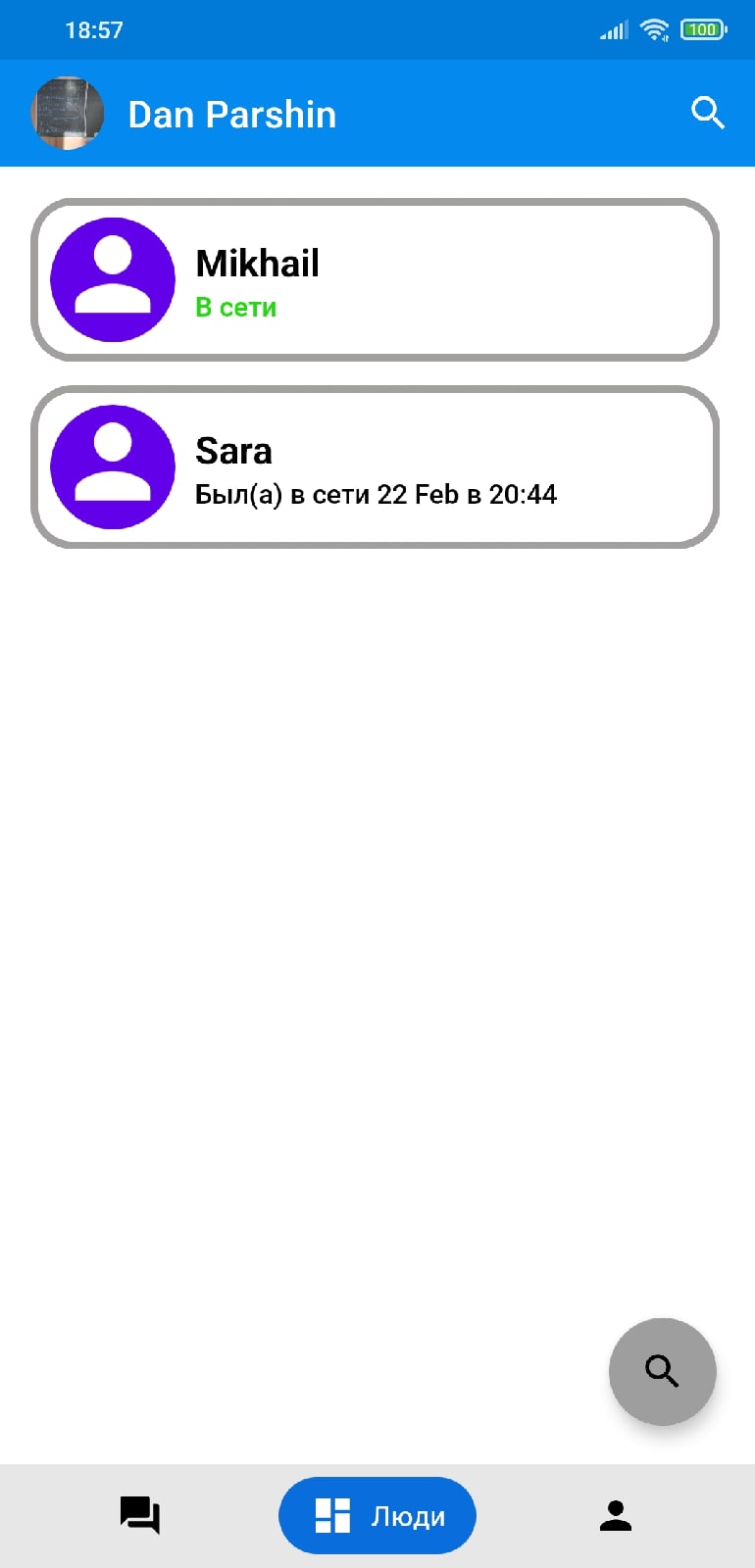


Рис.14

Рисунок 1: окно входа

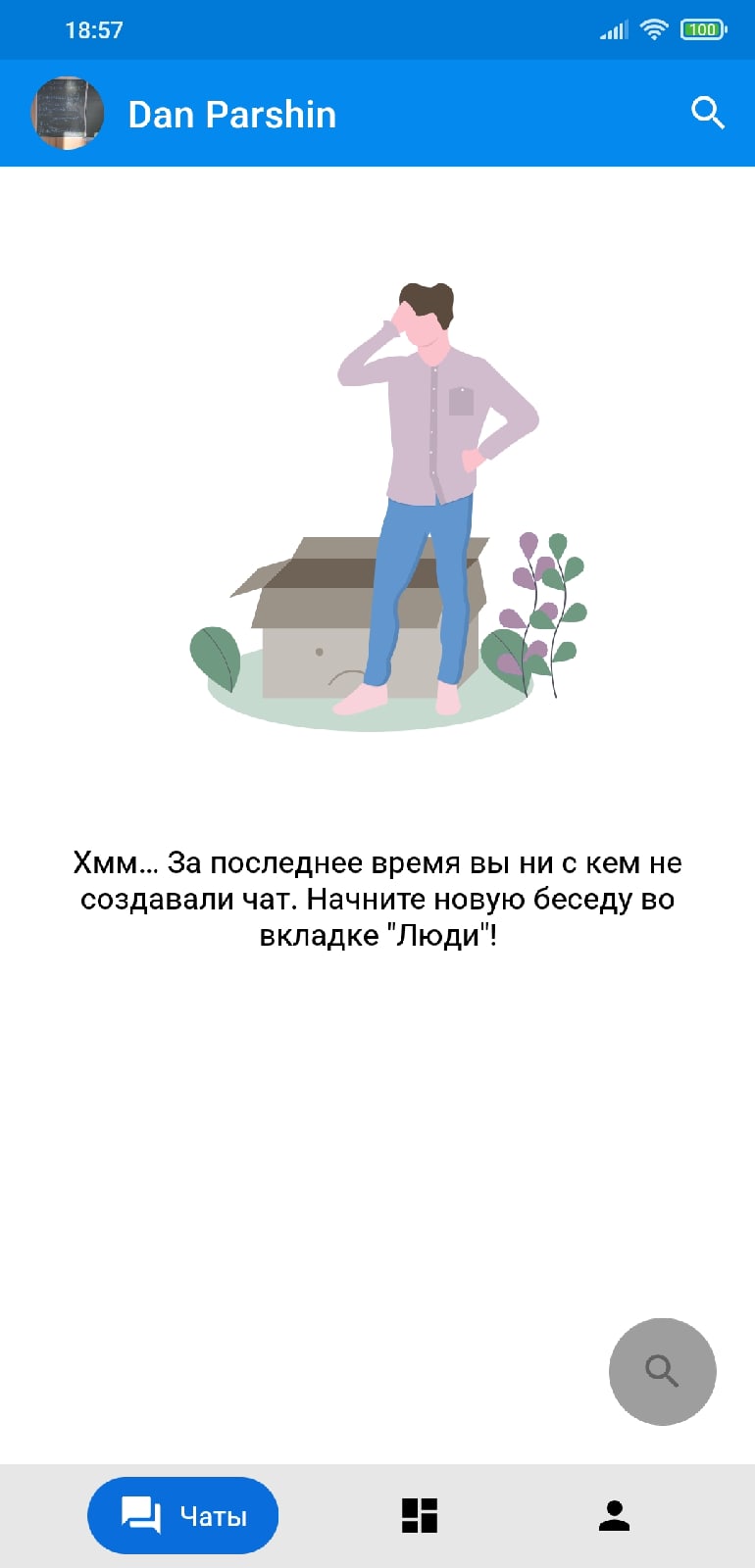


Рис.13

Рисунок 1: окно входа

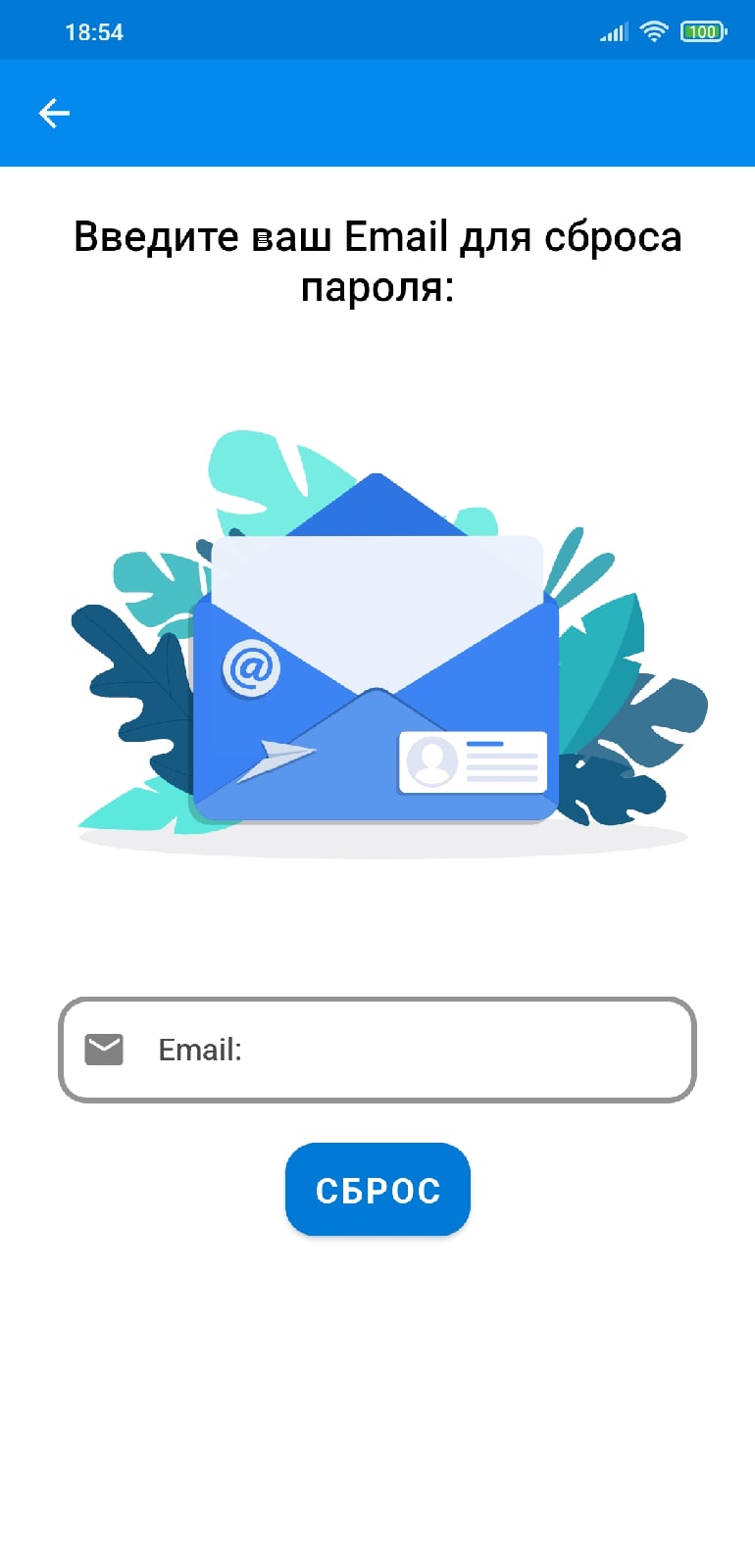


Рис.12

Рисунок 1: окно входа

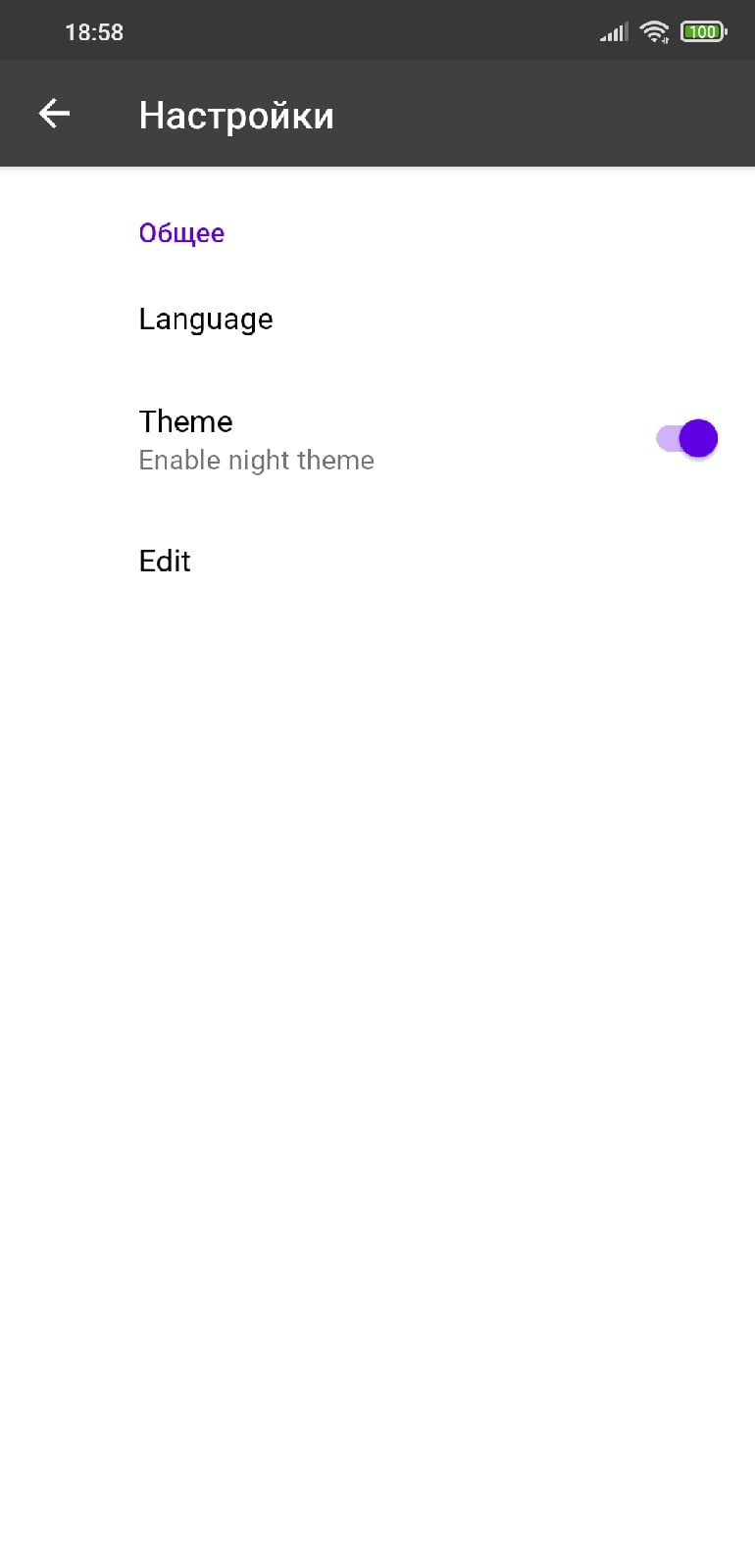


Рис.17

Рисунок 1: окно входа

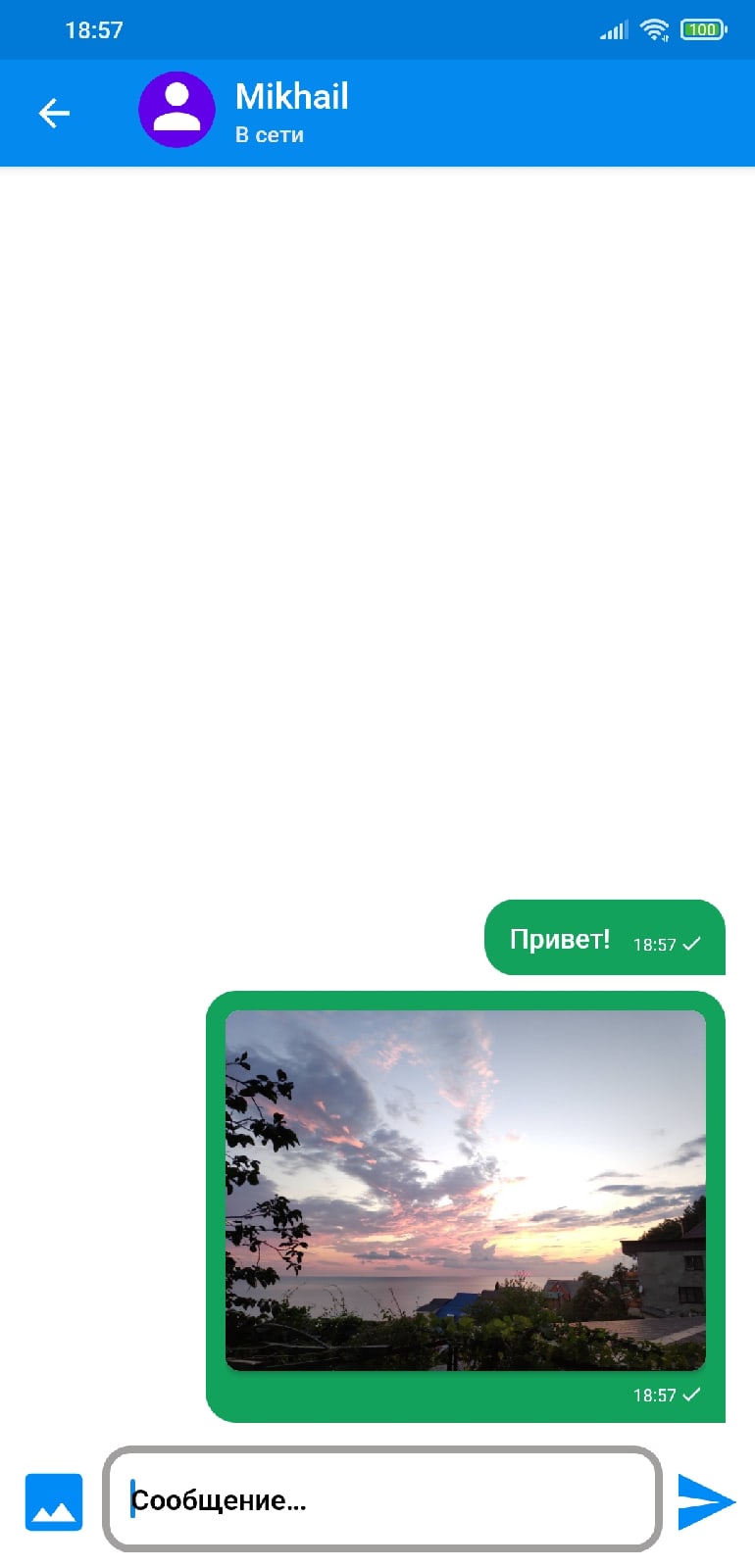


Рис.16

Рисунок 1: окно входа

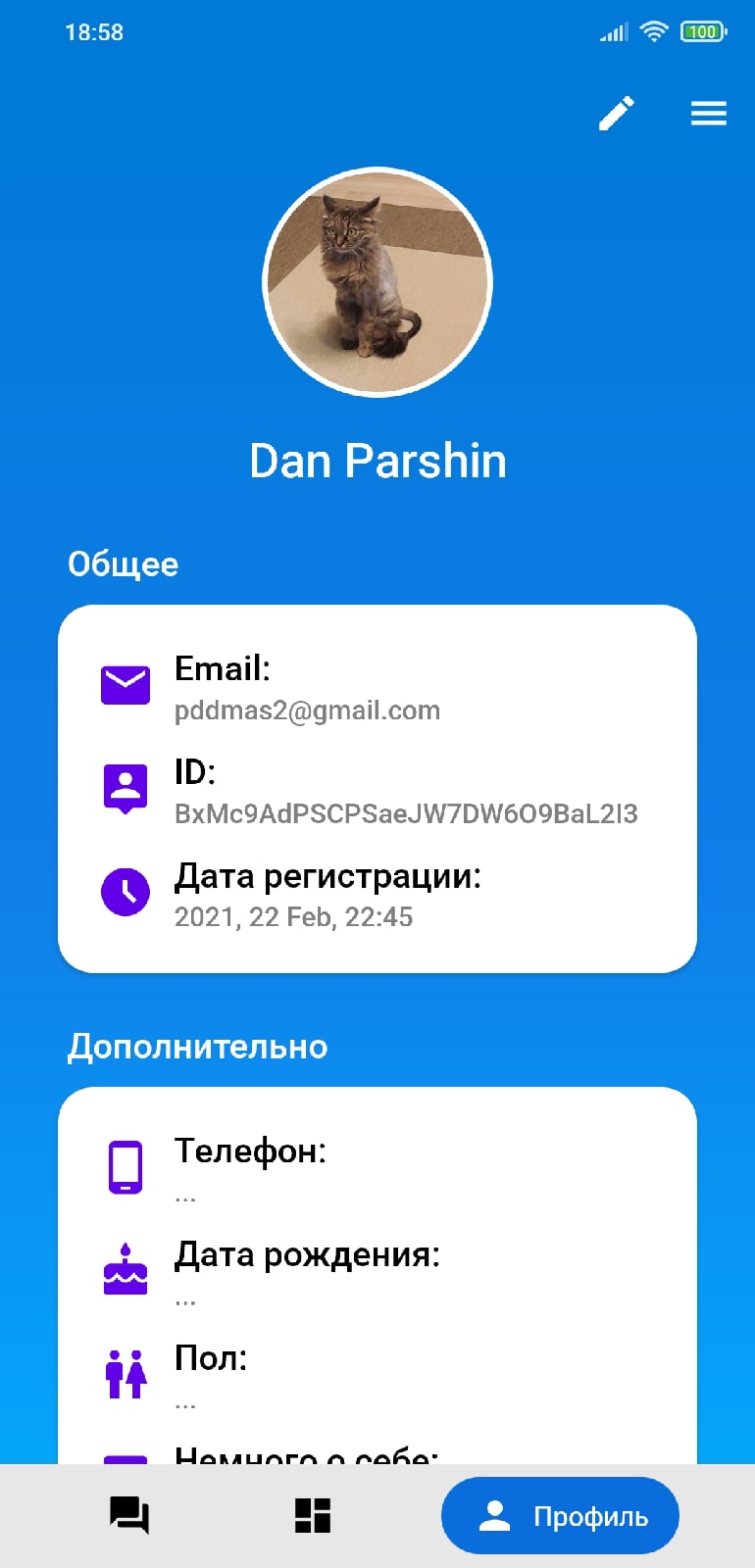


Рис.15

Рисунок 1: окно входа

1. акроним, обозначающий четыре базовые функции, используемые при работе с базами данных: создание (англ. create), чтение (read), модификация (update), удаление (delete). [↑](#footnote-ref-2)
2. Расширяемый язык разметки. [↑](#footnote-ref-3)
3. Американская компания, поставщик облачных услуг, основанная в 2011 году Эндрю Ли и Джеймсом Тэмплином, и поглощённая в 2014 году корпорацией Google. [↑](#footnote-ref-4)
4. Программно-аппаратная часть сервиса. [↑](#footnote-ref-5)
5. Клиентская сторона пользовательского интерфейса к программно-аппаратной части сервиса. [↑](#footnote-ref-6)
6. Объектно-ориентированное программирование — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования. [↑](#footnote-ref-7)