**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Механико-математический факультет

Кафедра веб-технологии и компьютерного моделирования

**Бобровко Валерий Иванович**

**МОБИЛЬНОЕ ПРИЛОЖЕНИЕ - ЧАТ**

Курсовая работа

студента III курса

Руководитель

**Дедков Даниил Юрьевич**

Минск 2017

ОГЛАВЛЕНИЕ

**ВВЕДЕНИЕ**

**1. Основные сведения о Android OS**

**2. Разработка мобильного приложения Чат**

**2.1 Используемые средства**

**2.2 Сервер**

**2.2.1 Библиотека Socket.IO**

**2.2.2 Создание сервера для мобильного приложения Чат**

**2.3 Клиент**

**3. Заключение**

**4. Список использованной литературы**

**Введение**

На сегодняшний день в сети Интернет используется огромное количество средств связи для общения: мессенджеры, IP-телефония, видеозвонки. Сейчас их существует большое количество. Самые популярные из них: Telegram, Skype, Discord, Hangouts, RaidCall, Viber, WhatsApp.

***Мессенджеры*** - службы мгновенных сообщений (Instant Messaging Service, IMS) для обмена сообщениями в реальном времени через Интернет. Могут передаваться текстовые сообщения, звуковые сигналы, изображения, видео, а также производиться такие действия, как совместное рисование или игры. Многие из таких программ-клиентов могут применяться для организации групповых текстовых чатов или видеоконференций.

Мессенджеры существуют для различных операционных систем: для десктопных - Windows, Linux и мобильных - Android, IOS, Windows Phone. Именно мобильные приложения для обмена сообщениями на сегодняшний день имеют наибольшую популярность. Наиболее популярная мобильная система в мире – Android (Рис. 1)

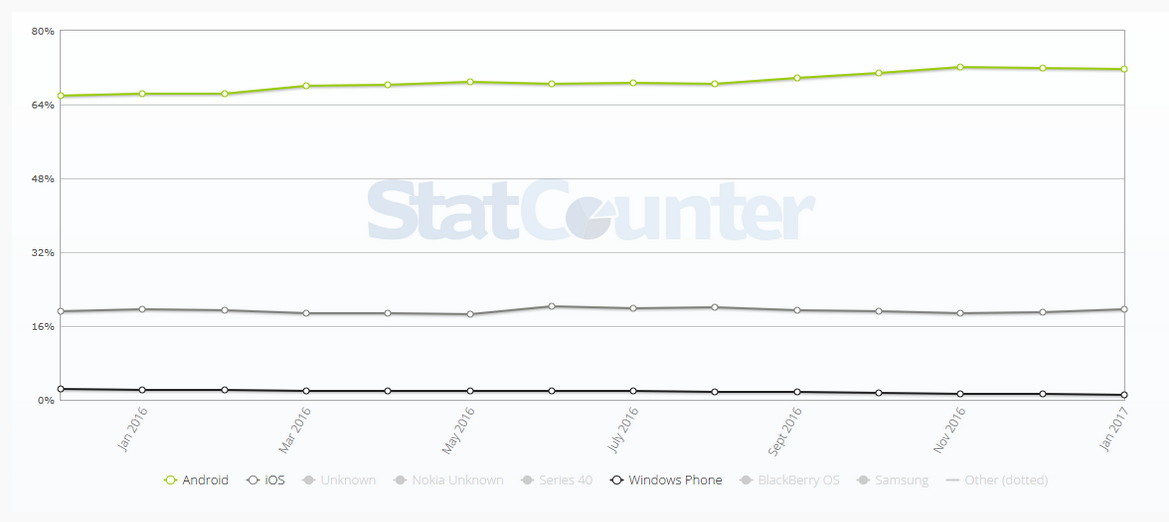


Рис. 1

Целью работы является изучение разработки Android-приложений типа клиент-сервер и создание мобильного приложения Чат.

* Изучение разработки Android-приложений типа клиент-сервер;
* Создание мобильного приложения Чат;

Для достижения данных целей были поставлены следующие задачи:

* Изучить компоненты Android SDK для создания приложений для мобильной системы Android.
* Разработать клиент-серверное приложение Чат

**Глава 1. Основные сведения о Android OS**

**Android**  - операционная система для смартфонов, интернет-планшетов, электронных книг, цифровых проигрывателей, наручных часов, игровых приставок, нетбуков, смартбуков, очков Google, телевизоров и других устройств. В 2015 году появилась поддержка автомобильных развлекательных систем и бытовых роботов. Основана на ядре Linux и собственной реализации виртуальной машины Java от Google. Изначально разрабатывалась компанией Android, Inc., которую затем купила Google. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки.

Одно из главных преимуществ платформы Android — ее открытость. Опе- рационная система Android построена на основе открытого исходного кода и находится в свободном распространении. Это позволяет разработчикам полу- чить доступ к исходному коду Android и понять, каким образом реализованы свойства и функции приложений. Любой пользователь может принять участие в совершенствовании операционной системы Android.

Использование Java стало логичным выбором для платформы Android, потому что это мощный, свободный и открытый язык, известный миллионам разработчиков. Опытные программисты Java могут быстро освоить Android-программирование, используя интерфейсы Google Android API (Application Programming Interface) и другие разработки независимых фирм. Язык Java является объектно-ориентированным, предоставляет разработчикам доступ к мощным библиотекам классов, ускоряющих разработку приложений.

Операционная система Android была разработана компанией Android, Inc., которая в 2005 году была приобретена компанией Google. В ноябре 2007-го был сформирован консорциум Open Handset Alliance™. В задачи этого консорциума входит разработка, сопровождение и развитие Android, внедрение инноваций в мобильных технологиях, а также повышение удобства работы с устройствами Android при одновременном снижении затрат.

Каждой новой версии Android присваивалось название, соответствующее на- званию какого-либо десерта (Рис. 2)

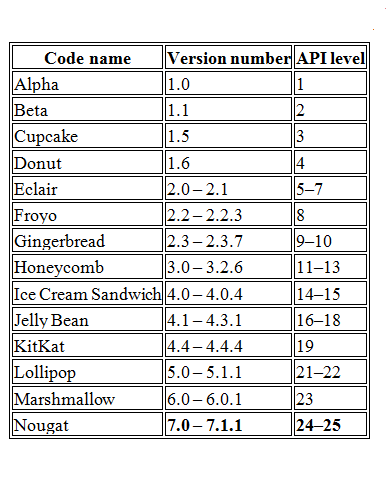


Рис. 2

Android SDK — универсальное средство разработки мобильных приложений для операционной системы Android. Отличительной чертой от обычных редакторов для написания кодов является наличие широких функциональных возможностей, позволяющих запускать тестирование и отладку исходных кодов, оценивать работу приложения в режиме совместимости с различными версиями ОС Андроид и наблюдать результат в реальном времени (опционально). Поддерживает большое количество мобильных устройств, среди которых выделяют: мобильные телефоны, планшетные компьютеры, умные очки (в том числе Google Glass), современные автомобили с бортовыми компьютерами на ОС Андроид, телевизоры с расширенным функционалом, особые виды наручных часов и многие другие мобильные гаджеты, габаритные технические приспособления.[9]

**2. Разработка мобильного приложения Чат**

**2.1 Используемые средства**

Для создания клиента для мобильного приложения Чат для операционной системы Android будет использовать Android Studio.

Впервые Android Studio был представлен публике ещё в 2013 году на конференции Google I/O и с тех пор считается одним из самых продвинутых инструментов для разработки приложений. Android Studio предоставляет своим пользователям максимально широкие возможности для программирования и создания мобильных приложений. Программист с его помощью может создать какое угодно приложение для абсолютно любой версии операционной системы Android, начиная от самой ранней и заканчивая последними новинками. [10]

Для написания приложения типа клиент/сервер, в случае данной курсовой работы – чата, необходимо изучить программные средства, позволяющие использовать сокеты.

**Со́кеты** (англ. *socket* — разъём) — название программного интерфейса для обеспечения обмена данными между процессами. Процессы при таком обмене могут исполняться как на одной ЭВМ, так и на различных ЭВМ, связанных между собой сетью. Различают клиентские и серверные сокеты.[3]

**2.2 Сервер**

***Сервер*** - программный компонент вычислительной системы, выполняющий сервисные (обслуживающие) функции по запросу клиента, предоставляя ему доступ к определённым ресурсам или услугам.[5]

В случае данной курсовой работы необходимо реализовать сервер с следующими функциями:

* Подключение пользователей
* Сохранение данных пользователей
* Передача сообщений между пользователями
* Передача действий пользователей другим пользователям
* Отключение пользователей

Для реализации этих функции мною была выбрана библиотека SocketIO, использующая для передачи информации сокеты.

**2.2.1 Библиотека Socket.IO**

**Socket.IO** — JavaScript-библиотека для веб-приложений и обмена данными в реальном времени. Состоит из двух частей: клиентской, которая запускается в браузере и серверной для node.js. Оба компонента имеют похожее API. Подобно node.js, Socket.IO событийно-ориентированная.

Socket.IO главным образом использует протокол WebSocket, но если нужно, использует другие методы, например Adobe Flash сокеты, JSONP запросы или AJAX запросы, предоставляя тот же самый интерфейс. Помимо того, что Socket.IO может быть использована, как оболочка для WebSocket, она содержит много других функций, включая вещание на несколько сокетов, хранение данных, связанных с каждым клиентом, и асинхронный ввод/вывод.[2]

**2.2.2 Создание сервера для мобильного приложения Чат**

Используя API Socket.IO необходимо реализовать серверное приложение.

Основной API, который понадобится при создании сервера[1]:

* io.on(‘connection’, function(){}) – обрабатывает новые подключения
* socket.on(‘event’, function(){}) – обрабатывает событие event
* socket.emit(‘event’, args…) – отправляет args на клиент подключённого socket-a

Для этого:

* Используя ExpressJS, создадим http-сервер для работы с сокетами.
* Подключим Socket.IO
* Реализуем подключение пользователей
* Реализуем коммуникацию пользователей
* Реализуем отключение пользователей

Для начала подключим все нужные библиотеки, создадим (Рис. 3):

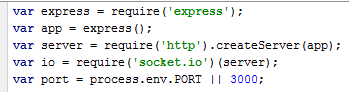


Рис. 3

Далее подключим библиотеку Socket.IO и установим его запуск на определённом порте (Рис. 3).

Создадим пустой объект users, в котором будем хранить ключ – никнейм пользователя и значение – id подключившегося сокета.

Далее необходимо создать обработчик подключения сокета (рис. 4) и его добавления в Подключённые пользователи(Рис. 5).



Рис. 4

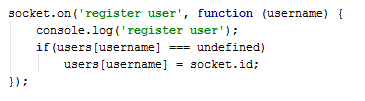


Рис. 5

Напишем обработчик принятия и отправки сообщений.

При получении сообщения ищем его получателя в users, и отправляем ему полученное сообщение (Рис. 6):

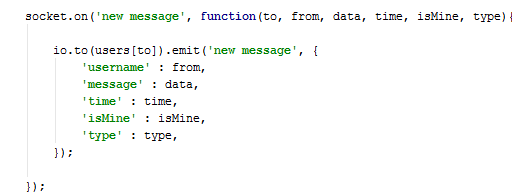


Рис. 6

Также реализуем удаление пользователя по его никнейму(Рис. 7):

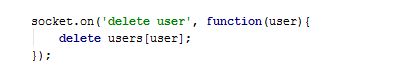


Рис. 7

**2.3 Клиент**

Создадим новый проект в Android Studio, зададим минимальную версию API – 15. Таким образом, приложение будет поддерживаться на устройствах с Android 4.0.3 и выше.

С помощью систему автоматической сборки Gradle подключим все необходимые библиотеки(Рис. 8):

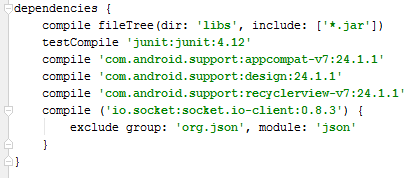


Рис. 8

* RecyclerView – для отображения списков объектов
* Socket.IO – для взаимодействия с сервером посредсвом сокетов

Клиент мобильного приложения будет состоять из 4 компонентов:

* Экран входа
* Экран регистрации
* Экран со списком диалогов
* Экран со списком сообщений

Необходимые элементы разметки экранов:

* EditText – поле для ввода текста.
* TextView – отображение текста.
* Button – кнопка.
* RecyclerView – отображение списка элементов.
* Floating Action Button – плавающая кнопка.
* ImageView – отображение графических файлов.

Моделью данных в случае данной курсовой работы являются Диалоги и Сообщения.

**Экран входа.**

Экран входа состоит из следующих элементов:

* Поле ввода Логина
* Поле ввода Пароля
* Кнопка входа
* Кнопка регистрации

При запуске приложения отображается экран входа (Рис. 9), на котором гость может авторизироваться или зарегестрироваться.

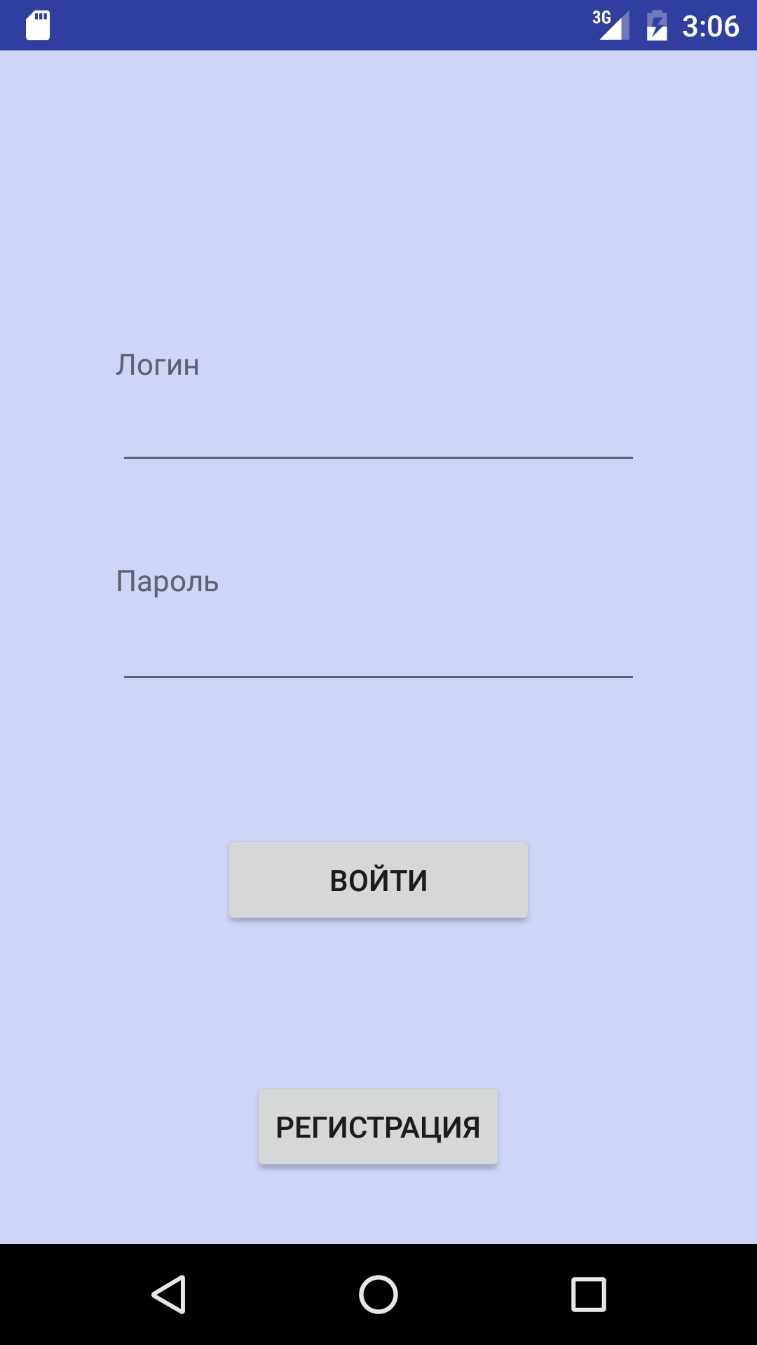


Рис. 9

В случае авторизации, если данные введены неверно либо пользователя с таким логином не существует, система предложит ввести данные заново либо зарегестрироваться(Рис. 10).

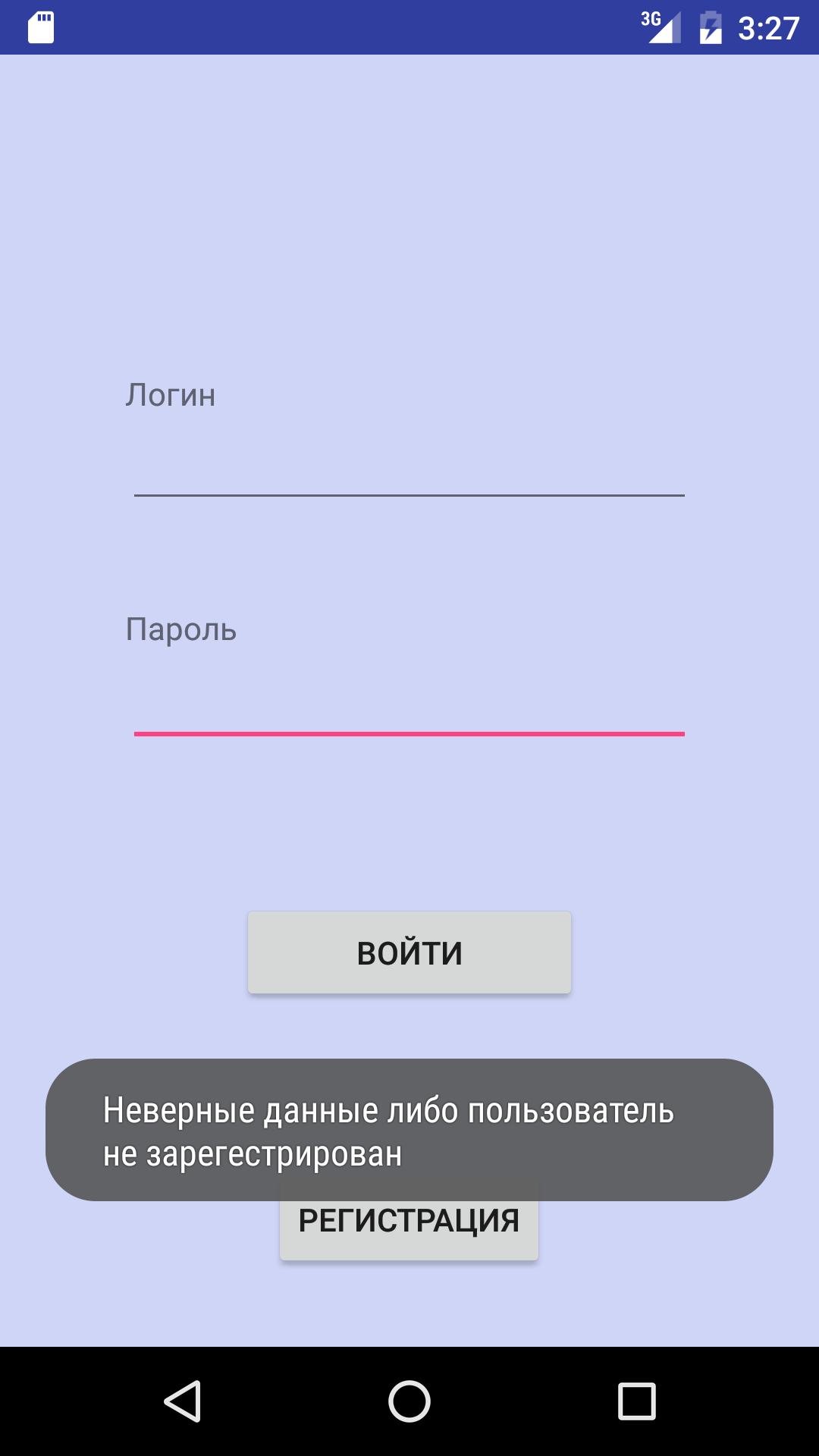


Рис. 10

При нажатии на кнопку Войти, система делает запрос в БД, где хранятся данные пользователей – логин и пароль. Если запрос выдал положительный результат, система переходит на экран со списком диалогов пользователя (Рис. 11).

При нажатии на кнопку Регистрация, пользователь переходит на экран регистрации (Рис.11).

**Экран регистрации.**

На экране регистрации (Рис. 11) гость вводит свои данные и при нажатии на кнопку Регистрация система сохраняет данные в базе данных. После чего система переходит на экран входа (Рис. 9), на котором гость может войти в систему используя свои данные.

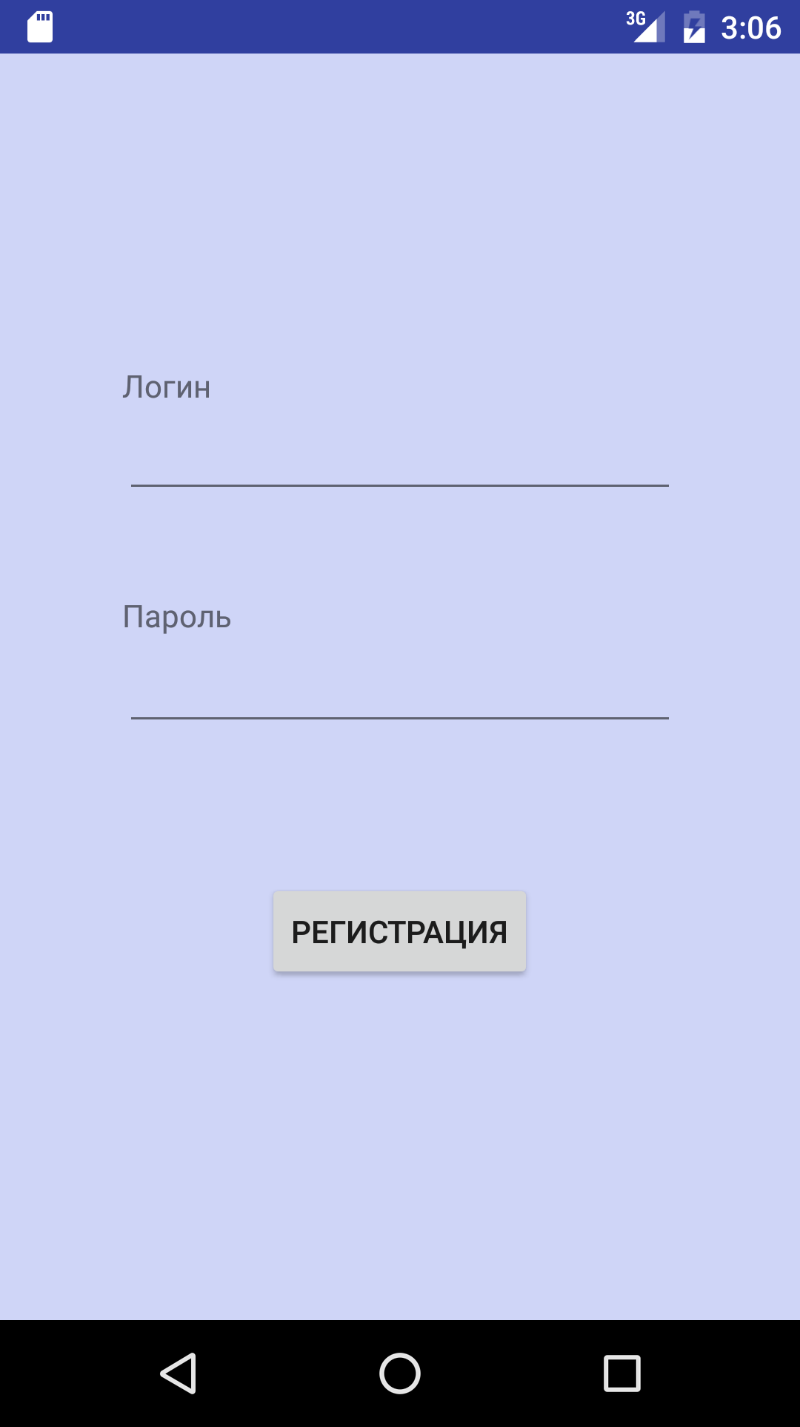


Рис. 10

**Экран со списком диалогов.**

На экране со списком диалогов (Рис. 11) отображаются текущие диалоги пользователя. Для этого используется RecyclerView.

RecyclerView, по сути, является эволюцией одного из самых необходимых в Android-разработке виджетов — ListView. Собственно, предназначение у него ровно то же самое — отображать список элементов, но в свою очередь данный виджет при большом количестве данных делает UI более отзывчивым, а также не использует лишней памяти, так как для каждого элемента списка создаётся объект, и при инициализации списка поиск деталей этого элемента осуществляется единственный раз.

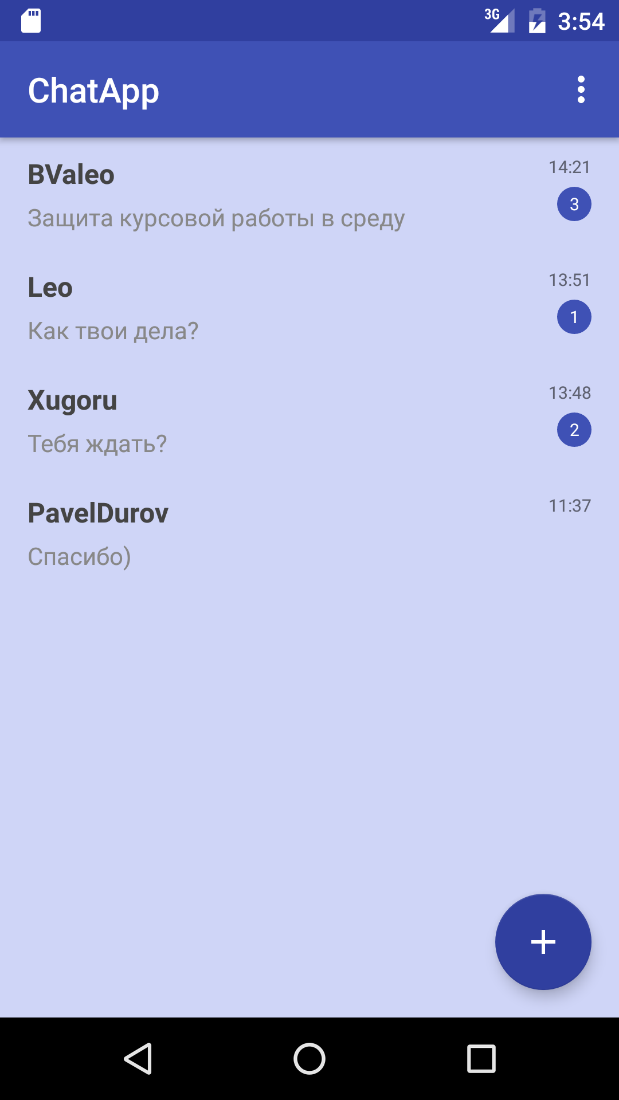


Рис. 11

При переходе на этот экран система делает запрос в базу данных и отображает диалоги текущего пользователя. Элемент RecyclerView отображает элементы в нужной разметке. Для этого необходимо создать реализацию адаптера, который при получении списка элементов настраивает элемент RecyclerView, а после отображает полученные данные.

Если же это новый пользователь, либо пользователь не вёл диалогов список будет пуст. Каждый элемент списка отображает следующую информацию:

* Никнейм пользователя
* Последнее сообщение
* Время последнего сообщения
* Количество непрочитанных сообщений

Пользователь имеет возможность создать новый диалог. При нажатии на элемент Floating Action Bar появляется диалоговое окно (Рис. 12), на котором пользователь должен ввести никнейм получателя, нажать на кнопку Отправить и перейти к экрану со списком сообщений либо нажать отмену, и вернуться к экрану со списком диалогов.

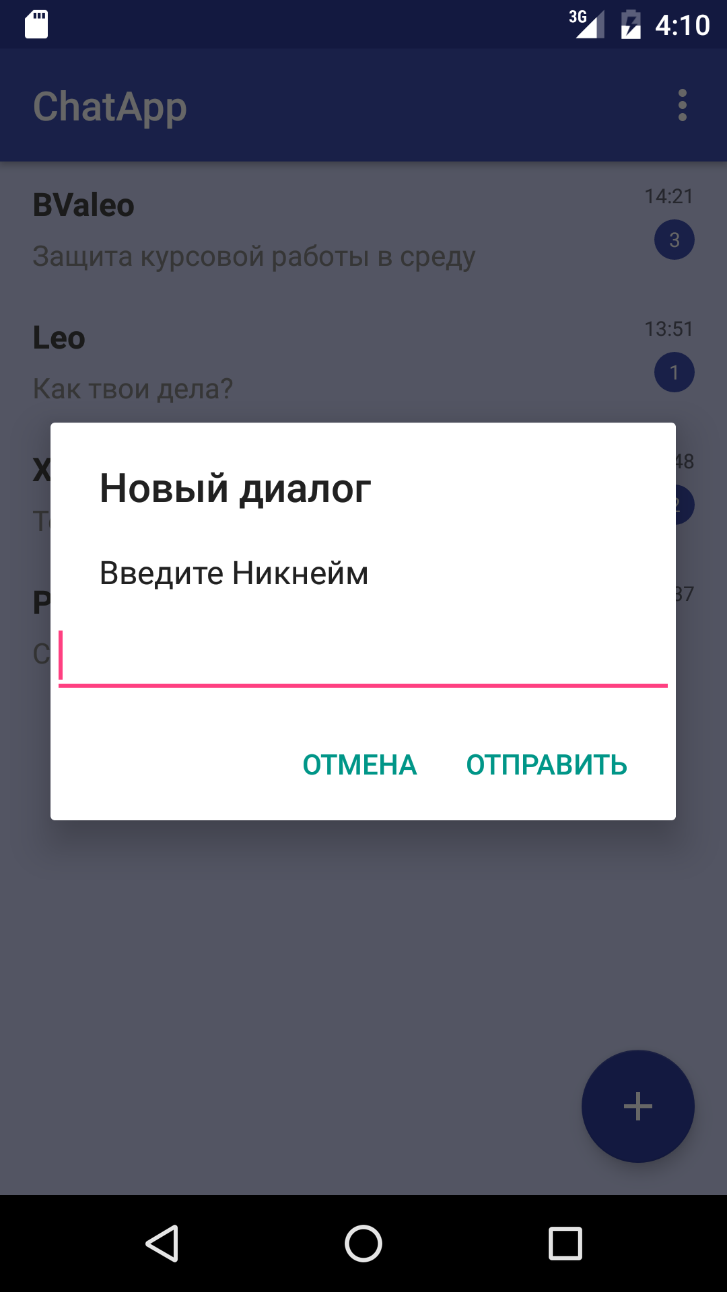


Рис. 12

Также при нажатии на контекстное меню в правом верхнем углу экрана, пользователь может выйти из своего профиля (Рис. 13).

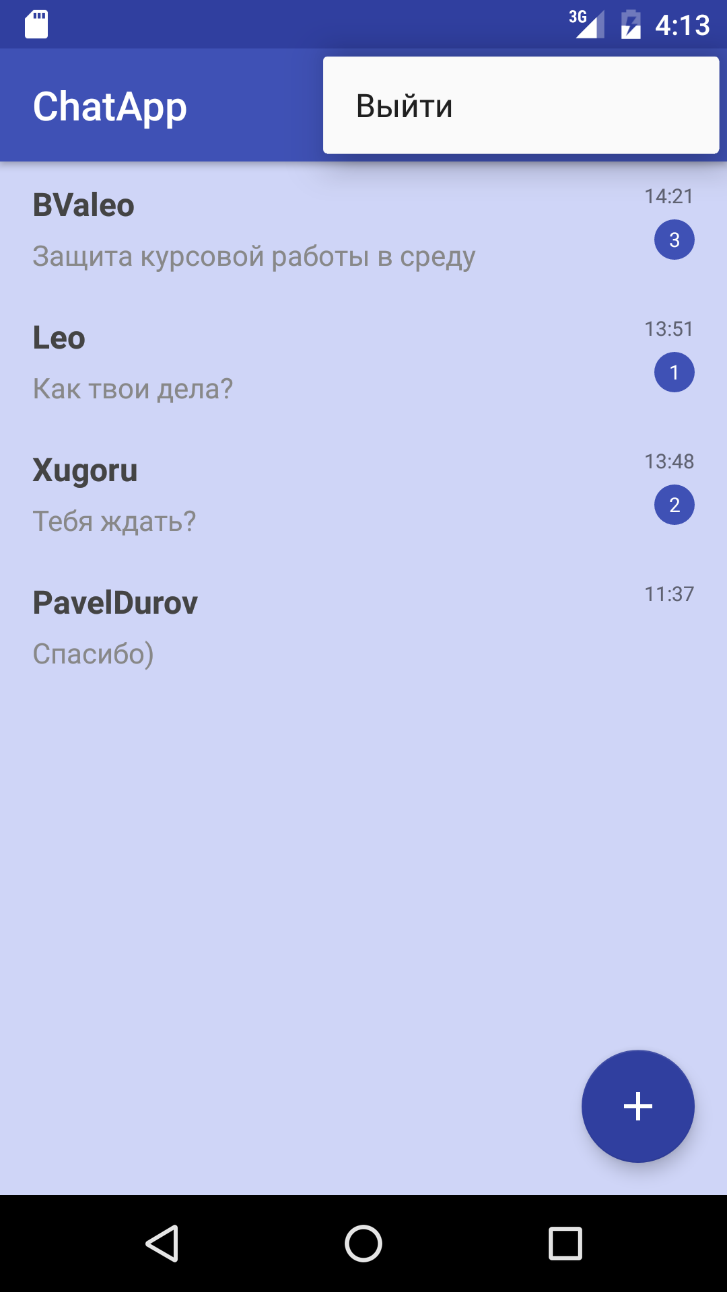


Рис. 13

При нажатии на элемент списка происходит переход на экран со списком сообщений данного диалога.

**Экран со списком сообщений.**

При переходе на данный экран, пользователь видит отображение списка сообщений (Рис. 14). Элемент Toolbar отображает никнейм пользователя, с которым ведётся диалог.

На данном экран пользователь может вводить сообщения, используя элемент отображения EditText, и отправлять их, используя элемент ImageView (Рис.14);

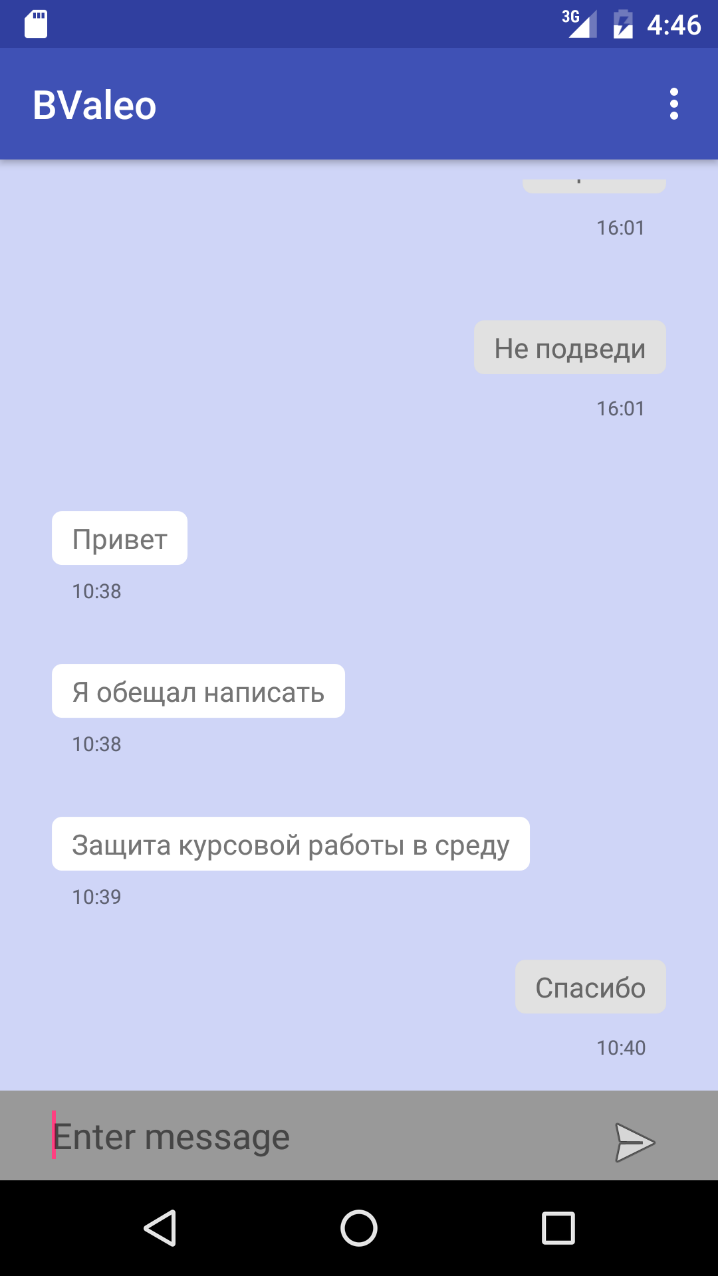


Рис. 14

**3. Заключение**

Таким образом, в результате выполнения курсовой работы были изучены средства, которые могут быть использованы при создании приложений для операционной системы Android. Практическую часть составляют программные средства, позволяющие создать приложения типа-клиент сервер.

**4. Список использованной литературы**

1. Socket.IO [Socket.io]
2. Socket.IO [https://ru.wikipedia.org/wiki/Socket.IO#cite\_note-3]
3. Wikipedia [https://ru.wikipedia.org/wiki/Сокет\_(программный\_интерфейс)]
4. Wikipedia [https://ru.wikipedia.org/wiki/Android]
5. Wikipedia[https://ru.wikipedia.org/wiki/Сервер]
6. AndroidHive [androidhive.com]
7. StartAndroid [startandroid.ru]
8. Head First. Программирование для Android
9. Android для разработчиков 3-е издание
10. Android Studio [http://softprime.net/razrabotchikam/454-android-studio.html]
11. StackOwerflow [stackowerflow.com]
12. Android Developers [https://developer.android.com]