

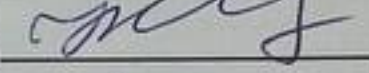
Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НАУКИ И  
ТЕХНОЛОГИЙ ИМ. АКАДЕМИКА М.Ф.РЕШЕТНЕВА

Институт информатики и телекоммуникаций  
Кафедра информационно-управляющих систем

Разработка мобильного приложения "GifStore"  
Пояснительная записка


Руководитель:

Киреев Н.В.   
(подпись)

удовл., 16.12. 2022г.  
(оценка, дата)

Выполнил:

Студент группы БИМ19-01

Здоровец П. С.   
(подпись)

Красноярск, 2022



**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ ПО ДИСЦИПЛИНЕ  
РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ**

Студент Здоровец Полина Сергеевна

Факультет ИИТК

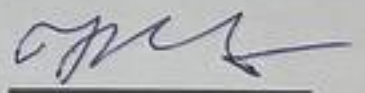
Группа БИМ 19-01

Тема курсового проекта: Разработка мобильного приложения  
“GifStore”

**Общая структурная схема курсового проекта**

1. Введение
2. Техническое задание
3. Анализ предметной области
4. Использование средств и инструментов разработки
5. Разработка мобильного приложения
6. Руководство пользователя
7. Заключение

Задание выдано



Руководитель Киреев Н.В.

## Реферат

Пояснительная записка включает в себя 14 страниц текста, 3 рисунка, 8 использованных литературных источников. Результатом данного курсового проекта является спроектированное и разработанное мобильное приложение по поиску и просмотру gif-изображений.

Целью курсового проекта является разработка технического задания и дальнейшей разработки мобильного android приложения по поиску и просмотру gif-изображений на основе знаний и навыков, приобретенных при изучении дисциплины, лекционных и лабораторно-практических занятий, а также использовании контроля версий Git и размещении проекта на GitHub.

В соответствии с поставленной целью были выявлены необходимые к реализации задачи:

1. Составление технического задания;
2. Приложение должно быть написано на языке Kotlin в программе Android Studio;
3. Возможность отображения элементов как в портретном, так и в ландшафтном режимах;
4. Использование технологии REST API для HTTP запросов;
5. Использование Room Database для хранения загруженных gif изображений;
6. Использовать обработку JSON;
7. Использование списков RecyclerView.

Средами разработки и эксплуатации программного продукта являются IDE Android Studio, Android OS.

Ключевые слова: Android, Android Studio, Java, Kotlin, Room, Интернет, Данные, Списки, Дизайн, ТЗ, Техническое задание, Язык программирования, Среда разработки, База данных, Мобильное приложение.

## Оглавление

Введение .....	5
1. Техническое задание.....	6
1.1 Описание.....	6
1.2 Интерфейс приложения .....	6
1.4 Описание используемых библиотек .....	7
2. Анализ предметной области. ....	8
2.1 Работа элементов в фоновом режиме. ....	8
2.2 Использование метода уведомлений «Notifications». ....	8
3. Разработка мобильного приложения .....	10
Заключение.....	11
Библиографический список.....	14

## Введение

С развитием и повсеместным внедрением информационных технологий современные смартфоны практически полностью способны заменить любую деятельность человека. Современные смартфоны обладают рядом преимуществ, такими как удельная стоимость хранения, размер носителя, удобство в изменении, копировании и передаче информации, возможность представления информации в удобном для конечного пользователя виде, хранение большого количества информации на относительно небольшом по размеру устройстве, защита информации от несанкционированного доступа.

Благодаря этим преимуществам электронные устройства успешно выполняют поставленные перед ними функции. Например, приложение для записи заметок легко заменит ежедневник. При этом записи можно просматривать на других устройствах, к примеру, на компьютере или ноутбуке. Информацию можно хранить в облачном хранилище, при этом даже потеряв свое устройство, данные можно восстановить.

В настоящее же время мобильные телефоны прочно укрепились в нашей повседневной жизни благодаря широкому спектру функций. Смартфон в наши дни это, прежде всего: камера, доступ в Интернет и звонки. Поскольку сейчас, в силу широкого распространения занятости людей, было бы не плохо создать приложение для отслеживания своих задач и дел. В этом и заключается актуальность такого продукта.

Смартфоны на базе Android становятся все более популярными. Широко известен тот факт, что самую большую долю мобильной индустрии занимают операционные системы (ОС) iOS и Android.

# 1. Техническое задание

## 1.1 Описание

Приложение задумывается как приложение для поиска и сохранения gif изображений, а также возможности поделиться ими.

В главном окне приложения представлены: поисковая строка, кнопка загрузки картинок по запросу, кнопка загрузки картинок из базы данных. При нажатии на картинку, загруженную с помощью онлайн запроса, она будет сохранена в базу данных, а также всплывёт соответствующее уведомление. При выходе из приложения в фоновом процессе начинается загрузка популярных картинок с возможностью будущего их просмотра.

## 1.2 Интерфейс приложения

На рисунке 1 изображен главный экран, на котором можно загрузить картинки по запросу из интернета или из базы данных.

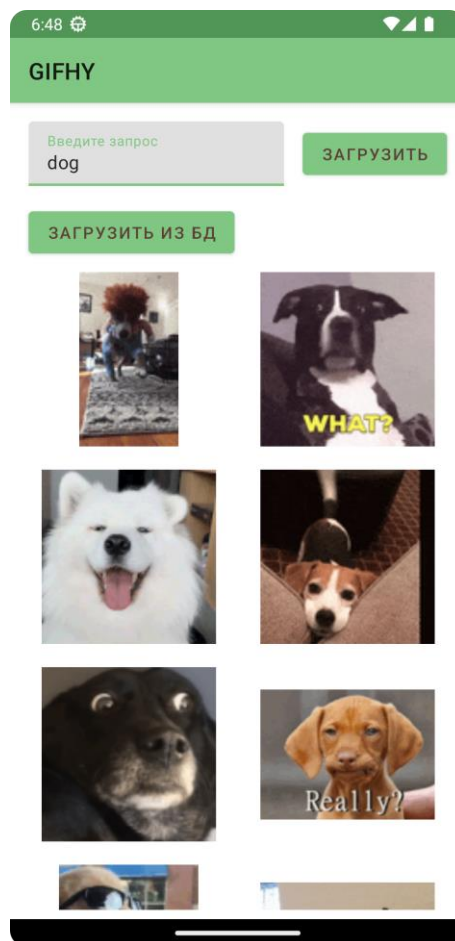


Рисунок 1. Главное экран приложения (горизонтальный режим)

Набрав в поиске необходимую тематику картинок необходимо нажать на кнопку загрузить. После чего появятся картинки, подходящие по тематике. Выбрав понравившиеся картинки, они автоматически добавляются в базу данных. Если необходимо вывести список картинок, который был добавлен в базу данных необходимо нажать «Загрузить из БД» (Рисунок 2).

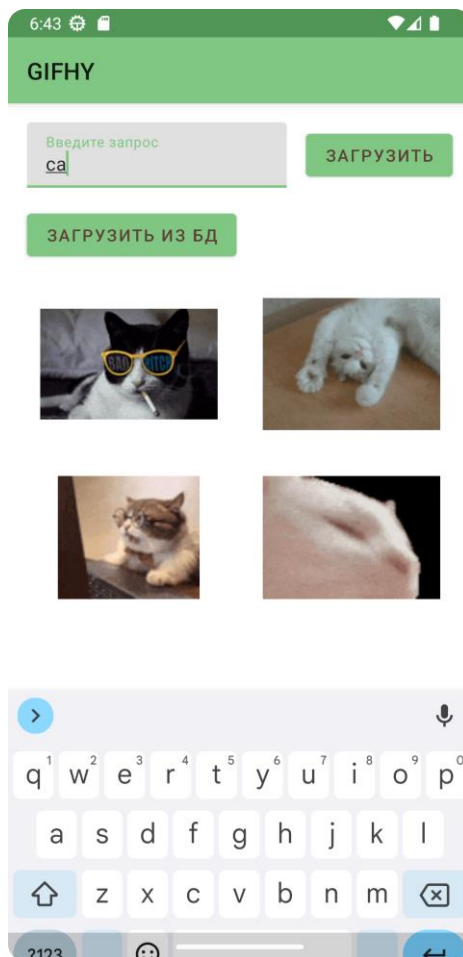


Рисунок 2. Экран с загруженными из интернета изображениями

### 1.3 Ключевые аспекты разработки приложения

Для разработки приложения использовались: язык Kotlin и Android Studio.

### 1.4 Описание используемых библиотек

Приложение использует коллекцию библиотек компонентов архитектуры Android: JSON, Glide, Room Database, RecyclerView, Lifecycle.

## 2. Анализ предметной области.

### 2.1 Использование метода Lifecycle

Lifecycle — класс, который хранит информацию про состояние жизненного цикла и разрешает другим объектам отслеживать его с помощью реализации LifecycleObserver. Состоит из методов: addObserver(LifecycleObserver), removeObserver(LifecycleObserver) и getCurrentState(). Как понятно из названий для добавления подписчика, удаления и соответственно получения текущего состояния.

Жизненный цикл приложения в Android жёстко контролируется системой и зависит от нужд пользователя, доступных ресурсов. Например, пользователь хочет запустить браузер. Решение о запуске приложения принимает система. Хотя последнее слово и остаётся за системой, она подчиняется определённым заданным и логическим правилам, позволяющим определить, можно ли загрузить, приостановить приложение или прекратить его работу. Если в данный момент пользователь работает с определённым окном, система даёт приоритет соответствующему приложению. И наоборот, если окно невидимо и система решает, что работу приложения необходимо остановить, чтобы мобилизовать дополнительные ресурсы, будет прекращена работа приложения, имеющего более низкий приоритет. В Android ресурсы более ограничены, поэтому Android более жёстко контролирует работу приложений.

### 2.2 Хранение данных в Room

Room — это новый способ сохранить данные приложений в Android-приложении. Это часть новой Android Architecture, группа библиотек от Google, которые поддерживают уместную архитектуру приложений. Room предлагается в качестве альтернативы Realm, ORMLite, GreenDao и многим другим.

Room — это высокоуровневый интерфейс для низкоуровневых привязок SQLite, встроенных в Android. Он выполняет большую часть своей работы во время компиляции, создавая API-интерфейс поверх встроенного SQLite API.

Room также позволяет наблюдать за изменениями данных, интегрируя их как с API LiveData, так и с RxJava 2. Это означает, что, если есть сложная схема, где изменения в базе данных должны появляться в нескольких местах приложения, Room делает уведомления об изменениях. Это мощное



дополнение может быть включено одной строкой. Все, что нужно сделать, это изменить тип возвращаемых значений.

### 2.3 Обработка потоков в Flow

Для асинхронной сборки приложения мы должны использовать RxJava, и это также одна из самых важных тем в разработке Android. Но теперь в Kotlin JetBrains придумали Flow API. С Flow в Kotlin теперь можно обрабатывать поток данных последовательно. Flow - это API обработки потоков в Kotlin, разработанный JetBrains. Это реализация спецификации реактивного потока, инициативы, целью которой является предоставление стандарта для асинхронного, который выполняется последовательно. JetBrains построил Kotlin Flow поверх сопрограмм Kotlin. В Kotlin сопрограмма - это просто планировщик, входящий в состав RxJava, но теперь, благодаря API-интерфейсам Flow, она может стать альтернативой RxJava в Android.

Kotlin Flow API - это лучший способ асинхронной обработки потока данных, который выполняется последовательно. Используя Flow для обработки потоков значений, можно преобразовывать данные сложными многопоточными способами, написав всего лишь небольшой фрагмент кода. В Kotlin.

### 3. Разработка мобильного приложения

Следующим этапом выполнения курсового проекта является процесс разработки. При выполнении всех этапов работы было получено большое количество теоретических и практических знаний, которые были применены в процессе создания программного продукта и которые в последствии поспособствовали эффективной разработке мобильного приложения.

В данном разделе будет описана разработка интерфейса мобильного приложения и будут описаны технологии, применяемые в процессе ее реализации.

#### 3.1 Разработка интерфейса

На рисунке 3 изображен основной layout главного окна приложения и размещение всех элементов на нём.

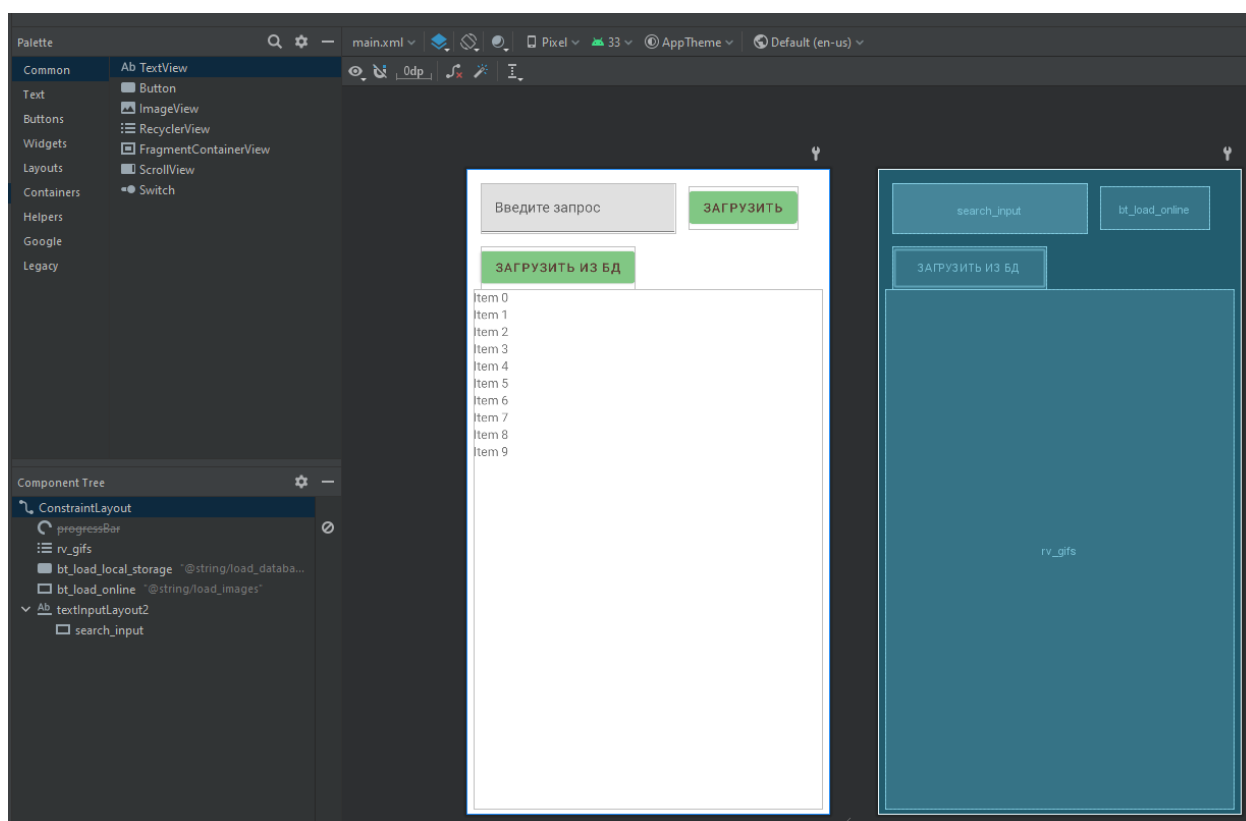


Рисунок 3. Структура основного layout.

## Далее приведен программный код Main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/ConstraintLayout"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:animateLayoutChanges="true"
tools:context=".Main.Main">

<ProgressBar
android:id="@+id/progressBar"
style="?android:attr/progressBarStyle"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:visibility="gone"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent" />

<androidx.recyclerview.widget.RecyclerView
android:id="@+id/rv_gifs"
android:layout_width="0dp"
android:layout_height="0dp"
android:layout_marginStart="8dp"
android:layout_marginEnd="8dp"
android:layout_marginBottom="8dp"
android:layout_marginTop="8dp"
android:animationCache="true"
android:saveEnabled="true"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/bt_load_local_storage"
/>

<Button
android:id="@+id/bt_load_local_storage"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="@string/load_database"
android:textColor="#5D4037"
```

```

app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.5"
app:layout_constraintStart_toEndOf="@+id/bt_load_online"
app:layout_constraintTop_toTopOf="@+id/bt_load_online" />

<com.google.android.material.button.MaterialButton
android:id="@+id/bt_load_online"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginStart="16dp"
android:text="@string/load_images"
android:textColor="#5D4037"
app:layout_constraintBottom_toBottomOf="@+id/textInputLayout2"
app:layout_constraintEnd_toStartOf="@+id/bt_load_local_storage"
app:layout_constraintHorizontal_bias="0.5"
app:layout_constraintStart_toEndOf="@+id/textInputLayout2"
app:layout_constraintTop_toTopOf="@+id/textInputLayout2" />

<com.google.android.material.textfield.TextInputLayout
android:id="@+id/textInputLayout2"
android:layout_width="220dp"
android:layout_height="wrap_content"
android:layout_marginStart="16dp"
android:layout_marginTop="16dp"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent">

<com.google.android.material.textfield.TextInputEditText
android:id="@+id/search_input"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="@string/hint_search" />
</com.google.android.material.textfield.TextInputLayout>
</androidx.constraintlayout.widget.ConstraintLayout>

```

## Заключение

В результате выполнения курсовой работы были изучены и применены на практике методы разработки технического задания, мобильных Android приложений, а также использования системы контроля версий Git.

В результате выполнения курсового проекта была реализована поставленная цель и сопутствующие ей задачи.

На этапе анализа предметной области были выявлены основные критерии, показатели и особенности работы с программными средствами и продуктами.

На этапе изучения программных продуктов были выявлены основные функциональные особенности, особенности работы каждой программной среды и технологии разработки мобильных приложений, которые были применены для успешного создания технического задания, разработки мобильного приложения и применения технологии контроля версий Git.

В процессе разработки была реализована программа по поиску Gif изображений и их просмотра, и сохранения.

Применена система контроля версий Git для хостинга IT-проектов и их совместной разработке.



## Библиографический список

1. Android [Электронный ресурс] // Википедия: свободная энциклопедия. - Режим доступа: <https://ru.wikipedia.org/wiki/Android> - Загл. с экрана.
2. Android Studio [Электронный ресурс] // Developers: сайт. - Режим доступа: <https://developer.android.com/studio> - Загл. с экрана.
3. Android Studio [Электронный ресурс] // Википедия: свободная энциклопедия. - Режим доступа: [https://ru.wikipedia.org/wiki/Android\\_Studio](https://ru.wikipedia.org/wiki/Android_Studio) - Загл. с экрана.
4. GitHub [Электронный ресурс] // Википедия: свободная энциклопедия. - Режим доступа: <https://ru.wikipedia.org/wiki/GitHub> - Загл. с экрана.
5. RecyclerView [Электронный ресурс] // Хабр. - Режим доступа: <https://habr.com/ru/post/237101/> - Загл. с экрана.
6. Адаптивный дизайн [Электронный ресурс] // Google поиск. - Режим доступа: <https://developers.google.com/search/mobile-sites/mobile-seo/responsive-design?hl=ru> - Загл. с экрана.
7. База данных [Электронный ресурс] // Википедия: свободная энциклопедия. - Режим доступа: [https://ru.wikipedia.org/wiki/База\\_данных](https://ru.wikipedia.org/wiki/База_данных) - Загл. с экрана.
8. Техническое задание [Электронный ресурс] // Википедия: свободная энциклопедия. - Режим доступа: [https://ru.wikipedia.org/wiki/Техническое\\_задание](https://ru.wikipedia.org/wiki/Техническое_задание) - Загл. с экрана.