КОМИТЕТ ОБРАЗОВАНИЯ И НАУКИ КУРСКОЙ ОБЛАСТИ  
областное бюджетное профессиональное образовательное учреждение  
«Курский государственный политехнический колледж»

Курск

2024

Специальность 09.02.07 Информационные системы и программирование

КУРСОВОЙ ПРОЕКТ

**по ПМ.05 «Проектирование и разработка информационных систем»**

**Тема: Тестирование информационной системы «Интернет-магазин 3D моделей»**

Выполнил:

Студент группы ИС-33 Колкнева А. И.

Руководитель:

Чаплыгин А.В.

Оценка за курсовой проект

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_ Чаплыгин А.В.

(подпись) (инициалы, фамилия)

**Содержание**

|  |  |
| --- | --- |
| Введение | 3 |
| 1 Техническое задание | 5 |
| 1.1 Обоснование требований к комплексу технических средств | 5 |
| 1.2 Описание функциональной структуры | 5 |
| 1.3 Характеристика программных комплексов для тестирования информационной системы | 6 |
| 2 Методы испытаний | 7 |
| 2.1 Интеграционное тестирование | 7 |
| 2.2 Модульное тестирование | 18 |
| 2.3 Нагрузочное тестирование | 19 |
| Заключение | 21 |
| Список литературы | 22 |
| Приложение А Интерфейс среды Visual Studio 2019 | 23 |
| Приложение Б Интерфейс среды Apace JMeter | 24 |
| Приложение В Листинг модульных тестов | 25 |

# ****Введение****

В современном мире информация стала одним из самых важных ресурсов, а информационные системы (ИС) стали необходимым инструментом во многих сферах деятельности. В реальных условиях проектирование ИС - это процесс поиска способа, который удовлетворяет требованиям функциональности системы с использованием имеющихся технологий и с учетом заданных ограничений.

Разнообразие задач, решаемых с помощью информационных систем (ИС), привело к появлению множества разнотипных систем, которые отличаются принципами построения и заложенными в них правилами обработки информации.

Таким образом, актуальность проектирования автоматизированной информационной системы для магазина заключается в том, что она:

* облегчает работу с огромными объемами информации;
* экономит ресурсы предприятия;
* приводит информацию к формату, более удобному для восприятия человеком.

Объектом исследования курсового проекта является тестирование основных функций, имеющихся в программе.

Предметом исследования является методика тестирования систем, подсистем и компонентов программы.

Основной целью курсового проекта является тестирование всех функций программного проекта, с последующем устранением найденных ошибок.

Для достижения данной цели необходимо решить следующие частные задачи:

* разработка структур данных;
* автоматизация процесса работы предприятия;
* выбор эффективных алгоритмов с учётом их точности, устойчивости, сходимости и т.д.;
* интеграция программных модулей, анализ полученных результатов работы модулей;
* нагрузочное тестирование базы данных программы;
* разработка плана тестирования;
* оформление результатов модульных тестов в TestCase-таблицы;
* тестирование всех методов программы;
* устранение ошибок, выявленных в процессе тестирования;
* тестирование программного обеспечения с целью поиска уязвимостей;
* составление отчётов о тестировании.

Источниками данных для тестирования ИС является исходный код программы, базы данных и планы тестирования.

## Техническое задание

## 1.1 Обоснование требований к комплексу технических средств

Для разработки программного обеспечения использовались следующие программно-технические средства:

Персональный компьютер с характеристиками:

* объем оперативной памяти: 32,0 Гб;
* объем накопителя: 500 Гб;
* 8-ми ядерный процессор с тактовой частотой 2.90GHz;

Требования к составу и параметрам периферийных устройств: клавиатура, мышь.

Требования к программному обеспечению:

* операционная система Windows 10 Профессиональная;
* средства разработки: Microsoft Windows, Android Studio, Microsoft SQL Server Management Studio, Microsoft Visio 2016, Apache JMeter.
* .NET Framework 4.7.2;

## 1.2 Описание функциональной структуры

Программное обеспечение предназначено для автоматизации деятельности ведения заявок. Программный продукт позволяет вести автоматизированную обработку данных о заявках, клиентах и товарах.

Разрабатываемый программный продукт предназначен для работников отдела менеджмента и клиентов.

Функциональные возможности программного обеспечения:

* формирование отчётов;
* поиск данных;
* фильтрация данных;
* добавление данных;
* редактирование данных;
* удаление данных.

## 1.3 Характеристика программных комплексов

Для разработки курсового проекта был выбран язык Kotlin и .Net Framework. Главная причина выбора Kotlin - он уделяет большое внимание удобочитаемости, что высоко ценится в отрасли. Синтаксис позволяет создавать алгоритмы, используя меньшее количество строк. Создание кода занимает меньше времени, что повышает эффективность работы программиста. Написанный код выглядит лаконичным и легко читается. Он разработан для функционального программирования, в нем реализованы возможности использования функций высшего порядка.

Из всех сред разработки мой выбор пал на Android Studio. Она превосходит конкурентов по многим параметрам, к которым можно отнести:

* гибкость среды разработки;
* больший набор функций;
* процесс разработки, который подстраивается под разработчика.

Android Studio – универсальная среда разработки, так как позволяет оптимизировать работу будущих приложения для работы не только на смартфонах, но и на планшетах, портативных ПК, которые работают на основе рассматриваемой операционной системы. Отличительная особенность эмулятора – просмотр приблизительных показателей производительности при запуске приложения на самых популярных устройствах.

Среда разработки для приложений Android Studio последней версии стала по настоящему удобной даже для начинающих разработчиков. В программе реализованы все современные средства для упаковки кода, его маркировки. Востребованная многими создателями ПО функция Drag-n-Drop, облегчающая перенос компонентов в среду разработки непосредственно.

Apache JMeter - инструмент для проведения нагрузочного тестирования, разрабатываемый Apache Software Foubdation.

– Легко установить;

– GUI и non-GUI режим работы (из консоли);

– Кроссплатформенность;

– Поддержка протоколов (HTTP(S), SOAP, FTP);

– Возможность интеграций с Selenium;

– Возможность автоматически генерировать тестовые скрипты;

**2 Методы испытаний**

2.1 Интеграционное тестирование

Испытаниям подвергается автоматизированная информационная система «Интернет-магазин 3D моделей».

Область применения – электронный метод оформления и обработки заявок на мойку. Обозначение программы – «TelekomNevaApp.exe».

Цель испытаний – проверка работоспособности автоматизированной информационной системы «Нева Телеком».

Требования к программе – проверка работоспособности приложения при выполнении функций:

* авторизация пользователя;
* регистрация пользователя;
* проведение расчётов;
* поиск данных;
* фильтрация данных;
* добавление данных;
* редактирование данных;
* удаление данных.

Методы и сценарии испытаний представлены в Таблицах 1 – 16

Таблица 1. Общая информация о тестировании

|  |  |
| --- | --- |
| **Название проекта** | TelekomNevaApp |
| **Рабочая версия** | 1.1v |
| **Имя тестирующего** | Колкнева Ангелина Игоревна |
| **Дата(ы) теста** | 12.04.2024 |
|  |  |

Таблица 2. Test Case #1:

|  |  |
| --- | --- |
| **Test Case #** | TC\_AuthorizationTest\_1 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка входа существующего пользователя (Логин, пароль) |
| **Резюме испытания** | 1. Проверка входа пользователя с корректными данными формы (Логин, пароль); 2. Проверка невозможности входа пользователя с некорректными данными формы (Логин, пароль) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Логин; 2. Ввод поля пароль; 3. Нажатие кнопки Войти   Для каждого случая некорректных полей формы   1. Ввод поля Логин; 2. Ввод поля пароль; 3. Нажатие кнопки Войти |
| **Данные тестирования** | Для корректных полей формы:   1. Логин: Правильный логин 2. Пароль: Правильный пароль   Некорректные поля формы случай 1:   1. Логин: Правильный логин 2. Пароль: Не правильный пароль |
| **Ожидаемый результат** | Для корректных полей формы:   1. Вход пользователя в приложение;   Просмотр опубликованных моделей;  Для некорректных полей формы:   1. Невозможность входа пользователя; 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Вход пользователя в приложение;   Просмотр опубликованных моделей;  Для некорректных полей формы:   1. Невозможность входа пользователя; 2. Вывод сообщение об ошибке; |
| **Предпосылки** | 1. Запуск приложения; |
| **Постусловия** | Выполнена навигация согласно роли авторизованного пользователя/форма авторизации |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 3. Test Case #2:

|  |  |
| --- | --- |
| **Test Case #** | TC\_AuthorizationTest\_2 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка входа существующего пользователя (Логин, пароль) |
| **Резюме испытания** | 1. Проверка входа пользователя с корректными данными формы (Логин, пароль); 2. Проверка невозможности входа пользователя с некорректными данными формы (Логин, пароль) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Логин; 2. Ввод поля пароль; 3. Нажатие кнопки Войти   Для каждого случая некорректных полей формы   1. Ввод поля Логин; 2. Ввод поля пароль; 3. Нажатие кнопки Войти |
| **Данные тестирования** | Для корректных полей формы:   1. Логин: Правильный логин 2. Пароль: Правильный пароль   Некорректные поля формы случай 2:   1. Логин: Не правильный логин 2. Пароль: Правильный пароль |
| **Ожидаемый результат** | Для корректных полей формы:   1. Вход пользователя в приложение; 2. Просмотр опубликованных моделей;   Для некорректных полей формы:   1. Невозможность входа пользователя; 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Вход пользователя в приложение; 2. Просмотр опубликованных моделей;   Для некорректных полей формы:   1. Невозможность входа пользователя; 2. Вывод сообщение об ошибке; |
| **Предпосылки** | 1. Запуск приложения; |
| **Постусловия** | Выполнена навигация согласно роли авторизованного пользователя/форма авторизации |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 4. Test Case #3:

|  |  |
| --- | --- |
| **Test Case #** | TC\_AuthorizationTest\_3 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка входа существующего пользователя (Логин, пароль) |
| **Резюме испытания** | 1. Проверка входа пользователя с корректными данными формы (Логин, пароль); 2. Проверка невозможности входа пользователя с некорректными данными формы (Логин, пароль) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Логин; 2. Ввод поля пароль; 3. Нажатие кнопки Войти   Для каждого случая некорректных полей формы   1. Ввод поля Логин; 2. Ввод поля пароль; 3. Нажатие кнопки Войти |
| **Данные тестирования** | Для корректных полей формы:   1. Логин: Правильный логин 2. Пароль: Правильный пароль   Некорректные поля формы случай 3:   1. Логин: Не правильный логин 2. Пароль: Не правильный пароль |
| **Ожидаемый результат** | Для корректных полей формы:   1. Вход пользователя в приложение; 2. Просмотр опубликованных моделей;   Для некорректных полей формы:   1. Невозможность входа пользователя; 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Вход пользователя в приложение; 2. Просмотр опубликованных моделей;   Для некорректных полей формы:   1. Невозможность входа пользователя; 2. Вывод сообщение об ошибке; |
| **Предпосылки** | 1. Запуск приложения; |
| **Постусловия** | Выполнена навигация согласно роли авторизованного пользователя/форма авторизации |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 5. Test Case #4:

|  |  |
| --- | --- |
| **Test Case #** | TC\_PublicationTest\_1 |
| **Приоритет тестирования (Малый/Средний/высокий)** | Высокий |
| **Название тестирования/Имя** | Публикация модели с валидацией полей формы (Название, Описание, Превью модели, Цена) |
| **Резюме испытания** | 1. Публикация модели с пройденной процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели); 2. Проверка невозможности публикации модели с не пройдённой процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Описание; 4. Ввод поля Стоимость модели; 5. Нажатие кнопки Опубликовать;   Для каждого случая некорректных полей формы   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Стоимость модели; 4. Нажатие кнопки Опубликовать; |
| **Данные тестирования** | Для корректных полей формы:   1. Название: Присутствует 2. Картинка: Присутствует 3. Стоимость модели: 50руб.   Некорректные поля формы случай 1:   1. Название: 2. Картинка: Присутствует 3. Стоимость модели: 50руб. |
| **Ожидаемый результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Предпосылки** | 1. Открытие сайта; 2. Переход по кнопке Опубликовать; |
| **Постусловия** | Выполнена навигация на форму «Публикация» |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 6. Test Case #5:

|  |  |
| --- | --- |
| **Test Case #** | TC\_PublicationTest\_2 |
| **Приоритет тестирования (Малый/Средний/высокий)** | Высокий |
| **Название тестирования/Имя** | Публикация модели с валидацией полей формы (Название, Описание, Превью модели, Цена) |
| **Резюме испытания** | 1. Публикация модели с пройденной процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели); 2. Проверка невозможности публикации модели с не пройдённой процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Описание; 4. Ввод поля Стоимость модели; 5. Нажатие кнопки Опубликовать;   Для каждого случая некорректных полей формы   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Стоимость модели; 4. Нажатие кнопки Опубликовать; |
| **Данные тестирования** | Для корректных полей формы:   1. Название: Присутствует 2. Картинка: Присутствует 3. Стоимость модели: 50руб.   Некорректные поля формы случай 2:   1. Название: Присутствует 2. Картинка: 3. Стоимость модели: 50руб. |
| **Ожидаемый результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Предпосылки** | 1. Открытие сайта; 2. Переход по кнопке Опубликовать; |
| **Постусловия** | Выполнена навигация на форму «Публикация» |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 7. Test Case #6:

|  |  |
| --- | --- |
| **Test Case #** | TC\_PublicationTest\_3 |
| **Приоритет тестирования (Малый/Средний/высокий)** | Высокий |
| **Название тестирования/Имя** | Публикация модели с валидацией полей формы (Название, Описание, Превью модели, Цена) |
| **Резюме испытания** | 1. Публикация модели с пройденной процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели); 2. Проверка невозможности публикации модели с не пройдённой процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Описание; 4. Ввод поля Стоимость модели; 5. Нажатие кнопки Опубликовать;   Для каждого случая некорректных полей формы   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Стоимость модели; 4. Нажатие кнопки Опубликовать; |
| **Данные тестирования** | Для корректных полей формы:   1. Название: Присутствует 2. Картинка: Присутствует 3. Стоимость модели: 50руб.   Некорректные поля формы случай 3:   1. Название: Присутствует 2. Картинка: Присутствует 3. Стоимость модели: -50руб. |
| **Ожидаемый результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Предпосылки** | 1. Открытие сайта; 2. Переход по кнопке Опубликовать; |
| **Постусловия** | Выполнена навигация на форму «Публикация» |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 8. Test Case #7:

|  |  |
| --- | --- |
| **Test Case #** | TC\_PublicationTest\_4 |
| **Приоритет тестирования (Малый/Средний/высокий)** | Высокий |
| **Название тестирования/Имя** | Публикация модели с валидацией полей формы (Название, Описание, Превью модели, Цена) |
| **Резюме испытания** | 1. Публикация модели с пройденной процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели); 2. Проверка невозможности публикации модели с не пройдённой процедурой валидации полей формы (Название, Описание, Превью модели, Стоимость модели) |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Описание; 4. Ввод поля Стоимость модели; 5. Нажатие кнопки Опубликовать;   Для каждого случая некорректных полей формы   1. Ввод поля Название; 2. Выбор картинки; 3. Ввод поля Стоимость модели; 4. Нажатие кнопки Опубликовать; |
| **Данные тестирования** | Для корректных полей формы:   1. Название: Присутствует 2. Картинка: Присутствует 3. Стоимость модели: 50руб.   Некорректные поля формы случай 4:   1. Название: 2. Картинка: 3. Стоимость модели: |
| **Ожидаемый результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Публикация модели на сайт;   Для некорректных полей формы:   1. Невозможность публикации модели на сайт 2. Вывод сообщения об ошибке; |
| **Предпосылки** | 1. Открытие сайта; 2. Переход по кнопке Опубликовать; |
| **Постусловия** | Выполнена навигация на форму «Публикация» |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** |  |

Таблица 9. Test Case #8:

|  |  |
| --- | --- |
| **Test Case #** | TC\_RegistrationTest\_1 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка регистрации нового пользователя с валидацией полей формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения) |
| **Резюме испытания** | 1. Проверка регистрации нового участника бегуна с корректными данными формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения); 2. Проверка невозможности регистрации нового пользователя с некорректными данными формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения); |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля E-mail; 2. Ввод поля пароль; 3. Ввод поля имя; 4. Нажатие кнопки Регистрация   Для каждого случая некорректных полей формы   1. Ввод поля E-mail; 2. Ввод поля пароль; 3. Ввод поля имя; 4. Нажатие кнопки Регистрация |
| **Данные тестирования** | Для корректных полей формы:   1. E-mail: rasska@mail.ru 2. Пароль: 2rR$z722 3. Имя: Иван   Некорректные поля формы случай 1:   1. E-mail: rasska 2. Пароль: 2rR$z722 3. Имя: Иван |
| **Ожидаемый результат** | Для корректных полей формы:   1. Регистрация нового пользователя;   Для некорректных полей формы:   1. Невозможность регистрации нового пользователя 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Регистрация нового пользователя;   Для некорректных полей формы:   1. Невозможность регистрации нового пользователя 2. Вывод сообщение об ошибке; |
| **Предпосылки** | 1. Запуск приложения; 2. Переход по кнопке Регистрация; |
| **Постусловия** | Выполнена навигация согласно роли авторизованного пользователя/форма авторизации |
| **Статус (Pass/Fail)** | pass |

Таблица 10. Test Case #9:

|  |  |
| --- | --- |
| **Test Case #** | TC\_RegistrationTest\_2 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка регистрации нового пользователя с валидацией полей формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения) |
| **Резюме испытания** | 1. Проверка регистрации нового участника бегуна с корректными данными формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения); 2. Проверка невозможности регистрации нового пользователя с некорректными данными формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения); |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля E-mail; 2. Ввод поля пароль; 3. Ввод поля имя; 4. Нажатие кнопки Регистрация   Для каждого случая некорректных полей формы   1. Ввод поля E-mail; 2. Ввод поля пароль; 3. Ввод поля имя; 4. Нажатие кнопки Регистрация |
| **Данные тестирования** | Для корректных полей формы:   1. E-mail: rasska@mail.ru 2. Пароль: 2rR$z722 3. Имя: Иван   Некорректные поля формы случай 2:   1. E-mail: rasska@mail.ru 2. Пароль: 132 3. Имя: Иван |
| **Ожидаемый результат** | Для корректных полей формы:   1. Регистрация нового пользователя;   Для некорректных полей формы:   1. Невозможность регистрации нового пользователя 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Регистрация нового пользователя;   Для некорректных полей формы:   1. Невозможность регистрации нового пользователя 2. Вывод сообщение об ошибке; |
| **Предпосылки** | 1. Запуск приложения; 2. Переход по кнопке Регистрация; |
| **Постусловия** | Выполнена навигация согласно роли авторизованного пользователя/форма авторизации |
| **Статус (Pass/Fail)** | pass |

Таблица 11. Test Case #10:

|  |  |
| --- | --- |
| **Test Case #** | TC\_RegistrationTest\_3 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка регистрации нового пользователя с валидацией полей формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения) |
| **Резюме испытания** | 1. Проверка регистрации нового участника бегуна с корректными данными формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения); 2. Проверка невозможности регистрации нового пользователя с некорректными данными формы (Email, пароль, повтор пароля, имя, фамилия, дата рождения); |
| **Шаги тестирования** | Для корректных полей формы:   1. Ввод поля E-mail; 2. Ввод поля пароль; 3. Ввод поля имя; 4. Нажатие кнопки Регистрация   Для каждого случая некорректных полей формы   1. Ввод поля E-mail; 2. Ввод поля пароль; 3. Ввод поля имя; 4. Нажатие кнопки Регистрация |
| **Данные тестирования** | Для корректных полей формы:   1. E-mail: rasska@mail.ru 2. Пароль: 2rR$z722 3. Имя: Иван   Некорректные поля формы случай 3:   1. E-mail: rasska@mail.ru 2. Пароль: 2rR$z722 3. Имя: Иван |
| **Ожидаемый результат** | Для корректных полей формы:   1. Регистрация нового пользователя;   Для некорректных полей формы:   1. Невозможность регистрации нового пользователя 2. Вывод сообщения об ошибке; |
| **Фактический результат** | Для корректных полей формы:   1. Регистрация нового пользователя;   Для некорректных полей формы:   1. Невозможность регистрации нового пользователя 2. Вывод сообщение об ошибке; |
| **Предпосылки** | 1. Запуск приложения; 2. Переход по кнопке Регистрация; |
| **Постусловия** | Выполнена навигация согласно роли авторизованного пользователя/форма авторизации |
| **Статус (Pass/Fail)** | pass |

## 2.2 Модульное тестирование

На данном этапе было произведено модульное тестирование отдельных методов и подсистем программы, с целью проверки полной работоспособности и отсутствия ошибок. Использовались стандартные средства Microsoft Visual Studio 2019 и план тестирования. Результаты тестирования представлены на Рисунке 1.

Листинг кода модульных тестов в Приложении В.

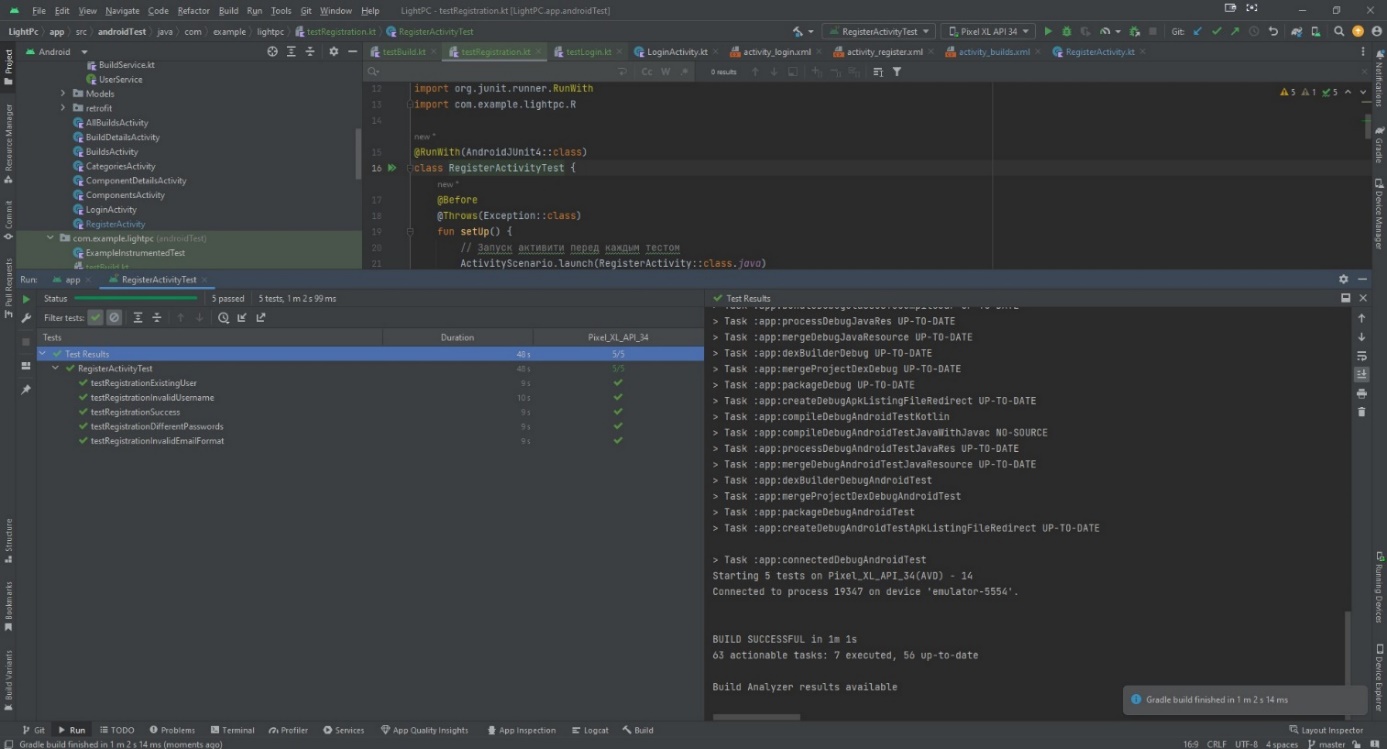


Рисунок 1. Результаты модульного тестирования

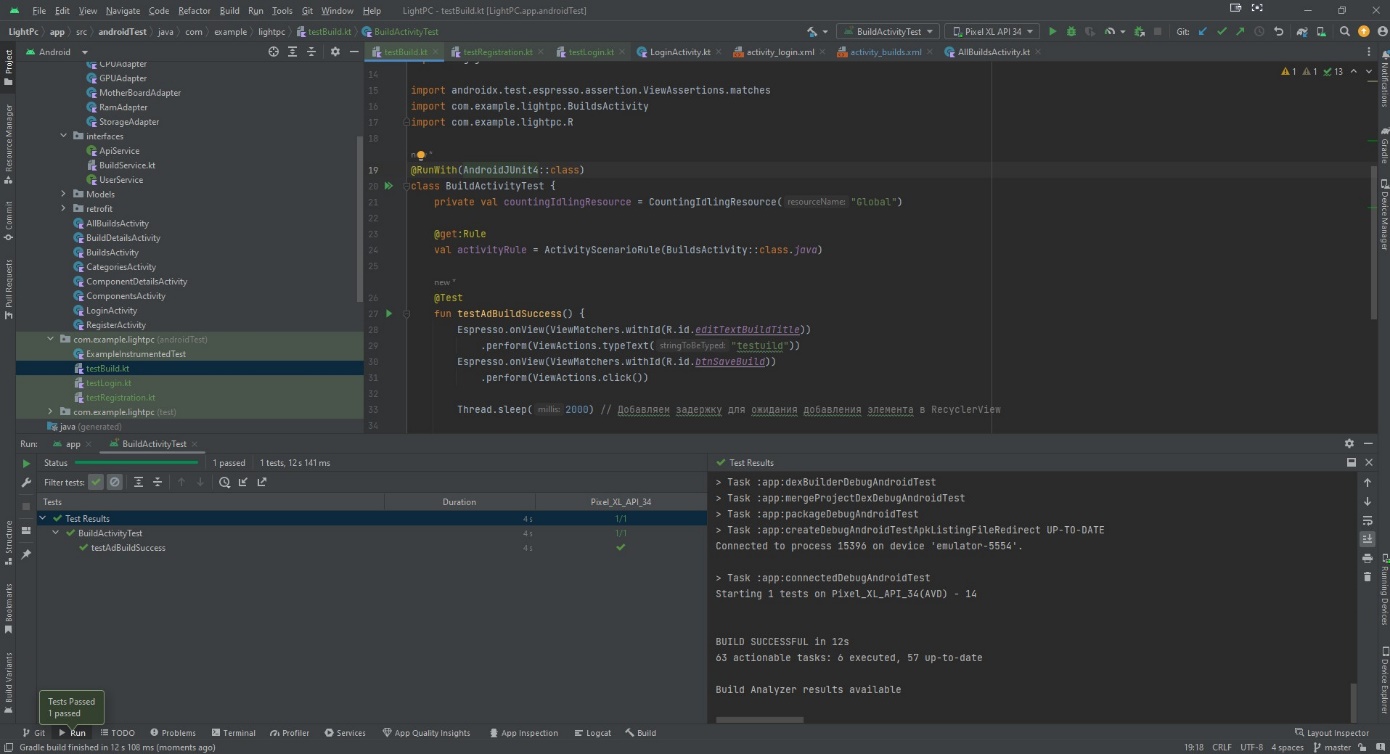


Рисунок 2. Результаты модульного тестирования

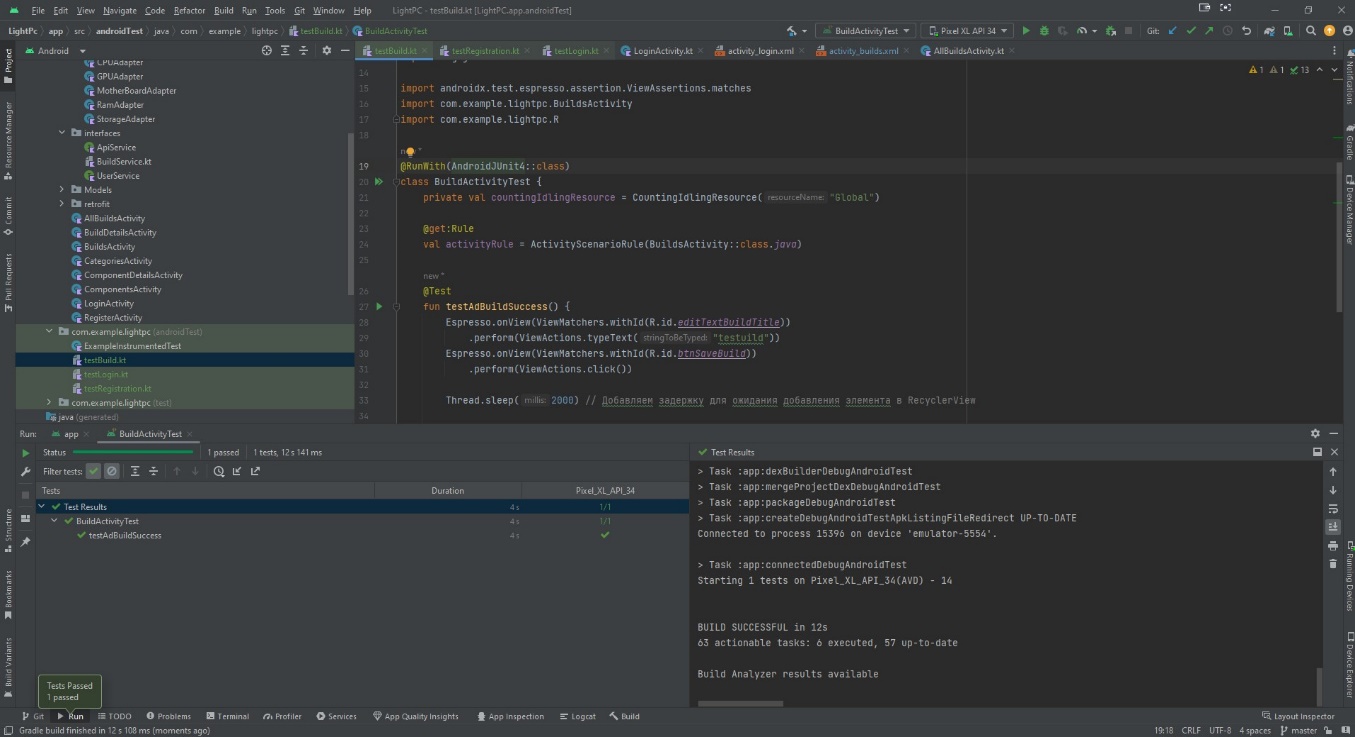


Рисунок 3. Результаты модульного тестирования

## 2.3 Нагрузочное тестирование

На данном этапе было произведено тестирование программного продукта в тяжёлых условиях, созданных искусственно. Был сымитирован сценарий большого количества одномоментных подключений к базе данных. В качестве программы для тестирования, использовалась бесплатная программа Apache JMeter.

План тестирования заключался в отправке тысяч запросов к базе, происходивших продолжительное время. Результаты тестирования предоставляются на Рисунках 2-4.

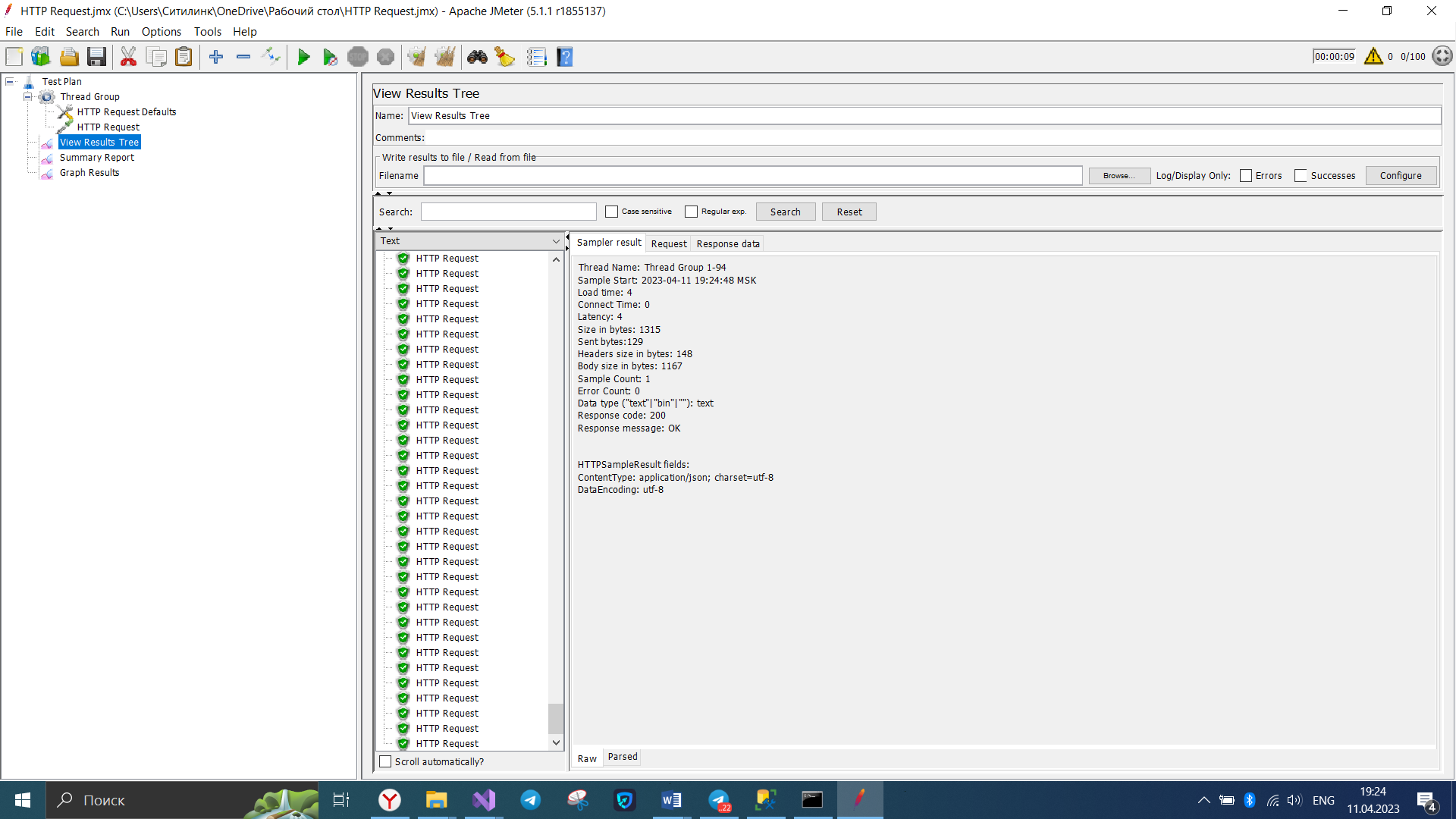


Рисунок 2. View Results Tree

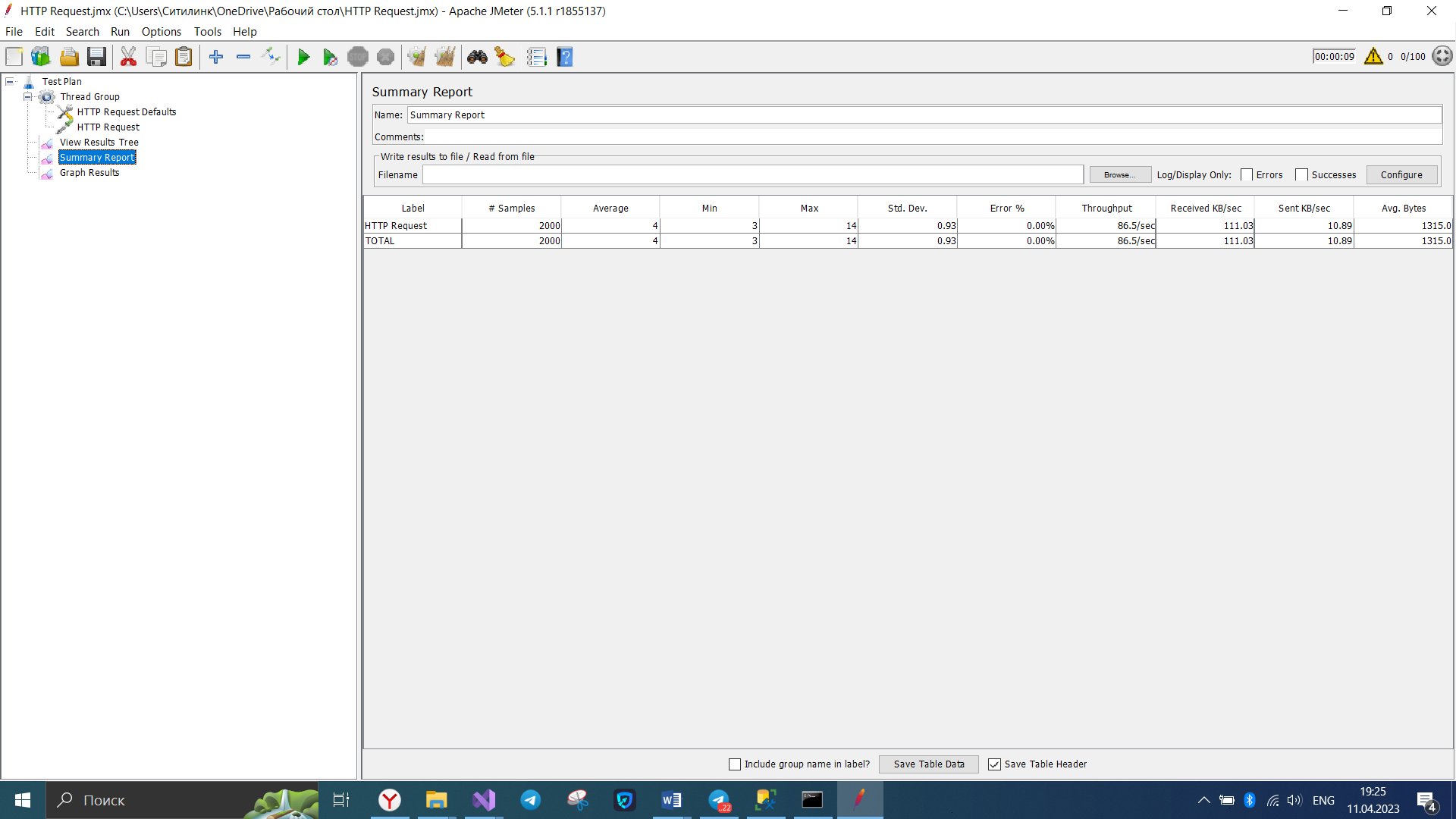


Рисунок 3. Summary Report

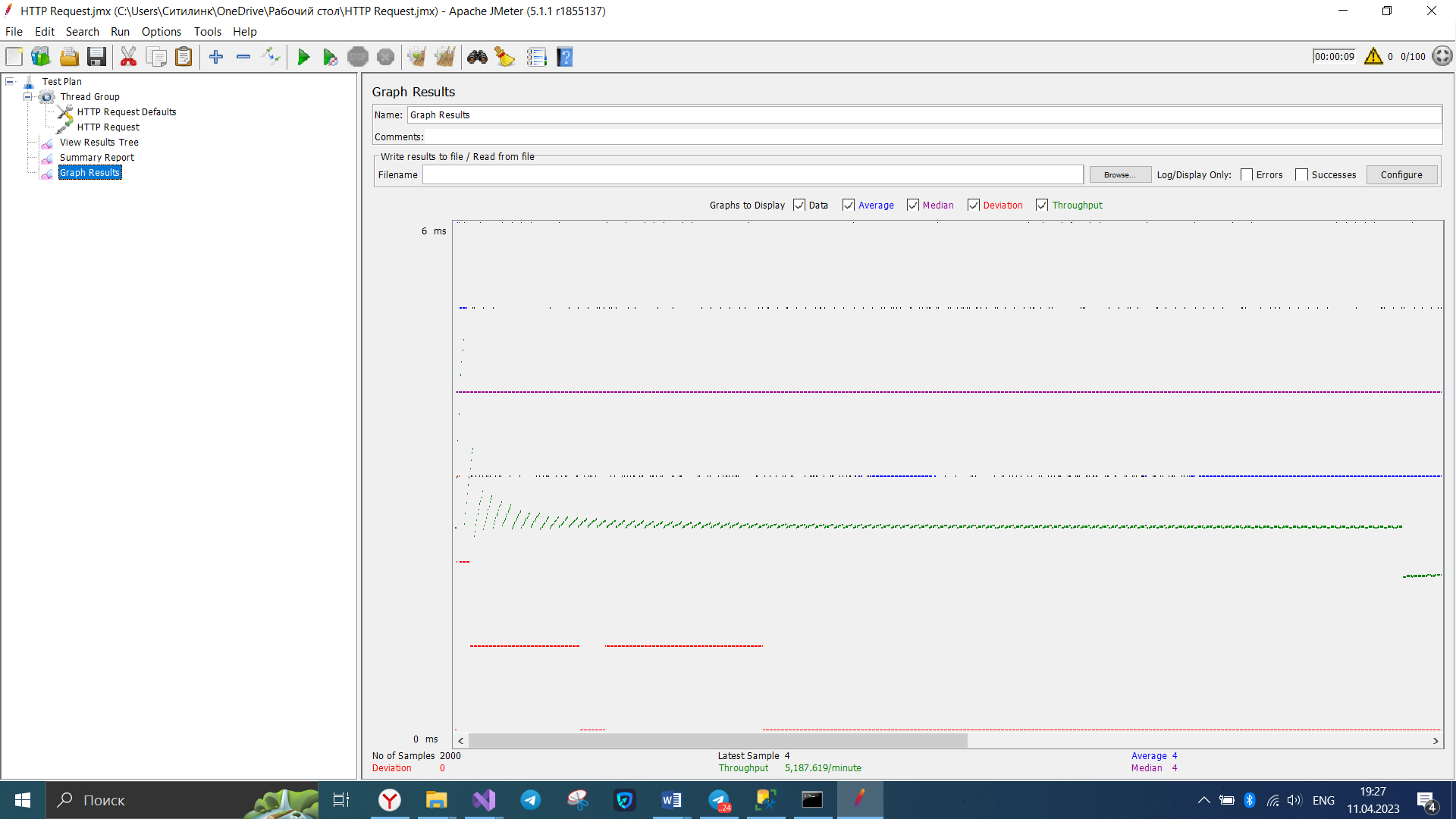


Рисунок 4. Graph Results

## Заключение

В ходе проектирования курсового проекта были решены все поставленные задачи:

* изучение особенностей конкретной предметной области, относящихся к теме «Система учета телекоммуникационного оборудования»
* анализ возможных подходов и методов решения с обоснованием выбранного метода, выбор или разработка модели (математической, структурной, информационной и т.д.), необходимой для достижения цели;
* выбор эффективных алгоритмов с учётом их точности, устойчивости, сходимости и т.д.;
* разработка структур данных;
* разработка программных модулей (авторизации, поиска и фильтрации данных, добавления, редактирования и удаления данных, формирования отчётной документации и т.д.);
* интеграция программных модулей, анализ полученных результатов работы модулей;
* тестирование программного обеспечения с целью найти уязвимости;
* разработка программной и эксплуатационной документации.

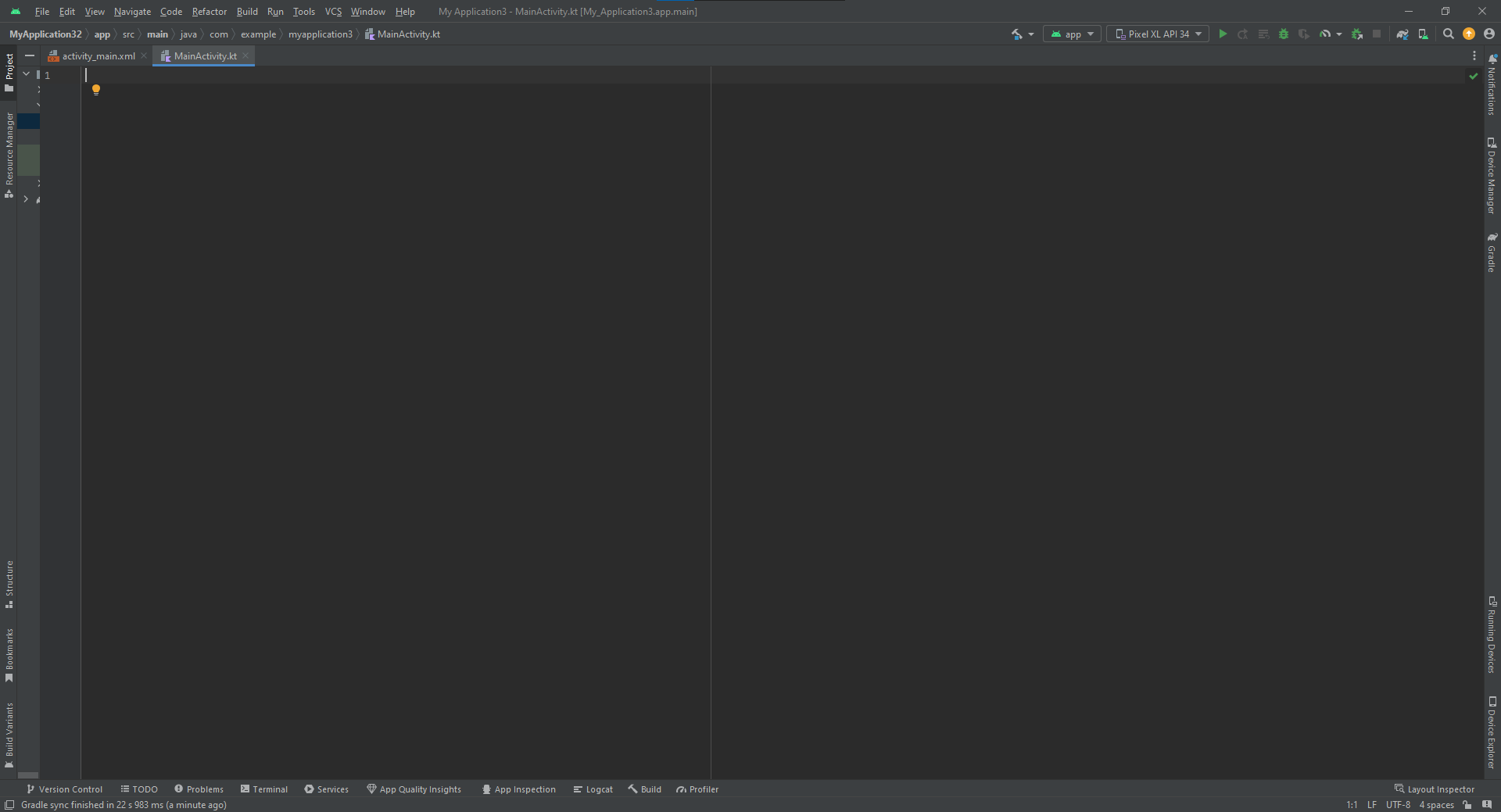
Подводя итоги, можно сказать, что разработанная программа не уступает по своим функциональным возможностям аналогам, а также в некоторых моментах превосходит их.

# Список использованной литературы

1. Ганенко А. П. и др. оформление текстовых и графических материалов при подготовке дипломных проектов и письменных экзаменационных работ (требования ЕСКД): Учеб. Пособие для нач. проф. Образования / А. П. Ганенко, Ю. В. Миловская, М. И. Лапсарь. – 2-е изд., стереотип. – М.: ИРПО; Изд. Центр «Академия», 2010. – 352 с.
2. Пьер-Ив Симон, «Волшебство Kotlin», 2020. - 418 с.
3. Пьер-Оливье Лоранс, Аманда Хинчман-Домингес, Дж.Блейк Мик, Майк Данн, «Программирование на Kotlin для Android», издание - БХВ, 2024. - 645 с.
4. Дон Гриффитс, Дэвид Гриффитс. «Head First. Kotlin», 2019. - 527 с.
5. Ияну Аделекан, «Котлин. Программирование на примерах», 2020. - 479 с.
6. Пол Дейтел, Харви Дейтел, Александер Уолд, «Android для разработчиков», 3-е издание, 2020. - 563с.
7. Зэев Фрайман, «Создание приложений для смартфонов и планшетов под ОС Android», 2023. - 387 с.
8. Денис Колисниченко, «Программирование для Android», 3-е издание, 2022. - 501 с.
9. Ян Дарвин, «Android. Сборник рецептов: задачи и решения для разработчиков приложений», 2022. - 309 с.
10. Джон Хортон, «Разработка Android-приложений с нуля», 3-е издание, 2021. - 576 с.

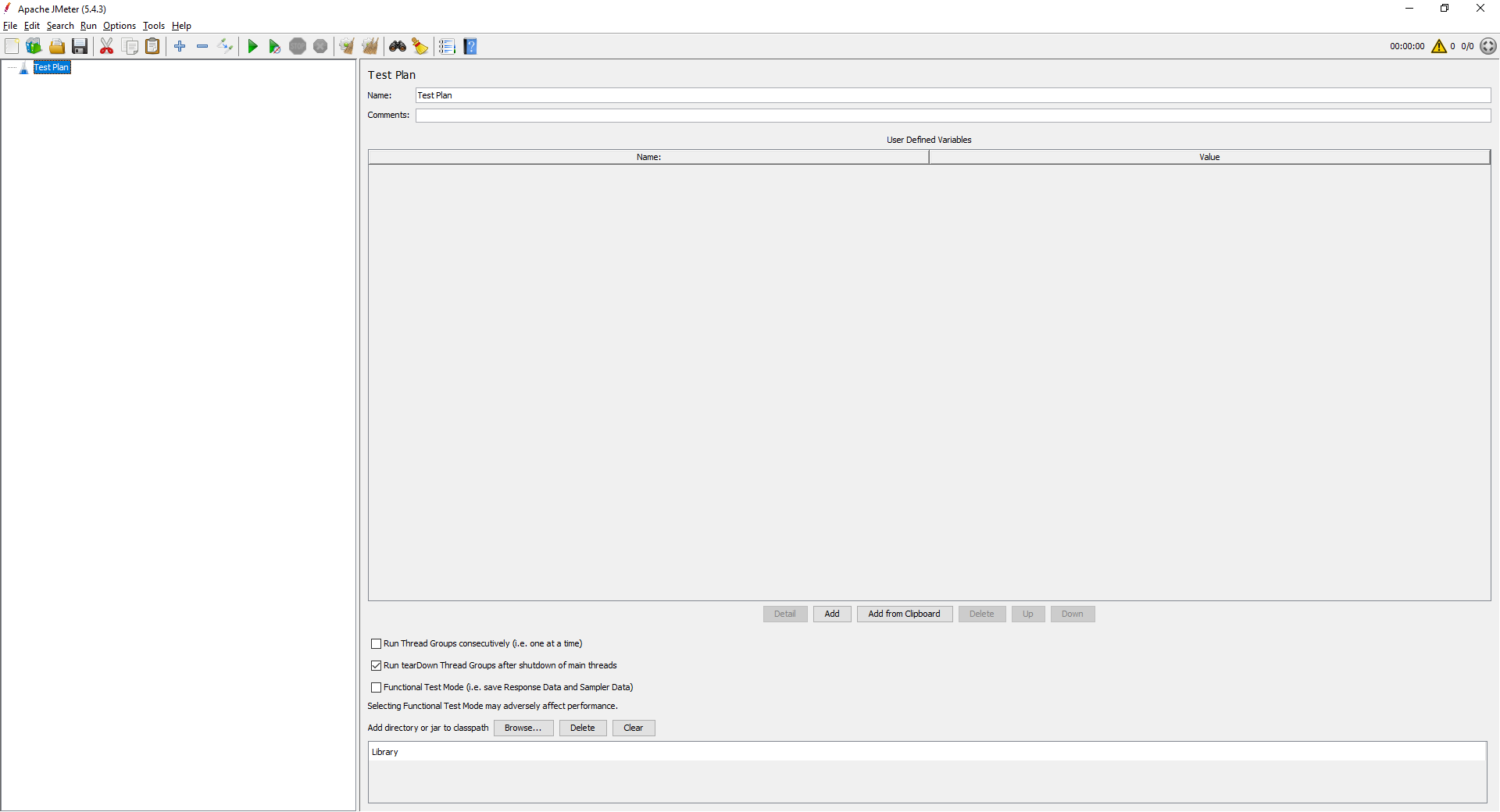
Приложение А

Интерфейс среды Android Studio



Приложение Б

Интерфейс среды Apache JMeter



Приложение В

# Листинг модульных тестов

@RunWith(AndroidJUnit4::class)  
class RegisterActivityTest {  
 @Before  
 @Throws(Exception::class)  
 fun setUp() {  
 // Запуск активити перед каждым тестом  
 ActivityScenario.launch(RegisterActivity::class.*java*)  
 }  
  
 @Test  
 fun testRegistrationSuccess() {  
 Espresso.onView(ViewMatchers.withId(R.id.nickname))  
 .perform(ViewActions.typeText("TestUser"))  
 Espresso.onView(ViewMatchers.withId(R.id.email))  
 .perform(ViewActions.typeText("test1@gmail.com"))  
 Espresso.onView(ViewMatchers.withId(R.id.username))  
 .perform(ViewActions.typeText("Test FullName1"))  
 Espresso.onView(ViewMatchers.withId(R.id.password))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.passwordConfirm))  
 .perform(ViewActions.typeText("light2d\_"), ViewActions.closeSoftKeyboard())  
 Espresso.onView(ViewMatchers.withId(R.id.radioButtonMale)).perform(ViewActions.click());  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister)).perform(ViewActions.click())  
  
 Thread.sleep(2000)  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
  
 @Test  
 fun testRegistrationExistingUser() {  
 Espresso.onView(ViewMatchers.withId(R.id.nickname))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.email))  
 .perform(ViewActions.typeText("test1@gmail.com"))  
 Espresso.onView(ViewMatchers.withId(R.id.username))  
 .perform(ViewActions.typeText("Test FullName1"))  
 Espresso.onView(ViewMatchers.withId(R.id.password))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.passwordConfirm))  
 .perform(ViewActions.typeText("light2d\_"), ViewActions.closeSoftKeyboard())  
 Espresso.onView(ViewMatchers.withId(R.id.radioButtonMale)).perform(ViewActions.click());  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister)).perform(ViewActions.click())  
  
 Thread.sleep(2000)  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
 @Test  
 fun testRegistrationInvalidEmailFormat() {  
 Espresso.onView(ViewMatchers.withId(R.id.nickname))  
 .perform(ViewActions.typeText("TestUser2"))  
 Espresso.onView(ViewMatchers.withId(R.id.email))  
 .perform(ViewActions.typeText("test2.com"))  
 Espresso.onView(ViewMatchers.withId(R.id.username))  
 .perform(ViewActions.typeText("Test FullName1"))  
 Espresso.onView(ViewMatchers.withId(R.id.password))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.passwordConfirm))  
 .perform(ViewActions.typeText("light2d\_"), ViewActions.closeSoftKeyboard())  
 Espresso.onView(ViewMatchers.withId(R.id.radioButtonMale)).perform(ViewActions.click());  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister)).perform(ViewActions.click())  
  
 Thread.sleep(2000)  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
 @Test  
 fun testRegistrationInvalidUsername() {  
 Espresso.onView(ViewMatchers.withId(R.id.nickname))  
 .perform(ViewActions.typeText("TestUser3"))  
 Espresso.onView(ViewMatchers.withId(R.id.email))  
 .perform(ViewActions.typeText("test3@gmail.com"))  
 Espresso.onView(ViewMatchers.withId(R.id.username))  
 .perform(ViewActions.typeText("Test FullName!;&?"))  
 Espresso.onView(ViewMatchers.withId(R.id.password))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.passwordConfirm))  
 .perform(ViewActions.typeText("light2d\_"), ViewActions.closeSoftKeyboard())  
 Espresso.onView(ViewMatchers.withId(R.id.radioButtonMale)).perform(ViewActions.click());  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister)).perform(ViewActions.click())  
  
 Thread.sleep(2000)  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
 @Test  
 fun testRegistrationDifferentPasswords() {  
 Espresso.onView(ViewMatchers.withId(R.id.nickname))  
 .perform(ViewActions.typeText("TestUser4"))  
 Espresso.onView(ViewMatchers.withId(R.id.email))  
 .perform(ViewActions.typeText("test4@gmail.com"))  
 Espresso.onView(ViewMatchers.withId(R.id.username))  
 .perform(ViewActions.typeText("Test FullName1"))  
 Espresso.onView(ViewMatchers.withId(R.id.password))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.passwordConfirm))  
 .perform(ViewActions.typeText("light2d"), ViewActions.closeSoftKeyboard())  
 Espresso.onView(ViewMatchers.withId(R.id.radioButtonMale)).perform(ViewActions.click());  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister)).perform(ViewActions.click())  
  
 Thread.sleep(2000)  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnRegister))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
}

@RunWith(AndroidJUnit4::class)  
class LoginActivityTest {  
 private val countingIdlingResource = CountingIdlingResource("Global")  
  
 @get:Rule  
 val activityRule = ActivityScenarioRule(LoginActivity::class.*java*)  
  
 @Test  
 fun testLoginSuccess() {  
 Espresso.onView(ViewMatchers.withId(R.id.loginUsername))  
 .perform(ViewActions.typeText("light2d\_"))  
 Espresso.onView(ViewMatchers.withId(R.id.loginPassword))  
 .perform(ViewActions.typeText("light2d!"))  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
 IdlingRegistry.getInstance().register(countingIdlingResource)  
  
 Thread.sleep(3000)  
  
 Espresso.onView(ViewMatchers.withId(R.id.loginImage))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 IdlingRegistry.getInstance().unregister(countingIdlingResource)  
 }  
  
 fun testLoginFailureLogin() {  
 Espresso.onView(ViewMatchers.withId(R.id.loginUsername))  
 .perform(ViewActions.typeText("wrong\_user"))  
 Espresso.onView(ViewMatchers.withId(R.id.loginPassword))  
 .perform(ViewActions.typeText("light2d\_"))  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
 }  
  
 fun testLoginFailurePassword() {  
 Espresso.onView(ViewMatchers.withId(R.id.loginUsername))  
 .perform(ViewActions.typeText("light"))  
 Espresso.onView(ViewMatchers.withId(R.id.loginPassword))  
 .perform(ViewActions.typeText("wrong\_password"))  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
  
 @Test  
 fun testLoginFailure() {  
 // Вводим неправильные имя пользователя и пароль  
 Espresso.onView(ViewMatchers.withId(R.id.loginUsername))  
 .perform(ViewActions.typeText("wrong\_user"))  
 Espresso.onView(ViewMatchers.withId(R.id.loginPassword))  
 .perform(ViewActions.typeText("wrong\_password"))  
  
 // Нажимаем на кнопку входа  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
 // Проверяем  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
  
 @Test  
 fun testLoginEmptyLogin() {  
 // Вводим неправильные имя пользователя и пароль  
 Espresso.onView(ViewMatchers.withId(R.id.loginPassword))  
 .perform(ViewActions.typeText("wrong\_password"))  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
  
 @Test  
 fun testLoginEmptyPassword() {  
 Espresso.onView(ViewMatchers.withId(R.id.loginUsername))  
 .perform(ViewActions.typeText("light"))  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
  
 @Test  
 fun testLoginEmpty() {  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .perform(ViewActions.click())  
  
 Espresso.onView(ViewMatchers.withId(R.id.btnLogin))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 }  
}

@RunWith(AndroidJUnit4::class)  
class BuildActivityTest {  
 private val countingIdlingResource = CountingIdlingResource("Global")  
  
 @get:Rule  
 val activityRule = ActivityScenarioRule(BuildsActivity::class.*java*)  
  
 @Test  
 fun testAdBuildSuccess() {  
 Espresso.onView(ViewMatchers.withId(R.id.editTextBuildTitle))  
 .perform(ViewActions.typeText("testuild"))  
 Espresso.onView(ViewMatchers.withId(R.id.btnSaveBuild))  
 .perform(ViewActions.click())  
  
 Thread.sleep(2000) // Добавляем задержку для ожидания добавления элемента в RecyclerView  
  
 // Проверяем, что элемент с указанным текстом действительно отображается в RecyclerView  
 Espresso.onView(ViewMatchers.withText("testuild")).check(matches(ViewMatchers.isDisplayed()))  
  
 IdlingRegistry.getInstance().register(countingIdlingResource)  
  
 Espresso.onView(ViewMatchers.withId(R.id.buildsLayout))  
 .check(ViewAssertions.matches(ViewMatchers.isDisplayed()))  
  
 IdlingRegistry.getInstance().unregister(countingIdlingResource)  
 }  
}