МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет вычислительной техники

Кафедра «Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

Профиль подготовки: «Вычислительные машины, комплексы, системы и сети»

БАКАЛАВРСКАЯ РАБОТА

на тему

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ИНТЕРАКТИВНОГО ОБУЧЕНИЯ

Студент 31.05.22 Коновалова Дарья Алексеевна
(подпись, дата) (ФИО полностью)

Руководитель 31.05.2022 Бикташев Р.А.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр 31.05.2022 Бычков А. С.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 02.06.2022г. № 13)

Заведующий кафедрой М.А. Митрохин
(подпись) (фамилия, инициалы)

Работа защищена с отметкой хорошо (протокол заседания ГЭК от 20 июня 2022 № 19)

Секретарь ГЭК Е.И. Гурин
(подпись) (фамилия, инициалы)

Пенза 2022

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет вычислительной техники

Кафедра «Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
«26» апреля 2022г.

ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ

1. Студент Коновалова Дарья Алексеевна гр. 18ВВ1 факультета вычислительной техники

2.Тема работы «Программное обеспечение для интерактивного обучения»

Тема утверждена приказом ПГУ № 357/о от "26" апреля 2022г.

3. Руководитель бакалаврской работы к.т.н. доцент кафедры ВТ ПГУ Бикташев Р.А.

4. Задание на работу
Разработать приложение для изучения английских слов, позволяющую запоминать слова в формате тестирования. В работе должны быть реализованы следующие требования: приложение должно быть написано на языке Java с использованием Android SDK, авторизация пользователя в программе, разработать темы слов для изучения на выбор пользователя, программа должна быть в формате тестирования и предоставлять выборы ответа на вопрос, отображение реакции приложения на выбранный пользователем ответ, вывод итогового счета после прохождения теста, отображение последнего результата прохождения теста.

5. Перечень подлежащих разработке вопросов
1) Анализ предметной области и изучение инструментов разработки Android Studio
2) Установка требований к программе
3) Разработка архитектуры разрабатываемого приложения
4) Создание мобильного приложения

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Под руково
1	Анализ предметной области и изучение инструментов разработки Android Studio	15%	26.04.22-29.04.22	
2	Установка требований к программе	20%	30.04.22-6.05.22	
3	Разработка архитектуры разрабатываемого приложения	20%	7.05.22-13.05.22	
4	Создание мобильного приложения	25%	14.05.22-22.05.22	
5	Тестирование и отладка	10%	22.05.22-26.05.22	
6	Оформление выпускной работы.	10%	26.05.22-31.05.22	

Дата выдачи задания "26" апреля 2022 г.

Руководитель бакалаврской работы _____. Бикташев Р.А
(подпись, дата)

Задание к исполнению принял студент _____. Коновалова
(подпись, дата)

Работу к защите допустить
Декан факультета 20.06.2022. Фионова Л.Р
(подпись, дата)

Реферат

Работа содержит 81 лист, 25 рисунков, 1 таблицу, 8 источников, 3 приложения.

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ИНТЕРАКТИВНОГО ОБУЧЕНИЯ.

Объектом разработки является приложение для изучения английских слов.

Цель ВКР – разработка приложения для изучения английских слов.

Для выполнения поставленной цели требуется реализовать ряд задач:

- анализ предметной области и изучение инструментов разработки Android Studio;

- установить требования к программе;

- смоделировать архитектуру разрабатываемого приложения;

- создать мобильное приложение;

- осуществить тестирование.

Функциональные требования к программе: создание приложения в формате тестирования, реализация авторизации по паролю, создание тем слов для изучения, вывод реакции на выбранный ответ, вывод счета по итогу прохождения теста.

В результате проделанной работы, было создано приложение для изучения иностранных слов в виде выбора ответа на заданный вопрос. Программа осуществляет вход пользователя под индивидуальной учетной записью, предлагает выбор темы слов для изучения и запоминает предыдущий счет прохождения темы.

Содержание

Содержание 3
Введение 5
1 Программное обеспечение для интерактивного обучения 7
1.1 Анализ предметной области 7
1.1.1 Анализ темы 7
1.1.2 Анализ аналогичных программ 7
1.1.3 Вывод из сравнения приложений 10
1.1.4 Постановка задачи 10
1.1.5 Выбор инструмента для решения поставленной задачи 10
1.1.6 Преимущество *Android Studio* 11
1.1.7 Выбор языка разработки 11
1.2 Анализ требований 12
1.2.1 Функциональные требования к проектируемой системе 12
1.2.2 Нефункциональные требования к проектируемой системе 13
1.2.3 Диаграмма прецедентов 13
1.2.4 Диаграмма деятельности 13
1.2.5 Диаграмма классов 14
1.2.6 Диаграмма последовательности 14
1.3 Технологии разработки 14
1.3.1 Разработка клиентской части 14
1.3.2 Разработка базы данных 19
1.4 Руководство пользователя 20
1.5 Тестирование 25
2 Безопасность жизнедеятельности 32
2.1 Введение 32
2.2 Методика оценки напряженности трудового процесса 33

2.2.1 Нагрузки интеллектуального характера 34
2.2.2 Сенсорные нагрузки 34
2.2.3 Эмоциональные нагрузки 35
2.2.4 Монотонность нагрузок 36
2.2.5 Режим работы 37
2.3 Общая оценка напряженности трудового процесса 37
2.4 Оценка напряженности труда пользователя ПЭВМ 38
2.5 Заключение 41
Заключение 42
Список использованных источников 43
Приложение А - *UML* - диаграммы 44
Приложение Б - Листинг программы 49
Приложение В - Презентация 75

Введение

Изучение иностранных языков представляется главным аспектом жизни современного человека. Оно дает нам способность более ясно понять традиции и культуру иных народов, содействует формированию мышления, воображения и памяти. тенденция изучения иностранных языков говорит о потребности людей путешествовать по миру, улучшать свои знания в общении с жителями разных стран и культур, изучать литературу на других языках, просмотр зарубежных фильмов, а без знания как минимум одного иностранного языка это сделать очень трудно. Самым распространенным иностранным языком, на котором говорит огромная часть населения земли, считается английский.

Значимость понимания иностранных языков трудно переоценить, так как в современном обществе мы ежедневно встречаемся с иностранными языками и часто бывает, что особенно их игнорирование значительно усложняет нам жизнь. С английским мы сталкиваемся на каждом шагу – в социальных сетях и в интернете в целом, в кинолентах, музыке, художественных книгах и научной литературе, в новостях, в описаниях продуктов, товаров и услуг, в руководстве к зарубежным товарам. Вдобавок владение иностранными языками выручает в условиях стремительного прогресса в технической сфере и компетентной организации трудового процесса.

Но без регулярных тренировок грамматики, изучения новых слов и повторения уже известной ему информации, человек быстро забывает изученный материал. Если запоминать иностранные слова путем механического зазубривания, то это не даст больших результатов в будущем и затратит внушительное количество времени на повторное изучение материала. Решением этой проблемы может являться ежедневная практика небольшого количества слов, что улучшит словарный запас человека.

Исходя из вышесказанного, тема ВКР посвящённая разработке *Android*-приложения для изучения английских слов посредством прохождения тестирования является актуальной.

Цель данной работы - разработка *Android*-приложения для изучения английских слов посредством прохождения тестирования.

Задачами данной выпускной квалификационной работы являются:

- изучение особенностей разработки *Android*-приложения;
- определение требований к программе;
-проектирование архитектуры разрабатываемой программы;
- создание мобильного приложения.

1 Программное обеспечение для интерактивного обучения

1.1 Анализ предметной области

1.1.1 Анализ темы

С современном мире изучение иностранных языков является не просто полезным навыком или развлечением, а жизненно необходимым знанием. Так же нельзя недооценивать влияние техники и информационных технологий в жизни человека и его рабочем пространстве. Изучение языков нужно для корректного понимания и взаимодействия между людьми, сотрудничества и выстраивания деловых отношений с иностранцами. Так же открывается доступ к огромному спектру информации и исследований.

У многих людей могут возникать сложности с запоминанием большого объема незнакомых и труднопроизносимых слов. Данную проблему можно решить путем многократного, ежедневного повторения, а игровой процесс сделает это занятие более простым и понятным для человека.

В рамках данного проекта предстоит разработка приложения, в функционал которого входит изучение иностранных слов путем выбора темы слов для изучения, подбор ответа на поставленный вопрос из представленных вариантов, реакция на выбранный ответ, отображение итогового счета и отображение последнего результата.

1.1.2 Анализ аналогичных программ

Проведя анализ рынка приложений для изучения иностранных слов платформы Android, было отобрано несколько популярных приложений.

Reword – приложение для изучения английских слов. Имеет встроенный словарь из тысячи английских слов и словосочетаний, разбитых по 50 категориям [1].

Плюсами данного приложения являются:

- встроенный словарь, представляющий собой тысячи английских слов и словосочетаний, разбитых на 50 категорий;

- простой и приятный для пользователя интерфейс;

- отслеживание выученных пользователем слов;
- оффлайн режим работы;
- статистика обучения.

Минусами программы являются:

- ограниченное время использования приложения в бесплатной версии;
- большое количество появляющейся рекламы;
- невозможно перенести прогресс на другое устройство;
- большое количество ненужных уведомлений от приложения.

Внешне приложение представлено на рисунке 1.

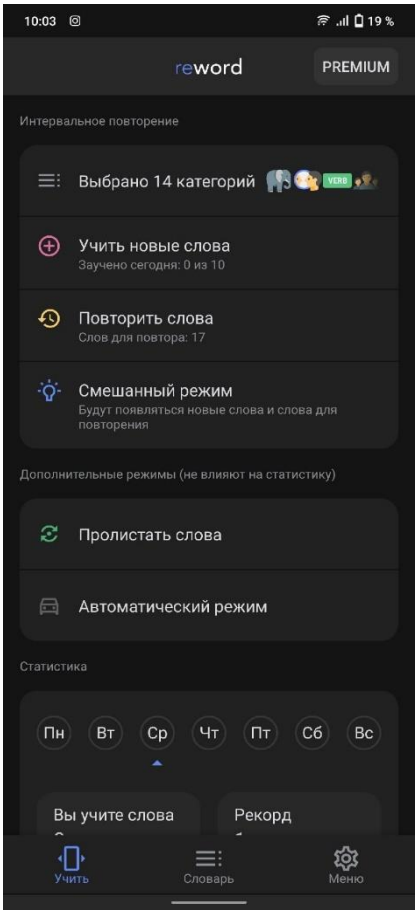


Рисунок 1– *Reword*

AnkiDroid - открытая мультиплатформенная программа для запоминания слов, основывающаяся на методике интервального повторения. Интервальное повторение - это техника обучения, при которой повторение материала (новых слов, формул и т.д.) совершается через определенные

промежутки времени, таким образом, обеспечивая долговременное запоминание [2].

Плюсами данного приложения можно назвать следующие моменты:

- пользователь самостоятельно может создать собственную коллекцию карт для изучения слов;

- полная совместимость на различных платформах;

- доступна возможность скачивания более 6000 готовых колод на множестве языков;

- сохранение прогресса обучения.

Минусами данного приложения являются:

- сложный интерфейс для скачивания колод карт;

- импорт данных в приложение занимает много времени;

- отсутствие упражнений;

- частые вылеты и ошибки в работе программы.

Приложение показано на рисунке 2.

Рисунок 2 – *AnkiDroid*

1.1.3 Вывод из сравнения приложений

После изучения аналогичных приложений были выявлены следующие недостатки:

- сложный интерфейс;
- трудность загрузки и создания тем слов для изучения;
- ограниченное время работы приложения;
- навязчивая реклама и уведомления.

Решением этих недостатков будет отказ от использования рекламы в приложении. В результате принято решение не ограничивать для пользователя время использования программы во время обучения. Большое внимание следует уделить разработке интерфейса и выбрать комфортную цветовую гамму, чтобы пользователю было удобно использовать приложение в любое время.

1.1.4 Постановка задачи

Целью работы является разработка программного обеспечения для изучения иностранных языков посредством запоминания новых слов, которое должно осуществлять следующие функции:

- авторизация пользователя;
- выбор упражнения для изучения слов;
- изучения слов выбранной группы посредством выбора ответа на заданный вопрос;
- отображение реакции на выбор ответа пользователя;
- вывод итогового счета;
- отображение последнего результата;
- простой и интуитивно понятный для пользователя графический интерфейс.

1.1.5 Выбор инструмента для решения поставленной задачи

На основании поставленной задачи был использован такой инструмент программирования приложения как *Android Studio*. Это интегрированная среда разработки (*IDE)* для работы с платформой *Android*. *Android Studio* -

программное обеспечение от компании *JetBrains,* официальное средство разработки *Android* приложений. Среда разработки доступна для работы на различных операционных системах, таких как *Windows, macOS* и *Linux*. Впоследствии была добавлена поддержка языка *Kotlin,* используемого в *Android Studio*, как официального языка программирования для платформы *Android* в добавок к *Java и C++.*

Операционная система *Android* пользуется большой популярностью среди пользователей мобильных устройств. На рынке представлено большое количество устройств различного форм-фактора и ценового диапазона, что привлекает внимание широкой аудитории.

1.1.6 Преимущество *Android Studio*

Интегрированная среда разработки *Android Studio* [3] поддерживает работу с популярными языками программирования – *C/C++, Java, Kotlin*. Является официальной средой разработки приложений платформы Android. Удобный редактор кода, большая библиотека с готовыми компонентами и шаблонами для разработки приложений. В *Android Studio* есть возможность разработки приложения для последней версии *Android N* и после создания приложения можно проверить на наличие ошибок благодаря большому набору инструментов для тестирования каждого элемента приложения. Имеются различные виды сборки и генерации нескольких *.apk* файлов. В Android Studio есть встроенный статический анализатор кода для проведения тестов производительности, несовместимость версий разрабатываемых приложений и при необходимости проводить оптимизацию. Также имеется встроенная поддержка *Google Cloud Platform*, которая позволяет разработчику использовать в своих приложениях интеграцию с сервисами *Google Cloud Messaging и App Engine*

1.1.7 Выбор языка разработки

Для разработки приложения необходимо определиться с тем, на каком языке оно будет написано. Рассмотрим два таких языка как *Java* и *Kotlin* [4].

Java является официальным и самым популярным языком программирования, его часто применяют при мобильной *Android*-разработке. Язык так же популярен для работы не только с *Android*. Он занимает лидирующие позиции в топе *TIOBE,* имеет большое количество исходных материалов на *GitHub.* Для новичков *Java* может показаться немного сложным. Это становится препятствием, с которым сталкиваются люди, которые планируют попасть в разработку приложений для *Android.*

Kotlin зарекомендовал себя в качестве более молодого официального языка для разработки *Android*. Как и *Java, Kotlin* работает на виртуальной машине *Java.* Он полностью совместим с *Java* и не вызывает никаких сложностей или изменения размера файлов. Главное отличие заключается в том, что *Kotlin* требует меньше времени, т.е. более простая для чтения система. Он также устраняет такие ошибки, как исключение нулевого указателя, и освобождает пользователя от необходимости заканчивать каждую строку точкой с запятой. Это отличный язык программирования для начинающего *Android* разработка.

Проанализировав выбранные языки, для разработки приложения был выбран *Java.* Большое количество готовой информации, библиотек для решения поставленных задач, много разнообразных инструмент для написания программ.

1.2 Анализ требований

1.2.1 Функциональные требования к проектируемой системе

Функциональные требования определяют опциональность создаваемого программного обеспечения, чтобы пользователь смог осуществить свои потребности. В процессе анализа были выдвинуты требования:

- авторизация пользователя в приложении;
- предоставление выбора тем для изучения;
- программа должна быть в формате тестирования;
- отображение реакции на выбранный ответ;

- вывод итогового счета после прохождения теста;

- отображение последнего результата.

1.2.2 Нефункциональные требования к проектируемой системе

Нефункциональные требования показывают ограничения системы. Для проектирования приложения были выдвинуты следующие требования:

- приложение должно быть написано на языке *Java* с использованием *Android SDK*;

- приложение должно работать на платформе *Android 7.0* и выше.

1.2.3 Диаграмма прецедентов

В процессе анализа выявлен основной клиент системы – Пользователь. Он имеет возможность использовать все функции приложения, указанные далее:

- пользователь может осуществить регистрацию и вход в приложение;

- пользователь может выбирать категорию слов из списка для дальнейшего изучения;

- пользователь может выполнять задания в выбранной категории выбирая вариант ответа из списка;

- пользователь может получить реакцию приложения на свои ответы;

- пользователь, после прохождения теста, получает счет в данной категории.

Диаграмма вариантов использования представлена в приложении А.

1.2.4 Диаграмма деятельности

Во время анализа был разработан основной алгоритм работы программы. Обработка событий должна происходить следующим образом:

- при запуске программы выводится окно с полями ввода логина и пароля, с помощью которых пользователь может зарегистрироваться и войти в приложение;

- после входа в учетную запись, выводится окно приложения, где представлены темы для изучения слов и последний счет;

- после выбора темы, происходит переход к окну с вопросом и выбором ответов на него;

- во время решения теста приложение выводит экран с реакцией на верный или неправильный ответ;

- после прохождения теста приложение выводит счет и комментарий;

- если не был произведен выбор другой группы слов, приложение может быть закрыто.

Диаграмма обработки событий представлена в приложении А.

1.2.5 Диаграмма классов

Также во время анализа была разработана схема классов. В нее входят активити содержащие задания и ответы на них, подсчет верных ответов на вопросы, реакция на ответы и вывод счета. Диаграмма классов представлена в приложении А.

1.2.6 Диаграмма последовательности

Была разработана диаграмма последовательности, которая показывает связь клиента и базы данных. Диаграмма последовательности представлена в приложении А.

1.3 Технологии разработки

1.3.1 Разработка клиентской части

В ходе разработки клиентской части были реализованы *XML* – файлы верстки для каждого из окон:

- *activity_login* предназначена для окна регистрации. В нем описываются два поля ввода для логина и пароля, при чем символы при вводе скрываются за знаками звездочки, две кнопки «Войти» и «Зарегистрироваться», которые позволяют осуществлять вход и регистрацию соответственно;

- *verticallayout* служит для отображения тем слов и вывода последнего набранного счета во время прохождения тестирования;

- *activity_qnumber* содержит в себе поле для вопроса в виде английского слова, четыре *radioButton* с вариантами ответов в виде русских слов и кнопка «Подтвердить»;

- *activity_right* имеет вид окна зеленого цвета с изображением галочки. Служит для уведомления пользователя о данном верном ответе;

-activity_wrong имеет вид окна красного цвета с изображенным на нем символа крестика. Уведомляет пользователя о ложном ответе на данный вопрос;

- *activity_show_score* содержит в себе поле для вывода счета после прохождения теста на экран, поле для вывода мотивационного сообщения.

Также для клиентской части были разработаны *JAVA*-классы, которые описывают функционирование приложения. Процесс авторизации в приложении описывается в классе *LoginActivity*. В нем обрабатываются события нажатия кнопок «Войти» и «Зарегистрироваться». В первом случае осуществляется запрос к базе данных [5] по поиску элемента со значением текстового поля «Логин». Если была найдена запись, то в полученной строке проверяется поле «Пароль» на соответствие с введенными данными. Если проверка выполнена успешно, то будет произведен переход к *MainActivity*. Иначе будет выведено сообщение о некорректности введенных данных. Фрагмент кода, выполняющий проверку данных по базе данных на этапе входа приведен на рисунке 3.

Во втором случае введённые данные пройдут проверку на соответствие определенному формату [6], а так будет выполнен запрос на существование записи для данного пользователя. Если данные такого пользователя в базе данных не найдены, то в ней создастся новая запись, иначе выведется сообщение о просьбе сменить логин.

В классе *MainActivity* для отображения списка тем используется *ArrayAdapter*. По нажатию на один из элементов списка осуществляется переход к классу *QNumberActivity* с передачей в него параметра, содержащего

в себе номер темы. Метод обработки нажатия на элемент списка представлен на рисунке 4.

Для отображения последнего набранного счета осуществляется запрос к базе данных для поиска результата последнего тестирования для данного пользователя.

```
db = databaseHelper.getReadableDatabase();
                query = db.rawQuery("SELECT * FROM "+
DataBaseContract.UserEntry.TABLE_NAME_1 + " where
"+DataBaseContract.UserEntry.T1_ATR2 +"=?", new String[]{strLogin});
  if (query.moveToNext()){
      String checkPassword = query.getString(2);
      if (strPassword.equals(checkPassword))
      {

          textView.setText("Ok");

          db.close();
          SharedPreferences.Editor editor = mSettings.edit();
          editor.putString(APP_PREFERENCES_LOGIN, strLogin);
          editor.putString(APP_PREFERENCES_PASS, strPassword);
          editor.apply();
          Intent user = new Intent(LoginActivity.this,
MainActivity.class);
          user.putExtra("login", strLogin);
          startActivity(user);
      }
      else{
          textView.setText("Неверный пароль");
          db.close();
      }
  }
  else{
      textView.setText("Пользователя не существует");
      db.close();
  }
```

Рисунок 3 - Проверка данных по базе данных

```
@Override
public void onItemClick(AdapterView<?> a, View v, int
position, long id) {
                Intent i;
                i = new Intent(MainActivity.this,
QNumberActivity.class);
                i.putExtra("login", login);
                i.putExtra("question", position +1);
startActivity(i);
            }
        });
```

Рисунок 4 - Метод обработки нажатия на элемент списка

В классе *QNumberActivity* на основе полученного входного параметра с номером темы выполняется формирование списка вопросов для пользователя. На экране отображается первый элемент списка в виде английского слова и вариантов его перевода. При выборе пользователем верного варианта ответа увеличивается счет и вызывается класс *RightActivity*. Если выбор был сделан неправильно, то счет не изменяется и вызывается класс *WrongActivity*. Метод, выполняющий переход к активностям верного и ложного ответа, представлен на рисунке 5.

Для отображения последнего набранного счета осуществляется запрос к базе данных для поиска результата последнего тестирования для данного пользователя.

В классе *QNumberActivity* на основе полученного входного параметра с номером темы выполняется формирование списка вопросов для пользователя [7]. На экране отображается первый элемент списка в виде английского слова и вариантов его перевода. При выборе пользователем верного варианта ответа увеличивается счет и вызывается класс *RightActivity*. Если выбор был сделан неправильно, то счет не изменяется и вызывается класс *WrongActivity*. Метод,

выполняющий переход к активностям верного и ложного ответа, представлен на рисунке 5.

```
public Intent isRightOrWrong(String Answer){
        Intent screen;


        if(Answer.equals(rightAnswer)) {
            score += 1;
            screen = new Intent(QNumberActivity.this,
RightActivity.class);
        }


else {
            screen = new Intent(QNumberActivity.this,
WrongActivity.class);
        }
```

Рисунок 5 - Метод перехода к *RightActivity* и *WrongActivity*

В независимости от верности ответа, вопрос удаляется из списка, что позволяет при каждом новом запуске активности отобразить последовательно все вопросы. Когда вопросы закончатся, набранный счет сохранится в базу данных и вызовется *ShowScoreActivity* с параметром, в котором содержится счет.

Классы *RightActivity и WrongActivity* ожидают от пользователя нажатия на экран. Когда пользователь его совершает, будет выполнен переход к предыдущей активности.

Класс *ShowScoreActivity* выполняет отображение полученного счета, его анализ, в результате которого выведется одно из трех мотивационных сообщений: «Ты очень умный!», «Поздравляем!», «Вам нужно больше учиться...». При нажатии пользователя на экран выполнится переход к *MainActivity*. Метод анализирующий результат тестирования представлен на рисунке 6.

```
private String setStatus(String scores){
        int score = Integer.parseInt(scores);

        if(score >= 8){
            return "Ты очень умный!";
        }

        if (score >= 5){
            return "Поздравляем!";
        }
        return "Вам нужно больше учиться...";
    }
```

Рисунок 6 - Метод анализа результата тестирования

1.3.2 Разработка базы данных

Для хранения данных пользователей, была разработана база данных, состоящая из двух таблиц [8]. Первая таблица предназначена для хранения логинов и паролей пользователей. Она содержит три поля:

- *U_ID* – идентификатор записи;
- *U_LOGIN* – поле логина;
- *U_PASSWORD* - поле пароля.

Во второй таблице хранятся последний набранный счет в результате тестирования для конкретных пользователей. Она содержит три поля:

- *D_ID* – идентификатор записи;
- *D_U_ID* – внешний ключ для связи с первой таблицей;
- *D_SCORE* – поле последнего результата тестирования.

Модель базы данных представлена на рисунке 7.

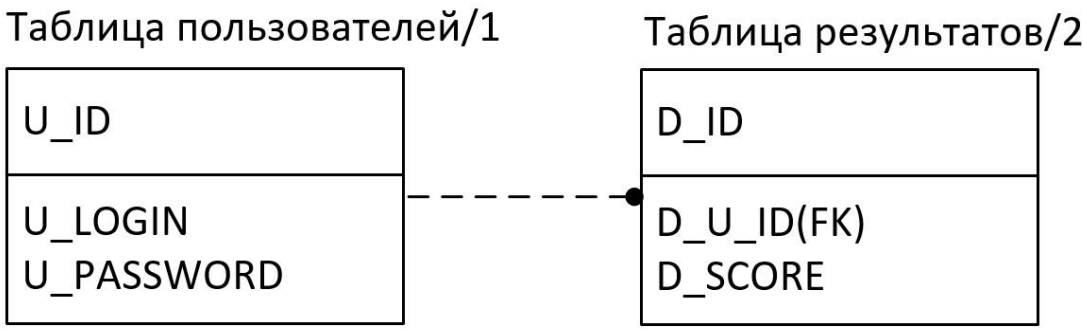


Рисунок 7 - Модель базы данных

1.4 Руководство пользователя

При запуске приложения пользователь видит окно авторизации, в котором он может зарегистрироваться в приложении или выполнить вход уже в имеющуюся учетную запись. Во время регистрации пароль проверяется на количество символов, их должно быть не меньше восьми. Если выбранный пользователем логин был уже занят ранее, выведется сообщение с ошибкой и учетная запись не будет допущена к регистрации. Во время входа по введенному логину осуществляется запрос к базе данных, если по данному логину имеется запись, то выполняется проверка пароля. Если данные введены корректно, то выводится сообщение и осуществляется вход. Окно регистрации представлено на рисунке 8.

После авторизации в приложении на экране пользователю отображается последний набранный счет и список тем для изучения, при нажатии на любую из которых производится переход к выбранной активности, где и происходит выбор ответов на выбранную категорию вопросов. Также на этом фрагменте будет представлены новые темы для изучения. Интерфейс показан на рисунке 9.

Рисунок 8 - Окно авторизации

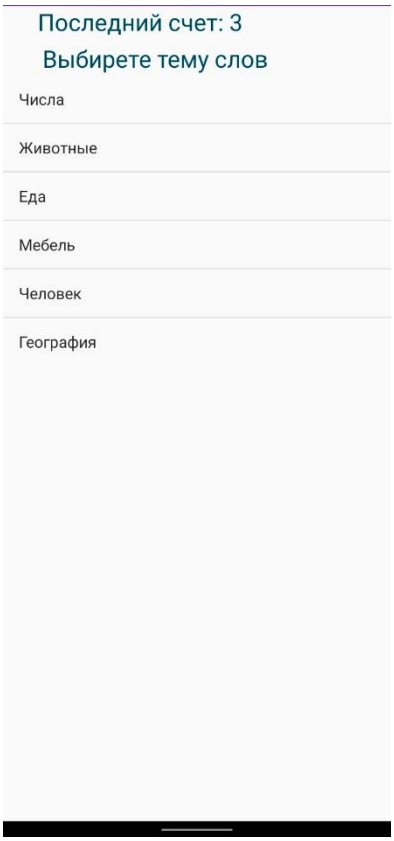


Рисунок 9 - Список тем для изучения

После выбора категории осуществляется переход к выбранной теме. В новом окне отображается вопрос, варианты ответа и кнопка «Подтвердить». Для перехода к следующему вопросу необходимо выбрать вариант ответа и нажать кнопку «Подтвердить». Количество слов в каждой теме составляет десять штук. Интерфейс показан на рисунке 10.

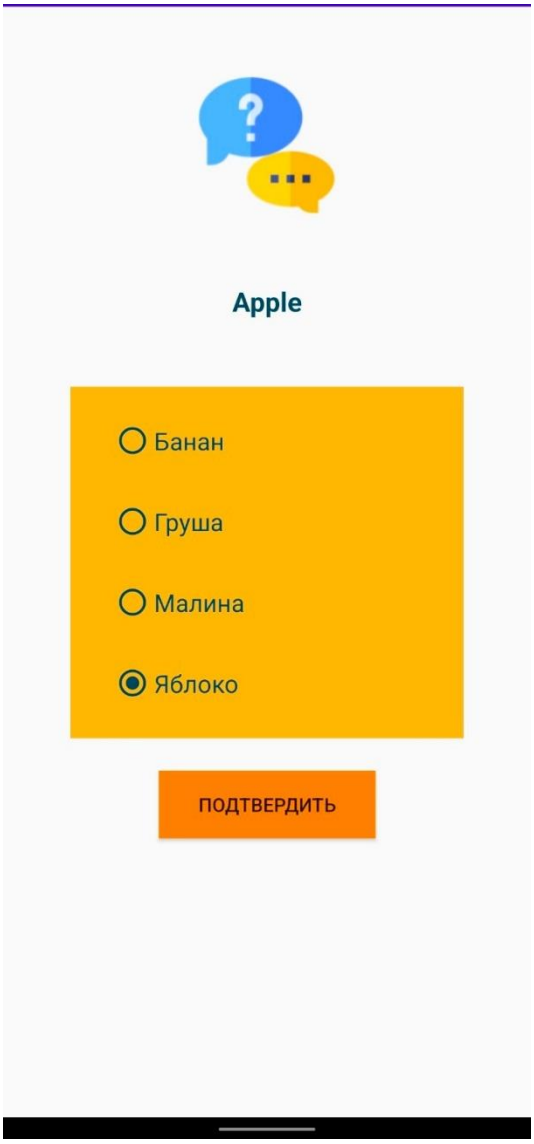


Рисунок 10- Интерфейс ответов на вопросы

Если был дан неверный ответ пользователь увидит активность, где отображается отрицательная реакция. Интерфейс показан на рисунке 11. Если был выбран верный вариант, отобразится положительная реакция на ответ пользователя. Интерфейс представлен на рисунке 12. По нажатию на

активности реакции на ответ пользователя осуществляется переход к следующему вопросу.

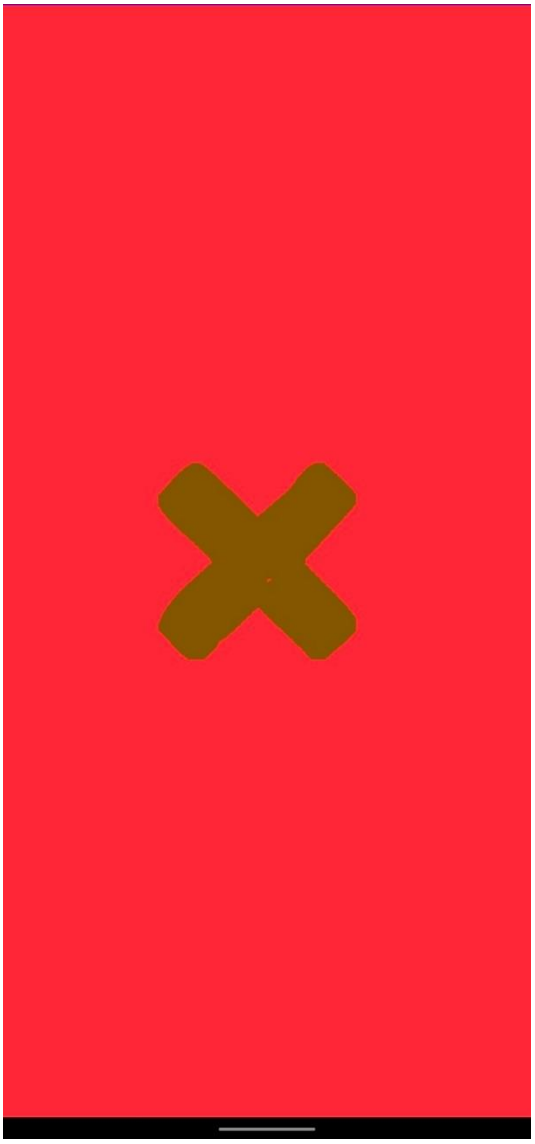


Рисунок 11- Неправильный ответ

Последний экран приложения служит для отображения результата прохождения теста в выбранной категории. Если было набрано восемь и более баллов, будет выведено сообщение «Ты очень умный!». Если было от семи до пяти баллов, то отобразится сообщение «Поздравляем!». В противном случае на экран выведется сообщение «Вам нужно больше учиться...». После прохождения теста и получения результатов, пользователь может вернуться к выбору новой темы и увидеть обновление поля последнего полученного результата. Интерфейс показан на рисунке 13.

Рисунок 12- Правильный ответ

Рисунок 13- Результат прохождения теста

1.5 Тестирование

Для демонстрации работы приложения было проведено тестирование. На рисунках 14- 23 представлены вопросы по теме «Животные» и реакции на ответ по ним. На рисунке 24 изображен итоговый счет и последний результат тестирования в окне выбора темы. На рисунке 25 представлены другие мотивационные комментарии на полученный счет.

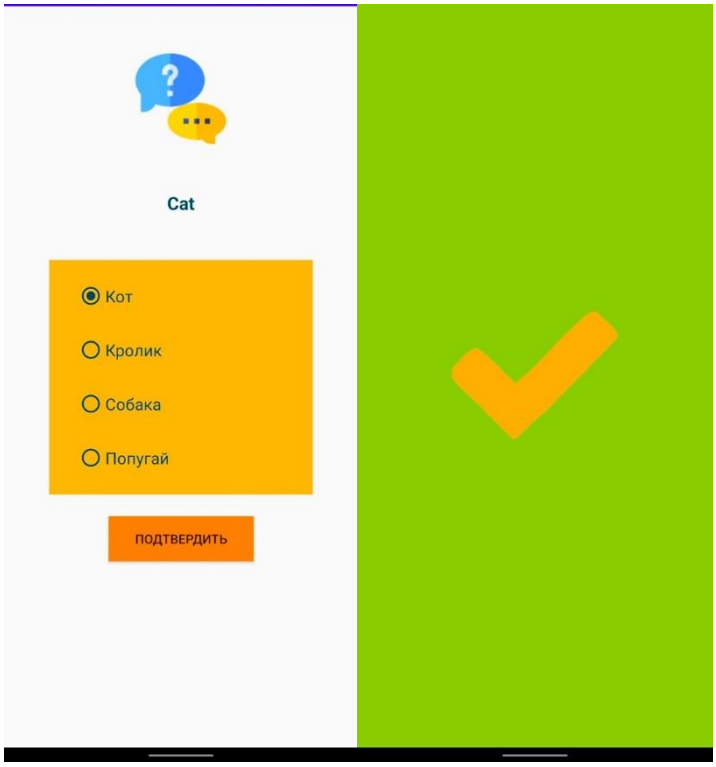


Рисунок 14- Вопрос 1 (верно)

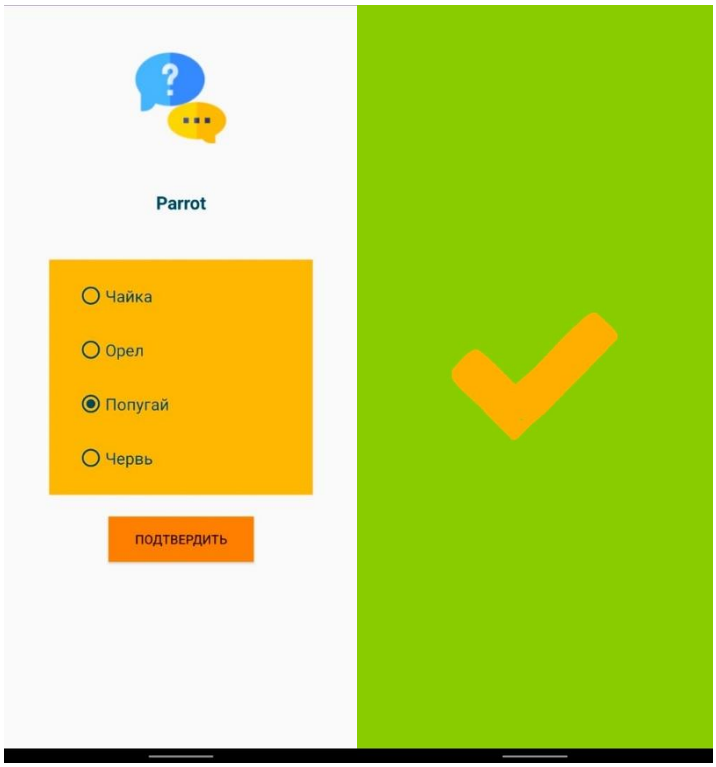


Рисунок 15 -Вопрос 2 (верно)

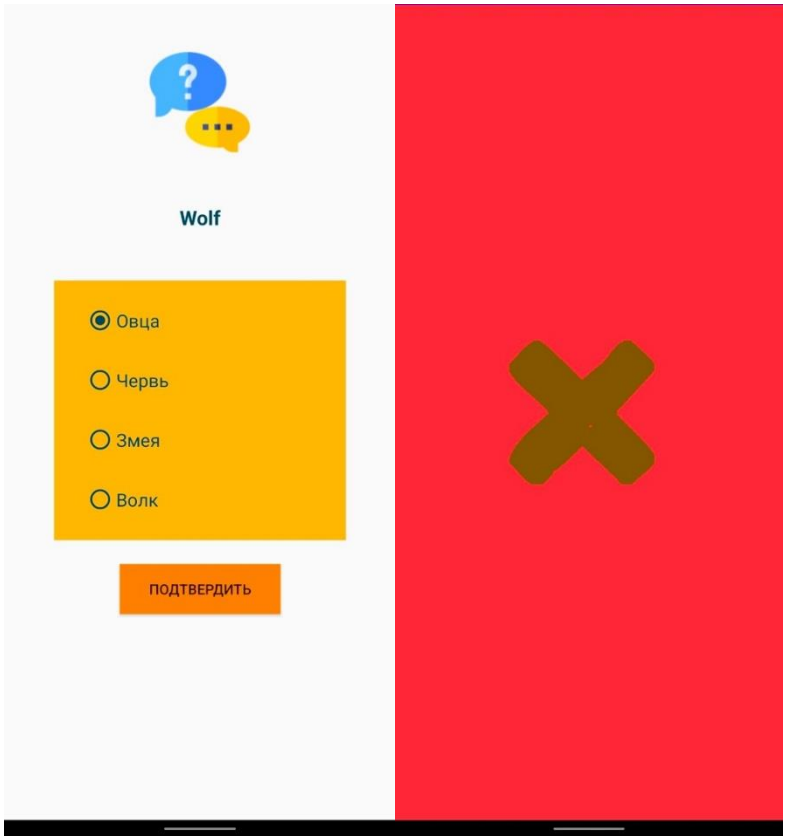


Рисунок 16 - Вопрос 3 (не верно)

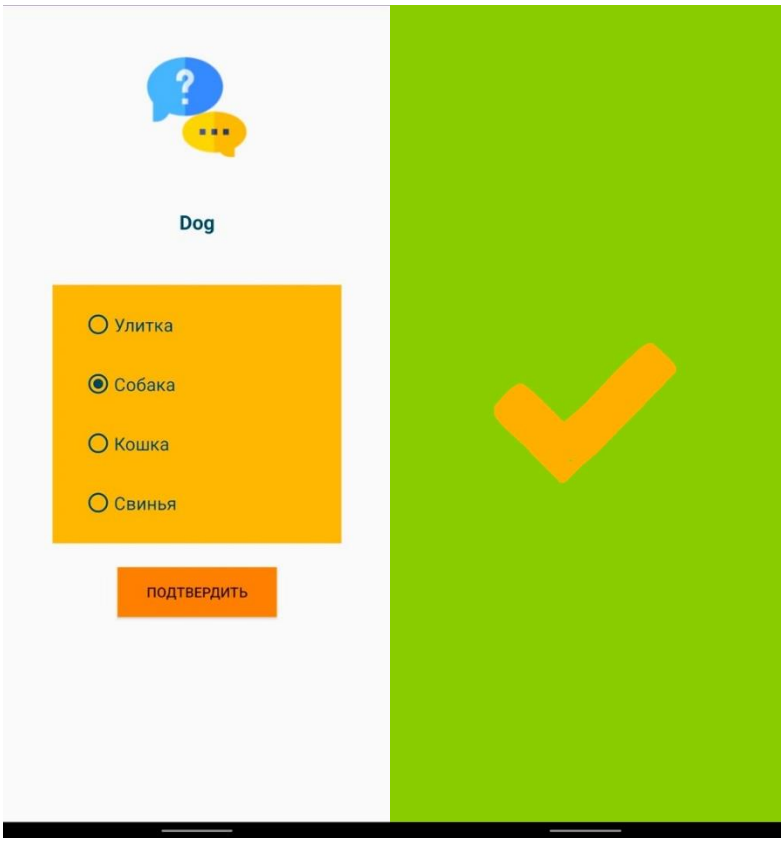


Рисунок 17 - Вопрос 4 (верно)

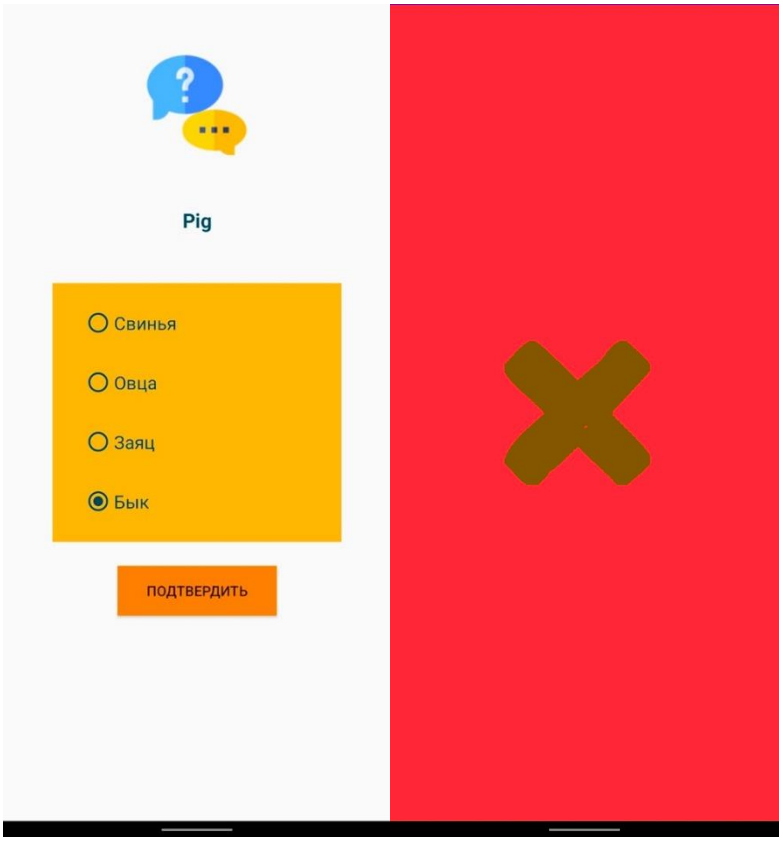


Рисунок 18 - Вопрос 5 (не верно)

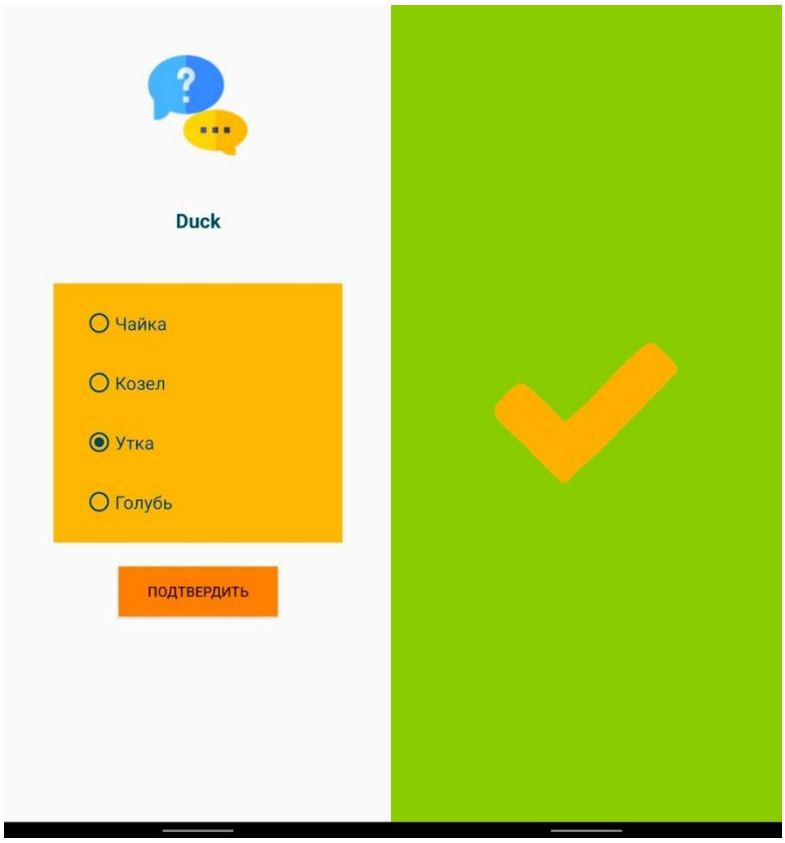


Рисунок 19 - Вопрос 6 (верно)

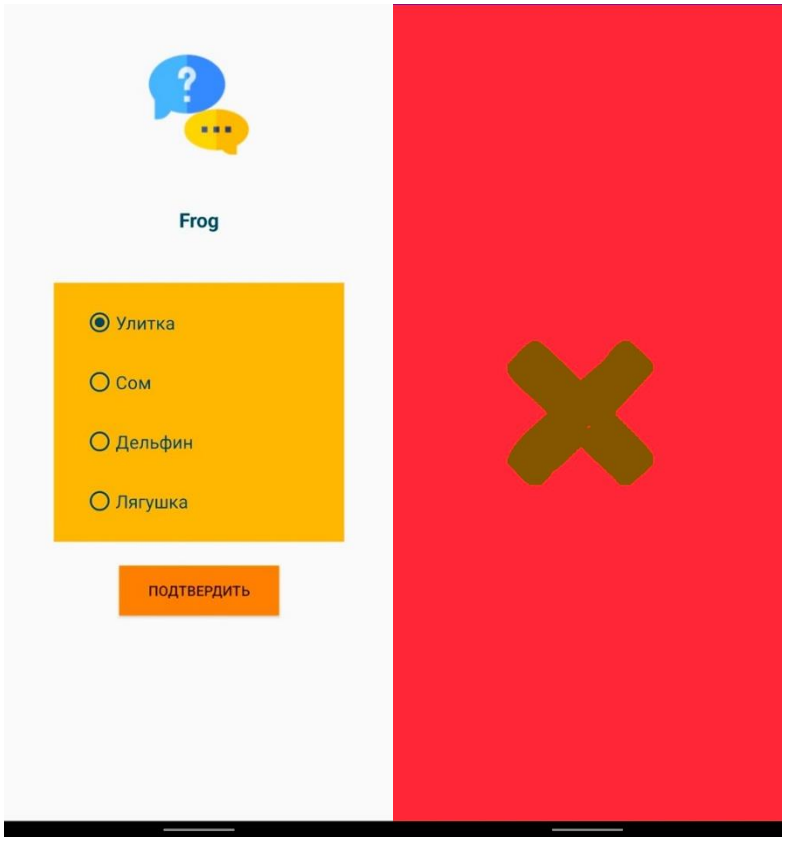


Рисунок 20 - Вопрос 7 (не верно)

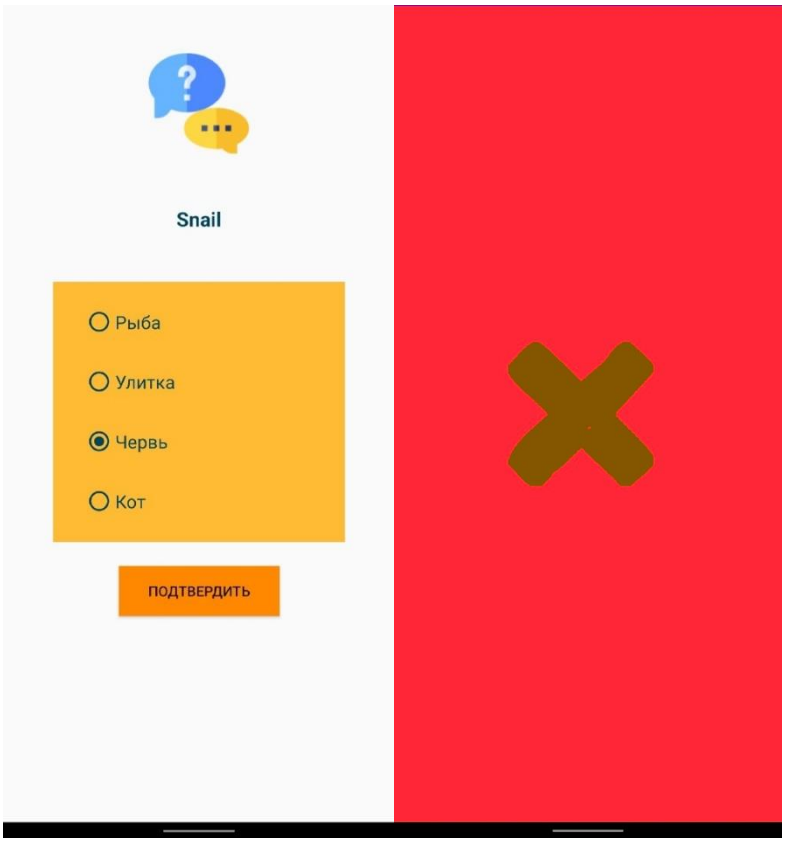


Рисунок 21 - Вопрос 8 (не верно)

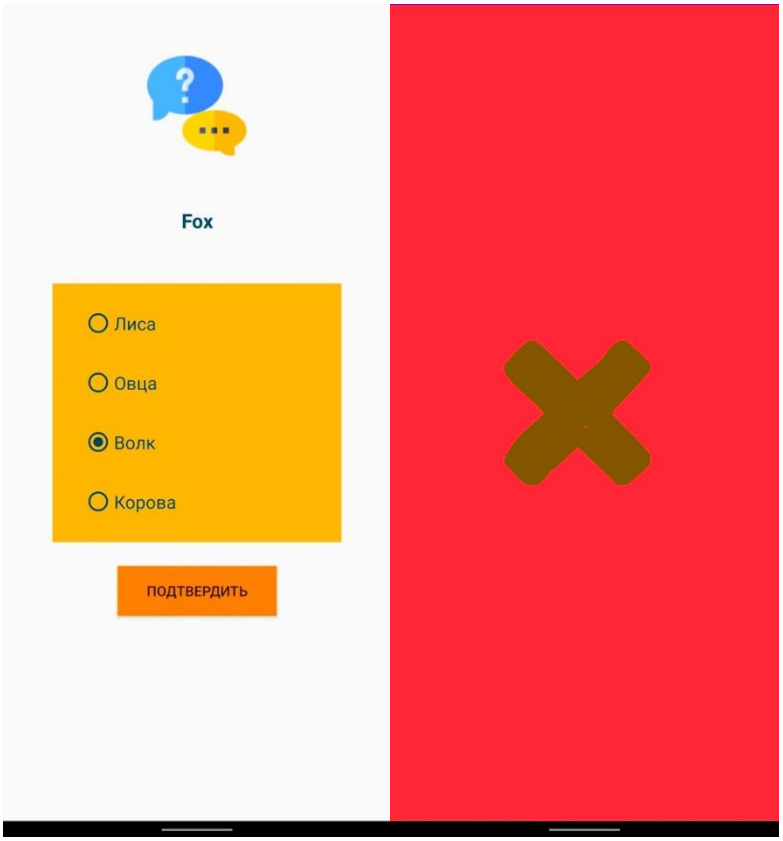


Рисунок 22 - Вопрос 9 (не верно)

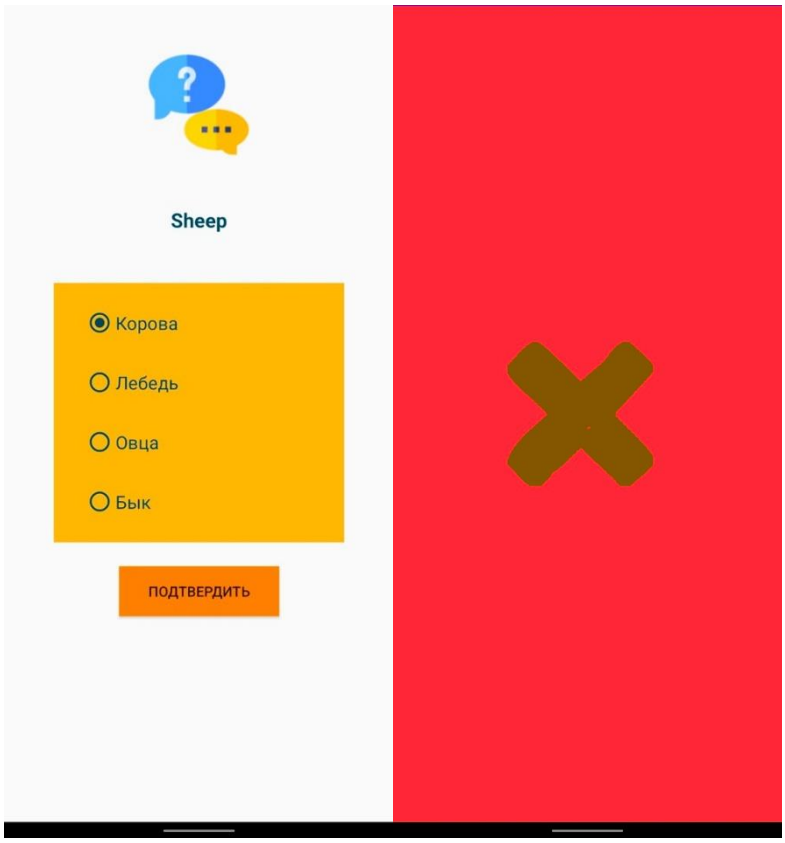


Рисунок 23 - Вопрос 10 (не верно)

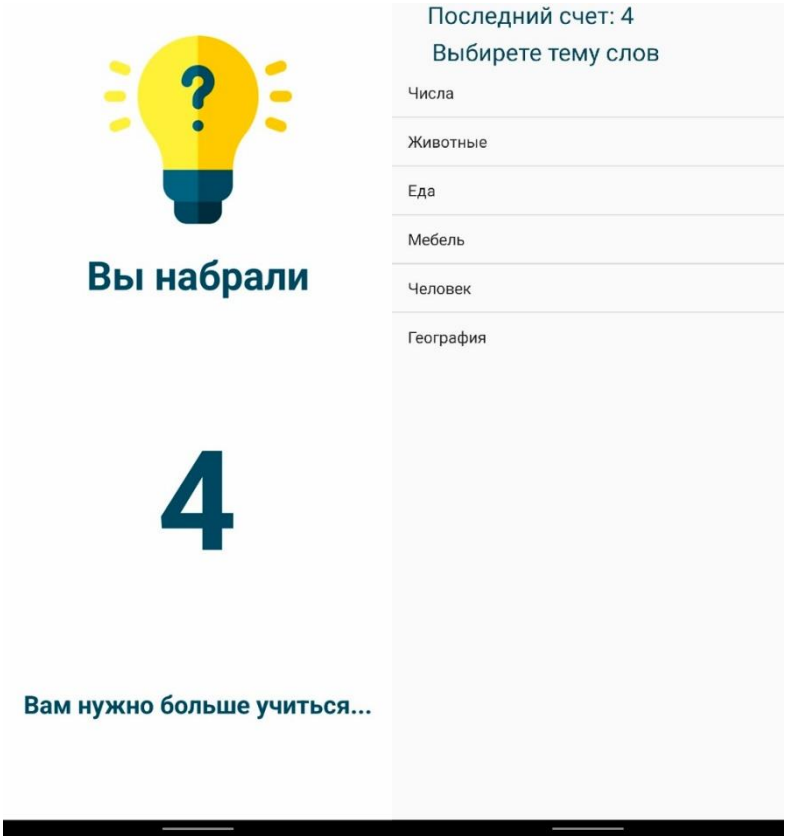


Рисунок 24 - Итоговый счет и его отображение в окне выбора темы

Рисунок 25 - Реакции на разное количество баллов

2 Безопасность жизнедеятельности

2.1 Введение

На фоне технологического прогресса, стремительно увеличивающегося производства, введения современных технических средств и техники, увеличения значительности человека на предприятии и общественной важности защищенных и благоприятных условий труда, проблема безопасности жизнедеятельности заслуживает особенную важность.

Введение вычислительной техники на производстве даёт положительный социально-экономический эффект, что выражается в росте производительности, понижении доли рутинного и монотонного труда, повышения быстроты расчётов, скорости обмена информацией. Отрицательное воздействие на человека вычислительной техники менее выражено, сглажено многими позитивными моментами. Впрочем, у людей длительно использующих ПЭВМ могут иметься такие реакции как нарушение функций зрения, быстрое общее утомление. Для того чтобы избежать вредного влияния при работе с вычислительной техникой необходимо придерживаться соответствующих мер безопасности, правильно планировать рабочее пространство и режим работы.

Напряженность труда – характеристика трудового процесса, отражающая нагрузку преимущественно на центральную нервную систему, органы чувств, эмоциональную сферу работника. К факторам, характеризующим напряженность труда, относятся: интеллектуальные, сенсорные, эмоциональные нагрузки, степень монотонности нагрузок, режим работы.

Постоянная работа в условиях повышенной напряженности труда может привести к нервному истощению работника, зарождению у него разнообразных заболеваний, сопряженных с расстройством центральной нервной системы, следовательно нужно определять уровень напряженности труда и при его повышенном значении предусмотреть мероприятия по его

снижению. Работодатель обязан организовать условия труда, отвечающие нормативам, так как превышение гигиенических нормативов представляется нарушением Законов Российской Федерации и основанием для использования органами и учреждениями Федеральной службы по надзору в сфере защиты прав потребителей и благосостояния человека и другими контролирующими организациями для применения санкций за вредные и опасные условия труда.

Чрезмерно напряженный трудовой процесс приводит к скорой утомляемости работника, что в свою очередь приводит к снижению его работоспособности, производительности труда, а также может привести к ошибкам и последствиям, приносящим множество вреда, вред на производстве и в последствии, что вдобавок не отвечает и интересам работодателя. Также в условиях недостатка квалифицированных кадров, от здоровья и работоспособности трудящегося зависит успех предприятия.

Таким образом напряженность труда представляется значимой характеристикой трудового процесса, которую надлежит отслеживать, учитывать и принимать соответствующие меры по ее снижению.

2.2 Методика оценки напряженности трудового процесса

Напряженность трудового процесса оценивают в соответствии с настоящими «Гигиеническими критериями оценки условий труда по показателям вредности и опасности факторов производственной среды, тяжести и напряженности трудового процесса».

Анализ напряженности труда профессиональной группы сотрудников основана на оценке трудовой деятельности и ее структуры, которые изучаются путем хронометражных наблюдений в динамике всего рабочего дня, в течение не менее одной недели. Оценка основана на учете всего комплекса производственных условий труда (стимулов, раздражителей), создающих предпосылки для возникновения негативных нервно-эмоциональных состояний (перенапряжения). Все факторы (показатели) трудового процесса

имеют качественную или количественную выраженность и сгруппированы по видам нагрузок:

- сенсорные;

- эмоциональные;

- монотонные;

- режимные нагрузки.

2.2.1 Нагрузки интеллектуального характера

Нагрузки интеллектуального характера оцениваются по следующим критериям:

- «Содержание работы» указывает на степень сложности выполнения задания: от решения простых задач до творческой (эвристической) деятельности с решением сложных заданий при отсутствии алгоритма;

- «Восприятие сигналов (информации) и их оценка». Критериальным с точки зрения различий между классами напряженности трудового процесса является установочная цель (или эталонная норма), которая принимается для сопоставления поступающей при работе информации с номинальными значениями, необходимыми для успешного хода рабочего процесса;

- «Распределение функций по степени сложности задания». Любая трудовая деятельность характеризуется распределением функций между работниками. Соответственно, чем больше возложено функциональных обязанностей на работника, тем выше напряженность его труда;

- «Характер выполняемой работы» - в том случае, когда работа выполняется по индивидуальному плану, то уровень напряженности труда невысок. Если работа протекает по строго установленному графику с возможной его коррекцией по мере необходимости, то напряженность повышается.

2.2.2 Сенсорные нагрузки

Сенсорные нагрузки оцениваются по следующим критериям:

- «Длительность сосредоточенного наблюдения (в % от времени смены)» -чем больше процент времени отводится в течение смены на сосредоточенное

наблюдение, тем выше напряженность. Общее время рабочей смены принимается за 100 %;

- «Плотность сигналов (световых, звуковых) и сообщений в среднем за 1 час работы» - количество воспринимаемых и передаваемых сигналов (сообщений, распоряжений) позволяет оценивать занятость, специфику деятельности работника;

- «Число производственных объектов одновременного наблюдения» - указывает, что с увеличением числа объектов одновременного наблюдения возрастает напряженность труда;

- «Размер объекта различения при длительности сосредоточенного внимания (% от времени смены)». Чем меньше размер рассматриваемого предмета (изделия, детали, цифровой или буквенной информации и т. п.) и чем продолжительнее время наблюдения, тем выше нагрузка на зрительный анализатор;

- «Работа с оптическими приборами (микроскоп, лупа и т.п.) при длительности сосредоточенного наблюдения (% от времени смены)». На основе хронометражных наблюдений определяется время (часы, минуты) работы за оптическим прибором;

- «Наблюдение за экраном видеотерминала (ч в смену)». Согласно этому показателю фиксируется время (ч, мин) непосредственной работы пользователя ВДТ с экраном дисплея в течение всего рабочего дня при вводе данных, редактировании текста или программ, чтении информации буквенной, цифровой, графической с экрана;

- «Нагрузка на слуховой анализатор». Степень напряжения слухового анализатора определяется по зависимости разборчивости слов в процентах от соотношения между уровнем интенсивности речи и «белого» шума;

- «Нагрузка на голосовой аппарат (суммарное количество часов наговариваемых в неделю)». Степень напряжения голосового аппарата зависит от продолжительности речевых нагрузок.

2.2.3 Эмоциональные нагрузки

Эмоциональные нагрузки оцениваются по следующим критериям:

- «Степень ответственности за результат собственной деятельности. Значимость ошибки» - указывает, в какой мере работник может влиять на результат собственного труда при различных уровнях сложности осуществляемой деятельности;

- «Степень риска для собственной жизни». Мерой риска является вероятность наступления нежелательного события, которую с достаточной точностью можно выявить из статистических данных производственного травматизма на данном предприятии и аналогичных предприятиях отрасли;

- «Ответственность за безопасность других лиц». При оценке напряженности необходимо учитывать лишь прямую, а не опосредованную ответственность, то есть такую, которая вменяется должностной инструкцией;

- «Количество конфликтных производственных ситуаций за смену». Наличие конфликтных ситуаций в производственной деятельности ряда профессий существенно увеличивают эмоциональную нагрузку и подлежат количественной оценке.

2.2.4 Монотонность нагрузок

Монотонность нагрузок оцениваются по следующим критериям:

- «Число элементов (приемов), необходимых для реализации простого задания или многократно повторяющихся операций»;

- «Продолжительность (с) выполнения простых производственных заданий или повторяющихся операций» - чем меньше число выполняемых приемов и чем короче время, тем, соответственно, выше монотонность нагрузок;

- «Время активных действий (в % к продолжительности смены)». Наблюдение за ходом технологического процесса не относится к «активным действиям»;

- «Монотонность производственной обстановки (время пассивного наблюдения за ходом техпроцесса, в % от времени смены)» - чем больше время

пассивного наблюдения за ходом технологического процесса, тем более монотонной является работа.

2.2.5 Режим работы

Режим работы оцениваются по следующим критериям:

- «Фактическая продолжительность рабочего дня» - выделен в самостоятельную рубрику, так как независимо от числа смен и ритма работы фактическая продолжительность рабочего дня колеблется от 6—8 ч до 12 ч и более;

- «Сменность работы» определяется на основании внутрипроизводственных документов, регламентирующих распорядок труда на данном предприятии;

- «Наличие регламентированных перерывов и их продолжительность (без учета обеденного перерыва)». К регламентированным перерывам следует относить только те перерывы, которые введены в регламент рабочего времени на основании официальных внутрипроизводственных документов, таких как коллективный договор, приказ директора предприятия или организации, либо на основании государственных документов - санитарных норм и правил, отраслевых правил по охране труда и других.

2.3 Общая оценка напряженности трудового процесса

Независимо от профессиональной принадлежности (профессии) учитываются все 23 показателя, перечисленные в таблице 1. Не допускается выборочный учет каких-либо отдельно взятых показателей для общей оценки напряженности труда.

По каждому из 23 показателей в отдельности определяется свой класс условий труда. В том случае, если по характеру или особенностям профессиональной деятельности какой-либо показатель не представлен (например, отсутствует работа с экраном видеотерминала или оптическими приборами), то по данному показателю ставится 1 класс (оптимальный) - напряженность труда легкой степени.

При окончательной оценке напряженности труда.

«Оптимальный» (1 класс) устанавливается в случаях, когда 17 и более показателей имеют оценку 1 класса, а остальные относятся ко 2 классу. При этом отсутствуют показатели, относящиеся к 3 (вредному) классу.

«Допустимый» (2 класс) устанавливается в следующих случаях:

- когда 6 и более показателей отнесены ко 2 классу, а остальные -к 1 классу;

- когда от 1 до 5 показателей отнесены к 3.1 и/или 3.2 степеням вредности, а остальные показатели имеют оценку 1-го и/или 2-го классов.

«Вредный» (3) класс устанавливается в случаях, когда 6 или более показателей отнесены к третьему классу (обязательное условие).

При соблюдении этого условия труд, напряженный 1-й степени (3.1):

- когда 6 показателей имеют оценку только класса 3.1, а оставшиеся показатели относятся к 1 и/или 2 классам;

- когда от 3 до 5 показателей относятся к классу 3.1, а от 1 до 3 показателей отнесены к классу 3.2.

Труд, напряженный 2-й степени (3.2):

- когда 6 показателей отнесены к классу 3.2;

- когда более 6 показателей отнесены классу 3.1;

- когда от 1 до 5 показателей отнесены к классу 3.1, а от 4 до 5 показателей -к классу 3.2;

- когда 6 показателей отнесены к классу 3.1 и имеются от 1 до 5 показателей класса 3.2.

В тех случаях, когда более 6 показателей имеют оценку 3.2, напряженность трудового процесса оценивается на одну степень выше - класс 3.3.

2.4 Оценка напряженности труда пользователя ПЭВМ

Напряженность трудового процесса оценивают в соответствии с настоящими «Гигиеническими критериями оценки условий труда по

показателям вредности и опасности факторов производственной среды, тяжести и напряженности трудового процесса» по таблице 1.

Таблица 1 – Оценка напряженности труда пользователя ПЭВМ

Показатели		Класс условий труда				
1		2	3	4	5	6
		1	2	3.1	3.2	3.3
1. Интеллектуальные нагрузки						
1.1	Содержание работы				+	
1.2	Восприятие сигналов и их оценка		+			
1.3	Распределение функции по степени сложности задания		+			
1.4	Характер выполняемой работы			+		
2. Сенсорные нагрузки						
2.1	Длительность сосредоточенного наблюдения	+				
2.2	Плотность сигналов за 1 час работы	+				
2.3	Число объектов одновременного наблюдения	+				
2.4	Размер объекта различения при длительности сосредоточенного внимания		+			
2.5	Работа с оптическими приборами при длительности сосредоточенного наблюдения	+				
2.6	Наблюдение за экраном видеотерминала		+			
2.7	Нагрузка на слуховой анализатор	+				
2.8	Нагрузка на голосовой аппарат	+				
3. Эмоциональные нагрузки						

Показатели		Класс условий труда				
1		2	3	4	5	6
		1	2	3.1	3.2	3.3
3.1	Степень ответственности за результат собственной деятельности. Значимость ошибки.		+			
3.2	Степень риска для собственной жизни	+				
3.3	Ответственность за безопасность других лиц	+				
3.4	Количество конфликтных производственных ситуаций за смену	+				
4. Монотонность нагрузок						
4.1	Число элементов, необходимых для реализации простого задания или многократно повторяющихся операций	+				
4.2	Продолжительность выполнения простых заданий или повторяющихся операций	+				
4.3	Время активных действий	+				
4.4	Монотонность производственной обстановки	+				
5. Режим работы						
5.1	Фактическая продолжительность рабочего дня	+				
5.2	Сменность работы	+				
5.3	Наличие регламентированных перерывов и их продолжительность		+			
Количество показателей в каждом классе		15	6	1	1	0
Общая оценка напряженности труда			+			

Один показатель относится к классу 3.1, Один показатель относится к классу 3.2, пяинадцать показателей к первому классу и 6 ко второму. Таким образом, общая оценка напряженности труда пользователя ПЭВМ соответствует классу 2 «Допустимый».

2.5 Заключение

Наибольшую напряженность пользователь ПЭВМ испытывает в связи с интеллектуальными нагрузками, чуть меньшую из-за сенсорных нагрузок и практически не испытывает напряженности от эмоциональных нагрузок, нагрузок напряженности и режима работы.

Хотя общая оценка напряженности труда пользователя ПЭВМ соответствует классу 2 «Допустимый», интеллектуальные нагрузки на пользователя ПЭВМ особо велики, поэтому следует предусмотреть мероприятия по снижению напряженности труда, для того, чтобы сохранить здоровье работника и избежать ухудшения качества его работы и нанесение ущерба от невнимательной, малоэффективной работы.

Заключение

В ходе выполнения выпускной квалификационной работы разработано программное обеспечение для интерактивного обучения. Были реализованы авторизация пользователя, выбор упражнения для изучения слов, изучение слов выбранной группы посредством выбора ответа на заданный вопрос, отображение реакции на выбор ответа пользователя, вывод итогового счета, отображение последнего результата, простой и интуитивно понятный для пользователя графический интерфейс.

В результате выполнения выпускной квалификационной работы был достигнут следующий ряд задач:

- анализ предметной области и изучение инструментов разработки *Android Studio*;
- установка требований к программе;
- смоделирована архитектура разрабатываемого приложения;
- создано мобильное приложение;
- осуществлено тестирование.

Направление дальнейших исследований – добавление в приложение следующих функций:

- увеличение количество тем для изучения;
- добавление словаря;
- возможность создания своих тем для повторения;
- функция озвучивания слова в вопросе.

Список использованных источников

1. Сайт *Reword.* [Электронный ресурс]: Режим доступа: *https://reword.app/ru/en.*

2. Сайт *AnkiDroid*. [Электронный ресурс]: Режим доступа: *https://apps.ankiweb.net/.*

3. Документация *Android Studio*. [Электронный ресурс]: Режим доступа: *https://developer.android.com/studio*.

4. Как выбрать язык программирования. [Электронный ресурс]: Режим доступа: *https://habr.com/ru/post/477578/.*

5. *SQLite – a software library that implements a self-contained, server- less, zero-configuration, transactional SQL database engine.* [Электронный ресурс] Режим доступа: *https://sqlite.org/index.html*.

6. Герберт Шилдт. *Java*, Руководство для начинающих, 7-е издание. – СПб.: ООО «Диалектика», 2019. — 816 с.

7. Кей С. Хорстманн. *Java*. Библиотека профессионала, том 1. Основы. 11-е изд.: СПб.: ООО «Диалектика», 2019. — 864 с.

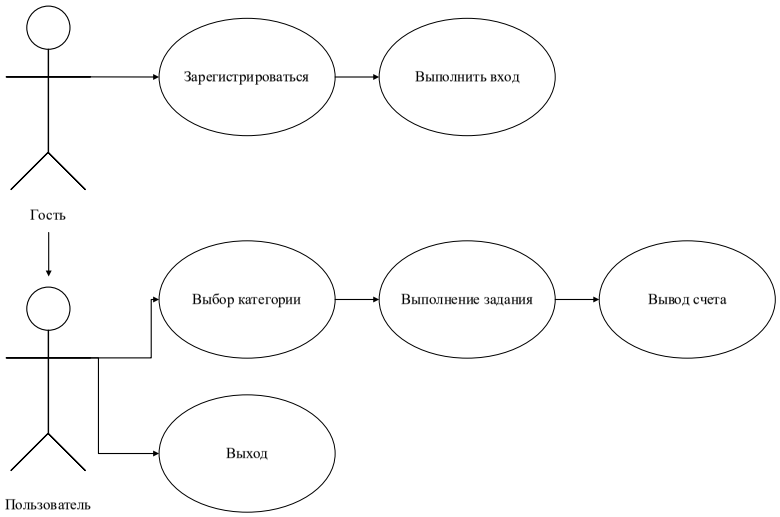
8. Основные понятия баз данных. [Электронный ресурс]: Режим доступа: *https://www.sites.google.com/site/razrabotkaizasitabazdannyh/osnovnye-ponatia-baz-dannyh.*

Приложение А

(Обязательное)

Программное обеспечение для интерактивного обучения

UML - диаграммы

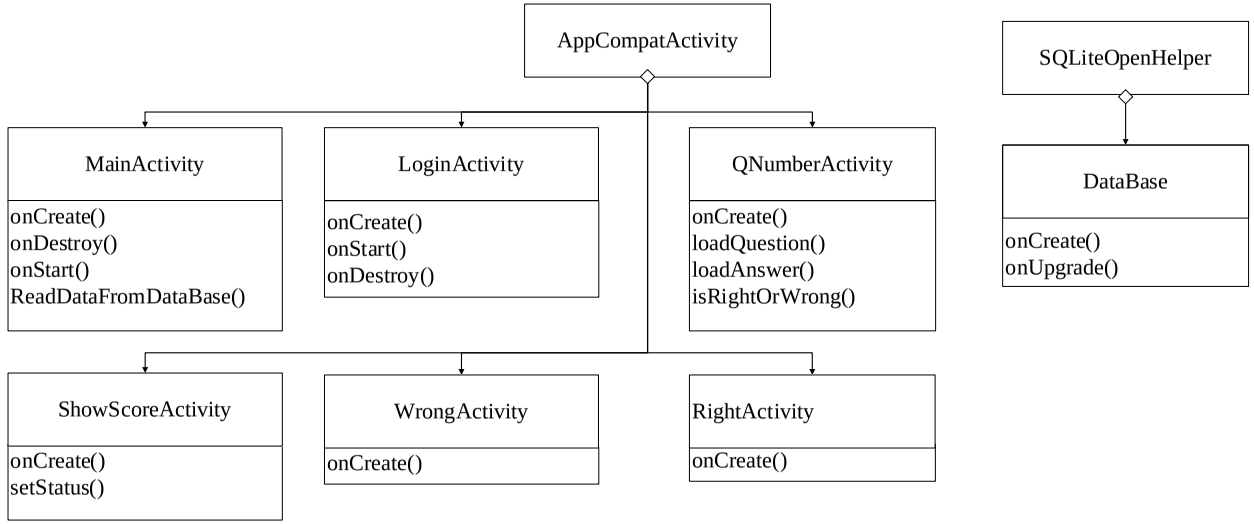


					ПГУ 1.090301.18.002 Д13	
Изм.	Лист	№ докум.	Подпись	Дата	Программное обеспечение для интерактивного обучения	Лит. / Масса / Масштаб
Разраб.		Кочкаткин Д.А.			Диаграмма прецедентов	
Пров.		Бикташев Р.А.				Лист 1 / Листов 1
Т. контр.						
Н. контр.		Бычков А.С.				ПГУ, каф.ВТ, гр. 18ВВ1
Утв.		Матросов М.А.				

Изм. | Лист | № докум. | Подпись | Дата

ПГУ 1.090301.18.001 ПЗ

Лист 45



ПГУ 1.090301.18.003 Д11

Программное обеспечение для интерактивного обучения

Диаграмма классов

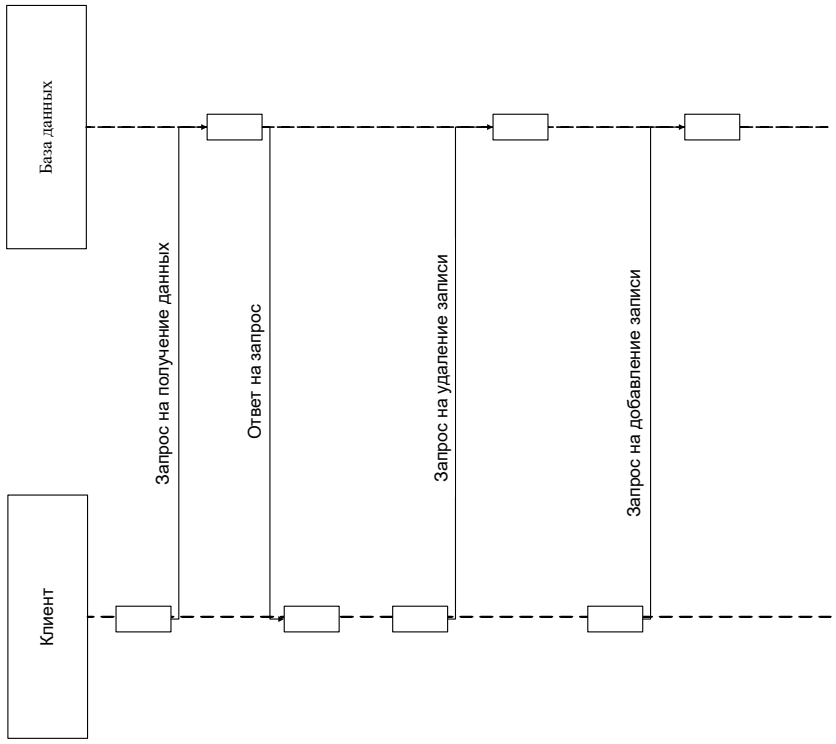
Изм.	Лист	№ докум.	Подпись	Дата
Разраб.	Ясноватова Д.А.			
Пров.	Бикташев Р.А.			
Т. контр.				
Н. контр.	Бычков А.С.			
У. в.	Митрохин М.А.			

Лит.	Масса	Масштаб
Лист 1	Листов 1	

ПГУ, каф.ВТ, гр. 18ВВ1

ПГУ 1.090301.18.001 ПЗ

Изм. | Лист | № докум | Подпись | Дата



ПГУ 1.090301.18.004 Д14

Программное обеспечение для системы технического обучения

Диаграмма последовательностей

Лит. | Масс. | Масштаб

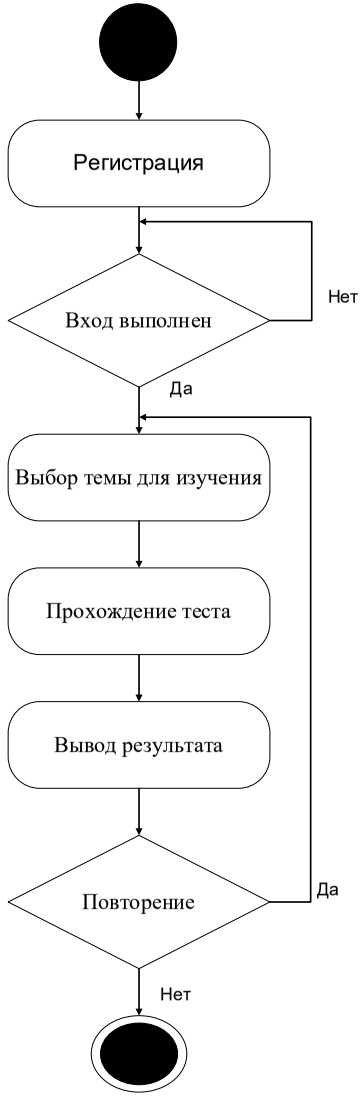
Лист 1 | Листов 1

ПГУ, каф. ВТ, гр. 18ВВ1

Изм.	Лист	№ докум	Подпись	Дата

ПГУ 1.090301.18.001 ПЗ

Лист 47



									ПГУ 1.090301.18.005 Д17
Изм.	Лист	№ докум.	Подпись	Дата					
Разраб.		Кононалов Д.А.			Программное обеспечение для интерактивного обучения		Лит.	Масса	Масштаб
Пров.		Бикташев Р.А.							
Т. контр.					Диаграмма деятельности		Лист 1	Листов1	
Н. контр.		Быченов А.С.							
Утв.		Митрохин М.А.					ПГУ, каф.ВТ, гр. 18ВВ1		

Приложение Б

(Обязательное)

Программное обеспечение для интерактивного обучения

Листинг программы

Файл DataBase.java

```
package com.example.myapplication;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;

public class ShowScoreActivity extends AppCompatActivity {
    TextView TxtScore;
    TextView TxtStatus;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_show_score);

        TxtScore = findViewById(R.id.txtscore);
        TxtStatus = findViewById(R.id.txtStatus);
        Intent intent = getIntent();
        String scores =
String.valueOf(intent.getIntExtra("score", 0));

        TxtScore.setText(scores);
        TxtStatus.setText(setStatus(scores));

        ConstraintLayout root = (ConstraintLayout)
findViewById(R.id.layout);
        root.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }});
    }
```

```
    private String setStatus(String scores){
        int score = Integer.parseInt(scores);

        if(score >= 8){
            return "Ты очень умный!";
        }

        if (score >= 5){
            return "Поздравляем!";
        }

        return "Вам нужно больше учиться...";

    }


    public void goToHome(View v){
        Intent home = new Intent(ShowScoreActivity.this,
MainActivity.class);
        startActivity(home);
        finish();
    }

}
```

Файл DataBaseContract.java

```
package com.example.myapplication;

import android.provider.BaseColumns;

public class DataBaseContract {
    private DataBaseContract() {}
    public static class UserEntry implements BaseColumns {
        public static final String TABLE_NAME_1 = "USERS";
        public static final String T1_ATR1 = "U_ID";
```

```
        public static final String T1_ATR2 = "U_LOGIN";
        public static final String T1_ATR3 = "U_PASSWORD";
    }
    public static class DataEntry implements BaseColumns {
        public static final String TABLE_NAME_2 = "DATA";
        public static final String T2_ATR1 = "D_ID";
        public static final String T2_ATR2 = "D_U_ID";
        public static final String T2_ATR3 = "D_SCORE";

    }
}
```

Файл LoginActivity.java

```
package com.example.myapplication;

import android.content.ContentValues;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class LoginActivity extends AppCompatActivity {

    DataBase databaseHelper;
    SQLiteDatabase db;
    Cursor query;

    public static final String APP_PREFERENCES = "settings";
    public static final String APP_PREFERENCES_LOGIN = "login";
```

```
    public static final String APP_PREFERENCES_PASS =
"password";
    public static boolean isShowAlert = true;
    SharedPreferences mSettings;
    EditText editlogin;
    EditText editpassword;
    TextView textView;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        databaseHelper = new DataBase(getApplicationContext());

        mSettings = getSharedPreferences(APP_PREFERENCES,
Context.MODE_PRIVATE);
        editlogin = findViewById(R.id.loginPlace);
        editpassword = findViewById(R.id.passwordPlace);

editlogin.setText(mSettings.getString(APP_PREFERENCES_LOGIN,
""));

editpassword.setText(mSettings.getString(APP_PREFERENCES_PASS,
""));


        final Button button =
(Button)findViewById(R.id.button_Enter);
        textView = findViewById(R.id.textView);


        // Кнопка войти
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

```
                String strLogin =
editlogin.getText().toString();
                String strPassword =
editpassword.getText().toString();

                db = databaseHelper.getReadableDatabase();
                query = db.rawQuery("SELECT * FROM "+
DataBaseContract.UserEntry.TABLE_NAME_1 + " where
"+DataBaseContract.UserEntry.T1_ATR2 +"=?", new
String[]{strLogin});
                if (query.moveToNext()){
                    String checkPassword = query.getString(2);
                    if (strPassword.equals(checkPassword))
                    {
                        textView.setText("Ok");
                        db.close();
                        SharedPreferences.Editor editor =
mSettings.edit();
                        editor.putString(APP_PREFERENCES_LOGIN,
strLogin);
                        editor.putString(APP_PREFERENCES_PASS,
strPassword);
                        editor.apply();
                        Intent user = new
Intent(LoginActivity.this, MainActivity.class);
                        user.putExtra("login", strLogin);
                        startActivity(user);
                    }
                    else{
                        textView.setText("Неверный пароль");
                        db.close();
                    }
                }
                else{
                    textView.setText("Пользователя не
существует");
```

```
                    db.close();
                }

            }
        });

        final Button button4 =
(Button)findViewById(R.id.button_Create);
        //Кнопка регистрации
        button4.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final String strLogin =
editlogin.getText().toString();
                final String strPassword =
editpassword.getText().toString();
                db = databaseHelper.getReadableDatabase();
                query = db.rawQuery("SELECT * FROM "+
DataBaseContract.UserEntry.TABLE_NAME_1 + " where
"+DataBaseContract.UserEntry.T1_ATR2 +"=?", new
String[]{strLogin});
                if (query.moveToNext()) {
                    textView.setText("Логин занят");
                    db.close();
                }
                else if(strLogin.length() > 0) {
                    db.close();
                    if (strPassword.length() >=8) {

                        Thread regThread = new Thread( //
создаём новый поток
                                new Runnable() { // описываем
объект Runnable в конструкторе
                                    public void run() {

regUser(strLogin,strPassword); // вызываем метод воспроизведения
```

```
                                    }
                                }

                            );
                            regThread.start();
                            try {
                                regThread.join();
                                textView.setText("Регистрация
успешна выполните вход");
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                        }

                        else{
                            textView.setText("Слабый пароль");
                        }
                    }
                    else{
                        db.close();
                        textView.setText("Некорректный логин");
                    }
                }
            });
        }
        @Override
        protected void onStart(){
            super.onStart();
            isShowAlert = true;

            Bundle arguments = getIntent().getExtras();
            if (arguments!= null){
                String sms = arguments.get("sms").toString();
                textView.setText(sms);
            }
```

```
    }
    @Override
    protected void onPause(){
        super.onPause();
        isShowAlert = false;
    }
    @Override
    protected void onStop(){
        super.onStop();

    }
    @Override
    protected void onDestroy(){
        databaseHelper.close();
        super.onDestroy();
    }

    public void regUser(String login, String password){
        SQLiteDatabase dbInThread =
databaseHelper.getWritableDatabase();
        ContentValues newUser = new ContentValues();
        newUser.put(DataBaseContract.UserEntry.T1_ATR2, login);
        newUser.put(DataBaseContract.UserEntry.T1_ATR3,
password);

dbInThread.insert(DataBaseContract.UserEntry.TABLE_NAME_1, null,
newUser);
        dbInThread.close();
    }
}
```

Файл MainActivity.java

```
package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
```

```
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.database.Cursor;
import android.widget.TextView;


public class MainActivity extends AppCompatActivity {


    private final static String TAG = "MainActivity";
    String login;
    DataBase databaseHelper;
    SQLiteDatabase db;
    TextView textView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.verticallayout);
        Log.d(TAG, "onCreate");
        Bundle arguments = getIntent().getExtras();
        login = arguments.get("login").toString();
        databaseHelper = new DataBase(getApplicationContext());
        //Вывод счета на экран
        textView = findViewById(R.id.textView5);
        String score = ReadDataFromDataBase(login);
        if (score != null)
        textView.setText("Последний счет: "+ score);
        else
            textView.setText("Последний счет: неизвестно");
        final String[] catNames = new String[] {
```

```
                "Числа", "Животные", "Еда", "Мебель", "Человек",
"География"
        };

        ArrayAdapter<String> adapter = new
ArrayAdapter<String>(this, R.layout.simple_list_item_1,
catNames);
        final ListView listView = (ListView)
findViewById(R.id.countriesList);

        listView.setAdapter(adapter);

        // When the user clicks on the ListItem
        listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> a, View v,
int position, long id) {
                Intent i;
                i = new Intent(MainActivity.this,
QNumberActivity.class);
                i.putExtra("login", login);
                switch(position)
                {
                    case 0:
                        i.putExtra("question", "1");
                        break;
                    case 1:
                        i.putExtra("question", "2");
                        break;
                    case 2:
                        i.putExtra("question", "3");
                        break;
                    case 3:
                        i.putExtra("question", "4");
```

```
                    break;
                case 4:
                    i.putExtra("question", "5");
                    break;
                case 5:
                    i.putExtra("question", "6");

                    break;
            }
            startActivity(i);
        }
    });

}

@Override
protected void onDestroy(){
    databaseHelper.close();
    super.onDestroy();
    Log.d(TAG, "onDestroy");
}
@Override
protected void onStop(){
    super.onStop();
    Log.d(TAG, "onStop");
}
@Override
protected void onStart(){
    super.onStart();
    Log.d(TAG, "onStart");
}
@Override
protected void onPause(){
    super.onPause();
    Log.d(TAG, "onPause");
}
```

```
    @Override
    protected void onResume(){
        super.onResume();
        Log.d(TAG, "onResume");
    }

    @Override
    protected void onRestart(){
        super.onRestart();
        String score = ReadDataFromDataBase(login);
        if (score != null)
            textView.setText("Последний счет: "+ score);
        else
            textView.setText("Последний счет: неизвестно");
        Log.d(TAG, "onRestart");
    }

    public String ReadDataFromDataBase(String login) {
        String count = null;
        db = databaseHelper.getReadableDatabase();
        Cursor query = db.rawQuery("SELECT * FROM "+
DataBaseContract.DataEntry.TABLE_NAME_2 +" where "+
DataBaseContract.DataEntry.T2_ATR2+"=?", new String[]{login});
        if(query.moveToFirst()){
                int id = query.getInt(0);
                count = query.getString(2);
        }
        query.close();
        db.close();
        return count;
    }
}
```

Файл QNumberActivity.java

```
package com.example.myapplication;
```

```
import androidx.appcompat.app.AppCompatActivity;

import android.content.ContentValues;
import android.content.Intent;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;

import java.util.ArrayList;
import java.util.List;

public class QNumberActivity extends AppCompatActivity {
    DataBase databaseHelper;
    SQLiteDatabase db;
    RadioGroup radioGroup;
    TextView lblQuestion;
    RadioButton optionA;
    RadioButton optionB;
    RadioButton optionC;
    RadioButton optionD;
    Button confirm;
    String rightAnswer;
    String Answer;
    List<Question> questions;
    String login;
    int score;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_qnumber);
        lblQuestion = findViewById(R.id.lblQuestion);
        optionA = findViewById(R.id.optionA);
```

```
        optionB = findViewById(R.id.optionB);
        optionC = findViewById(R.id.optionC);
        optionD = findViewById(R.id.optionD);
        score = 0;
        radioGroup = findViewById(R.id.radioGroup);
        Bundle arguments = getIntent().getExtras();
        String question = arguments.get("question").toString();
        login = arguments.get("login").toString();
        switch (question){
            case "1":
                questions = new ArrayList<Question>(){
                    {
                        add(new Question("One", "C", "Два",
"Четыре","Один", "Ноль"));
                        add(new Question("Two", "A", "Два",
"Семь","Три", "Пять"));
                        add(new Question("Three", "D", "Один",
"Четыре","Шесть", "Три"));
                        add(new Question("Four", "B", "Девять",
"Четыре", "Восемь", "Пять"));
                        add(new Question("Five", "A", "Пять",
"Восемь", "Один", "Три"));
                        add(new Question("Six", "D", "Девять",
"Семь", "Два", "Шесть"));
                        add(new Question("Seven", "C", "Пять",
"Один", "Семь", "Три"));
                        add(new Question("Eight", "B", "Ноль",
"Восемь", "Четыре", "Два"));
                        add(new Question("Nine", "B", "Три",
"Девять", "Четыре", "Один"));
                        add(new Question("Zero", "D", "Четыре",
"Семь", "Два", "Ноль"));
                    }
                };
                break;
            case "2":
```

```
                questions = new ArrayList<Question>(){
                    {
                        add(new Question("Cat", "A", "Кот",
"Кролик", "Собака", "Попугай"));
                        add(new Question("Parrot", "C", "Чайка",
"Орел", "Попугай", "Червь"));
                        add(new Question("Wolf", "D", "Овца",
"Червь", "Змея", "Волк"));
                        add(new Question("Dog", "B", "Улитка",
"Собака", "Кошка", "Свинья"));
                        add(new Question("Pig", "A", "Свинья",
"Овца", "Заяц", "Бык"));
                        add(new Question("Duck", "C", "Чайка",
"Козел", "Утка", "Голубь"));
                        add(new Question("Frog", "D", "Улитка",
"Сом", "Дельфин", "Лягушка"));
                        add(new Question("Snail", "B", "Рыба",
"Улитка", "Червь", "Кот"));
                        add(new Question("Fox", "A", "Лиса",
"Овца", "Волк", "Корова"));
                        add(new Question("Sheep", "C", "Корова",
"Лебедь", "Овца", "Бык"));
                    }
                };
                break;
            case "3":
                questions = new ArrayList<Question>(){
                    {
                        add(new Question("Apple", "D", "Банан",
"Груша","Малина", "Яблоко"));
                        add(new Question("Bread", "B", "Масло",
"Хлеб","Суп", "Курица"));
                        add(new Question("Apricot", "A",
"Абрикос", "Яблако","Виноград", "Персик"));
                        add(new Question("Cheese", "C",
"Колбаса", "Мясо", "Сыр", "Курица"));
```

```
                        add(new Question("Juice", "D", "Соус",
"Яблако", "Майонез", "Сок"));
                        add(new Question("Chicken", "A",
"Курица", "Мясо", "Сыр", "Хлеб"));
                        add(new Question("Potato", "B",
"Авокадо", "Картофель", "Помидор", "Капуста"));
                        add(new Question("Milk", "D", "Сок",
"Масло", "Вода", "Молоко"));
                        add(new Question("Sugar", "A", "Сахар",
"Соль", "Перец", "Чеснок"));
                        add(new Question("Oil", "C", "Вода",
"Абрикос", "Масло", "Сок"));
                    }
                };
                break;
            case "4":
                questions = new ArrayList<Question>(){
                    {
                        add(new Question("Clock", "D", "Стол",
"Шкаф", "Ковер", "Часы"));
                        add(new Question("Fireplace", "C",
"Диван", "Ковер", "Камин", "Стул"));
                        add(new Question("Chair", "A", "Стул",
"Стол", "Диван", "Кровать"));
                        add(new Question("Sofa", "C", "Дверь",
"Окно", "Диван", "Камин"));
                        add(new Question("Bed", "B", "Стол",
"Кровать", "Холодильник", "Зеркало"));
                        add(new Question("Mirror", "D", "Дверь",
"Окно", "Лампа", "Зеркало"));
                        add(new Question("Door", "A", "Дверь",
"Часы", "Комната", "Ковер"));
                        add(new Question("Lamp", "B", "Ковер",
"Лампа", "Стул", "Кровать"));
                        add(new Question("Carpet", "C",
"Кровать", "Стол", "Ковер", "Часы"));
```

```
                        add(new Question("Table", "D", "Камин",
"Шкаф", "Зеркало", "Стол"));
                    }
                };
                break;
            case "5":
                questions = new ArrayList<Question>(){
                    {
                        add(new Question("Arm", "C", "Нога",
"Голова", "Рука", "Палец"));
                        add(new Question("Eye", "D", "Рот",
"Нос", "Ухо", "Глаз"));
                        add(new Question("Body", "D", "Голова",
"Шея", "Нос", "Тело"));
                        add(new Question("Head", "C", "Ухо",
"Тело", "Голова", "Шея"));
                        add(new Question("Leg", "B", "Рука",
"Нога", "Палец", "Плечо"));
                        add(new Question("Face", "B", "Голова",
"Лицо", "Рот", "Глаз"));
                        add(new Question("Nose", "C", "Ухо",
"Шея", "Нос", "Глаз"));
                        add(new Question("Neck", "C", "Нога",
"Нос", "Шея", "Ухо"));
                        add(new Question("Tooth", "A", "Зуб",
"Нога", "Тело", "Палец"));
                        add(new Question("Finger", "A", "Палец",
"Лицо", "Рука", "Нога"));
                    }
                };
                break;
            case "6":
                questions = new ArrayList<Question>(){
                    {
                        add(new Question("Desert", "D", "Поле",
"Дерево", "Море", "Пустыня"));
```

```
                        add(new Question("City", "B", "Деревня",
"Город", "Страна", "Столица"));
                        add(new Question("Mountain", "A",
"Гора", "Поле", "Мыс", "Карта"));
                        add(new Question("Island", "C", "Холм",
"Море", "Остров", "Река"));
                        add(new Question("River", "D", "Гора",
"Джунгли", "Болото", "Река"));
                        add(new Question("Lake", "A", "Озеро",
"Устье", "Долина", "Город"));
                        add(new Question("Road", "C", "Дом",
"Поле", "Дорога", "Пик"));
                        add(new Question("Field", "B", "Гора",
"Поле", "Море", "Река"));
                        add(new Question("Volcano", "D",
"Город", "Страна", "Холм", "Вулкан"));
                        add(new Question("Village", "A",
"Деревня", "Холм", "Пригорок", "Гора"));
                    }
                };
                break;
            default:
                finish();
        }

        databaseHelper = new DataBase(getApplicationContext());

        loadQuestion();

    }


    @Override
    protected void onRestart(){
        super.onRestart();
        loadQuestion();
```

```
    }


    private void loadQuestion(){
        if(questions.size() > 0) {
            Question q = questions.remove(0);
            lblQuestion.setText(q.getQuestion());
            List<String> answers = q.getAnswers();

            optionA.setText(answers.get(0));
            optionB.setText(answers.get(1));
            optionC.setText(answers.get(2));
            optionD.setText(answers.get(3));

            rightAnswer = q.getRightAnswer();
        } else {
            //Сохранение счета
            final ContentValues newValues = new ContentValues();
            newValues.put(DataBaseContract.DataEntry.T2_ATR2,
login);
            newValues.put(DataBaseContract.DataEntry.T2_ATR3,
score);
            Thread insertThread = new Thread( // создаём новый
поток
                    new Runnable() { // описываем объект
Runnable в конструкторе
                        public void run() {
                            SQLiteDatabase dbw;
                            dbw =
databaseHelper.getWritableDatabase();

dbw.delete(DataBaseContract.DataEntry.TABLE_NAME_2,
DataBaseContract.DataEntry.T2_ATR2+" =" + login, null);

dbw.insert(DataBaseContract.DataEntry.TABLE_NAME_2, null,
newValues);
```

```
                            dbw.close();
                        }
                    }

            );
            insertThread.start();
            try {
                insertThread.join();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            //Отображение счета
            Intent intent = new Intent(this,
ShowScoreActivity.class);
            intent.putExtra("score", score);
            startActivity(intent);
            finish();
        }
    }

    public void loadAnswer(View view) {
        int op = radioGroup.getCheckedRadioButtonId();

        switch (op){
            case R.id.optionA:
                Answer="A";
                break;

            case R.id.optionB:
                Answer="B";
                break;

            case R.id.optionC:
                Answer="C";
                break;
```

```
            case R.id.optionD:
                Answer="D";
                break;

            default:
                return;
        }

        radioGroup.clearCheck();
        startActivity(isRightOrWrong(Answer));

    }

    public Intent isRightOrWrong(String Answer){
        Intent screen;
        if(Answer.equals(rightAnswer)) {
            score += 1;
            screen = new Intent(QNumberActivity.this,
RightActivity.class);

        }else {
            screen = new Intent(QNumberActivity.this,
WrongActivity.class);
        }

        return screen;
    }
}
```

Файл Question.java

```
package com.example.myapplication;

import java.util.ArrayList;
import java.util.List;
```

```
public class Question {

    private String question;
    private List<String> answers = new ArrayList<>();
    private String rightAnswer;

    public Question(String question, String rightAnswer, String
... answers ) {
        this.question = question;
        this.rightAnswer = rightAnswer;
        this.answers.add(answers[0]);
        this.answers.add(answers[1]);
        this.answers.add(answers[2]);
        this.answers.add(answers[3]);
    }

    public String getQuestion() {
        return question;
    }

    public List<String> getAnswers() {
        return answers;
    }

    public String getRightAnswer() {
        return rightAnswer;
    }
}
```

Файл RightActivity.java

```
package com.example.myapplication;

import android.os.Bundle;
import android.view.View;

import androidx.appcompat.app.AppCompatActivity;
```

```
import androidx.constraintlayout.widget.ConstraintLayout;

public class RightActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_right);

        ConstraintLayout root = (ConstraintLayout)
findViewById(R.id.layout);
        root.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }});
    }
}
```

Файл WrongActivity.java

```
package com.example.myapplication;

import android.os.Bundle;
import android.view.View;
import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;

public class WrongActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_wrong);

        ConstraintLayout root = (ConstraintLayout)
findViewById(R.id.layout);
```

```
        root.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }});
    }
}
```

Файл ShowScoreActivity.java

```
package com.example.myapplication;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;

public class ShowScoreActivity extends AppCompatActivity {
    TextView TxtScore;
    TextView TxtStatus;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_show_score);

        TxtScore = findViewById(R.id.txtscore);
        TxtStatus = findViewById(R.id.txtStatus);
        Intent intent = getIntent();
        String scores =
String.valueOf(intent.getIntExtra("score", 0));

        TxtScore.setText(scores);
        TxtStatus.setText(setStatus(scores));
```

```
        ConstraintLayout root = (ConstraintLayout)
findViewById(R.id.layout);
        root.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }});
    }

    private String setStatus(String scores){
        int score = Integer.parseInt(scores);

        if(score >= 8){
            return "Ты очень умный!";
        }

        if (score >= 5){
            return "Поздравляем!";
        }

        return "Вам нужно больше учиться...";

    }


    public void goToHome(View v){
        Intent home = new Intent(ShowScoreActivity.this,
MainActivity.class);
        startActivity(home);
        finish();
    }

}
```

Приложение В

(Обязательное)

Программное обеспечение для интерактивного обучения

Презентация

Программное обеспечение для интерактивного обучения

Выполнил: студент группы 18ВВ1
Коновалова Д.А.
Руководитель: Бикташев Р.А.

Актуальность

- Знание иностранных языков трудно переоценить, так как в современном обществе мы ежедневно встречаемся с иностранными языками, что значительно усложняет жизнь. С английским мы сталкиваемся на каждом шагу – в интернете, кино и музыке, книгах и новостях, в описаниях товаров и услуг.
- Без регулярного повторения материала человек быстро забывает информацию. Если запоминать иностранные слова путем механического зазубривания, то это не даст больших результатов в будущем и затратит большое количество времени на повторное изучение материала.
- Решением этой проблемы может являться ежедневная практика небольшого количества слов, что улучшит словарный запас человека. Интервальное повторение – это техника обучения, при которой повторение материала совершается через определенные промежутки времени, таким образом, обеспечивая долговременное запоминание.

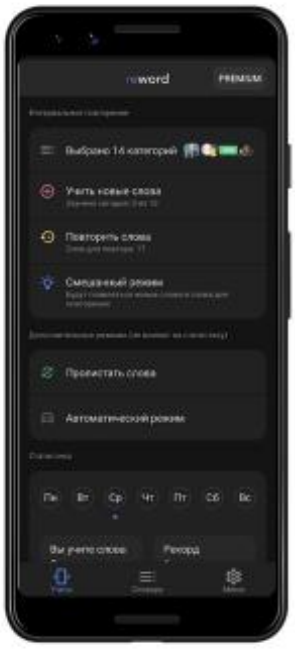
Цель и задачи

Цель ВКР – разработка приложения для изучения английских слов.

Для выполнения поставленной цели требуется реализовать ряд задач:

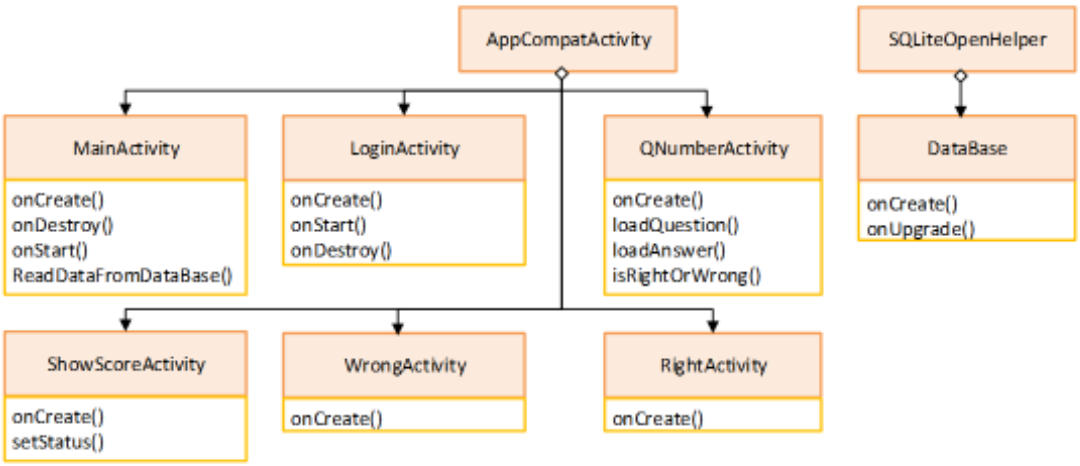
- Авторизация пользователя;
- Выбор упражнения для изучения слов;
- Изучения слов выбранной группы посредством выбора ответа на заданный вопрос;
- Отображение реакции на выбор ответа пользователя;
- Вывод итогового счета;
- Отображение последнего результата;
- Простой и интуитивно понятный для пользователя графический интерфейс.

Аналоги

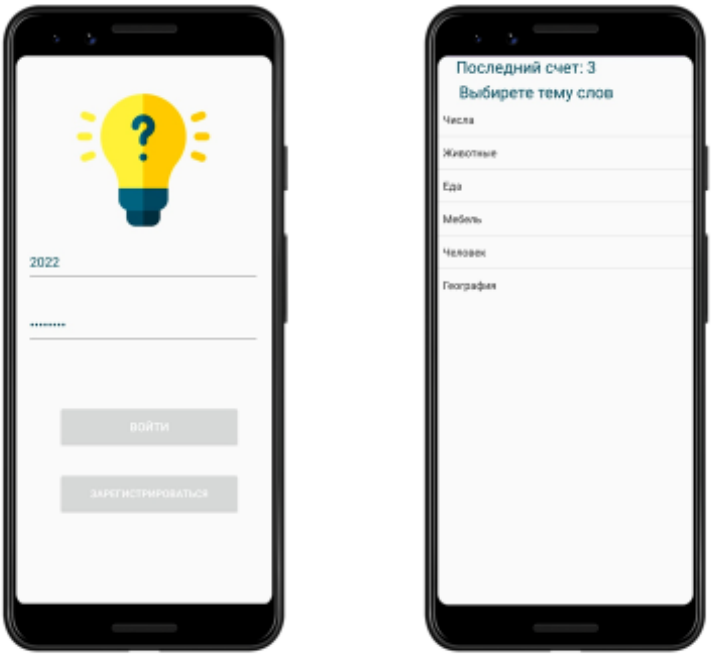


UML- диаграмма прецедентов

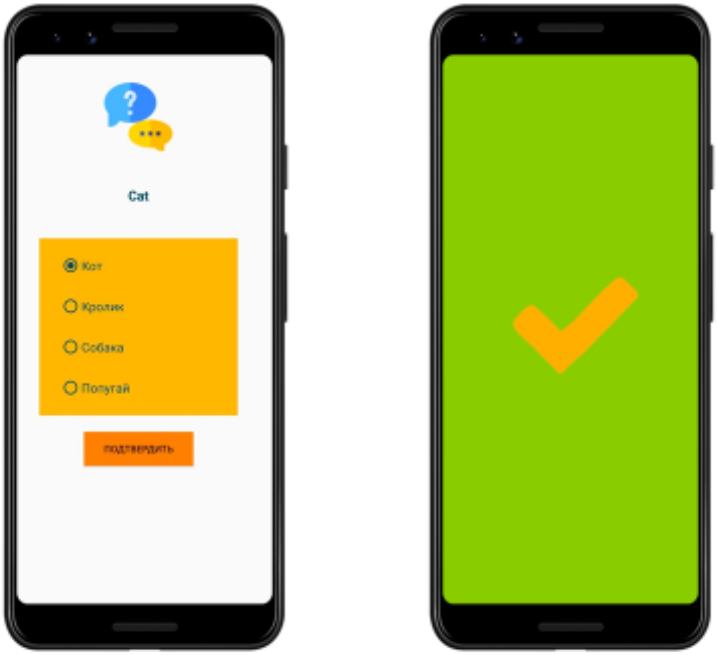
UML-диаграмма классов



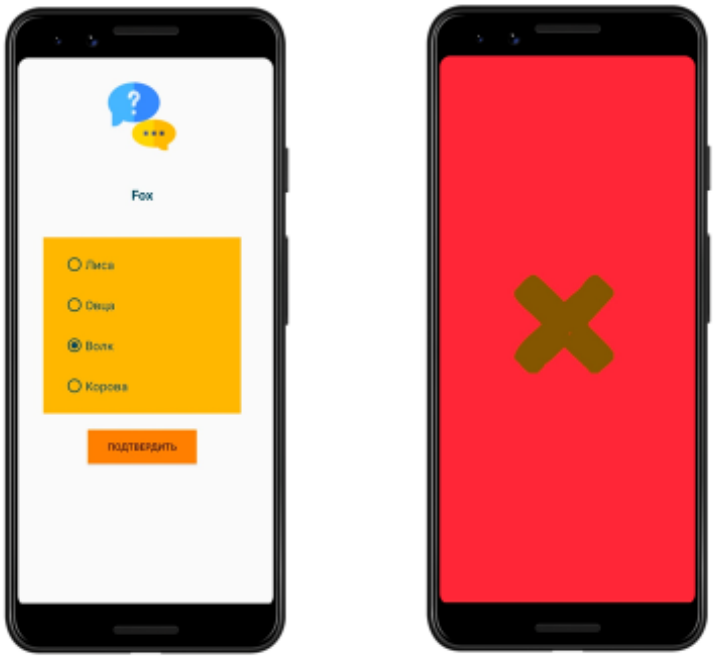
Интерфейс авторизации и выбора темы



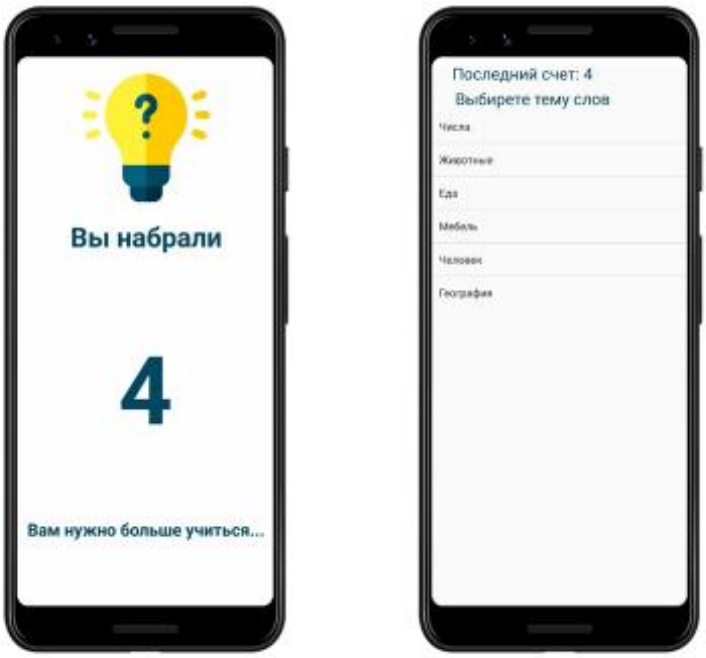
Интерфейс правильного ответа на вопрос



Интерфейс не верного ответа на вопрос



Итоговый счет и его отображение в окне выбора темы



Заключение

- Приложение обладает простым и понятным интерфейсом
- Приложение позволяет пользователю изучать английские слова по темам и отслеживать статистику прохождения тестов
- Приложение полностью соответствует поставленной цели

Спасибо за внимание