Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Колледж информатики и программирования

Специальность 09.02.07 Информационные системы и программирование

**ОТЧЕТ**

**ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

Профессиональный модульПМ.02 Осуществление интеграции

программных модулей

*(наименование профессионального модуля)*

Выполнил:

обучающийся учебной группы № 4ИСИП-619

П. А. Подлегаев

*(И.О. Фамилия)*

Проверил:

руководитель практики от колледжа:

О.Г. Пташкин

*(И.О. Фамилия)*

**Москва**

**20­­­­23**

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc134505933)

[ГЛАВА 1. ПРЕДПРОЕКТНОЕ ИССЛЕДОВАНИЕ 5](#_Toc134505934)

[1.1 Описание предметной области 5](#_Toc134505935)

[1.2 Сравнительный анализ программ аналогов 5](#_Toc134505936)

[1.3 Постановка задачи 7](#_Toc134505937)

[1.4 Характеристика инструментальных средств 8](#_Toc134505938)

[1.5 Обоснование выбора инструментальных средств разработки 10](#_Toc134505939)

[ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ 12](#_Toc134505940)

[2.1 Анализ требований и определение спецификаций ПО 12](#_Toc134505941)

[2.2 Проектирование программного обеспечения 15](#_Toc134505942)

[2.3 Разработка программного обеспечения 16](#_Toc134505943)

[2.4 Отладка и тестирование программы 35](#_Toc134505944)

[2.5 Руководство по использованию программы 35](#_Toc134505945)

[ЗАКЛЮЧЕНИЕ 38](#_Toc134505946)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 39](#_Toc134505947)

[ПРИЛОЖЕНИЕ 40](#_Toc134505948)

# ВВЕДЕНИЕ

В современном мире люди всё чаще сталкиваются с необходимостью общаться на разных языках. Связанное с этим повышение спроса на услуги переводчиков определяет важность создания новых инструментов, которые помогут людям переводить тексты быстрее и эффективнее, а мультиплатформенность переводчика позволит пользователям получать перевод на разных устройствах и в любой ситуации. К тому же функция распознавания текста становится все более востребованной, так как даёт возможность легко переводить текст, который находится вне интернета, например, фотографируя текст на бумажном документе. Исходя из вышеизложенного, можно заключить, что разработка мультиплатформенного переводчика с распознаванием текста является актуальной задачей, которая будет полезна для широкой аудитории пользователей в современном мире.

Целью проекта является разработка мультиплатформенного переводчика с распознаванием текста.

Для достижения поставленной цели были определены следующие задачи:

* проанализировать существующие мультиплатформенные переводчики и системы распознавания текста;
* определить инструментальную базу для разработки приложения;
* изучить требования и спецификацию ПО;
* разработать архитектуру мультиплатформенного переводчика с распознаванием текста;
* реализовать разработанную архитектуру;
* провести отладку и тестирование программы;
* разработать руководство по использованию программы.

Объектом дипломного проекта являются мультиплатформенные приложения-переводчики.

Предметом проекта являются инструменты и технологии разработки программного обеспечения мультиплатформенного переводчика с распознаванием текста.

Одной из особенностей предлагаемого переводчика является его мультиплатформенность. Это означает, что может быть разработано под различные операционные системы, таких как Windows и Android, что значительно расширяет круг потенциальных пользователей.

Для поставленных задач будут применяться следующие

инструментальные средства, языки проектирования и разработки

программного обеспечения:

* язык программирования C#;
* язык программирования Kotlin;
* интегрированная среда разработки Visual Studio 2022;
* интегрированная среда разработки Android Studio;
* компактная встраиваемая СУБД SQLite;
* сервис для проектирования диаграмм Draw.io;
* текстовый редактор Microsoft Word;
* программа подготовки и просмотра презентаций Microsoft PowerPoint.

Таким образом будут описываться процессы разработки мультиплатформенного переводчика с распознаванием текста, используя .net framework для desktop-приложения и Android SDK для мобильного приложения. В данной работе будут использоваться следующие инструменты:

* Visual Studio(.net framework) для desktop-приложения;
* API Google Переводчика;
* Android studio для мобильного приложения;
* ML Kit для машинного перевода текста
* Методы распознавания текста.

# ГЛАВА 1. ПРЕДПРОЕКТНОЕ ИССЛЕДОВАНИЕ

## 1.1 Описание предметной области

В качестве предметной области данной работы выбрано приложение переводчик для операционной системы Android, основная функция которого - перевод текста на один из выбранных языков. Пользователями приложения могут быть как частные лица, так и представители бизнес-среды. Для перевода можно использовать тексты частной переписки мессенджеров и электронной почты, материалы сайтов, тексты документов, и другие типы текстов.

Переводчик предоставляет возможность людям понимать друг друга даже если они являются носители разных языков, благодаря чему языковой барьер между ними может быть преодолен.

Крайне актуальным функционалом представляется оперативный перевод текста с голоса, что сокращает время в процессе перевода и не требует отдельного ввода текста вручную. Таким образом, можно сделать вывод, что приложение должно включать в себя следующие функции:

* перевод текста;
* распознавание текста с голоса.

## 1.2 Сравнительный анализ программ аналогов

Для проведения сравнительного анализа были выбраны несколько программ-аналогов, которые имеют похожие функции и предназначены для использования на мобильных устройствах. В данном разделе будет проведен анализ каждой из этих программ по нескольким критериям.

* Качество перевода: будет проанализировано качество перевода каждой программы, с учетом точности перевода и способности распознавать различные языки.
* Удобство использования: будет оценено удобство использования каждой программы, включая интерфейс пользователя и доступность функций.
* Совместимость: будет рассмотрена совместимость каждой программы с различными операционными системами и устройствами.
* Стоимость: будет оценена стоимость каждой программы, включая цену за использование и возможные дополнительные расходы.
* Дополнительные функции: будет рассмотрено наличие дополнительных функций в каждой программе, таких как возможность использования голосового ввода или функций машинного обучения.

Программный продукт «Яндекс Переводчик»

Онлайн-переводчик «Яндекс Переводчик», который распространяется бесплатно и способен не только к переводу введенного текста, но и любого выделенного. Просматривая сайты на иностранном и, как правило, малознакомом языке очень сложно обойтись без бесплатного онлайн-переводчика.

Приложение генерирует достаточно точный перевод. Он выполняется быстро для текста небольшого объёма. Программа удобная в использовании, совместима со всеми версиями Android, начиная с Android 6. К тому же приложение бесплатное.

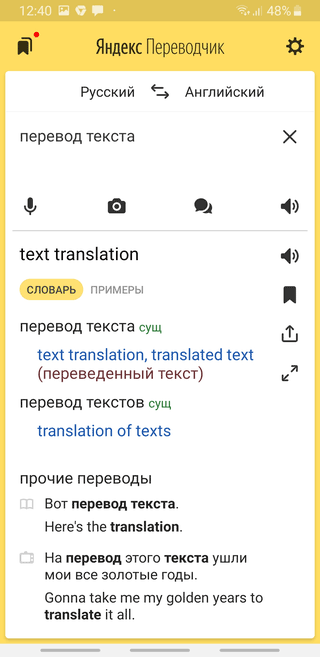


Рисунок 1. Скриншот программы «Яндекс Переводчик»

Программный продукт «DeepL»

DeepL - это программа для перевода текста и распознавания речи на различных языках, доступная для использования на Windows и Android.

Основной функцией DeepL является перевод текста с одного языка на другой. Программа поддерживает более 100 языков, включая многие редкие языки.

В приложении скорость и точность перевода зависит от объема текста. Программа компактная и удобная в использовании и совместима с Windows и Android. К дополнительным функциям можно отнести возможность пользоваться офлайн, воспроизведение переводимого и переведённого текста.

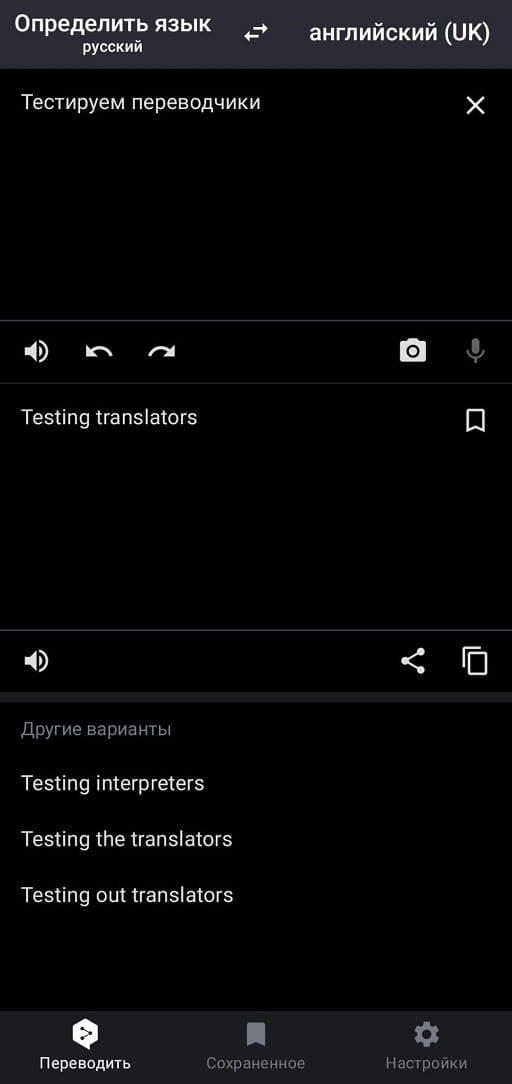


Рисунок 2. Скриншот программы-аналога «DeepL»

## Постановка задачи

При первом запуске приложения пользователя будет отображаться главное окно. По центру находится главная страница, на которой будет функция перевода, справа-сверху – кнопка троеточия, в которой будет доступно 3 раздела: раздел истории и раздел настроек.

В разделе истории пользователю будет отображаться страница, на которой будет история переводов и доступные языки. Историю переводов он сможет просматривать и/или удалять не нужную.

В разделе настроек будет доступно пользователю изменение размера шрифта, а также изменение цветовой темы приложения

Эксплуатационные требования для разрабатываемого приложения:

операционная система: Android 13,

минимальная система: Android 5,

требуемые разрешения: Доступ в интернет.

## 1.4 Характеристика инструментальных средств

Kotlin (Ко́тлин) — статически типизированный, объектно-ориентированный язык программирования, работающий поверх Java Virtual Machine и разрабатываемый компанией JetBrains. Также компилируется в JavaScript и в исполняемый код ряда платформ через инфраструктуру LLVM.

Язык назван в честь острова Котлин в Финском заливе, на котором расположен город Кронштадт. Авторы ставили целью создать язык более лаконичный и типобезопасный, чем Java, и более простой, чем Scala. Следствием упрощения по сравнению со Scala стали также более быстрая компиляция и лучшая поддержка языка в IDE. Язык полностью совместим с Java, что позволяет Java-разработчикам постепенно перейти к его использованию; в частности, язык также встраивается Android, что позволяет для существующего Android-приложения внедрять новые функции на Kotlin без переписывания приложения целиком.

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментов. Данные продукты позволяют разрабатывать как консольные приложения, так и игры и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, UWP а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, .NET Core, .NET, MAUI, Xbox, Windows Phone .NET Compact Framework и Silverlight. После покупки компании Xamarin корпорацией Microsoft появилась возможность разработки IOS и Android программ.

Android Studio — интегрированная среда разработки (IDE) для работы с платформой Android, анонсированная 16 мая 2013 года на конференции Google I/O. В последней версии Android Studio поддерживается Android 4.1 и выше. Данная IDE находилась в свободном доступе начиная с версии 0.1, опубликованной в мае 2013, а затем перешла в стадию бета-тестирования, начиная с версии 0.8, которая была выпущена в июне 2014 года. Первая стабильная версия 1.0 была выпущена в декабре 2014 года, тогда же прекратилась поддержка плагина Android Development Tools (ADT) для Eclipse.

Android Studio, основанная на программном обеспечении IntelliJ IDEA от компании JetBrains, — официальное средство разработки Android приложений[2]. Данная среда разработки доступна для Windows, macOS и GNU/Linux[3]. 17 мая 2017, на ежегодной конференции Google I/O, Google анонсировал поддержку языка Kotlin, используемого в Android Studio, как официального языка программирования для платформы Android в дополнение к Java и C++.

Draw.io — это бесплатное кроссплатформенное программное обеспечение для рисования диаграмм с открытым исходным кодом, разработанное на HTML5 и JavaScript. Его интерфейс можно использовать для создания диаграмм, таких как блок-схемы, каркасы, диаграммы UML, организационные диаграммы и сетевые диаграммы.

Draw.io доступна как онлайн в виде кросс-браузерного веб-приложения, так и в виде автономного настольного приложения для Linux, macOS и Windows.

MicrosoftWord (часто — MSWord,WinWord или просто Word) — текстовый процессор, предназначенный для создания, просмотра, редактирования и форматирования текстов статей, деловых бумаг, а также иных документов, с локальным применением простейших форм таблично-матричных алгоритмов. Выпускается корпорацией Microsoft в составе пакета Microsoft Office. Первая версия была написана Ричардом Броди (Richard Brodie) для IBM PC, использующих DOS, в 1983 году. Позднее выпускались версии для Apple Macintosh (1984), SCO UNIX и Microsoft Windows (1989). Текущей версией является Microsoft Office Word 2021 для Windows и macOS, а также веб-версия Word Online (Office Online), не требующая установки программы на компьютер.

PowerPoint (полное название — Microsoft Office PowerPoint, от англ. power point — убедительный доклад) — программа подготовки презентаций и просмотра презентаций, являющаяся частью Microsoft Office и доступная в редакциях для операционных систем Microsoft Windows и macOS, а также для мобильных платформ Android и IOS. Материалы, подготовленные с помощью PowerPoint, предназначены для отображения на большом экране — через проектор, либо телевизионный экран большого размера.

## 1.5 Обоснование выбора инструментальных средств разработки

Для разработки мультиплатформенного переводчика с распознаванием текста были выбраны инструменты и технологии, которые позволяют решать поставленные задачи и соответствуют требованиям проекта. Основные требования к проекту включают разработку приложения, работающего на нескольких платформах, высокую точность распознавания текста и качество перевода.

Для разработки приложения на нескольких платформах были выбраны такие фреймворки как: .Net Framework и Android SDK. .Net Framework был выбран из-за поддержки языка C#, на котором будет реализована desktop версия приложения «KIP Translator». Android SDK – из-за поддержки языка Kotlin для мобильной версии приложения.

Фреймворки .NET Framework и Android SDK были выбраны из-за доступности для разработчиков с разным уровнем опыта и знаний, что является важным фактором, который определил выбор данных инструментов. .NET Framework является одним из наиболее популярных инструментов для разработки приложений под Windows, обладает большим сообществом пользователей и имеет обширную документацию. Android SDK, в свою очередь, является стандартным комплектом разработки для Android-приложений и имеет множество учебных материалов и документации, что облегчает разработку для этой платформы.

В связи с тем, что фреймворки .NET Framework и Android SDK являются инструментами разработки в средах Vusial Studio и Android Studio, то для реализации проекта были выбраны именно эти среды разработки.

# ГЛАВА 2. ПРАКТИЧЕСКАЯ ЧАСТЬ

## 2.1 Анализ требований и определение спецификаций ПО

Спецификации данного проекта определяются следующими диаграммами из списка:

* ER-диаграмма;
* диаграмма классов;
* диаграмма вариантов использования;
* детализированная диаграмма потоков данных верхнего уровня;
* детализированная диаграмма потоков данных;
* краткая функциональная диаграмма.

На рисунке 3 представлена ER-диаграмма дипломного проекта по теме «Разработка мультиплатформенного переводчика с распознаванием текста», построенная с использованием Draw.io.

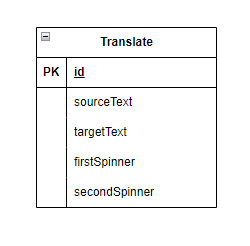


Рисунок 3. ER-диаграмма

На рисунке 4 представлена диаграмма классов дипломного проекта по теме «KIP Translator».

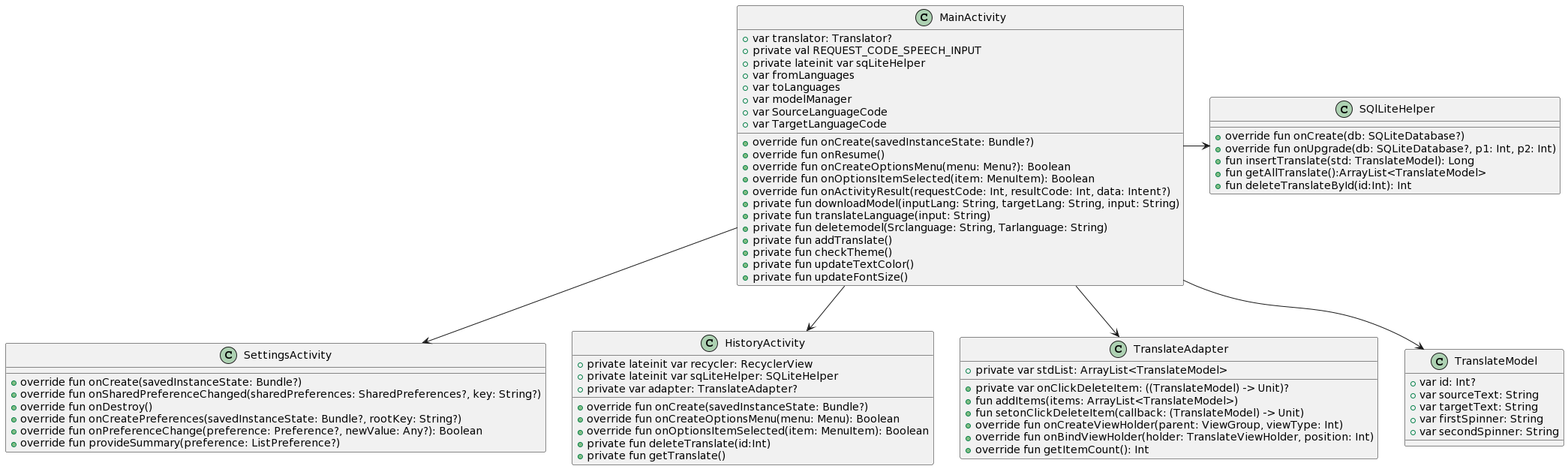


Рисунок 4. Диаграмма классов

Диаграмма вариантов использования (сценариев поведения, прецедентов) является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения (рисунок 5).

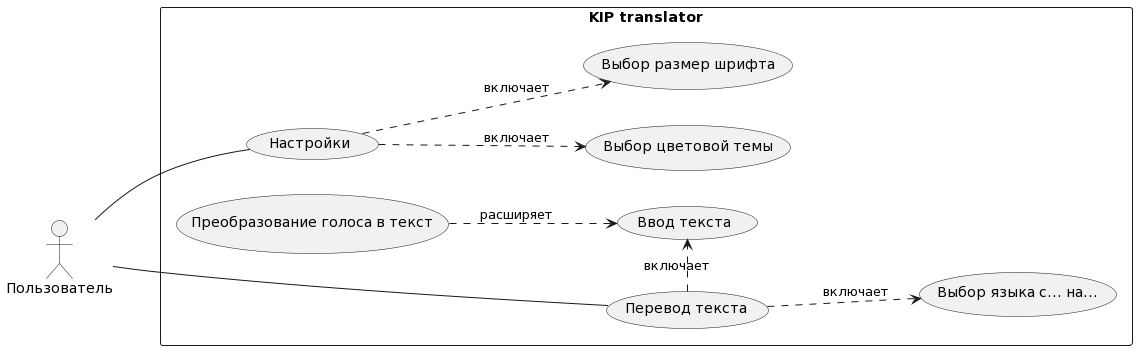


Рисунок 5. Диаграмма вариантов использования

На рисунке 6 изображена диаграмма потоков данных верхнего уровня для программного обеспечения предметной области «KIP Translator».

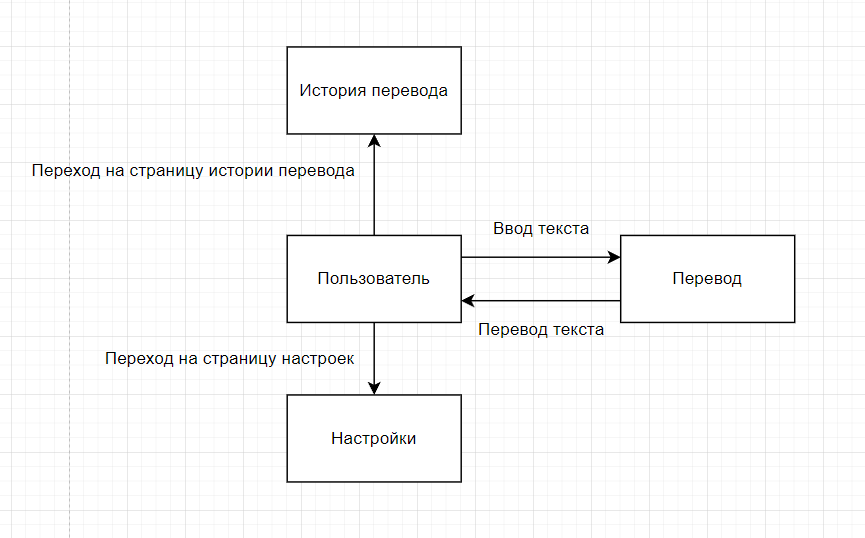


Рисунок 6. Диаграмма потоков данных верхнего уровня

На рис. 7 изображена детализированная диаграмма потоков данных для программного обеспечения предметной области «KIP Translator».



Рисунок 7. Детализированная диаграмма потоков данных

На рисунке 8 представлена краткая функциональная схема.

**Изображение выглядит как диаграмма

Автоматически созданное описание**

Рисунок 8. Краткая функциональная диаграмма

## 2.2 Проектирование программного обеспечения

На рисунке 9 изображена структурная схема программного продукта «KIP Translator».

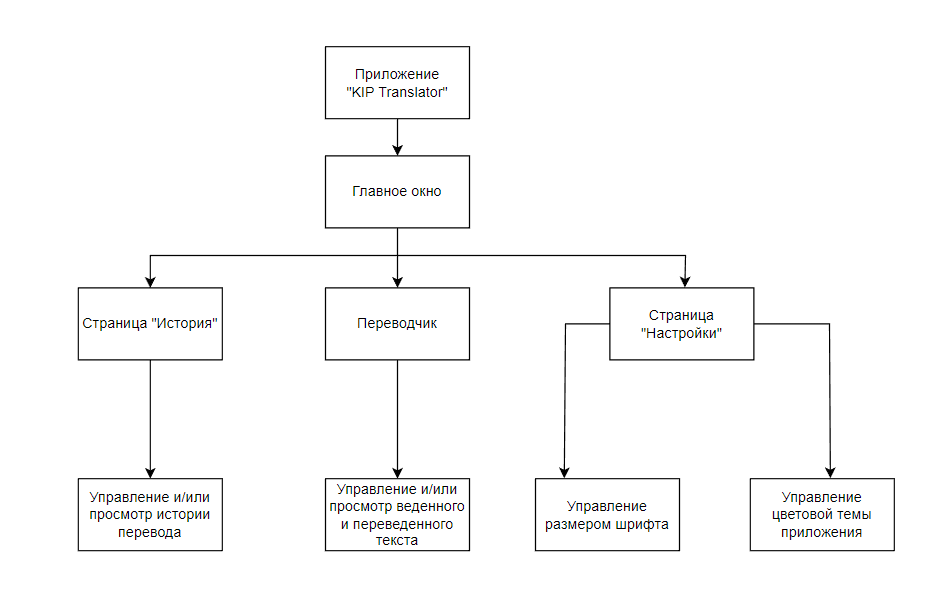


Рисунок 9. Структурная схема

Ниже представлена функциональная схема приложения «KIP Translator».

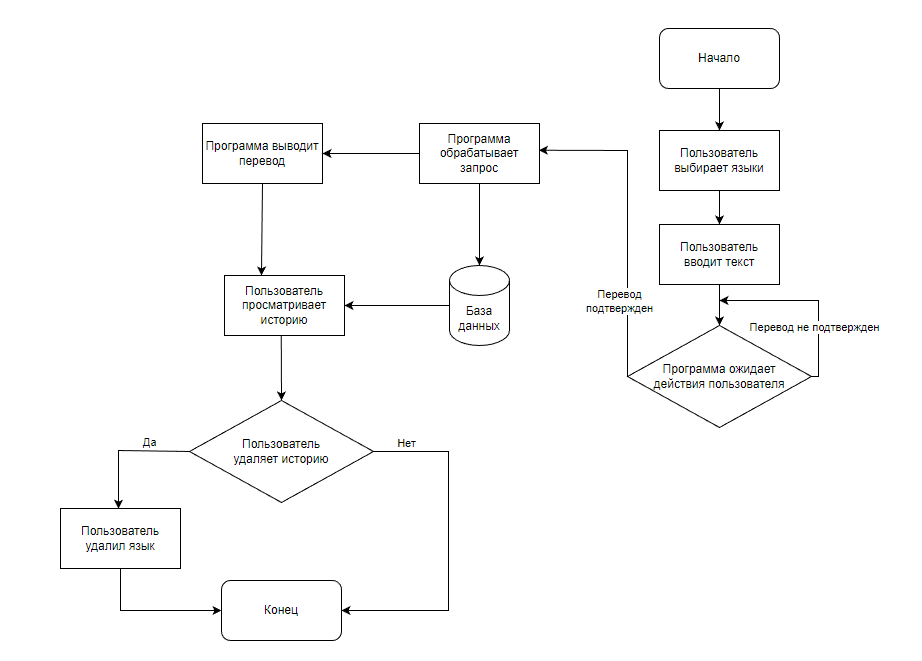


Рисунок 10. Функциональная схема

## 2.3 Разработка программного обеспечения

Программный продукт «KIP Translator» – это мультиплатформенное приложение-переводчик, разработанное на базе платформ Android и Windows. Среди функций «KIP Translator» можно выделить основную функцию – перевод текст на разные языки, а также дополнительные функции, такие как: сохранение истории переводов, озвучивание текста (PC), распознавание текста с изображения (PC), преобразование голоса в текст(android), добавление дополнительных языков перевода и некоторые другие.

При запуске приложения «KIP Translator» появляется главное окно, на котором представлены 6 кнопок, EditText, TextView и два Spinner, кнопка троеточия предназначена для перехода на страницу «История» или «Настройки».

Кнопка «История» при нажатии переводит пользователя на страницу сохраненного перевода.

Кнопка «Перевести» скачивает два языка, выбранных из двух Spinner, и переводит текст.

Кнопка «Удалить языки» удаляет два языка, выбранных из двух Spinner.

Кнопка микрофона предназначена для ввода текста голосом. При ее нажатии открывается небольшое окошко, где ожидается ввод пользователя (рисунок 12).

Кнопка в форме крестика предназначена для стирания текста из EditText и TextView.

Кнопка в форме двух листов предназначена для копирования переведенного текста.

Функции изменения цветовой темы приложения, а также изменение размера и цвета шрифта доступны в меню «настройки» (Кнопка «Настройки»).

Представленный на главной странице EditText предназначен для ввода теста сверху. Снизу TextView предназначен для вывода переведенного текста

К числу элементов, представленных на главной странице, относятся Spinner и кнопки Spinner содержит восемь языков: русский, английский, испанский, китайский, немецкий, французский, итальянский и японский.

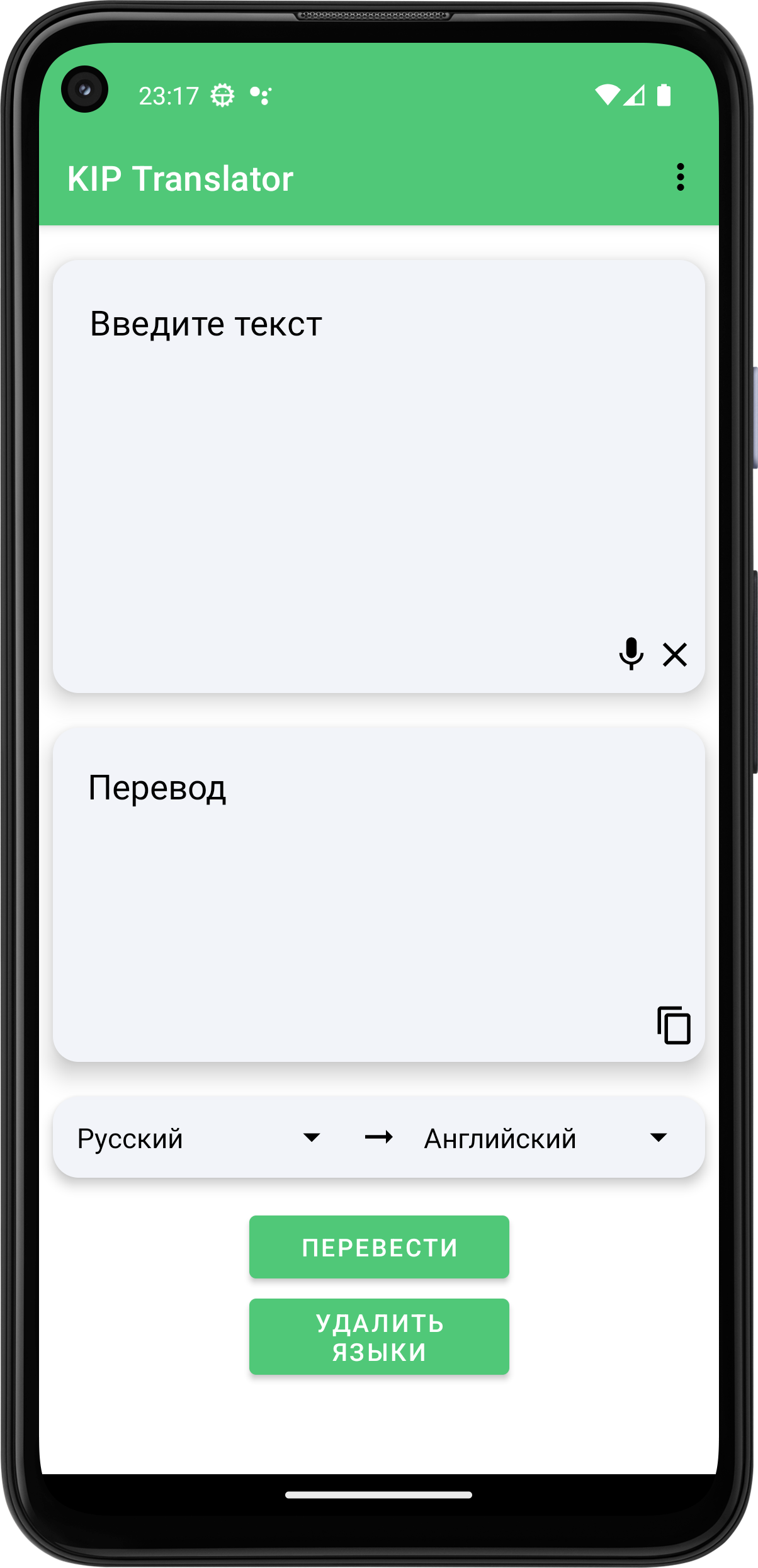


Рисунок 11. Главное окно приложения

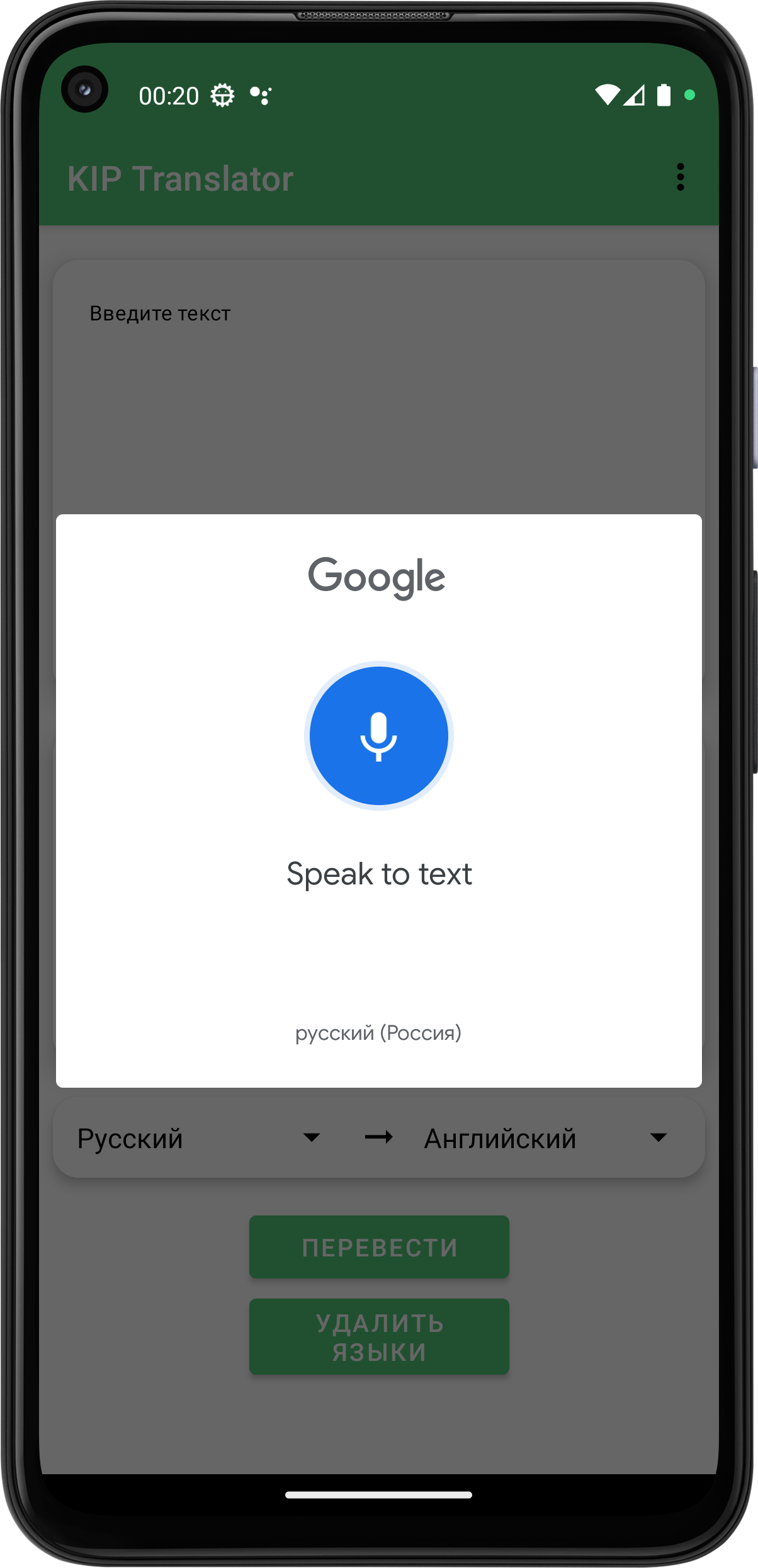


Рисунок 12. Окно микрофона

Ниже представлены вёрстка и код окна:

<?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:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
  
 <LinearLayout  
 android:id="@+id/linearLayout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="47dp"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginTop="20dp"  
 android:layout\_marginEnd="8dp"  
 android:background="@drawable/background"  
 android:elevation="4dp"  
 android:orientation="horizontal"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textInputLayout2">  
  
 <Spinner  
 android:id="@+id/firstLangSelector"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:layout\_margin="3dp"  
 android:layout\_weight="1"  
 android:padding="3dp" />  
  
 <ImageView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:src="@drawable/ic\_arrow\_right" />  
  
 <Spinner  
 android:id="@+id/secondLangSelector"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:layout\_margin="3dp"  
 android:layout\_weight="1"  
 android:padding="3dp" />  
  
 </LinearLayout>  
  
  
 <Button  
 android:id="@+id/btnTranslate"  
 android:layout\_width="150dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="16dp"  
 android:text="Перевести"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/linearLayout" />  
  
 <Button  
 android:id="@+id/btnDeleteTranslate"  
 android:layout\_width="150dp"  
 android:layout\_height="wrap\_content"  
 android:text="Удалить\nЯзыки"  
 app:layout\_constraintEnd\_toEndOf="@+id/btnTranslate"  
 app:layout\_constraintStart\_toStartOf="@+id/btnTranslate"  
 app:layout\_constraintTop\_toBottomOf="@+id/btnTranslate" />  
  
 <com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/textInputLayout"  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="250dp"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginTop="20dp"  
 android:layout\_marginEnd="8dp"  
 android:background="@drawable/background"  
 android:elevation="8dp"  
 android:inputType="textMultiLine"  
 android:orientation="vertical"  
 android:padding="5dp"  
  
 app:boxBackgroundMode="none"  
 app:hintEnabled="false"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent">  
  
 <com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/sourceText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="210dp"  
 android:gravity="top"  
 android:hint="@string/source\_text\_hint"  
 android:maxLines="100"  
 android:scrollbars="vertical"  
 android:textColor="?android:textColorPrimary"  
 android:textColorHint="?android:textColorHint"  
 android:visibility="visible" />  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="end">  
  
 <ImageButton  
 android:id="@+id/btnMic"  
 android:layout\_width="25dp"  
 android:layout\_height="25dp"  
 android:layout\_gravity="right"  
 android:background="?android:selectableItemBackground"  
 android:src="@drawable/ic\_mic" />  
  
 <ImageButton  
 android:id="@+id/btnCancel"  
 android:layout\_width="25dp"  
 android:layout\_height="25dp"  
 android:layout\_gravity="right"  
 android:background="?android:selectableItemBackground"  
 android:src="@drawable/ic\_cancel" />  
 </LinearLayout>  
  
  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/textInputLayout2"  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="193dp"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginTop="20dp"  
 android:layout\_marginEnd="8dp"  
 android:background="@drawable/background"  
 android:elevation="8dp"  
 android:padding="5dp"  
 app:boxBackgroundMode="none"  
 app:hintEnabled="false"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textInputLayout">  
  
 <TextView  
 android:id="@+id/targetText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="154dp"  
 android:gravity="top"  
 android:hint="@string/target\_text\_hint"  
 android:maxLines="100"  
 android:paddingLeft="15dp"  
 android:paddingTop="15dp"  
 android:scrollbars="vertical"  
 android:textColor="?android:textColorPrimary"  
 android:textColorHint="?android:textColorHint" />  
  
 <ImageButton  
 android:id="@+id/btnCopy"  
 android:layout\_width="25dp"  
 android:layout\_height="25dp"  
 android:layout\_gravity="end"  
 android:background="?android:selectableItemBackground"  
 android:src="@drawable/ic\_copy" />  
 </com.google.android.material.textfield.TextInputLayout>  
  
  
</androidx.constraintlayout.widget.ConstraintLayout>

class MainActivity : AppCompatActivity() {  
 var translator: Translator? = null  
  
 private val REQUEST\_CODE\_SPEECH\_INPUT = 1  
 private lateinit var sqLiteHelper: SQLiteHelper  
  
 var fromLanguages = *arrayOf*(  
 "Русский",  
 "Английский",  
 "Испанский",  
 "Китайский",  
 "Немецкий",  
 "Французский",  
 "Итальянский",  
 "Японский"  
 )  
  
 var toLanguages = *arrayOf*(  
 "Английский",  
 "Русский",  
 "Испанский",  
 "Китайский",  
 "Немецкий",  
 "Французский",  
 "Итальянский",  
 "Японский"  
 )  
  
 var modelManager = RemoteModelManager.getInstance()  
  
 var sourceLanguageCode = "ru"  
 var targetLanguageCode = "en"  
  
 private lateinit var translate: Button  
 private lateinit var delete: Button  
 private lateinit var sourceText: EditText  
 private lateinit var targetText: TextView  
 private lateinit var microphone: ImageButton  
 private lateinit var copy: ImageButton  
 private lateinit var firstLangSelector: Spinner  
 private lateinit var secondLangSelector: Spinner  
 private lateinit var clearText: ImageButton  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 initView()  
 sqLiteHelper = SQLiteHelper(this)  
 val fromAdapter = ArrayAdapter(this, android.R.layout.*simple\_spinner\_item*, fromLanguages)  
 fromAdapter.setDropDownViewResource(android.R.layout.*simple\_list\_item\_single\_choice*)  
 firstLangSelector.*adapter* = fromAdapter  
  
 firstLangSelector.*onItemSelectedListener* = object : AdapterView.OnItemSelectedListener {  
 override fun onItemSelected(  
 adapterView: AdapterView<\*>?,  
 view: View?,  
 i: Int,  
 l: Long  
 ) {  
 sourceLanguageCode = when (fromLanguages[i]) {  
 "Английский" -> TranslateLanguage.*ENGLISH* "Испанский" -> TranslateLanguage.*SPANISH* "Китайский" -> TranslateLanguage.*CHINESE* "Немецкий" -> TranslateLanguage.*GERMAN* "Французский" -> TranslateLanguage.*FRENCH* "Итальянский" -> TranslateLanguage.*ITALIAN* "Russian" -> TranslateLanguage.*RUSSIAN* "Японский" -> TranslateLanguage.*JAPANESE* else -> TranslateLanguage.*RUSSIAN* }  
 }  
  
 override fun onNothingSelected(adapterView: AdapterView<\*>?) {  
 //Toast.makeText(this@MainActivity, "Enter language to translate", Toast.LENGTH\_SHORT) .show()  
 }  
  
 }  
  
 val toAdapter = ArrayAdapter(this, android.R.layout.*simple\_spinner\_item*, toLanguages)  
 toAdapter.setDropDownViewResource(android.R.layout.*simple\_list\_item\_single\_choice*)  
 secondLangSelector.*adapter* = toAdapter  
 secondLangSelector.*onItemSelectedListener* = object : AdapterView.OnItemSelectedListener {  
 override fun onItemSelected(  
 adapterView: AdapterView<\*>?,  
 view: View?,  
 i: Int,  
 l: Long  
 ) {  
 targetLanguageCode = when (toLanguages[i]) {  
 "Русский" -> TranslateLanguage.*RUSSIAN* "Испанский" -> TranslateLanguage.*SPANISH* "Китайский" -> TranslateLanguage.*CHINESE* "Немецкий" -> TranslateLanguage.*GERMAN* "Французский" -> TranslateLanguage.*FRENCH* "Итальянский" -> TranslateLanguage.*ITALIAN* "Russian" -> TranslateLanguage.*RUSSIAN* "Японский" -> TranslateLanguage.*JAPANESE* else -> TranslateLanguage.*ENGLISH* }  
 }  
  
 override fun onNothingSelected(adapterView: AdapterView<\*>?) {  
 //Toast.makeText(this@MainActivity, "Enter language target", Toast.LENGTH\_SHORT).show()  
 }  
 }  
  
 clearText.setOnClickListener **{** sourceText.*text*.clear()  
 targetText.*text* = ""  
 **}** val clipboardManager = getSystemService(*CLIPBOARD\_SERVICE*) as ClipboardManager  
 copy!!.setOnClickListener **{** if (targetText.*text*.*isNotEmpty*()) {  
 val clipData = ClipData.newPlainText("label", targetText!!.*text*.toString())  
 clipboardManager.setPrimaryClip(clipData)  
 Toast.makeText(this@MainActivity, "Текст Скопирован", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}** microphone.setOnClickListener **{** val intent = Intent(RecognizerIntent.*ACTION\_RECOGNIZE\_SPEECH*)  
  
 intent.putExtra(  
 RecognizerIntent.*EXTRA\_LANGUAGE\_MODEL*,  
 RecognizerIntent.*LANGUAGE\_MODEL\_FREE\_FORM* )  
  
 intent.putExtra(  
 RecognizerIntent.*EXTRA\_LANGUAGE*,  
 Locale.getDefault()  
 )  
  
 intent.putExtra(RecognizerIntent.*EXTRA\_PROMPT*, "Speak to text")  
  
 try {  
 startActivityForResult(intent, REQUEST\_CODE\_SPEECH\_INPUT)  
 } catch (e: Exception) {  
 Toast  
 .makeText(  
 this@MainActivity, " " + e.message,  
 Toast.*LENGTH\_SHORT* )  
 .show()  
 }  
 **}** translate.setOnClickListener **{** val string: String = sourceText.*text*.toString()  
 downloadModel(sourceLanguageCode, targetLanguageCode, string)  
 **}** delete.setOnClickListener(View.OnClickListener **{** deleteModel(sourceLanguageCode, targetLanguageCode)  
 Toast.makeText(this@MainActivity, "Языки удалены", Toast.*LENGTH\_SHORT*).show()  
 **}**)  
  
 targetText.*movementMethod* = ScrollingMovementMethod()  
  
 }  
  
 override fun onResume() {  
 super.onResume()  
 updateFontSize()  
 checkTheme()  
 }  
  
 override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
 *menuInflater*.inflate(R.menu.*settings\_menu*, menu)  
 return super.onCreateOptionsMenu(menu)  
 }  
  
 override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 when (item.*itemId*) {  
 R.id.*action\_history* -> {  
 startActivity(Intent(this, HistoryActivity::class.*java*))  
 true  
 }  
 R.id.*action\_settings* -> {  
 startActivity(Intent(this, SettingsActivity::class.*java*))  
 true  
 }  
 }  
 return super.onOptionsItemSelected(item)  
 }  
  
 override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  
 super.onActivityResult(requestCode, resultCode, data)  
  
 if (requestCode == REQUEST\_CODE\_SPEECH\_INPUT) {  
 if (resultCode == *RESULT\_OK* && data != null) {  
 val res: ArrayList<String> =  
 data.getStringArrayListExtra(RecognizerIntent.*EXTRA\_RESULTS*) as ArrayList<String>  
 sourceText.setText(  
 Objects.requireNonNull(res)[0]  
 )  
 }  
 }  
 }  
  
 private fun initView() {  
 translate = findViewById(R.id.*btnTranslate*)  
 delete = findViewById(R.id.*btnDeleteTranslate*)  
 sourceText = findViewById(R.id.*sourceText*)  
 targetText = findViewById(R.id.*targetText*)  
 microphone = findViewById(R.id.*btnMic*)  
 copy = findViewById((R.id.*btnCopy*))  
 firstLangSelector = findViewById(R.id.*firstLangSelector*)  
 secondLangSelector = findViewById(R.id.*secondLangSelector*)  
 clearText = findViewById(R.id.*btnCancel*)  
 }  
  
 private fun downloadModel(inputLang: String, targetLang: String, input: String) {  
 val options = TranslatorOptions.Builder()  
 .setSourceLanguage(inputLang)  
 .setTargetLanguage(targetLang)  
 .build()  
 translator = Translation.getClient(options)  
 val conditions = DownloadConditions.Builder()  
 .requireWifi()  
 .build()  
 translator!!.downloadModelIfNeeded(conditions)  
 .addOnSuccessListener(*OnSuccessListener*<Void?> **{** //Toast.makeText(this@MainActivity, "Language model is being downloaded.", Toast.LENGTH\_SHORT).show()  
 translateLanguage(input)  
  
 **}**).addOnFailureListener(*OnFailureListener* **{** Toast.makeText(  
 this@MainActivity,  
 "Ошибка при установке языка",  
 Toast.*LENGTH\_SHORT* ).show()  
 **}**)  
 }  
  
 private fun translateLanguage(input: String) {  
  
 translator!!.translate(input).addOnSuccessListener **{** s **->** targetText.*text* = s  
 addTranslate()  
 **}** .addOnFailureListener **{** Toast.makeText(this@MainActivity, "Ошибка при переводе", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
  
 private fun deleteModel(srcLanguage: String, tarLanguage: String) {  
 val srcModel = TranslateRemoteModel.Builder(srcLanguage).build()  
 val tarModel = TranslateRemoteModel.Builder(tarLanguage).build()  
 modelManager.deleteDownloadedModel(srcModel)  
 .addOnSuccessListener(*OnSuccessListener*<Any?> **{ }**)  
 .addOnFailureListener(*OnFailureListener* **{ }**)  
 modelManager.deleteDownloadedModel(tarModel)  
 .addOnSuccessListener(*OnSuccessListener*<Any?> **{ }**)  
 .addOnFailureListener(*OnFailureListener* **{ }**)  
 }  
  
 private fun addTranslate() {  
 val source = sourceText.*text*.toString()  
 val target = targetText.*text*.toString()  
 val firstLang = firstLangSelector.*selectedItem*.toString()  
 val secondLang = secondLangSelector.*selectedItem*.toString()  
 if (source.*isNotEmpty*() && target.*isNotEmpty*()) {  
 val std = TranslateModel(  
 sourceText = source,  
 targetText = target,  
 firstSpinner = firstLang,  
 secondSpinner = secondLang  
 )  
 val status = sqLiteHelper.insertTranslate(std)  
  
 if (status > -1) {  
 //Toast.makeText(this,"Перевод добавлен",Toast.LENGTH\_SHORT).show()  
  
 }  
 }  
 }  
  
 private fun checkTheme() {  
 val darkModeValues = *resources*.getStringArray(R.array.*dark\_mode\_values*)  
 when (PreferenceManager.getDefaultSharedPreferences(this)  
 .getString(getString(R.string.*dark\_mode*), getString(R.string.*dark\_mode\_def\_value*))) {  
 darkModeValues[0] -> AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.*MODE\_NIGHT\_FOLLOW\_SYSTEM*)  
 darkModeValues[1] -> AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.*MODE\_NIGHT\_NO*)  
 darkModeValues[2] -> AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.*MODE\_NIGHT\_YES*)  
 }  
 }  
  
 private fun updateFontSize() {  
 val sharedPrefs = PreferenceManager.getDefaultSharedPreferences(this)  
 val fontSize = sharedPrefs.getString("font\_size", "16")?.*toFloat*() ?: 16f  
 sourceText.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, fontSize)  
 targetText.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, fontSize)  
 }  
}

На рисунке 13 изображена страница «История», отображающая сохранённую историю.



Рисунок 13. Страница истории переводов

Ниже представлены вёрстка и код страницы:

<?xml version="1.0" encoding="utf-8"?>  
<androidx.cardview.widget.CardView 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:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="8dp"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginRight="8dp"  
 android:backgroundTint="?android:color"  
 app:cardCornerRadius="15dp">  
  
 <androidx.constraintlayout.widget.ConstraintLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:padding="8dp">  
  
 <TextView  
 android:id="@+id/firstTranslateRecycle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:paddingBottom="5dp"  
 android:text="Перевод 1"  
 android:textColor="?android:textColorPrimary"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/firstLanguageRecycle" />  
  
 <TextView  
 android:id="@+id/secondTranslateRecycle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:paddingTop="5dp"  
 android:text="Перевод 2"  
 android:textColor="?android:textColorPrimary"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/firstTranslateRecycle" />  
  
 <ImageButton  
 android:id="@+id/btnDeleteRecycle"  
 android:layout\_width="24dp"  
 android:layout\_height="30dp"  
 android:layout\_marginEnd="8dp"  
 android:background="@android:color/transparent"  
 android:src="@drawable/ic\_delete"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintStart\_toEndOf="@+id/secondLanguageRecycle"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.0"  
 app:tint="?tint" />  
  
 <TextView  
 android:id="@+id/firstLanguageRecycle"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="30dp"  
 android:gravity="right|center"  
 android:text="Язык 1"  
 android:textColor="?android:textColorPrimary"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <TextView  
 android:id="@+id/secondLanguageRecycle"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="30dp"  
 android:gravity="left|center"  
 android:text="Язык 2"  
 android:textColor="?android:textColorPrimary"  
 app:layout\_constraintStart\_toEndOf="@+id/imageView"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="30dp"  
 app:layout\_constraintStart\_toEndOf="@+id/firstLanguageRecycle"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:srcCompat="@drawable/ic\_arrow\_right"  
 app:tint="?tint" />  
  
 <View  
 android:layout\_width="match\_parent"  
 android:layout\_height="1dp"  
 android:background="#c0c0c0"  
 android:backgroundTint="?tint"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/firstTranslateRecycle" />  
  
 </androidx.constraintlayout.widget.ConstraintLayout>  
</androidx.cardview.widget.CardView>

class HistoryActivity : AppCompatActivity() {  
 private lateinit var recycler: RecyclerView  
 private lateinit var sqLiteHelper: SQLiteHelper  
  
 private var adapter: TranslateAdapter? = null  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_history*)  
  
 sqLiteHelper = SQLiteHelper(this)  
  
 recycler = findViewById(R.id.*translateList*)  
 recycler.*layoutManager* = LinearLayoutManager(this)  
  
 adapter = TranslateAdapter()  
 recycler.*adapter* = adapter  
  
 // Получаем данные о переводах из базы данных и отображаем их в списке  
 getTranslate()  
  
 // Обработчик для удаления элемента из списка  
 adapter?.setonClickDeleteItem **{  
 it**.id?.*let* **{** it1 **->** deleteTranslate(it1) **}  
 }** // Отображение элементов в обратном порядке  
 val linearLayoutManager = LinearLayoutManager(this)  
 linearLayoutManager.*reverseLayout* = true  
 linearLayoutManager.*stackFromEnd* = true  
 recycler.setLayoutManager(linearLayoutManager)  
 }  
  
 // Создаем метод onCreateOptionsMenu для отображения меню  
 override fun onCreateOptionsMenu(menu: Menu): Boolean {  
 *menuInflater*.inflate(R.menu.*all\_delete*, menu)  
 return true  
 }  
  
 // Создаем метод onOptionsItemSelected для обработки выбранных пунктов меню  
 override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 return when (item.*itemId*) {  
 // Если выбран пункт "Удалить все записи?", показываем диалоговое окно с подтверждением  
 R.id.*action\_delete\_all* -> {  
 val builder = AlertDialog.Builder(this)  
 builder.setTitle("Удалить все записи?")  
 builder.setMessage("Вы действительно хотите удалить все записи?")  
 builder.setPositiveButton("Да") **{** \_, \_ **->** val dbHelper = SQLiteHelper(this)  
 // Удаляем все записи из базы данных  
 dbHelper.*writableDatabase*.delete("TBL\_TRANSLATE", null, null)  
 // Обновляем список переводов  
 getTranslate()  
 **}** builder.setNegativeButton("Отмена", null)  
 val dialog = builder.create()  
 dialog.show()  
 true  
 }  
 else -> super.onOptionsItemSelected(item)  
 }  
 }  
  
 // Метод для удаления выбранного перевода из базы данных и обновления списка  
 private fun deleteTranslate(id: Int) {  
 sqLiteHelper.deleteTranslateById(id)  
 getTranslate()  
 }  
  
 // Метод для получения списка всех переводов из базы данных и их отображения в списке  
 private fun getTranslate() {  
 val stdList = sqLiteHelper.getAllTranslate()  
 adapter?.addItems(stdList)  
 }  
}

Страница «Настройки» (рисунок 14) содержит функции изменения размера шрифта и изменение цветовой темы. Функция «изменения размера шрифта» предназначена для изменения размера текста в EditText и TextView на главной странице (рисунок 15 и 16). Функция «изменение цветовой темы» изменяет тему приложения (рисунок 17 и 18).

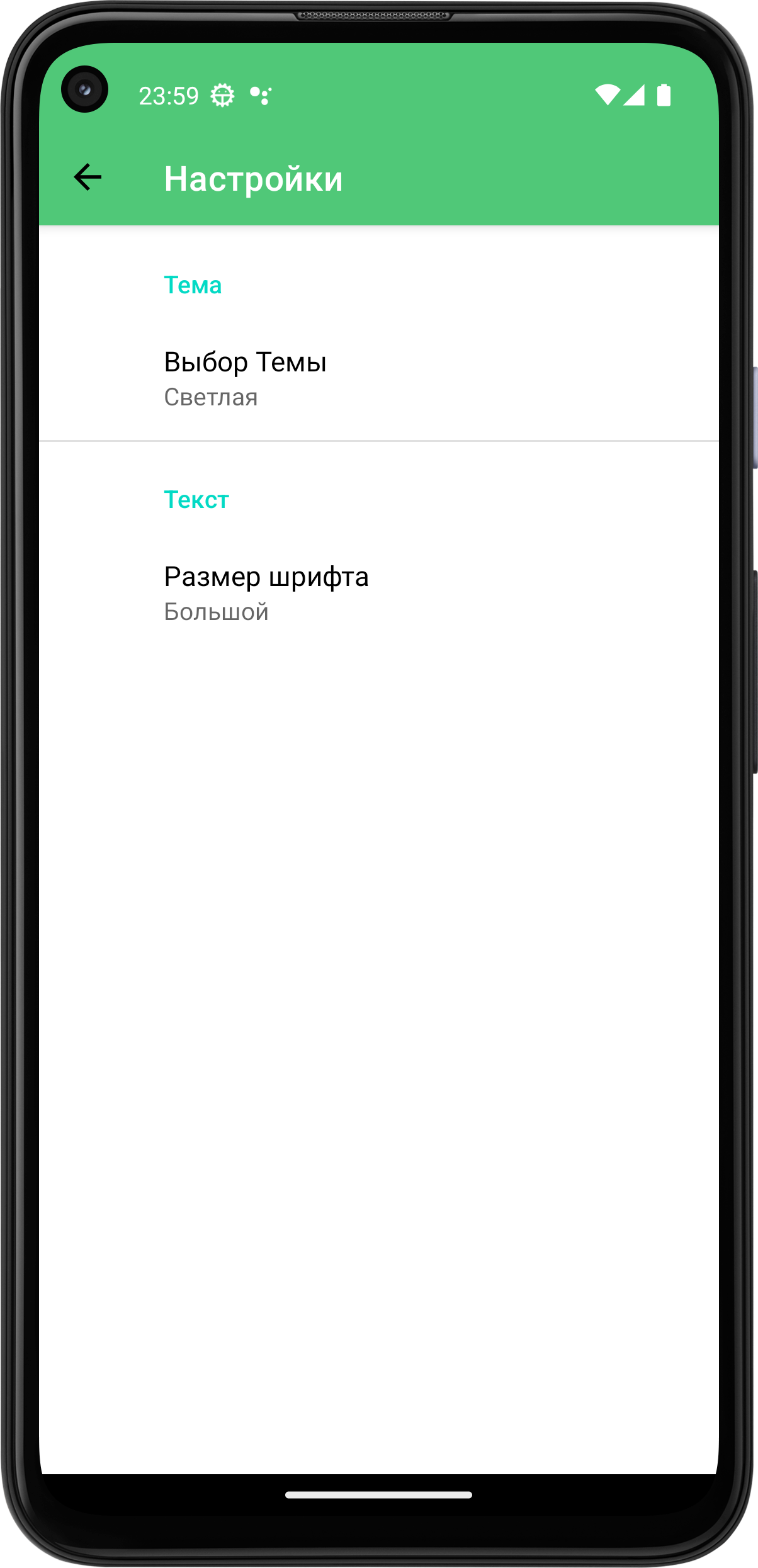


Рисунок 14. Страница настроек

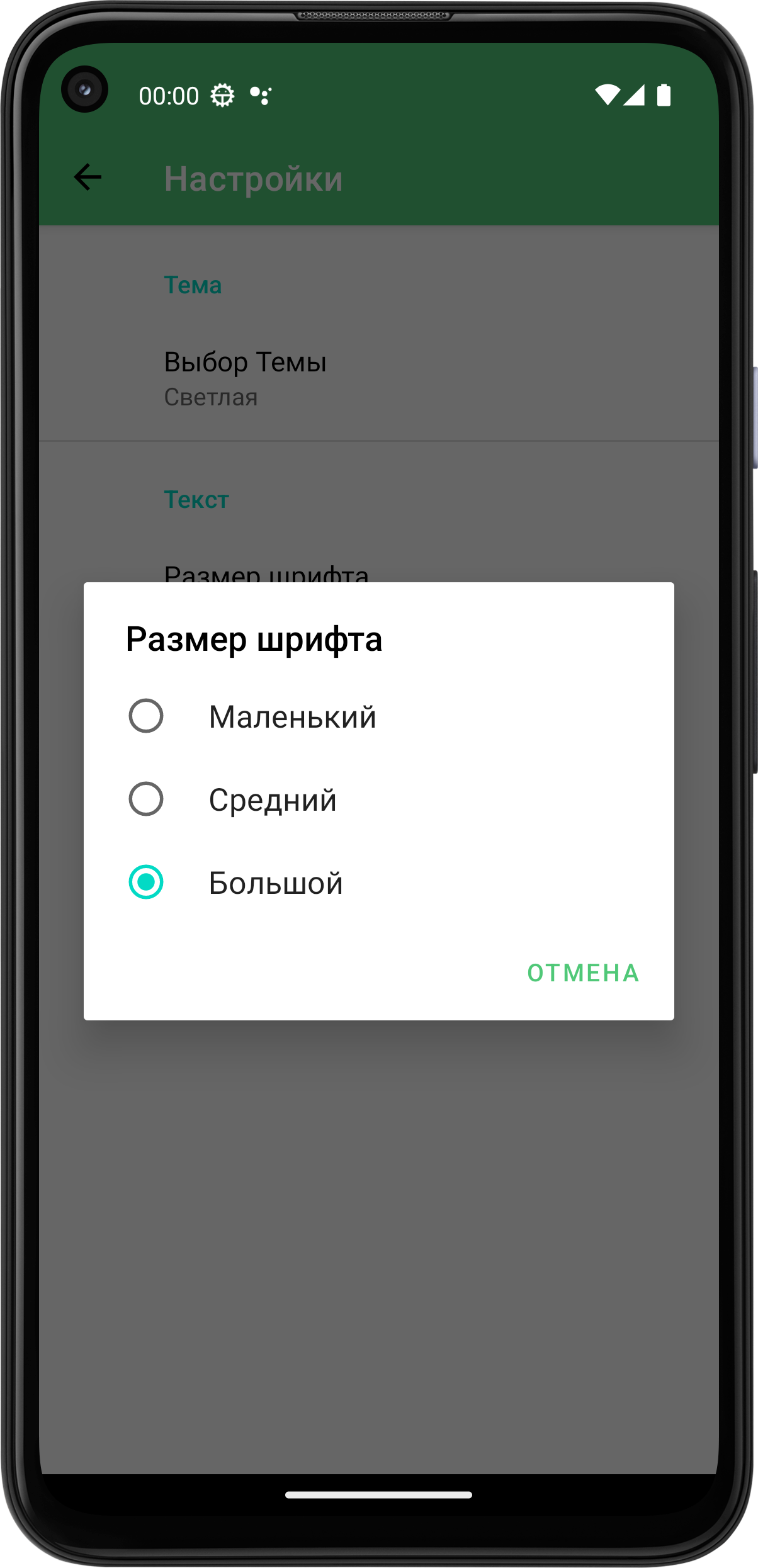


Рисунок 15. Окно изменения размера шрифта



Рисунок 16. Изменения шрифта на главной странице

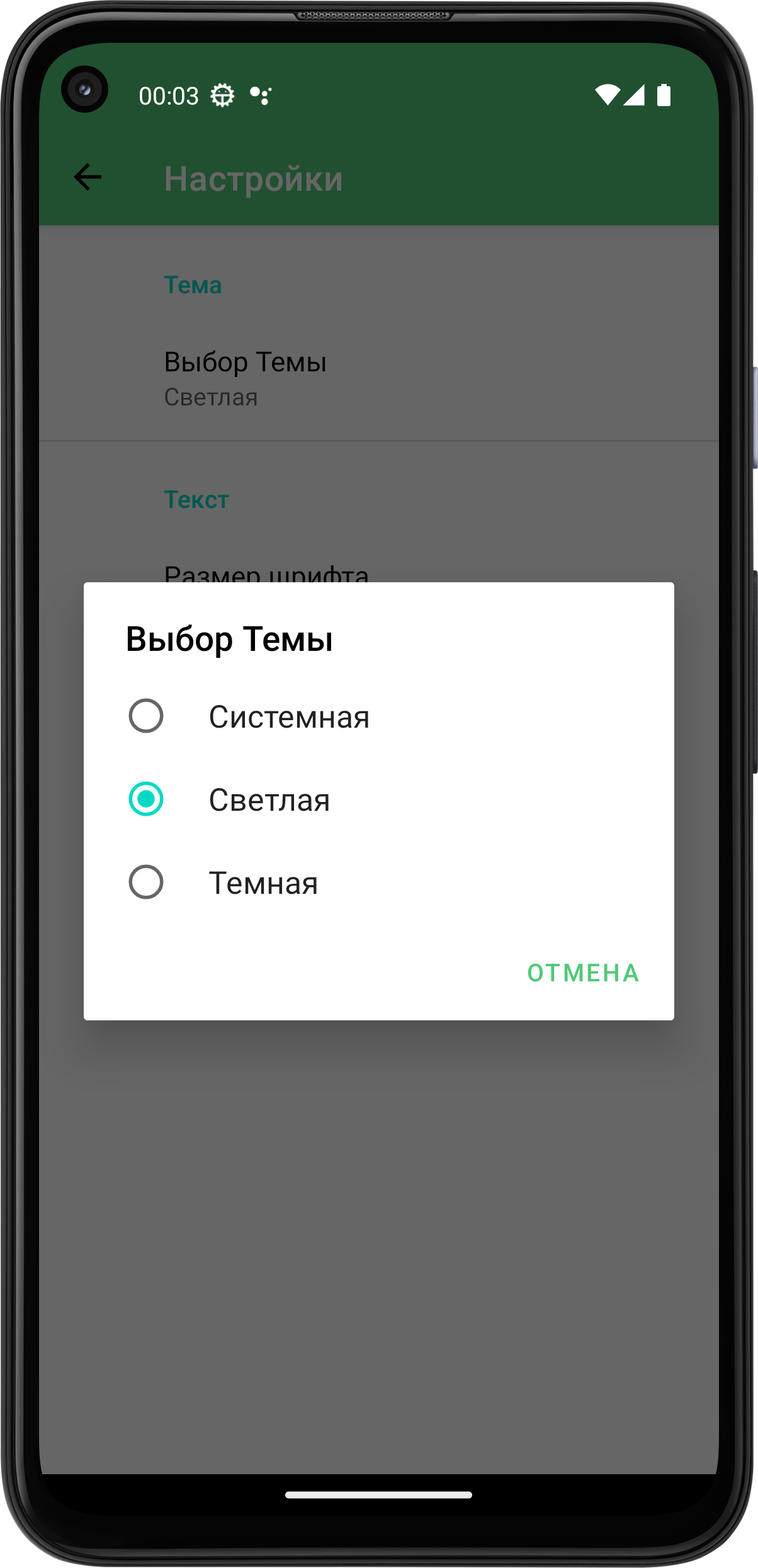


Рисунок 17. Окно изменения цветовой темы

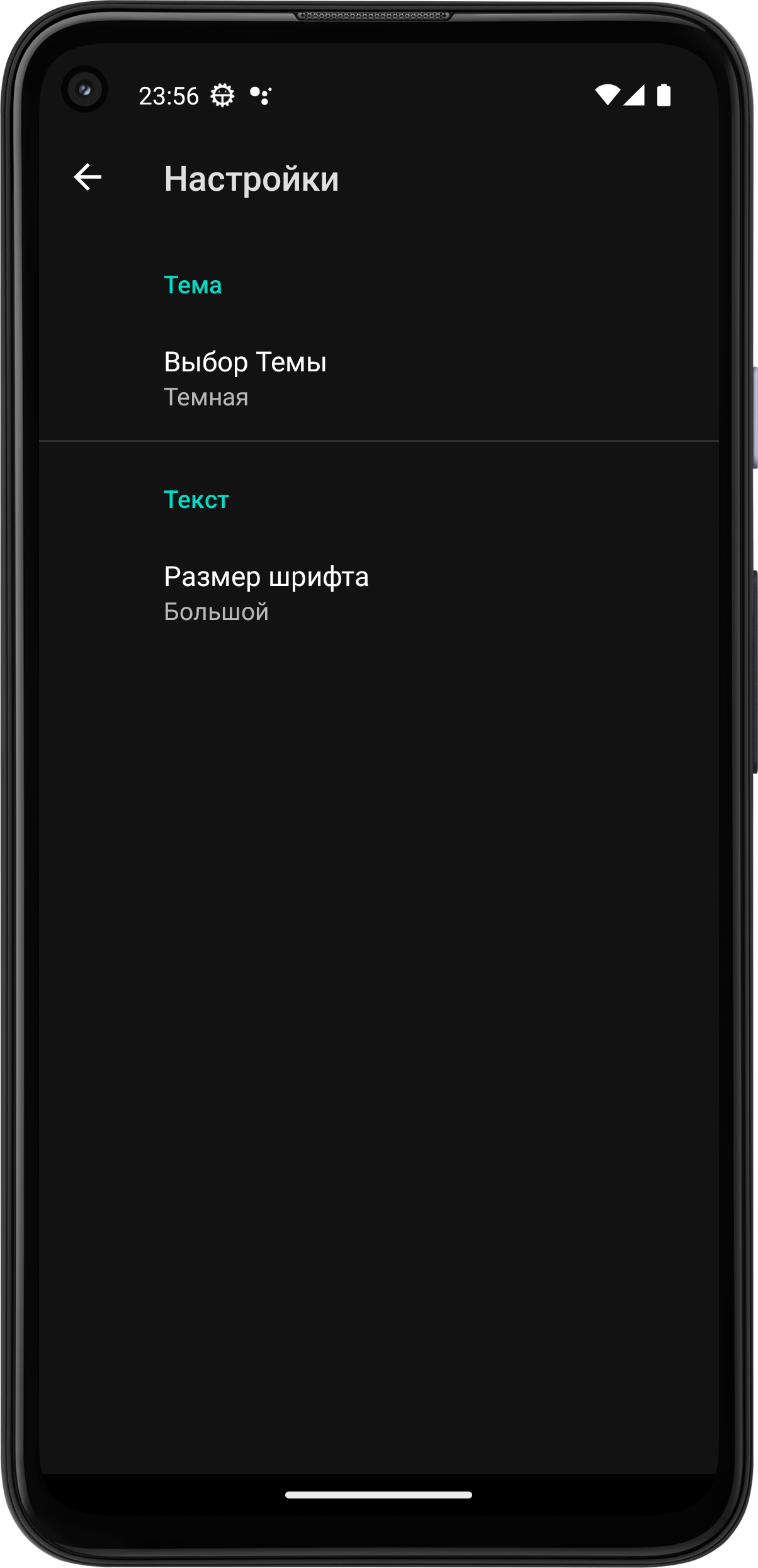


Рисунок 18. Темная тема приложения

Ниже представлены вёрстка и код страницы:

<PreferenceScreen xmlns:app="http://schemas.android.com/apk/res-auto">  
  
 <PreferenceCategory app:title="@string/theme\_header">  
  
 <ListPreference  
 app:defaultValue="@string/dark\_mode\_def\_value"  
 app:entries="@array/dark\_mode\_entries"  
 app:entryValues="@array/dark\_mode\_values"  
 app:key="@string/dark\_mode"  
 app:title="@string/dark\_mode"  
 app:useSimpleSummaryProvider="true" />  
  
 </PreferenceCategory>  
  
 <PreferenceCategory app:title="@string/text\_header">  
  
 <ListPreference  
 app:defaultValue="16"  
 app:entries="@array/font\_sizes"  
 app:entryValues="@array/font\_values"  
 app:key="font\_size"  
 app:summary="%s"  
 app:title="@string/size\_title" />  
  
 </PreferenceCategory>  
  
</PreferenceScreen>

class SettingsActivity : AppCompatActivity(), SharedPreferences.OnSharedPreferenceChangeListener,  
 Preference.SummaryProvider<ListPreference> {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*settings\_activity*)  
 *supportFragmentManager* .beginTransaction()  
 .replace(R.id.*settings*, SettingsFragment())  
 .commit()  
 *supportActionBar*?.setDisplayHomeAsUpEnabled(true)  
 PreferenceManager.getDefaultSharedPreferences(this)  
 .registerOnSharedPreferenceChangeListener(this)  
 }  
  
 override fun onSharedPreferenceChanged(sharedPreferences: SharedPreferences?, key: String?) {  
 key?.*let* **{** if (**it** == getString(R.string.*dark\_mode*)) sharedPreferences?.*let* **{** pref **->** val darkModeValues = *resources*.getStringArray(R.array.*dark\_mode\_values*)  
 when (pref.getString(getString(R.string.*dark\_mode*), darkModeValues[0])) {  
 darkModeValues[0] -> AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.*MODE\_NIGHT\_FOLLOW\_SYSTEM*)  
 darkModeValues[1] -> AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.*MODE\_NIGHT\_NO*)  
 darkModeValues[2] -> AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.*MODE\_NIGHT\_YES*)  
 }  
 **}  
 }** }  
  
 override fun onDestroy() {  
 super.onDestroy()  
 PreferenceManager.getDefaultSharedPreferences(this)  
 .unregisterOnSharedPreferenceChangeListener(this)  
 }  
  
 class SettingsFragment : PreferenceFragmentCompat(), Preference.OnPreferenceChangeListener {  
 override fun onCreatePreferences(savedInstanceState: Bundle?, rootKey: String?) {  
 setPreferencesFromResource(R.xml.*root\_preferences*, rootKey)  
 val fontSizePreference = findPreference<SeekBarPreference>("font\_size\_preference")  
 fontSizePreference?.*onPreferenceChangeListener* = this  
 }  
  
 override fun onPreferenceChange(preference: Preference?, newValue: Any?): Boolean {  
 if (preference?.*key* == "font\_size\_preference") {  
 val fontSize = newValue as Int  
 val sharedPrefs = PreferenceManager.getDefaultSharedPreferences(requireContext())  
 sharedPrefs.edit().putInt("font\_size", fontSize).apply()  
 return true  
 }  
 return false  
 }  
 }  
  
 override fun provideSummary(preference: ListPreference?): CharSequence =  
 if (preference?.*key* == getString(R.string.*dark\_mode*)) preference.*entry* else "Unknown Preference"  
}

## 2.4 Отладка и тестирование программы

Для проверки реализованного функционала необходимо провести отладку и тестирование программного обеспечения.

В таблице представлены результаты отладки и тестирования программного обеспечения.

Таблица 1. Тестирование и отладка программного обеспечения

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №  теста | Входные данные | Вводимое значение | Ожидаемая реакция программы | Фактический результат | Ошибка выявлена |
| 1 | Текст на перевод | Привет мир | Перевод на английский язык |  | Нет |
| 2 | Текст на перевод | Привет мир | Перевод на испанский язык |  | Нет |
| 3 | Изменение цветовой темы | Темный | Изменение темы приложения на темный |  | Нет |

Проведя работу с отладкой и тестированием, можно утверждать, что программа соответствует как функциональным, так и нефункциональным требованиям.

## 2.5 Руководство по использованию программы

Назначением программы является перевод текста.

Функции программы:

* перевод текста;
* добавление перевода в базу данных;
* отображение сохраненного перевода;
* удаление перевода;
* изменение размера шрифта;
* изменение цветовой темы приложения.

Ниже представлены минимальные требования к мобильному устройству для установки приложения:

* Платформа [OS]: Android 5.0 и более поздние версии
* Размер приложения: 27,60 МБ

Программное решение «KIP Translator» представляет из себя интерфейс переводчика с дополнительными функциями. В данной программе представлены такие функции как: перевод текста, преобразование голоса в текст, история, изменение размера шрифта, изменение цветовой темы. Функция «перевода текста» – функция, позволяющая пользователю выбрать исходный язык, вписать или вставить интересующий текст и выбрать целевой язык, язык на который будет переведён текст. Функция «преобразование голоса в текст» – функция, позволяющая пользователю используя свой голос преобразовать сказанную фразу или слово в текст. Функция «история» – функция, предоставляющая пользователю вариант просмотреть историю переводов или удалить ненужную. Функция «изменение размера шрифта», позволяющая изменить шрифт.

Для установки приложения необходимо скачать файл-установщик, далее нужно запустить установщик и подтвердить установку во всплывающем окне (рисунок 16). По завершению установки в появившемся окне нажать на кнопку «готово» или «открыть».

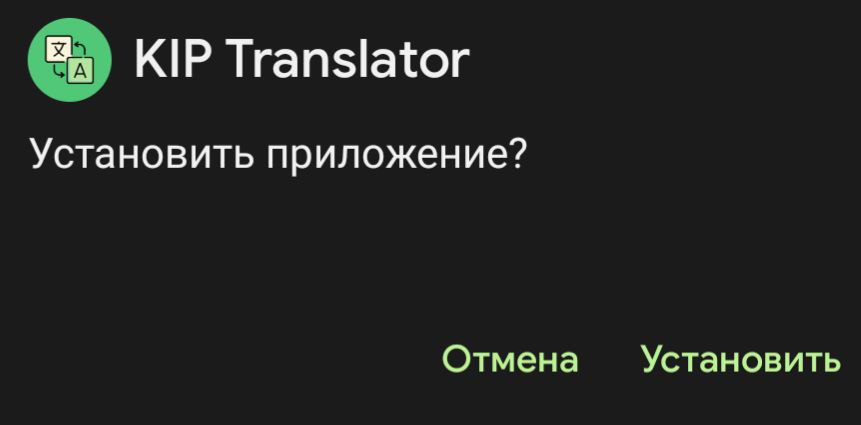


Рисунок 18. Установщик «KIP Translator»

После нажатия на кнопку «Установить» произойдет установка.

После установки программу можно будет открыть на вашем мобильном устройстве через ярлык на рабочем столе.

После запуска программы пользователь сможет как перевести текст, так и перейти на дополнительные функции, путём нажатия нужной кнопки, расположенной в кнопке троеточия на главном окне программы. После перехода на нужную функцию, для возвращения на страницу перевода нужно нажать стрелку, которая расположена слева-сверху на странице нужной функции.

Удаления программы происходит через меню, где нужно удерживать приложения и потянуть вверх экрана на надпись «Удалить». После этого будет открыто окно, представленное на рисунке 17, на котором нужно нажать на кнопку «Ок».

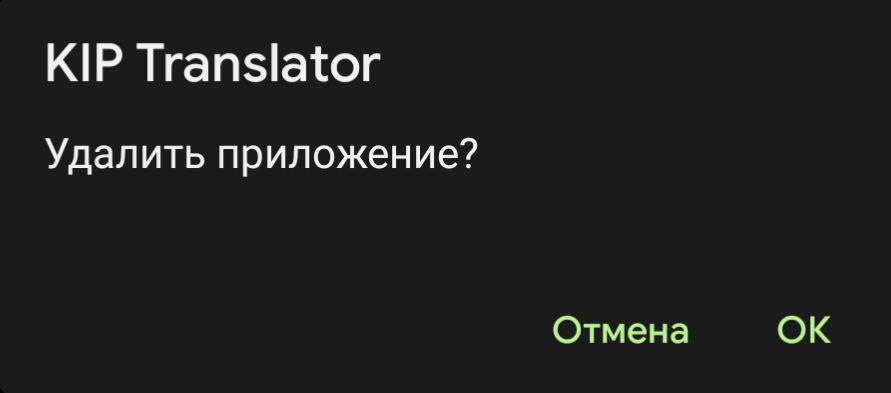


Рисунок 19. Удаление «KIP Translator»

ЗАКЛЮЧЕНИЕ

В данном проекте производилась разработка мультиплатформенного переводчика с распознаванием текста, который может использоваться на различных устройствах, в том числе на компьютерах и мобильных устройствах. Для разработки desktop-приложения был использован .net framework, а для мобильного приложения - Android Studio.

Для достижения этой цели были использованны современные методы и технологии, включая .NET Framework для разработки desktop-приложения и Android Studio для мобильной версии. Разработанный переводчик имеет удобный интерфейс и прост в использовании, что делает его доступным для широкого круга пользователей.

В результате дипломного проекта был создан инструмент, способный переводить текст на различных языках и распознавать его с помощью машинного обучения. Это позволяет ускорить процесс перевода и повысить его точность.

Кроме того, разработанный программный продукт имеет большой потенциал для дальнейшего развития и улучшения. Возможным направлением для будущих улучшений компьютерной версии является введение функции голосового ввода и распознавания речи, а для мобильной версии – распознавание текста по изображению, что еще больше упростит использование переводчика в повседневной жизни.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

*Законодательные и нормативные акты:*

1. ГОСТ Р 7.0.12-2011 Библиографическая запись. Сокращение слов и словосочетаний на русском языке. Общие требования и правила. – М.: Стандартинформ, 2012. – 61 с.
2. ГОСТ 7.32-2017 Отчет о научно-исследовательской работе. Структура и правила оформления. – М.: Стандартинформ, 2017. – 47 с.
3. ГОСТ 7.82-2001 Библиографическая запись. Библиографическое описание электронных ресурсов. Общие требования и правила составления. – М.: ИПК Издательство стандартов, 2001. – 39 с.
4. ГОСТ Р 7.0.100-2018 Библиографическая запись. Библиографическое описание. Общие требования и правила составления. – М.: Стандартинформ, 2018. – 122 с.
5. ГОСТ Р 7.0.5-2008 Библиографическая ссылка. Общие требования и правила составления. – М.: Стандартинформ, 2008. – 32 с.

*Интернет-документы:*

1. Интернет-сервис для построения схем и диаграмм Draw.io. – [Электронный ресурс]. – URL: <https://www.draw.io/>
2. Implement Dark/Night Mode in an Android app with a ListPreference toggle – [сайт] – URL: <https://dev.to/aurumtechie/implement-dark-night-mode-in-an-android-app-with-a-listpreference-toggle-2k5i>
3. Android Copy Text to Clipboard Programmatically – JAVA/KOTLIN – [сайт] – URL: <https://protocoderspoint.com/android-copy-text-to-clipboard-programmatically-java-kotlin/>
4. Android SQLite Database in Kotlin – [сайт] – URL: <https://www.geeksforgeeks.org/android-sqlite-database-in-kotlin/>
5. Step by step guide to create App Settings using Preferences in Android (Part -2) – [сайт] – URL: <https://medium.com/@bhavyakaria/step-by-step-guide-to-create-app-settings-using-preferences-in-android-part-2-9dd3466d1e2>
6. Перевод – [сайт] – URL: <https://developers.google.com/ml-kit/language/translation?hl=ru>
7. How to Create Language Translator Android App? – [Электронный ресурс] – URL: <https://www.youtube.com/watch?v=leG7IHRPZB8&t=145s>
8. SQLite: Кошкин дом. Часть первая. – [сайт] – URL: <https://developer.alexanderklimov.ru/android/sqlite/cathouse.php>
9. SQLite и Android. Кошкин дом. Часть вторая – [сайт] – URL: <https://developer.alexanderklimov.ru/android/sqlite/cathouse2.php>
10. Cursor – [сайт] – URL: <https://developer.alexanderklimov.ru/android/sqlite/cursor.php>