|  |  |
| --- | --- |
|  | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | |
| Институт информационных технологий | |
| Кафедра математического обеспечения и стандартизации информационных систем(МОСИТ)  **КУРСОВАЯ РАБОТА**  по дисциплине  «Функциональное и логическое программирование»  **Тема курсовой работы**: «Анализ предметной области и создание мобильного приложения согласно теме варианта» | |
| **Студент** группы ИКБО-07-18 | Зейналов Магеррам Гилал Оглы |
|  |  |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |
|  |  |
| **Руководитель курсовой работы** | к.т.н, доцент Смольянинова В.А |
|  |  |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись руководителя) |
|  |  |
| Работа представлена к защите | «» 2020 г |
|  |  |
| Допущен к защите | «» 2020 г. |

|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

Утверждаю

Заведующий кафедрой МОСИТ

Головин С.А.

« » 2020 г.

## ЗАДАНИЕ

**на выполнение курсовой работы по дисциплине**

«Функциональное и логическое программирование»

|  |  |
| --- | --- |
| Студент Зейналов Магеррам Гилал Оглы | Группа ИКБО-07-18 |

**Тема работы:** Анализ предметной области и создание мобильного решения согласно теме варианта.

**Исходные данные:** Вариант № 20. Предметная область – Общение людей

## Перечень вопросов, подлежащих разработке, и обязательного графического материала:

1. Анализ предметной области для нахождения в ней проблем и «болей рынка»
2. Изучения проблематики предметной области на конкретном бизнес процессе в данной предметной области
3. Подготовка минимального функционала по проанализированному бизнес процессу;
4. Разработка схем IDEF0 и DFD для анализа внедрения приложения в работающий бизнес процесс и нагрузки системы данными.
5. Разработка и проектирование архитектуры создаваемой информационной системы.
6. Создание портретов пользователей разрабатываемого приложения
7. Создание информационной схемы мобильного приложения
8. Создание интерактивного макета разрабатываемого мобильного приложения
9. Разработка мобильного приложения;
10. Разработка или подключение остальных элементов разрабатываемой информационной системы.

**Срок представления к защите курсовой работы:** до « » 2020 г.

## Задание на курсовую работу выдал (Смольянинова В.А)

« » 2020 г.

## Задание на курсовую работу получил (Зейналов М.Г.)

# Содержание

[Содержание 4](#_Toc59643488)

[Введение 5](#_Toc59643489)

[Основная часть 6](#_Toc59643490)

[Раздел 1. Теоретическое введение 6](#_Toc59643491)

[Раздел 2. Проектная часть 7](#_Toc59643492)

[Раздел 3. Экспериментальная часть 16](#_Toc59643493)

[Заключение 19](#_Toc59643494)

[Список используемых источников 20](#_Toc59643495)

# ВВедение

В современном обществе важно иметь способ быстрой коммуникации. Данным способом является – мессенджеры. Актуальность данной темы подтверждается последними событиями в мире, где нам нужно было передавать друг другу информацию на расстоянии.

В ходе выполнения курсовой работы была рассмотрена предметная область «общение людей» и были выдвинуты главные функции системы.

В основной части работы будут представлены бизнес-процессы, диаграммы классов

# основная часть

## Раздел 1. Теоретическое введение

Общение всегда играло важную роль в жизни человека. Человек – существо социальное, и нам всегда было необходимо общаться с друг другом. С течением времени, многие процессы в жизни менялись, но потребность в общении никуда не исчезла, менялись только способы.

Благодаря интернету и доступности мобильных устройств человечество перешло в новый этап общения. Начали развиваться мессенджеры, социальные сети, видеохостинги и иные способы мобильной передачи информации. Биты информации летят по проводам, передавая данные в самые разные уголки мира за считанные миллисекунды.

В качестве реализации социальных сетей и мессенджеров используется клиент-серверная архитектура. Ее основной смысл состоит в разделении ролей на пользователей и серверов, которые будут обрабатывать запросы от пользователей. Преимуществом такого подхода является то, что клиент не знает реализацию функций приложения, тем самым защищая сервис от некоторых видов атак. Также разгружается устройство клиента отсутствием необходимости хранить реализации функций и поддерживать последнюю версию кода с данными приложениями.

Основной функционал таких приложений – отправка и прием сообщений. С эволюцией мессенджеров добавились такие функции, как голосовые сообщения, расшифровка голосовых сообщений, истории и каналы.

## Раздел 2. Проектная часть

#### **2.1 Постановка задачи**

Задачей моей курсовой работы является реализовать альтернативный способ общения людей со списком контактов и сообщений.

#### **2.2 Проектное решение**

В качестве архитектуры был взят клиент-серверный подход с реализацией REST API.

Реализации клиентской части реализовано с помощью языка программирования Java с Android SDK [3].

Реализация серверной части реализовано с помощью языка программирования Java и Spring Boot [1]. В качестве БД решено использовать MySQL.

Функционал разрабатываемого приложения:

1. Вход;
2. Регистрация;
3. Добавление контактов;
4. Отправка сообщений;
5. Просмотр сообщений;
6. Просмотр аккаунтов.

Среди возможных доработок системы возможны:

1. Улучшение безопасности программы;
2. Возможность отправлять фотографии и видео;
3. Добавление голосовых сообщений;
4. Написание отдельных сервисов.

Рассмотрим реализацию серверной части приложения. Как было ранее сказано, серверная часть реализована с помощью Spring Boot.

Наш backend состоял из 3 контроллеров:

1. UserController
2. MessageController
3. ContactController

**UserController**

В этом контроллере были реализованы вход и регистрация. При регистрации мы обращались к БД и проверяли, существует ли пользователь с таким ником в системе, если ответ приходил утвердительный, то мы выкидывали исключение, в котором указывали, что пользователь с таким ником уже есть. Пароли шифруются с помощью алгоритма BCrypt.

Вход же реализован в виде симуляции, просто проверяя есть ли такой пользователь с паролем в БД и возвращая в качестве ответа сущность с true/false. Такой подход было решено использовать из-за трудоемкости настройки безопасности в серверной части. Конечно, можно было использовать JWT авторизацию с refresh token, которая бы решила проблему повторного входа при запуске приложения, но это слишком трудоемко для 3 контроллеров.

**MessageController**

Основными функциями этого контроллера являются получение и сохранение сообщений. Получение сообщений реализовано довольно просто: сервер обращается к базе данных и возвращает список сообщений для конкретного пользователя. Сохранение тоже сильно не отличается сложностью реализации. Сервер получает данные от клиента, упаковывает в объект и сохраняет в БД.

**ContactController**

Реализовано аналогично MessageController. В привязке аккаунта добавлена проверка на существование такого пользователя.

**Хранение данных**

Для хранения данных используются принциаы работы ORM. Сервер не работает напрямую через SQL скрипты, а использует прослойку в виде Spring Data + Hibernate. Создаются классы-домены, которые помечаются специальными аннотациями, чтобы Spring мог понять, какие классы использовать для создания таблиц. Далее создаются интерфейсы-репозитории, в которых с помощью кода прописываются запросы. Данные интерфейсы отвечают за манипуляцию с БД.

Весь код проекта будет доступен в GitHub [5].

На рис.1 представлена архитектура создаваемой информационной системы

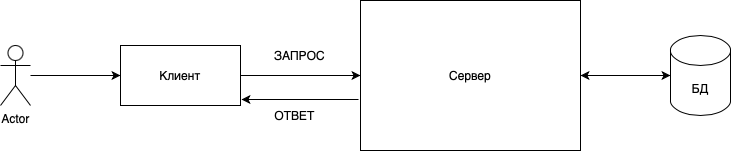


Рисунок 1 – Архитектура создаваемой информационной системы

На рис.2 представлена IDEF0 – диаграмма внедрения приложения в работающий бизнес-процесс

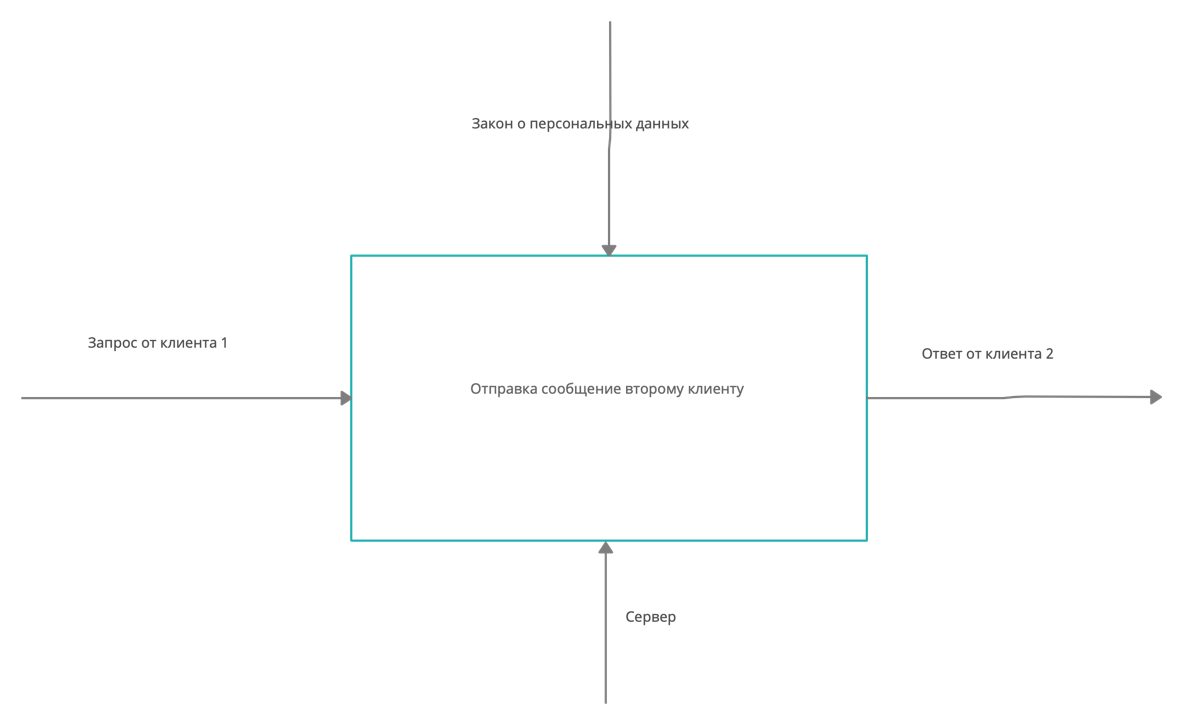


Рисунок 2.1 – Бизнес-процесс в нотации IDEF0

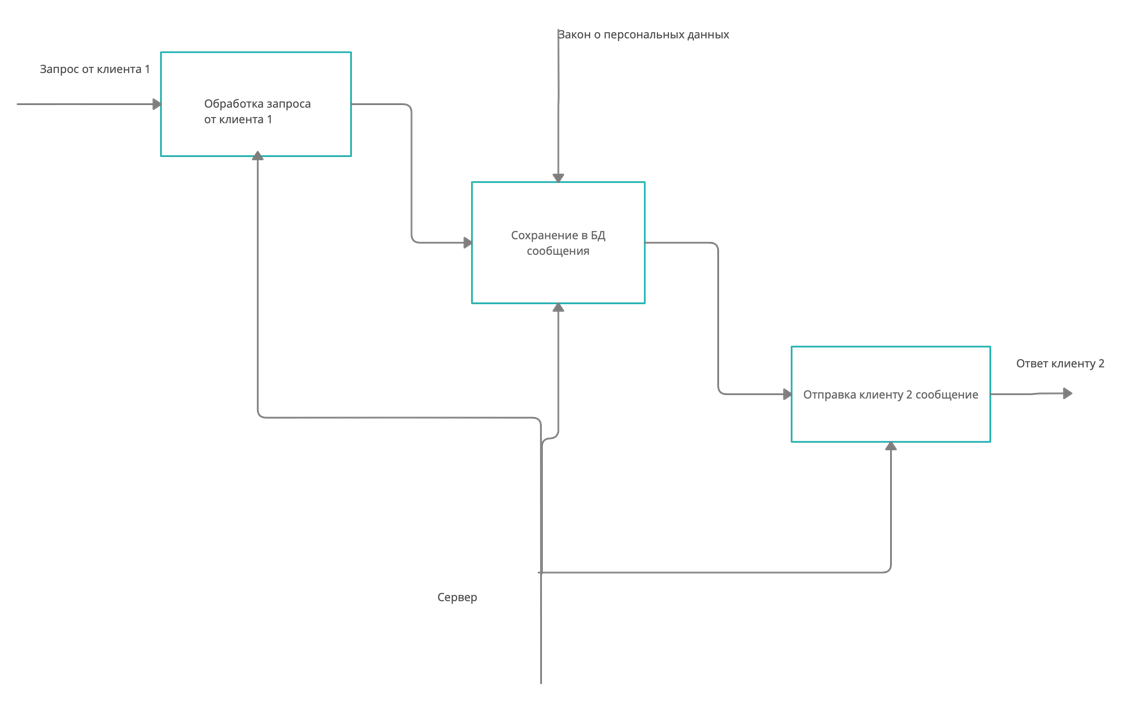


Рисунок 2.2 – Бизнес-процесс в нотации IDEF0

На рис.3 представлена DFD-диаграмма потоков данных

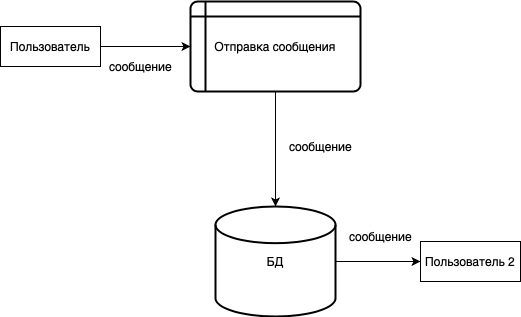


Рисунок 3 – Движение данных в DFD-диаграмме

На рис.4 представлена информационная схема мобильного приложения.

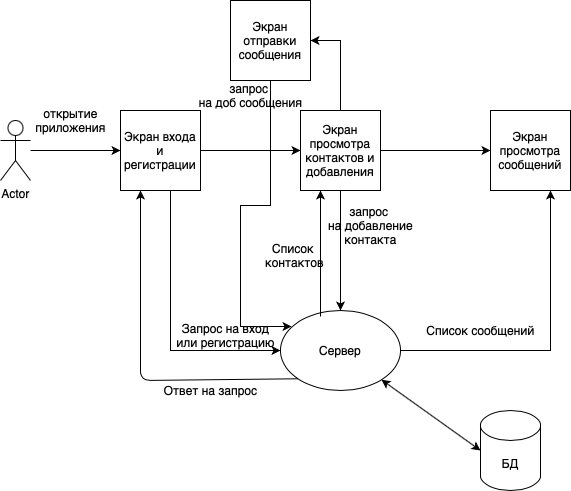


Рисунок 4 – Информационная схема мобильного приложения

В качестве основы была взята Figma, но в последствии остальные части приложения были уже задизайнены в редакторе дизайна Android Studio.

Все разработанные макеты представлены на рис.5 – 8.

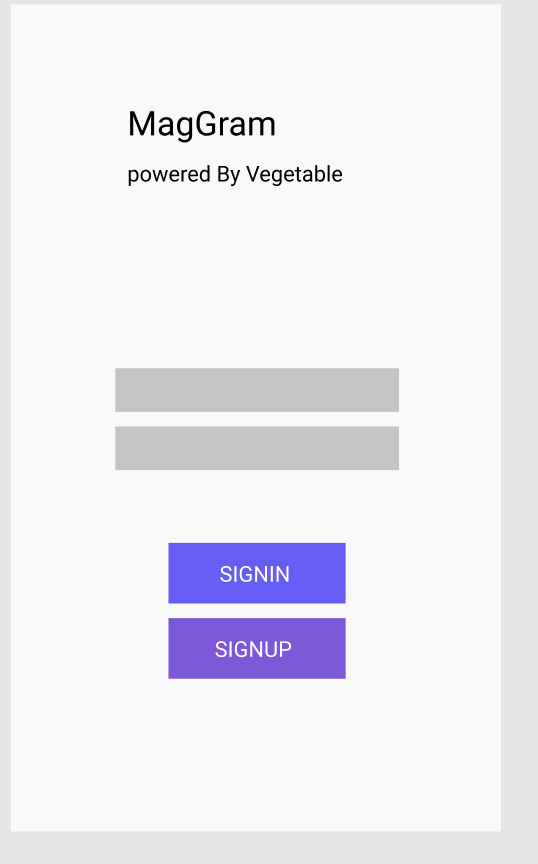


Рисунок 5 – Стартовый экран в Figma

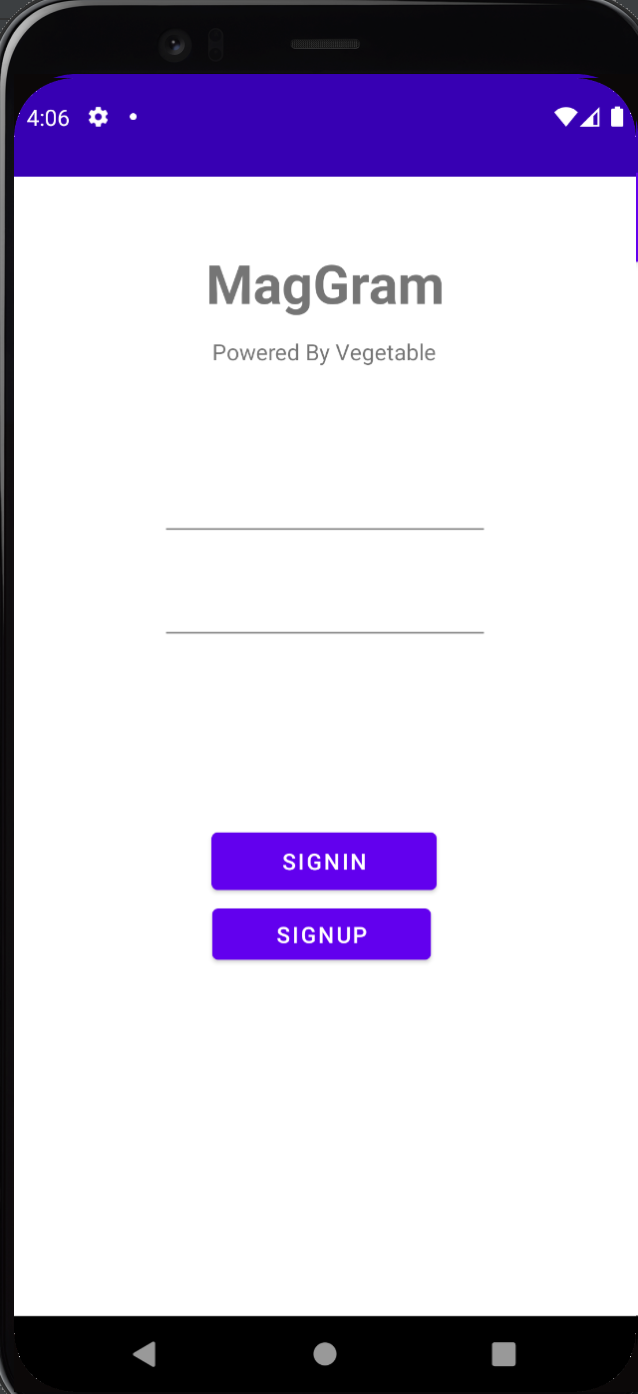


Рисунок 6 – Экран с входом

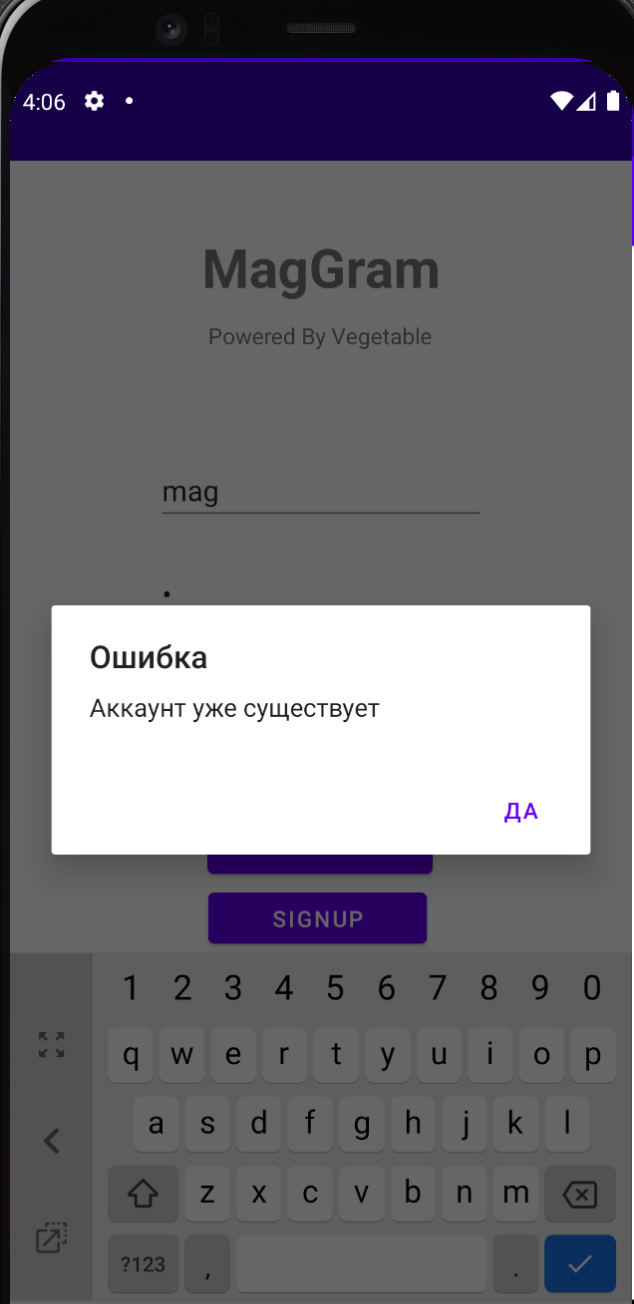


Рисунок 7 – Экран с ошибкой

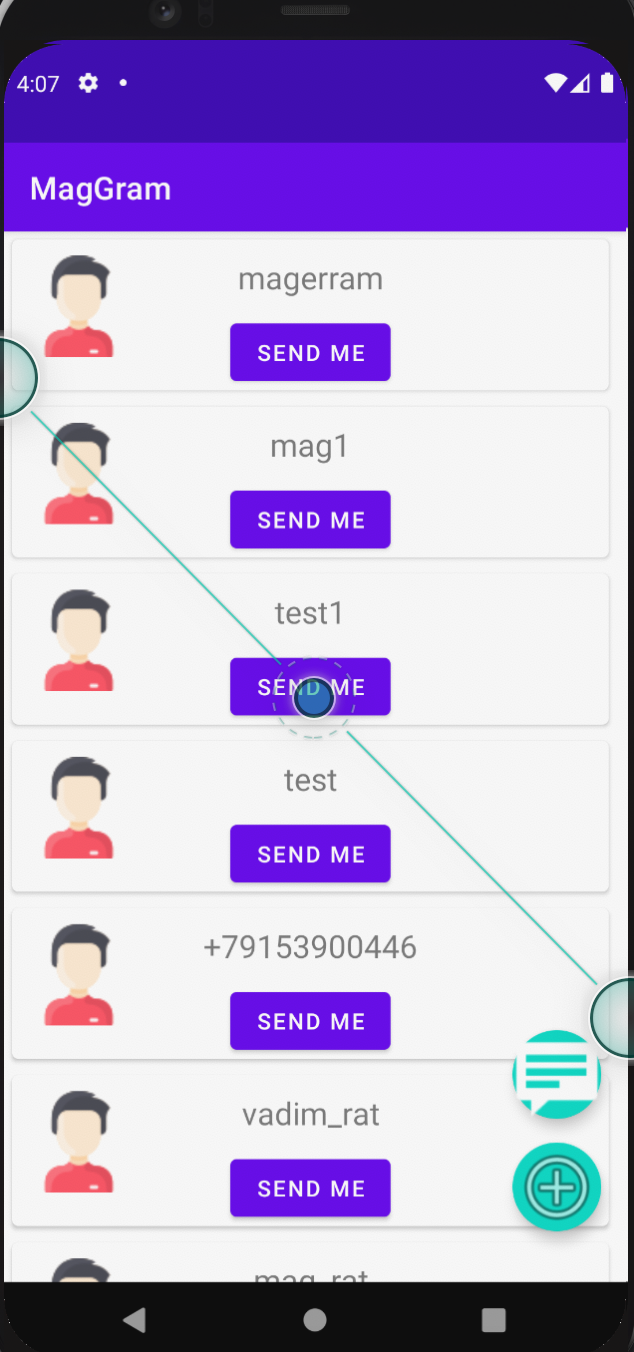


Рисунок 8 – Экран с контактами

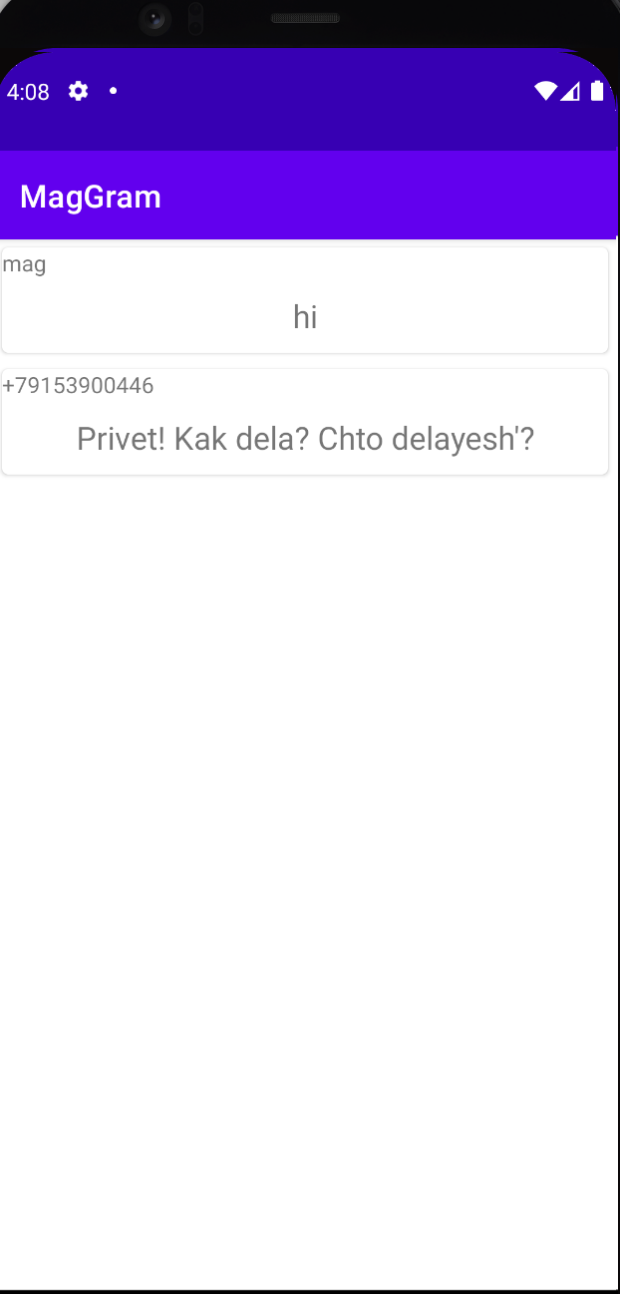


Рисунок 9 – Экран со списком сообщений

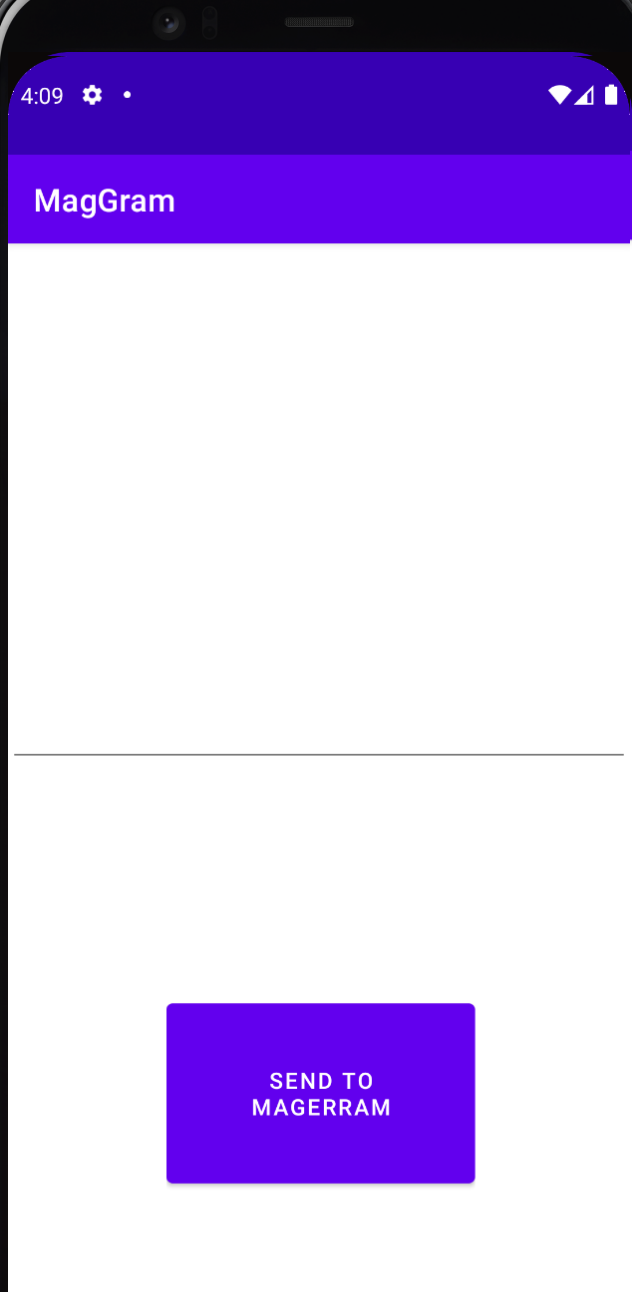


Рисунок 10 – Экран отправкой сообщения

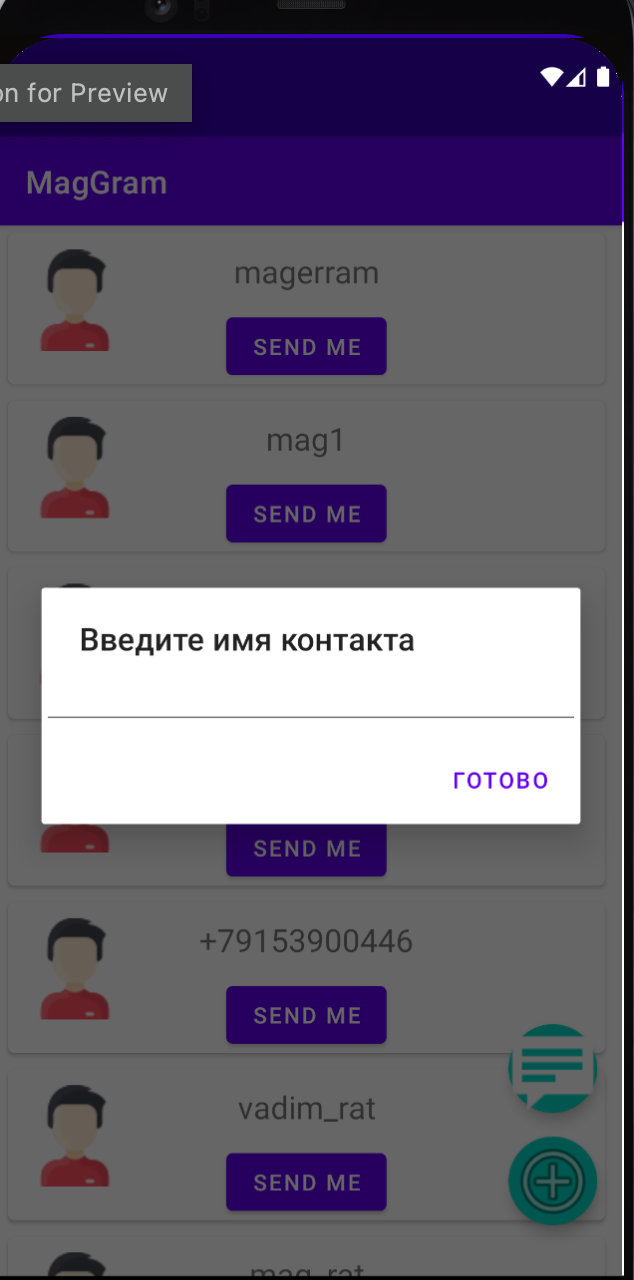


Рисунок 11 – Экран с добавлением контактов

## Раздел 3. Экспериментальная часть

#### **3.1 Тестирование бэкенда**

Бэкенд был протестирован с помощью приложения Postman. Ниже тестирование backend.

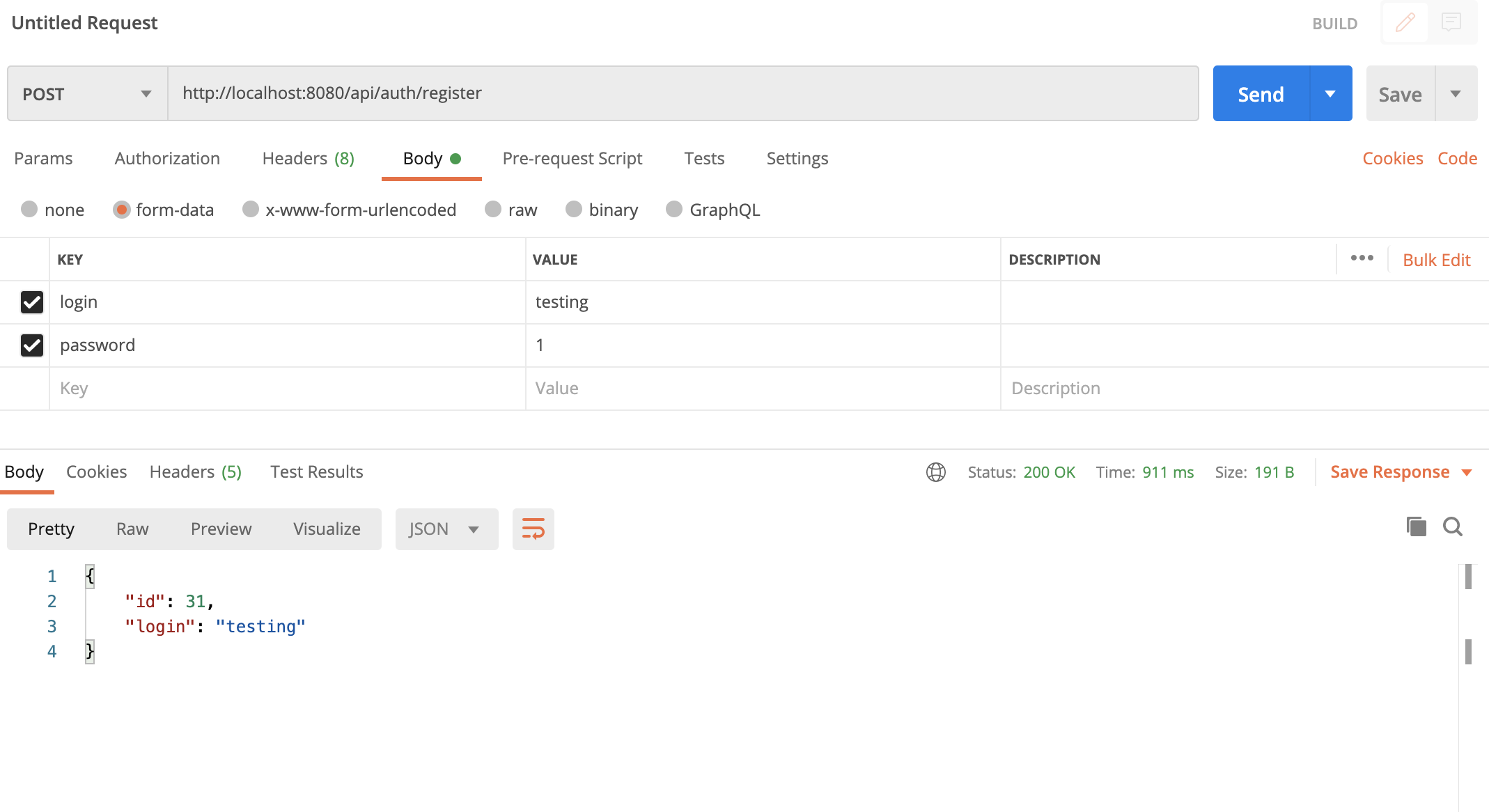


Рисунок 12. Тестирование регистрации

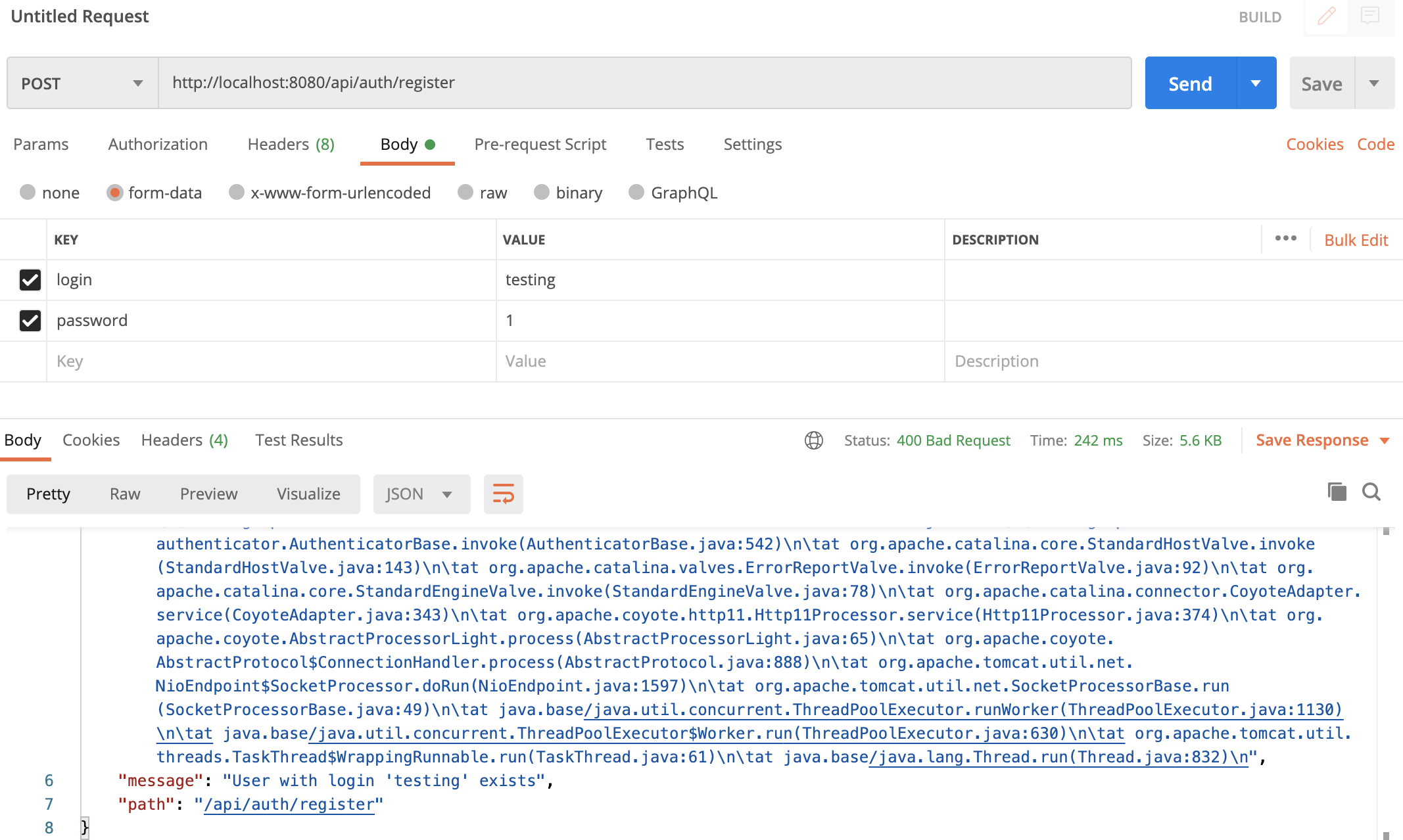


Рисунок 13. Тестирование повторной регистрации

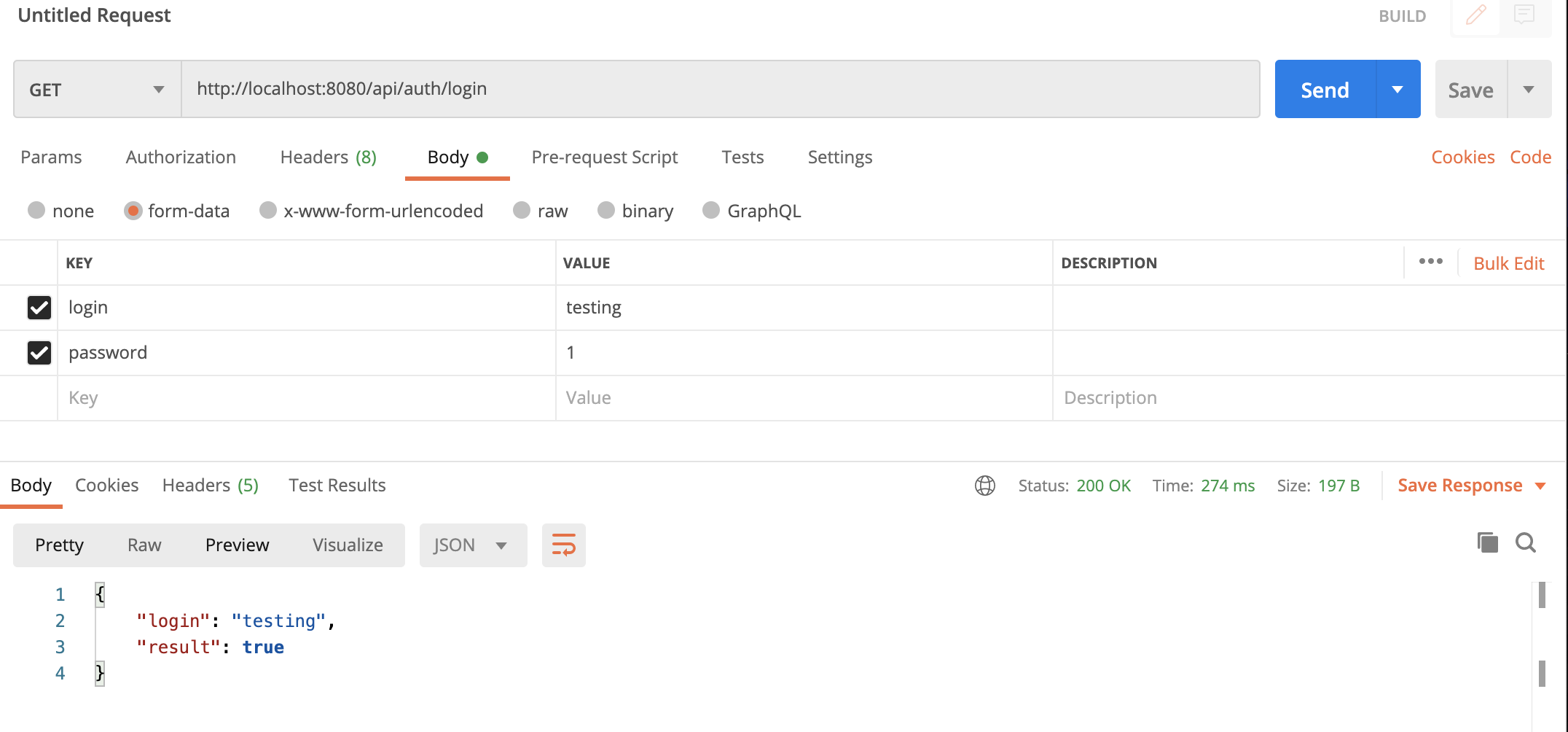


Рисунок 14. Тестирование входа

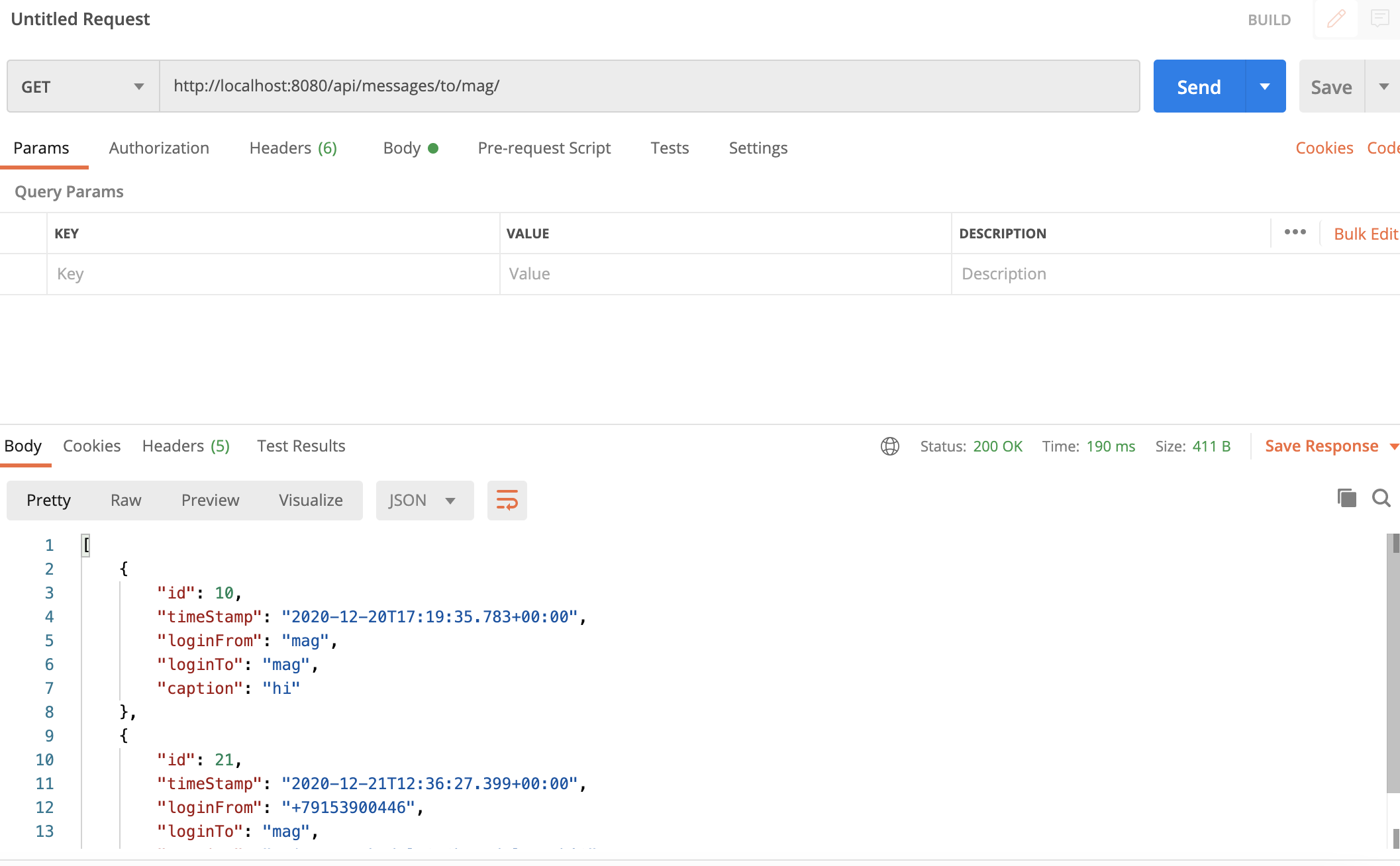


Рисунок 15. Тестирование сообщений

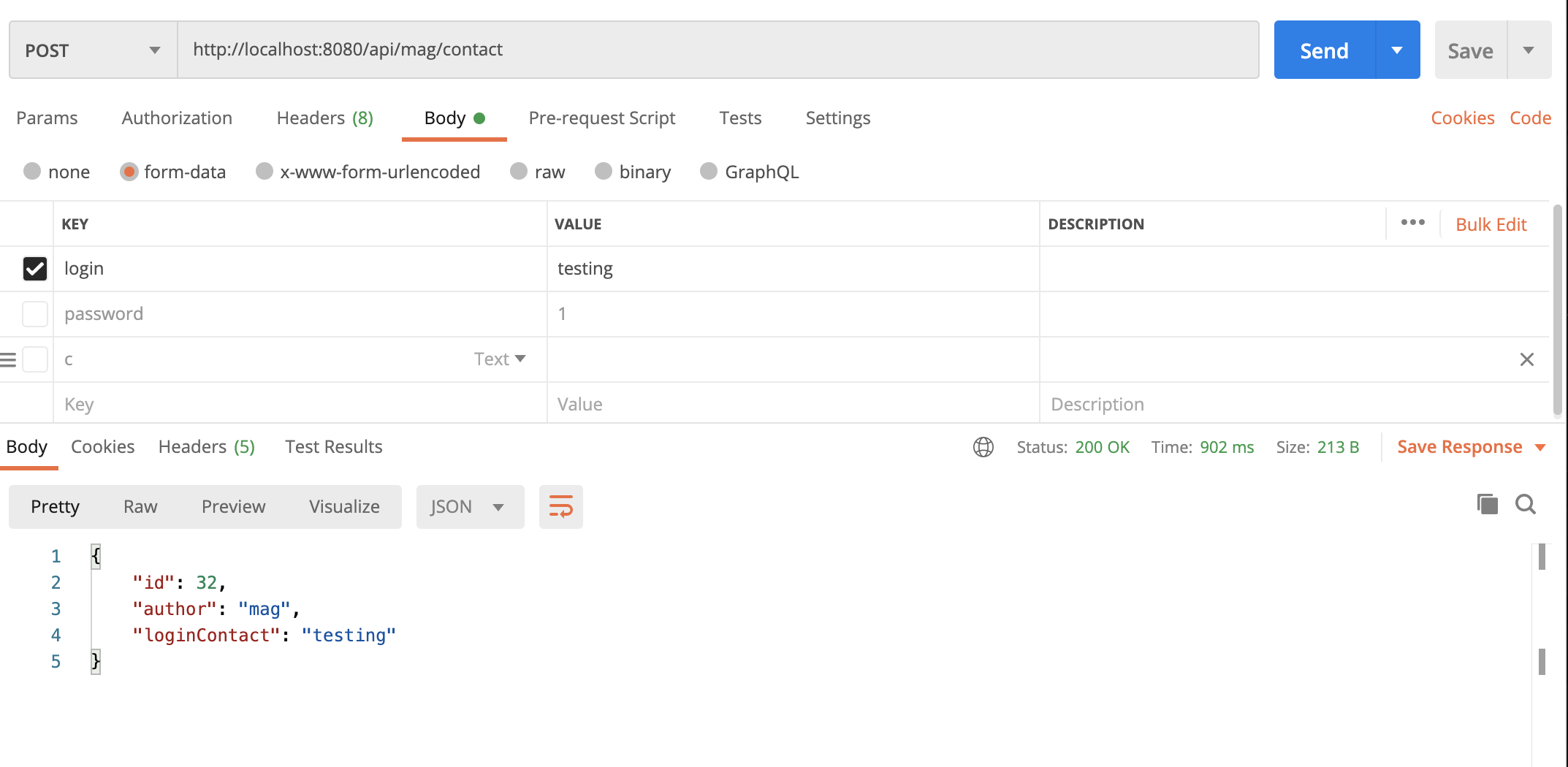


Рисунок 16. Тестирование добавления аккаунта.

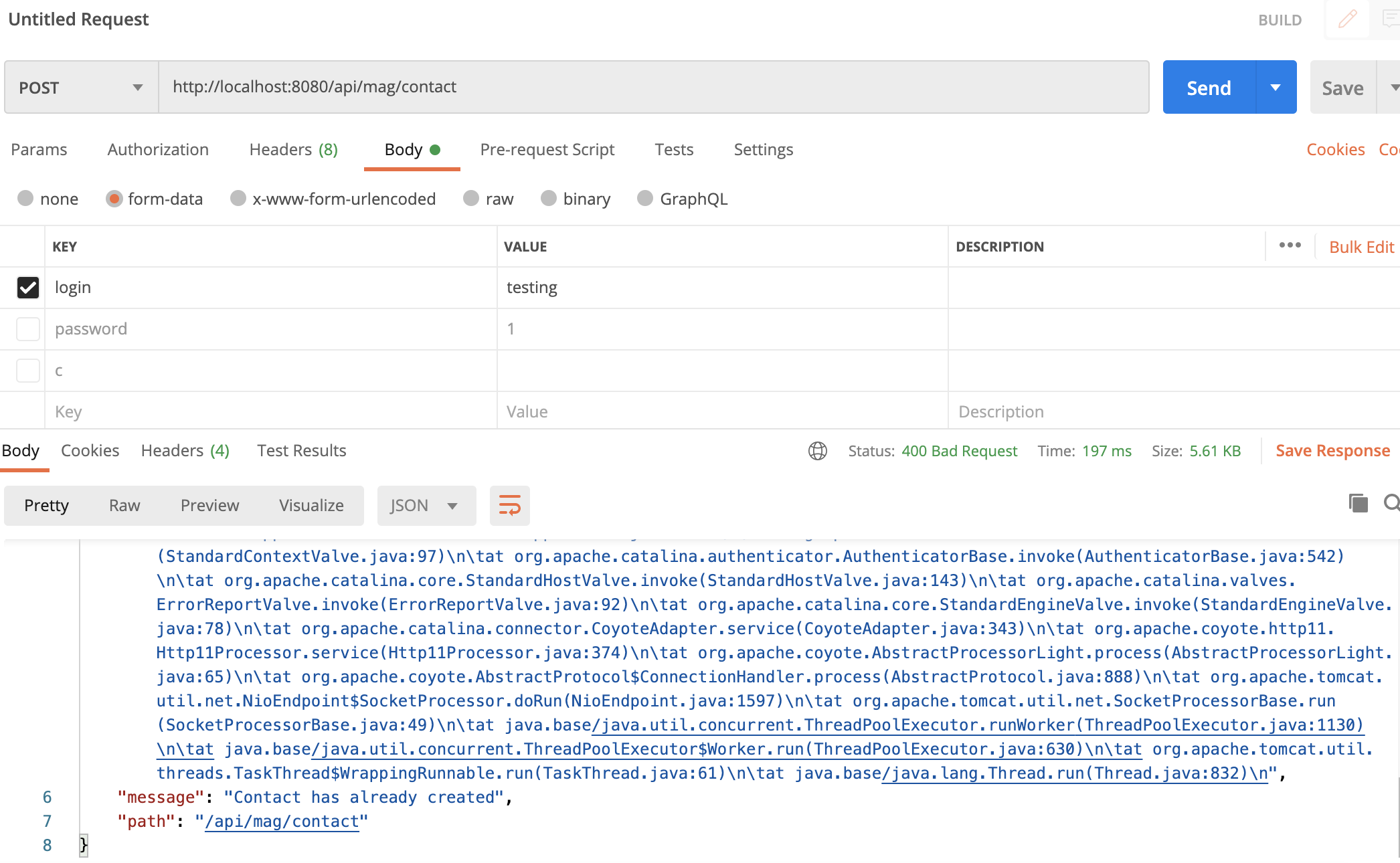


Рисунок 17. Тестирование повторного добавления аккаунта

#### **3.2 Управление экранов**

Для реализации управлении экранов использовали классы, отнаследованные от AppCompatActivity. В данных классах прописывается вся логика управления элементов на экране, ввод данных, проверка на корректность. Также в этих классах отправляются запросы на сервер.

#### **3.3 Связь данных**

Для передачи запроса использовалась библиотека RetroFit [2]. С помощью нее мы отправляли запросы на серверы и получали ответы сразу в объектах. Для этого нужно было настроить сериализацию в параметрах RetroFit.

В качестве библиотеки для десериализации и сериализации использовался GSON [4]. В классах-получателях для корректной работы десериализации настраивались имена полей с помощью аннотаций.

# заключение

В результате выполнения курсовой работы были изучены методы программирования на операционную систему Android. Подтянуты навыки работы с дизайном, рассмотрен такой инструмент как Figma. Были улучшены навыки написания серверной части приложения. В последствии, получены навыки связки клиента и сервера с помощью библиотеки Retrofit.

Также были рассмотрены и использованы такие методологии как DFD и IDEF0. C помощью них были построены диаграммы бизнес-процессов и потоков данных.

В заключении, можно заметить, что в результате работы были подняты общие навыки проектирования систем, где используются как клиент, так и сервер, повышена общая грамотность разработчика. Все заявление функции были реализованы в полной мере и протестированы.

# список используемых источников

1. Spring. Spring Boot [Электронный ресурс]. – Режим доступа: https://spring.io/projects/spring-boot, (свободный).
2. Retrofit [Электронный ресурс]. – Режим доступа: https://square.github.io/retrofit/, (свободный).
3. Android Studio. SDK [Электронный ресурс]. – Режим доступа: https://developer.android.com/studio, (свободный).
4. GitHub. GSON [Электронный ресурс]. – Режим доступа: [https://github.com/google/gson, (свободный)](https://github.com/google/gson,%20(свободный)).
5. GitHub. MagGram [Электронный ресурс]. – Режим доступа: <https://github.com/vgtstptlk/MagGram> , свободный.