МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

КУРСОВАЯ РАБОТА

по дисциплине «Разработка приложений для мобильных платформ»

Тема: IT interview trainer

Студент гр. 7303 _____ Петров С.А.
Студент гр. 7303 _____ Юсковец А.В.
Студент гр. 7303 _____ Швец А.А.
Преподаватель _____ Заславский М.М.

Санкт-Петербург
2020

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студенты

Швец А.А.

Петров С.А.

Юсковец А.А.

Группа 7303

Тема курсовой работы: Разработка мобильного приложения для подготовки к интервью по программированию.

Исходные данные:

Необходимо реализовать приложение IT interview trainer для Android платформы на языке Kotlin.

Содержание пояснительной записки:

«Содержание»

«Введение»

«Сценарии использования»

«Пользовательский интерфейс»

«Модель данных»

«Разработанное приложение»

«Последовательность действий для осуществления сценариев использования»

«Выводы»

«Заключение»

Предполагаемый объем пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания: 00.00.2000

Дата сдачи реферата: 00.00.2000

Дата защиты реферата: 00.00.2000

Студент гр. 7303	_____	Петров С.А.
Студент гр. 7303	_____	Юсковец А.В.
Студент гр. 7303	_____	Швец А.А.
Преподаватель	_____	Заславский М.М.

АННОТАЦИЯ

В данной курсовой работе представляется приложение для подготовки к интервью по программированию, написанное на языке программирования Kotlin под платформу Android. В работе прорабатываются сценарии использовании, представляется спроектированная модель данных. Также представляется разработанное решение: мобильное приложение, которое содержит набор тестов по различным языкам программирования (с оценкой прохождения каждого теста), и позволяет сохранять историю пройденных тестов. Приводятся скриншоты работы приложения, описываются использованные технологии. Приводится исследование разработанного решения и выводы по дальнейшему улучшению.

Исходный код решения: https://github.com/moevm/adfmp1h21-interview/

SUMMARY

This course paper presents an application for preparing for an interview on programming, written in the Kotlin programming language for the Android platform. In this paper, the usage scenarios are worked out, and the designed data model is presented. The developed solution is also presented: Android mobile application, which contains a set of tests on different programming languages (with scoring each test), and also keeps history of taken tests. Screenshots of the application are provided, and the technologies used are described. A study of the developed solution and conclusions for further improvement are presented.

The source code of the solution: https://github.com/moevm/adfmp1h21-interview/

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность решаемой проблемы

Изучение различных языков программирования является неотъемлемой частью подготовки к работе в IT-индустрии, связанной с написанием программного кода. Именно для достижения их целей и получения знаний, которые могут понадобиться как во время собеседования, так и в процессе работы, может помочь специальное приложения, тренирующие полезные навыки и умения.

Постановка задачи

Разработка мобильного приложения для подготовки к интервью по программированию.

Предлагаемое решение

Мобильное приложение, которое содержит набор тестов по различным языкам программирования. Тесты разделяются по категориям, а также уровням сложности. Приложение дает возможность как пройти заранее собранный тест, так и сгенерировать случайный набор вопросов исходя из выбранных категорий и уровня сложности. Приложение дает определенное время на прохождение каждого теста, а также сохраняет историю пройденных тестов.

Почему решение необходимо реализовывать как мобильное приложение

Мобильное приложение в определенных случаях выступает наиболее доступным способом. Например, пользователь едет в метро на собеседование, и хочет проверить свои знания напоследок. В данном случае, даже имея ноутбук, пользоваться им в метро неудобно, а вот смартфон в таких условиях наиболее доступен, причем пользоваться им можно даже одной рукой.

1. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

1.1. Сценарии использования

Ниже приведены сценарии использования мобильного приложения.

Выбор быстрого теста

1. Нажать на кнопку "Take test".

В появившемся окне настроек создаваемого теста:

2. Выбрать ЯП теста.
3. Выбрать количество вопросов теста.
4. Выбрать время, отведенное на прохождение теста.
5. Выбрать уровень сложности теста.
6. Нажать на кнопку "Start".

Выбор теста по категории + уровню

1. Нажать на кнопку "Categories".
2. В появившемся окне выбрать желаемую категорию тестирования.

В следующем окне, соответствующем выбранной ранее категории:

3. Выбрать вкладку с требуемым уровнем сложности теста.
4. Выбрать тест.

Прохождение теста

1. Выбрать варианта ответа.
2. Нажать на кнопку "Submit".
3. Просмотреть результат и нажать на кнопку "Next" (переход к п.1).

После окончания прохождения теста на экране итогов доступно две опции:

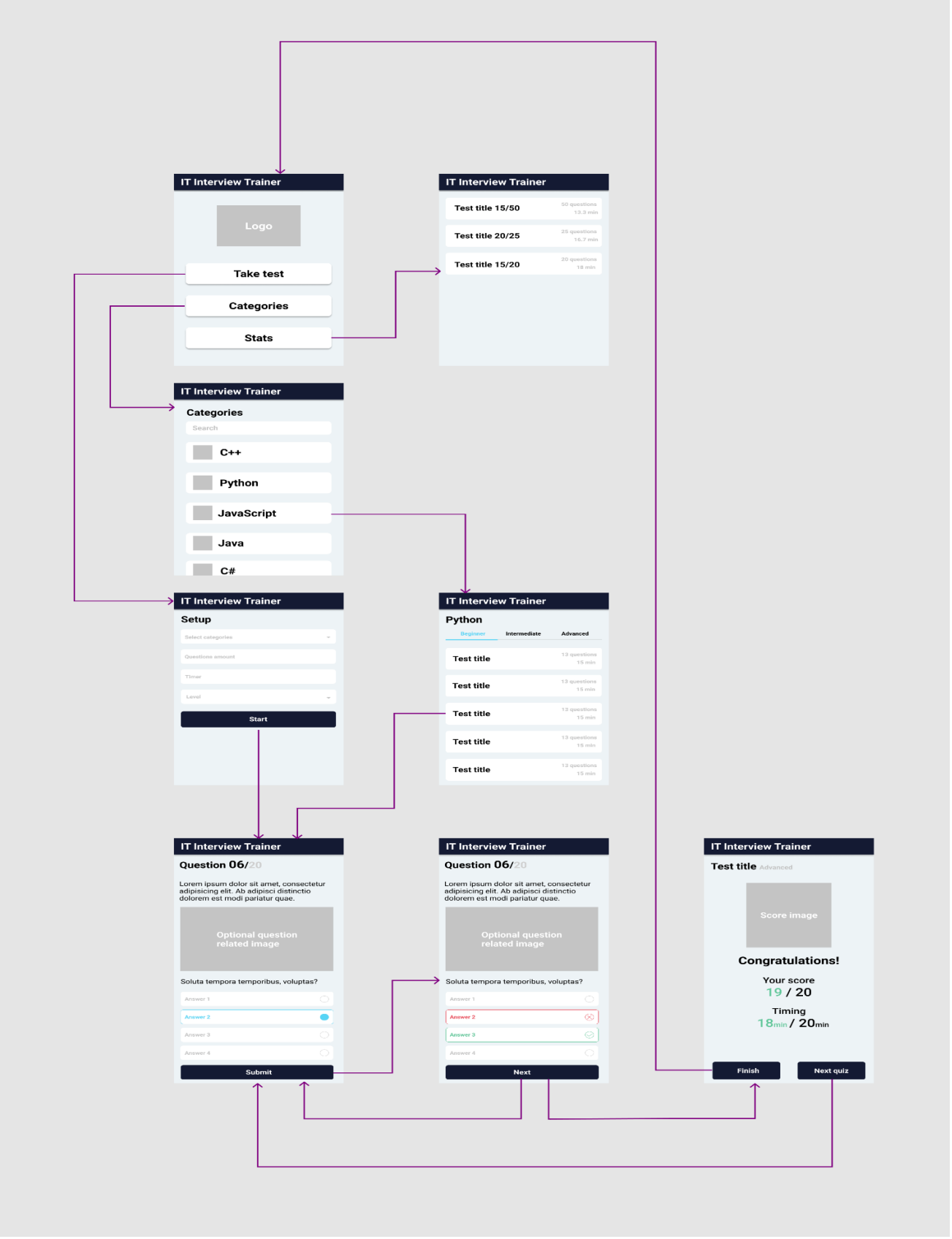
1. Нажать кнопку "Finish" и закончить тест (перейти на стартовый экран).
2. Нажать кнопку "Next quiz" и перейти к следующему по тесту.

Просмотр истории прохождения тестов

1. Нажать на кнопку "Stats".

2. ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

2.1 Макет интерфейса



2.2. Целевые устройства

2.2.1. Тип устройств

Целевыми устройствами разрабатываемого приложения являются смартфоны на ОС Android, минимальная версия - 6.0 (Marshmallow).

2.2.2. Аппаратная составляющая

Эмулятор:

Экран:

- hw.lcd.width: 480
- hw.lcd.height: 800
- hw.lcd.density: 240

Прочее:

- CPU/ABI: Google APIs Intel Atom (x86_64)
- Path: /home/vood/.android/avd/Pixel_XL_API_28.avd
- Target: google_apis [Google APIs] (API level 28)
- Skin: nexus_one
- SD Card: 512 MB
- fastboot.chosenSnapshotFile:
- runtime.network.speed: full
- hw.accelerometer: yes
- hw.initialOrientation: Portrait
- image.androidVersion.api: 28
- tag.id: google_apis
- hw.mainKeys: yes
- hw.camera.front: None
- hw.gpu.mode: auto
- hw.ramSize: 512
- PlayStore.enabled: false
- fastboot.forceColdBoot: no
- hw.cpu.ncore: 2
- hw.keyboard: yes
- hw.sensors.proximity: yes
- hw.dPad: no
- vm.heapSize: 48
- skin.dynamic: yes
- hw.device.manufacturer: Google
- hw.gps: yes
- hw.audioInput: yes

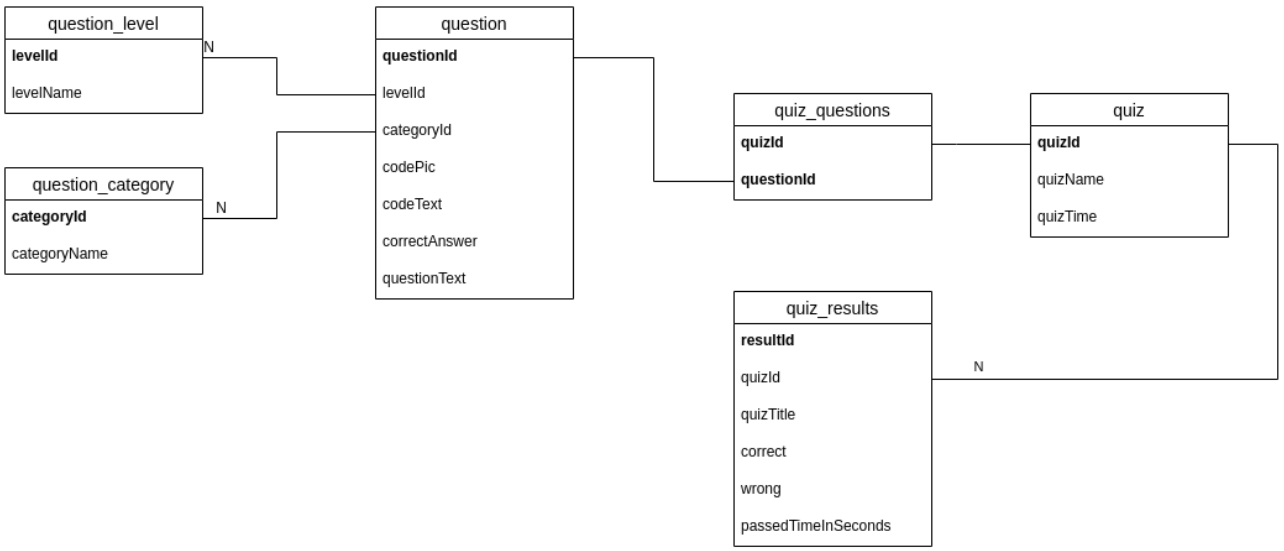
- image.sysdir.1: system-images/android-28/google_apis/x86_64/
- showDeviceFrame: yes
- hw.camera.back: virtualscene
- hw.arc: false
- hw.device.hash2: MD5:ef39e456bf2cab397201c2ac251f35fc
- fastboot.forceChosenSnapshotBoot: no
- fastboot.forceFastBoot: yes
- hw.trackBall: yes
- hw.battery: yes
- hw.sdCard: yes
- tag.display: Google APIs
- runtime.network.latency: none
- disk.dataPartition.size: 800M
- hw.sensors.orientation: no
- avd.ini.encoding: UTF-8
- hw.gpu.enabled: yes

3. МОДЕЛЬ ДАННЫХ

3.1 Хранимые данные

- вопросы
- уровни сложности
- категории вопросов,
- тесты (наборы вопросов)
- история пройденных тестов

3.2. Графическое представление



4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

4.1 Краткое описание

Мобильное приложение, содержащее в себе набор подготовленных вопросов и тестов по различным категориям (языкам программирования). Имеется 3 категории: Python, C++, JavaScript, и 3 уровня сложности - easy, medium, hard. Приложение позволяет пройти как предустановленный тест, так и сгенерировать случайный на основании выбранных категорий, уровня сложности, и с выбранным временем на прохождение. Приложение также сохраняет историю пройденных тестов, которая доступна в любой момент из главного экрана приложения.

4.2 Схема архитектуры

Согласно сценариям использования приложение было разделено на экраны:

- главное меню
- генерация случайного теста
- просмотр категорий
- выбор теста в категории
- прохождение теста
- история пройденных тестов

Логически данные экраны можно разделить на две категории - настройка/выбор теста, и прохождение теста. В соответствии с данным разделением приложение состоит из 2-х Activity - MainActivity и QuizActivity.

MainActivity состоит из следующих фрагментов:

- MainFragment (главный экран приложения) - состоит из трех кнопок и картинки (лого), по кнопкам осуществляется переход в другие экраны в соответствии с макетом и сценариям использования
- StatisticsFragment (история тестов) - состоит из списка элементов, отражающих пройденные тесты. Каждый элемент содержит название теста,

количество правильных ответов, количество вопросов, затраченное на прохождение время. Список поддерживает вертикальное пролистывание, реализован через RecyclerView.

- QuizSetupFragment (настройка случайного теста) - представляет из себя форму, состоит из поля выбора категории (по клику всплывающий диалог с категориями, выбор по чекбоксам, поддерживается выбор нескольких категорий), поля выбор сложности (аналогично выбору категории, но можно выбрать только 1 уровень сложности), выбора количества вопросов (слайдер, верхняя граница адаптивно изменяется в зависимости от выбранной сложности и категорий), выбора времени (аналогично слайдер) и кнопки для запуска теста.
- CategoryChoiceFragment (просмотр и выбор категории) - состоит из поля для быстрого поиска категории (фильтрация по введенной строке, независимо от регистра) и списка категорий. Список поддерживает вертикальное пролистывание, реализован через RecyclerView. Элемент (категория) кликабелен, по клику осуществляется переход в экран выбор теста в соответствующей категории.
- QuizChoiceFragment (выбор теста в категории) - состоит из 3-х вкладок (tabs), соответствующих уровням сложности, каждая вкладка представляет из себя список тестов (каждый элемент содержит название теста, отведенное время на прохождение, количество вопросов).

QuizActivity состоит из следующих элементов:

- поле с названием теста
- поле с оставшимся временем
- Номер вопроса в формате “<номер вопроса>/<количество вопросов>”
- Текст вопроса
- Опционально: картинка с кодом, если вопрос касается кода
- Варианты ответа (через RadioButton)
- Кнопка перехода к следующему вопросу с текстом “submit”

При нажатии на кнопку перехода к следующему вопросу:

- Если не выбран ни один ответ, высвечивается соответствующее сообщение
- Если выбран ответ, кнопка меняет текст с “submit” на “next”, правильный вариант ответа окрашивается в зеленый цвет, а если выбранный вариант неправильный, то он окрашивается в красный цвет. Повторное нажатие инициирует переход к следующему вопросу
- Если пройденный вариант последний, то высвечивается соответствующий диалог с количеством правильных ответов, результаты сохраняются в базу и осуществляется переход
  - на главный экран, если тест был сгенерирован случайно
  - на экран тестов категории, откуда был запущен тест, если тест не сгенерирован случайно

База данных приложения инициализируется при первом запуске приложения.

4.3 Использованные технологии

- Volley [4] - библиотека для web-запросов, используется для получения картинок с кодом для определенных вопросов, поскольку картинки было решено хранить на хостинге, а не локально, для снижения размера приложения
- Room [5] - библиотека, дающая уровень абстракции над SQLite, используется для работы с локальной базой данных приложения.
- Gson [6] - библиотека для сериализации/десериализации объектов, используется для чтения подготовленных вопросов из json файла при инициализации базы данных.

4.4 Стратегия для обеспечения кроссплатформенности приложения

Приложение разрабатывалось разрабатывалось на языке Kotlin, в котором имеются механизмы для кроссплатформенной разработки (Kotlin Multiplatform Mobile).

5. ПОСЛЕДОВАТЕЛЬНОСТЬ ДЕЙСТВИЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ СЦЕНАРИЕВ ИСПОЛЬЗОВАНИЯ

5.1 Измерение последовательности действий

Выбор быстрого теста

Согласно сценарию использования необходимо выполнить следующие действия:

1. Нажать на кнопку "Take test" (1 клик).

В появившемся окне настроек создаваемого теста:

2. Выбрать ЯП теста (2-4 клика). Выбрать количество вопросов теста (1 слайдер).
3. Выбрать время, отведенное на прохождение теста (1 слайдер).
4. Выбрать уровень сложности теста (2 клика).
5. Нажать на кнопку "Start" (1 клик).

Итого данный сценарий использования требует 8-10 действий пользователя.

Выбор теста по категории + уровню

1. Нажать на кнопку "Categories" (1 клик).
2. В появившемся окне выбрать желаемую категорию тестирования (1 клик).

В следующем окне, соответствующем выбранной ранее категории:

3. Выбрать вкладку с требуемым уровнем сложности теста (1 клик).
4. Выбрать тест (1 клик).

Таким образом, данный сценарий использования требует 4 действия пользователя.

Прохождение теста

Данный сценарий использования зависит от каждого конкретного теста, так как количество вопросов в тесте для разных тестов будет различным, поэтому можно посчитать количество действий пользователя на одной итерации (вопросе) теста.

1. Выбрать варианта ответа (1 клик).
2. Нажать на кнопку "Submit" (1 клик).
3. Просмотреть результат и нажать на кнопку "Next" (переход к п.1, 1 клик).

Таким образом, данный сценарий использования требует 3 действия пользователя на каждом вопросе теста.

Просмотр истории прохождения тестов

1. Нажать на кнопку "Stats" (1 клик).

Таким образом, просмотр истории прохождения тестов доступен пользователю в 1 клик.

Если проанализировать все возможные действия пользователя, необходимые для выполнения вышеуказанных сценариев использования, то на их основе можно составить диаграмму, отображающую общую статистику взаимодействий пользователя и приложения. Данная диаграмма приведена на рис. 1.

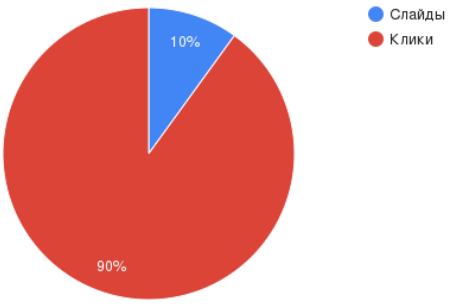


Рисунок 1 - статистика взаимодействий пользователя и приложения

5.2 Пути для сокращения последовательности

Для сокращения последовательности действий пользователя в соответствующих сценариях использования можно избавиться от некоторых параметров, которые пользователю необходимо задавать самому. Например, можно внутри приложения самостоятельно (программно) вычислять время, необходимое на решения составленного приложением теста по формуле

$$T = E * T_e + M * T_m + H * T_h,$$

где

T_e - время, выделенное на вопрос легкой сложности (easy);

T_m - время, выделенное на вопрос средней сложности (medium);

T_h - время, выделенное на вопрос высокой сложности (hard);

E, M, H - количество вопросов соответствующей сложности;

T - общее время опроса.

Также можно избавиться от отдельных окон, возникающий при выборе желаемого уровня сложности теста (см. первый сценарий использования). В данном случае при выборе сложности открывается отдельное окно и после выбора пользователем сложности оно закрывается.

6. ВЫВОДЫ

Достигнутые результаты

В ходе выполнения поставленной задачи было создано приложение позволяющее генерировать тесты по различным ЯП на основе предпочтений пользователя: количества вопросов теста, его длительности, а также сложности вопросов.

Недостатки и пути улучшения полученного решения

В текущем состоянии приложение не поддерживает просмотр подробной информации о пройденных тестах (текст вопросов, выбранные ответы).

База данных приложения является в небольшой степени избыточной - поле quizTitle таблицы quiz_results является дубликатом поля quizName таблицы quiz.

Как было выяснено на основе измерения последовательности действий пользователя, интерфейс приложения не является оптимальным: количество действий все еще можно свести к минимуму.

Будущее развитие решения

- Добавление подробной информации о пройденных тестах
- Оптимизация базы данных
- Добавление возможности создать тест (будет полезно в ситуации, когда пользователь после собеседования запомнит некоторые вопросы и захочет потренироваться на них позднее), с возможностью ввода программного кода к вопросу при необходимости

ЗАКЛЮЧЕНИЕ

В ходе данной курсовой работы было разработано приложение IT interview trainer. В ходе разработки были выделены требования к решению, проработаны сценарии использования и макеты пользовательского интерфейса, спроектирована модель данных. Было разработано мобильное приложение для подготовки к вопросам по языкам программирования, содержащее набор вопросов и тестов по различным категориям. Было проведено исследование разработанного решения, из которого был сделан вывод о существовании возможности для развития решения: добавление новых возможностей, оптимизация базы данных.

СПИСОК ЛИТЕРАТУРЫ

1. Android Developer Guides: [сайт]. URL: https://developer.android.com/guide.
2. Официальный документация Kotlin: [сайт]. URL: https://kotlinlang.org/docs/home.html.
3. Официальная документация Material Components для Android: [сайт]. URL: https://material.io/components?platform=android.
4. Volley overview: [сайт]. URL: https://developer.android.com/training/volley.
5. Save data in a local database using Room: [сайт]. URL: https://developer.android.com/training/data-storage/room.
6. Gson User Guide: [сайт]. URL: https://github.com/google/gson/blob/master/UserGuide.md.

ПРИЛОЖЕНИЕ А.
СКРИНШОТЫ РАЗРАБОТАННОГО ПРИЛОЖЕНИЯ

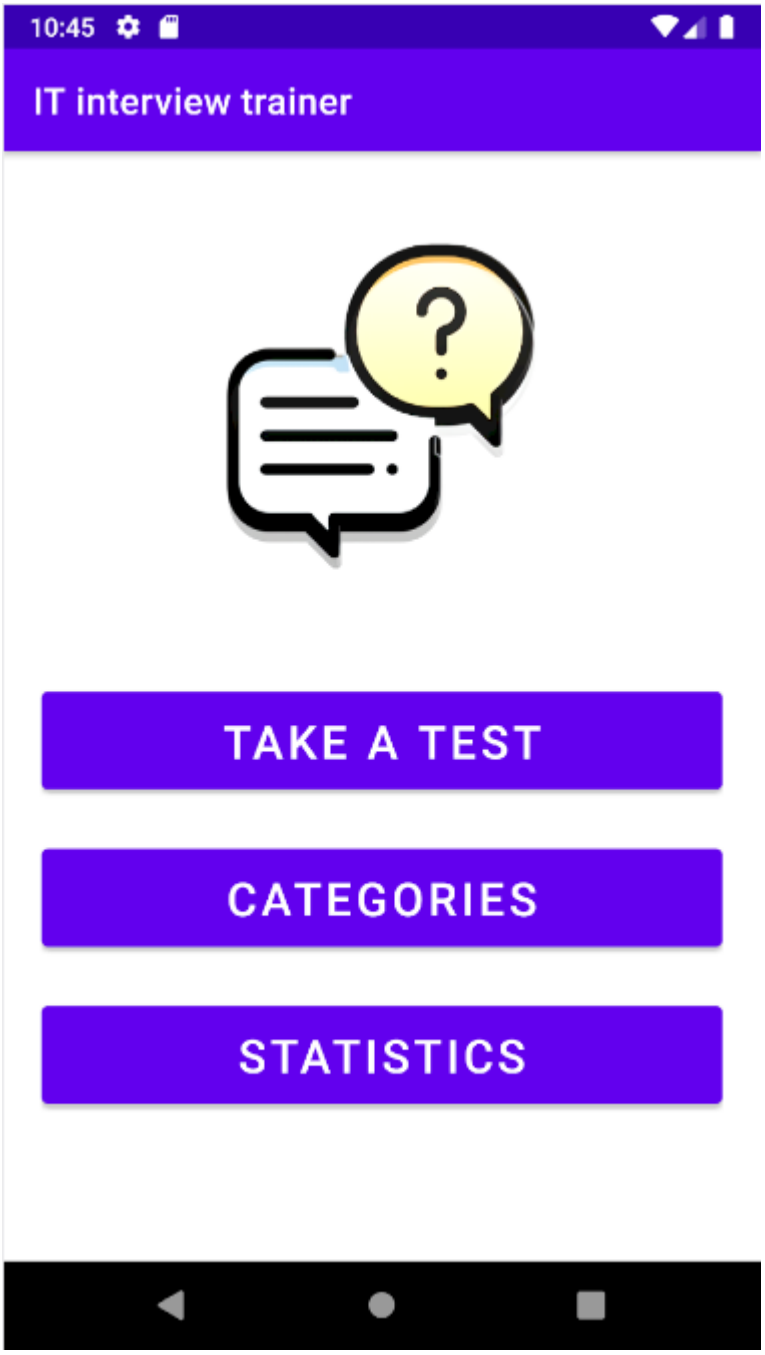


Рисунок 1 – стартовый экран приложения

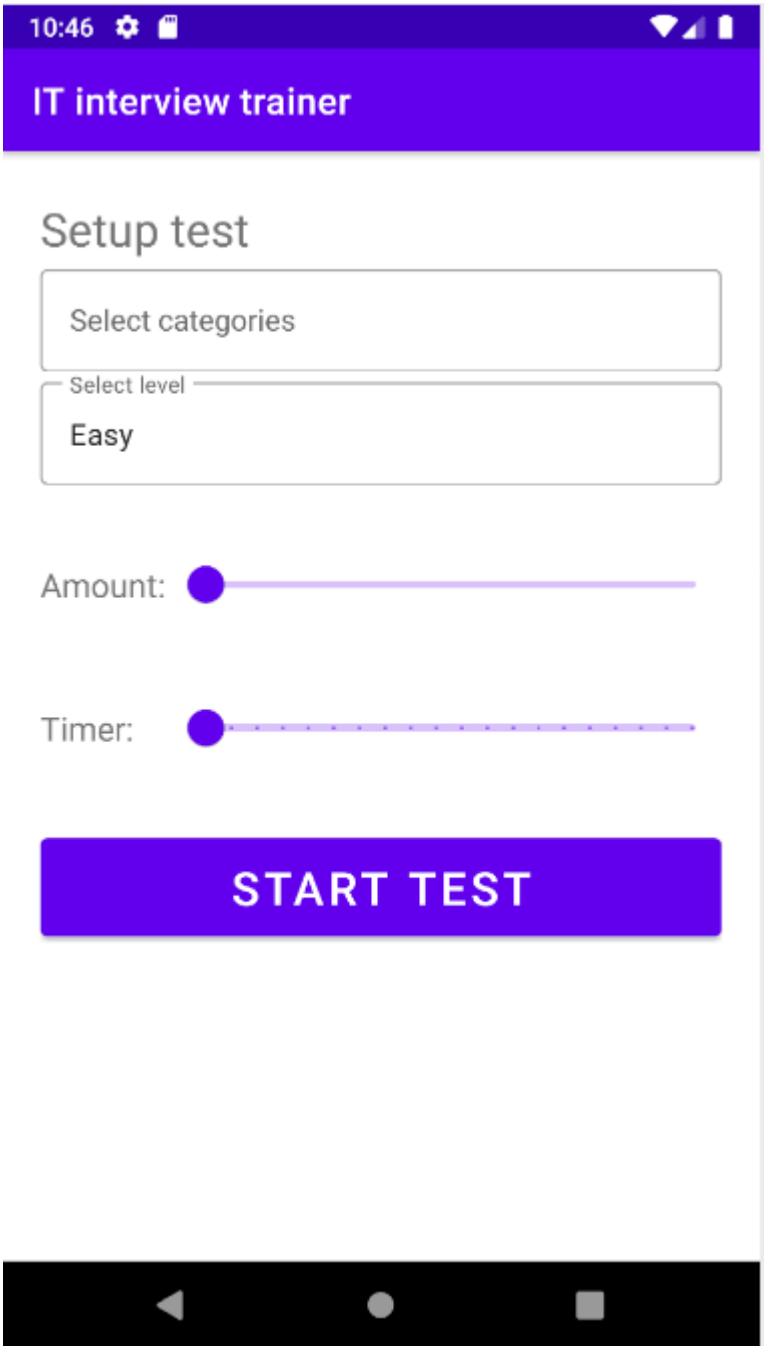


Рисунок 2 – Экран настройки теста, общий вид (состояние после нажатие кнопки “Take test”)

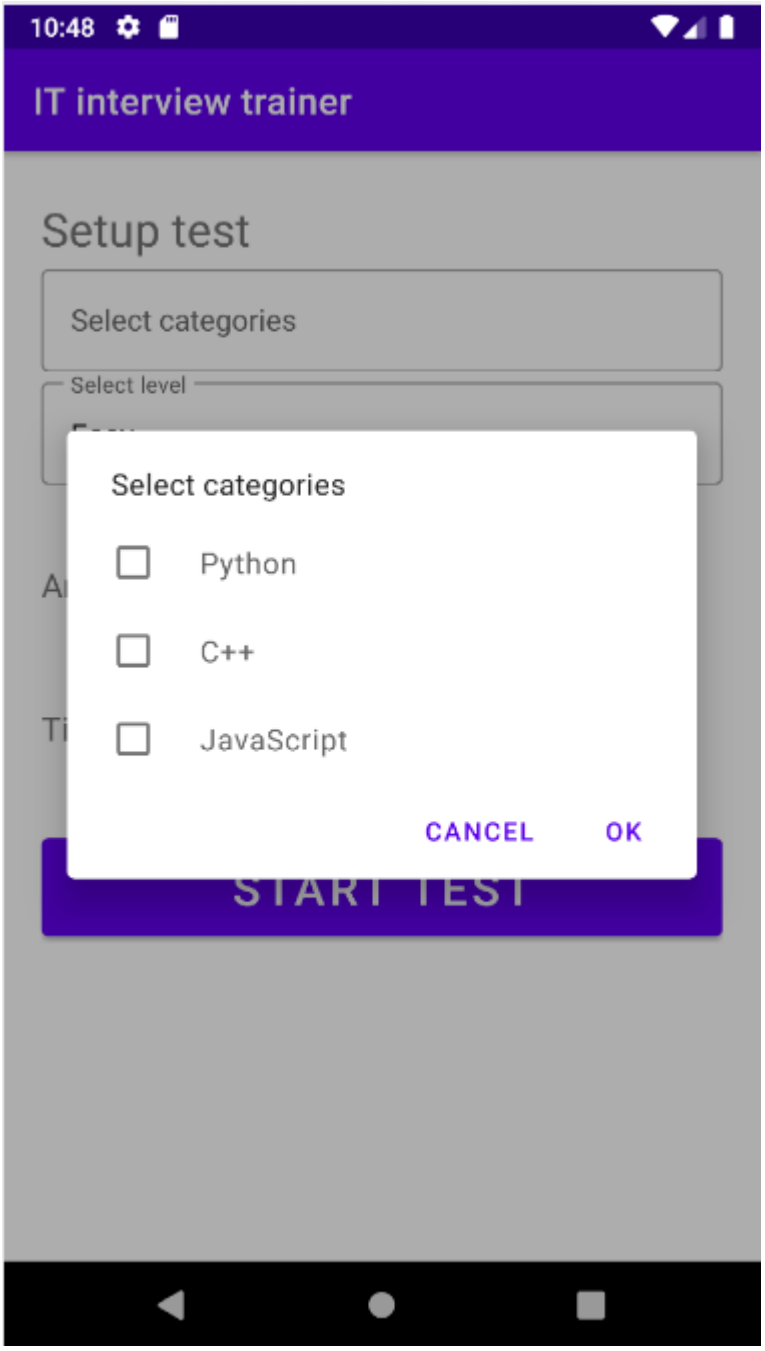


Рисунок 3 – Экран настройки теста, выбор категории

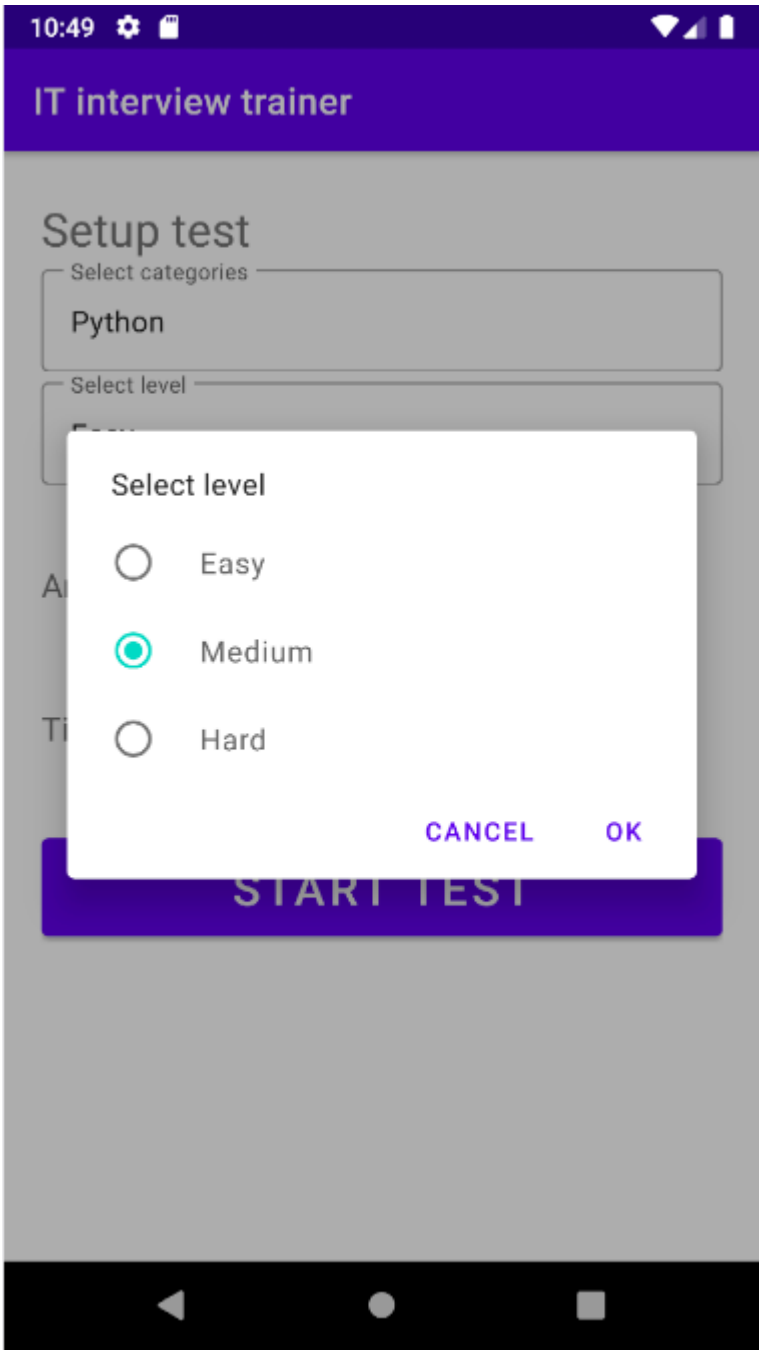


Рисунок 4 – Экран настройки теста, выбор сложности

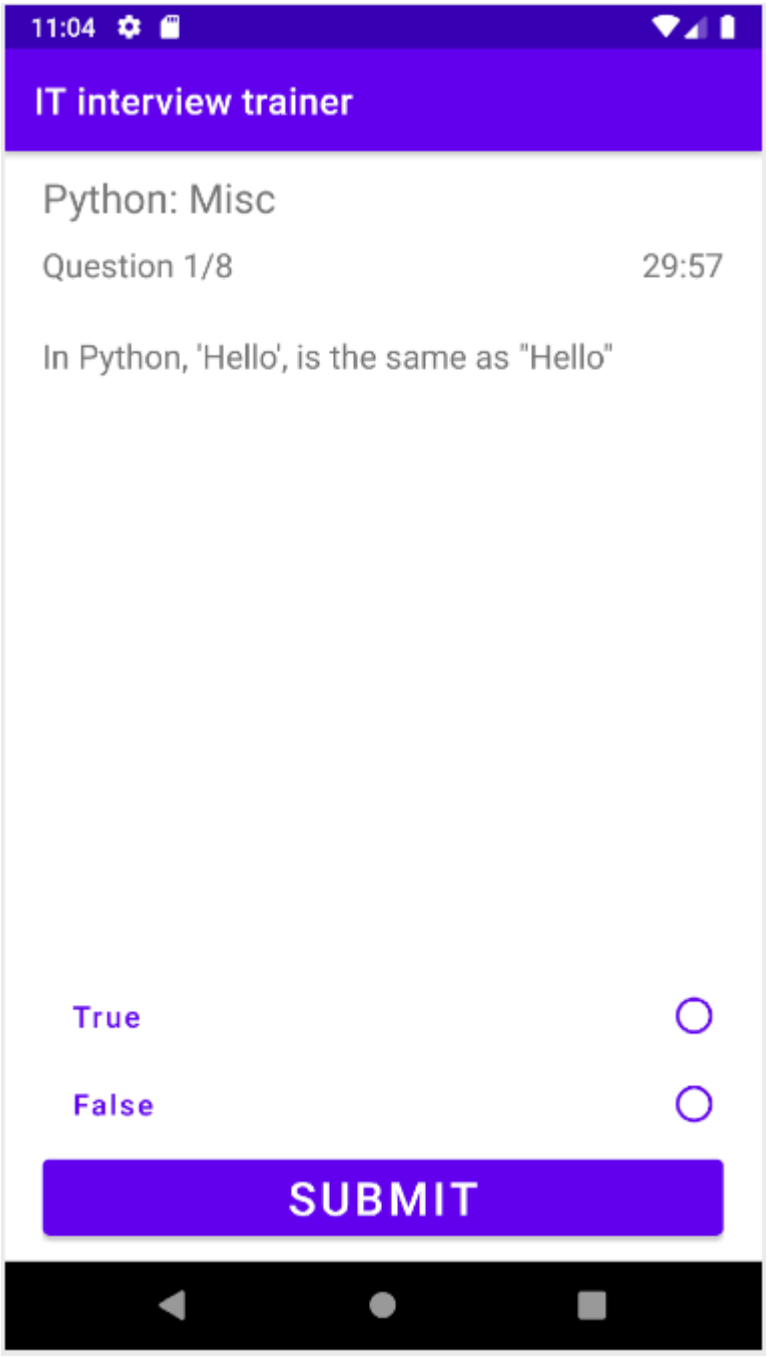


Рисунок 5 – Процесс решения теста, до выбора ответа

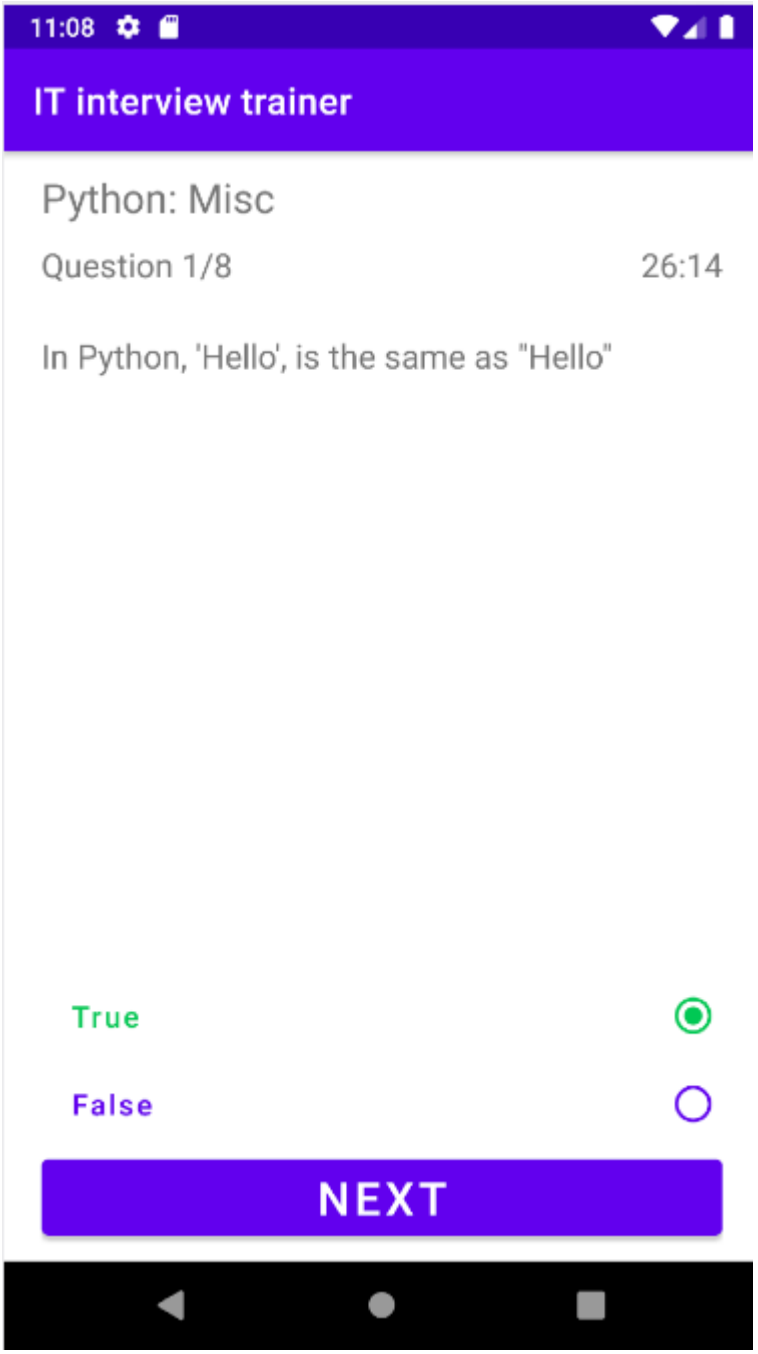


Рисунок 6 – Процесс решения теста, верный ответ

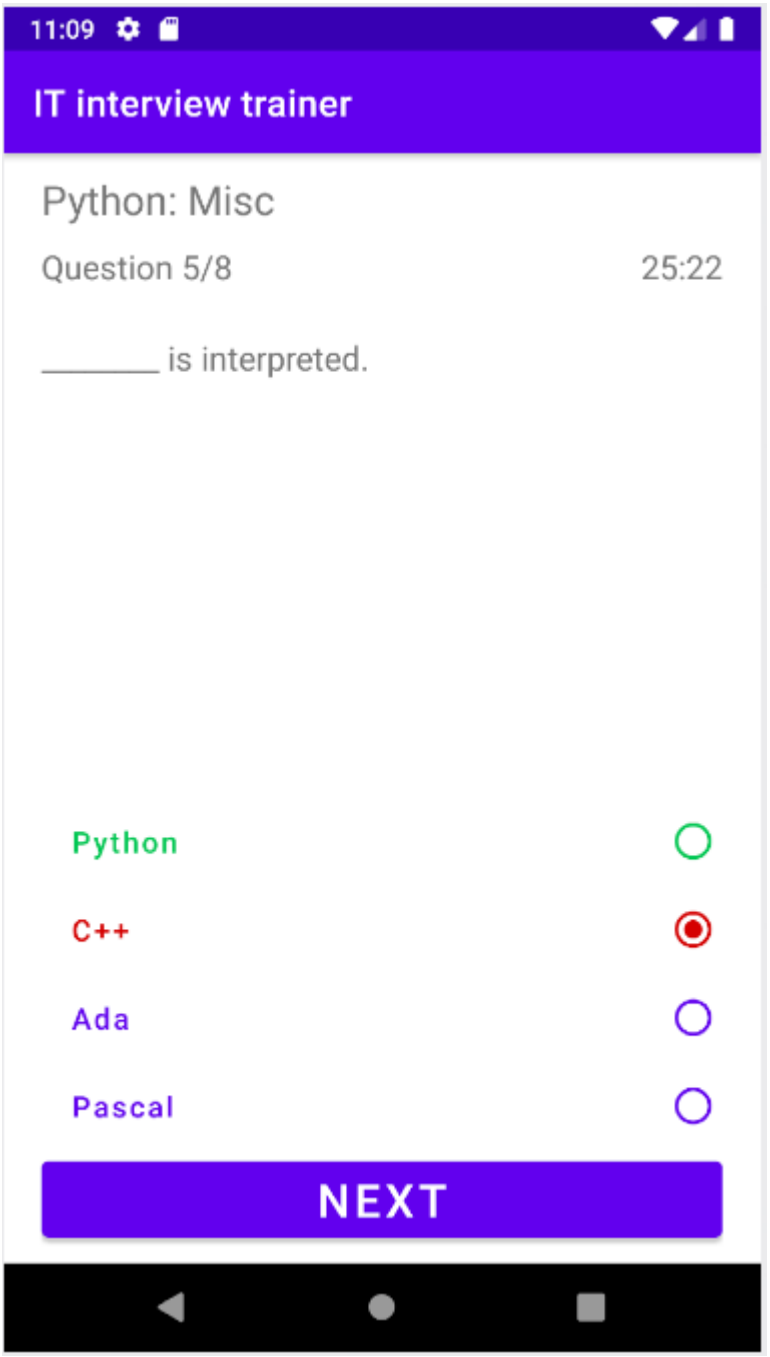


Рисунок 7 – Процесс решения теста, неверный ответ

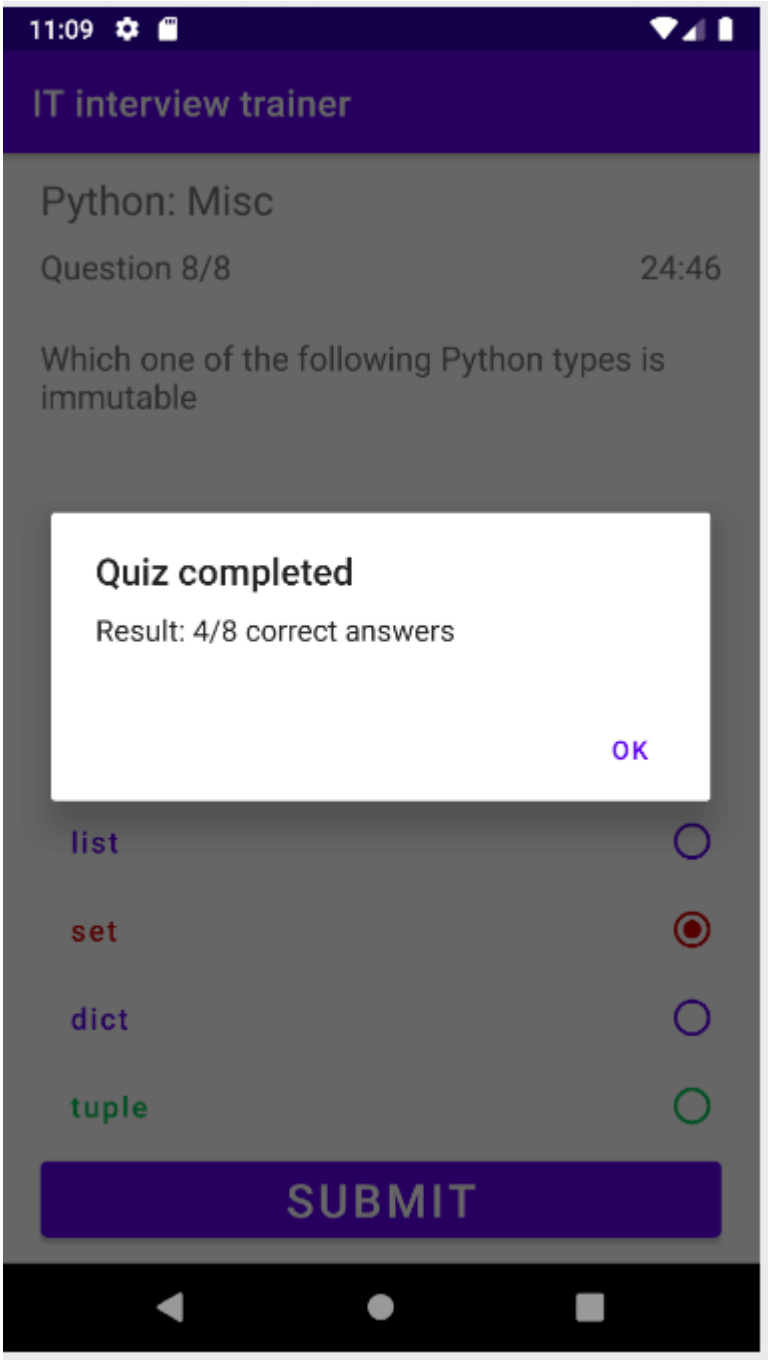


Рисунок 8 – Результаты теста

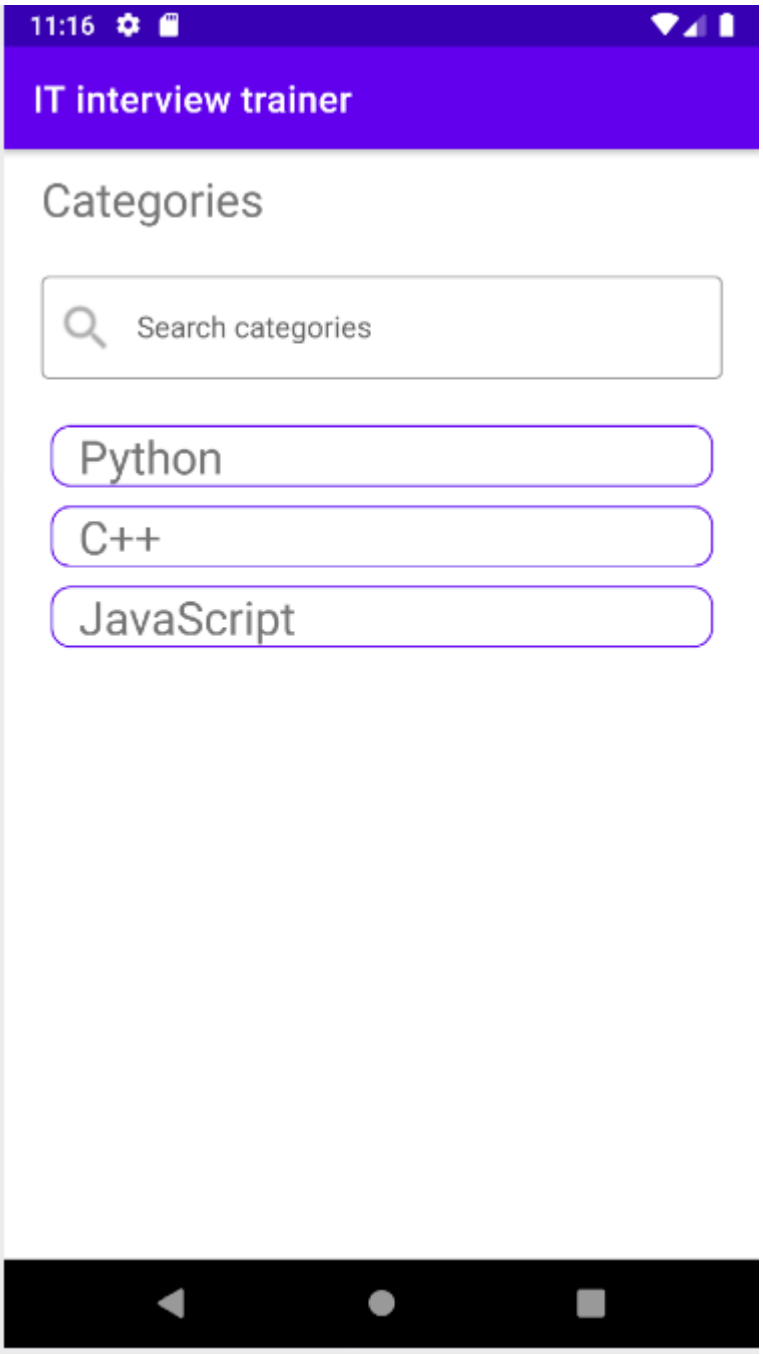


Рисунок 9 – Раздел категории (после перехода из главного экрана)

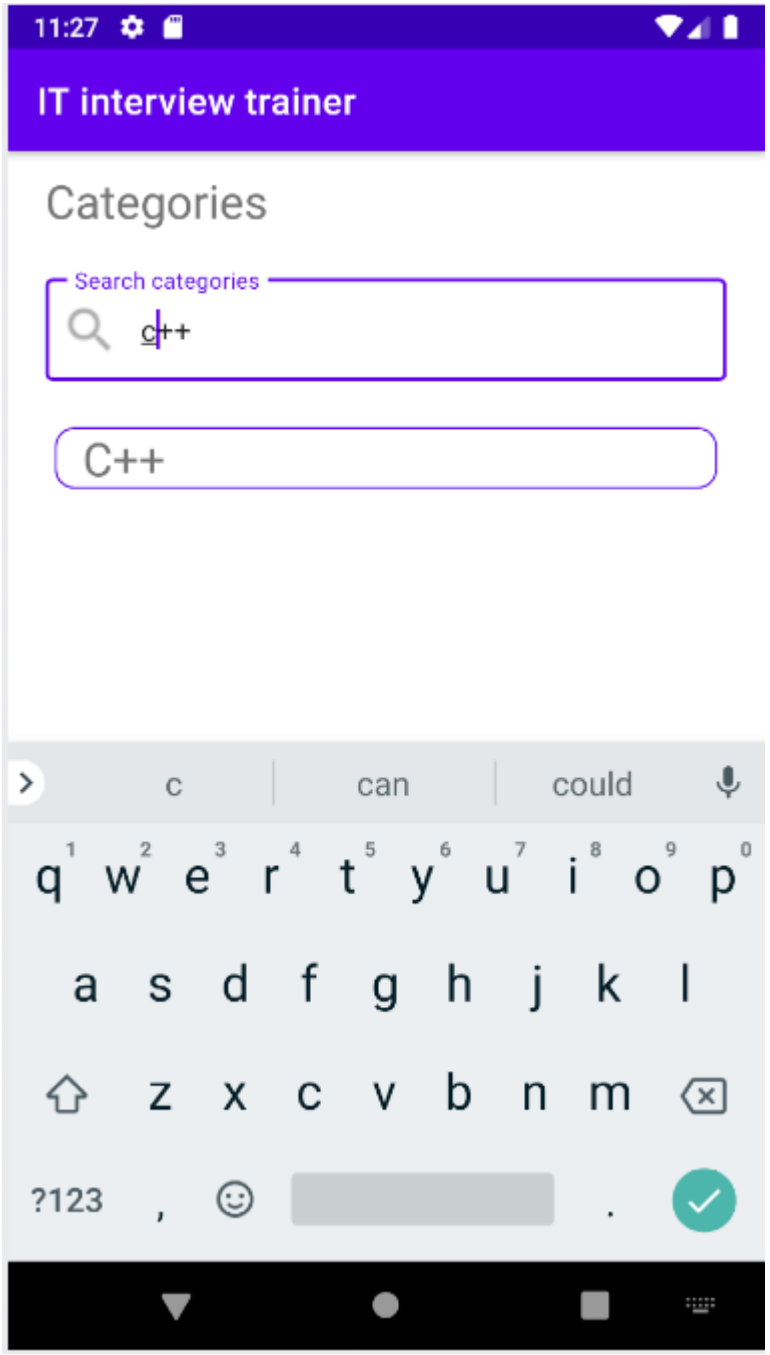


Рисунок 10 – Поиск категорий по названию

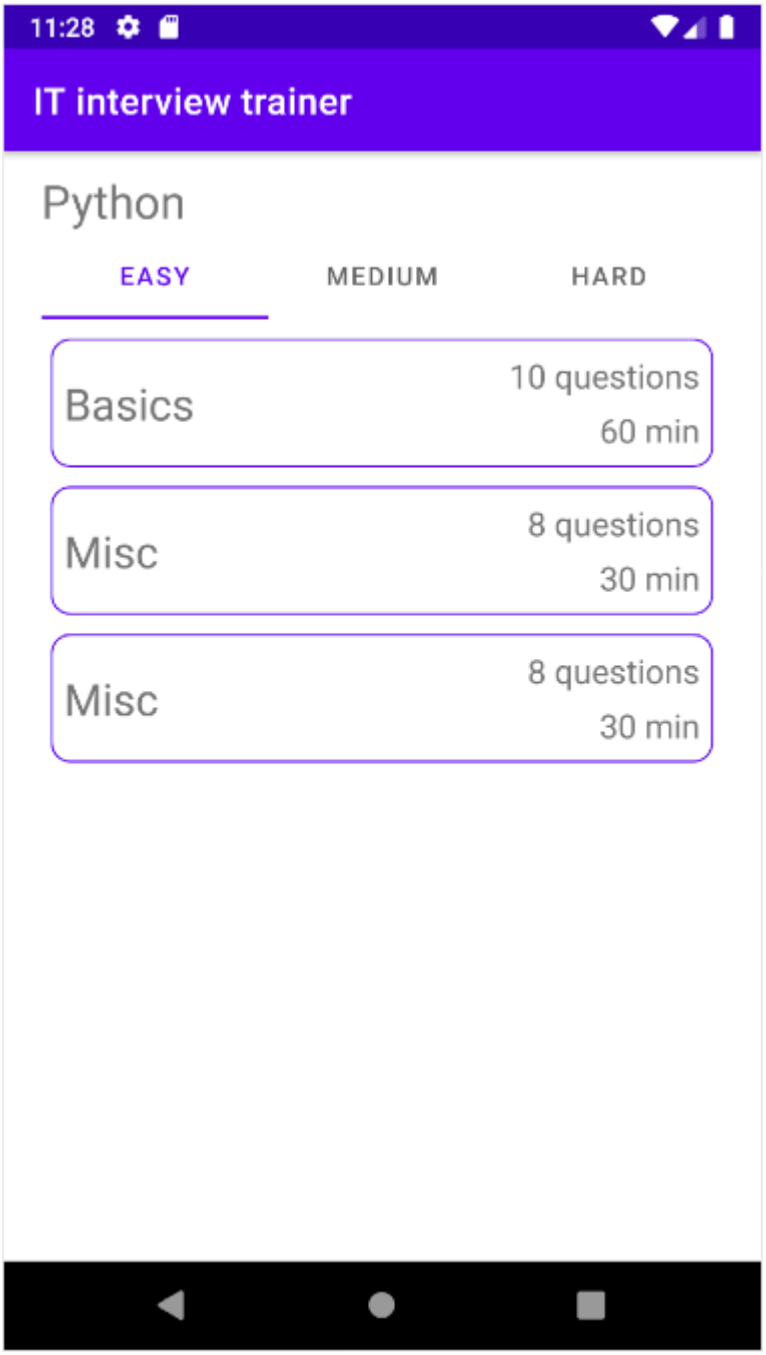


Рисунок 11 – выбор конкретного теста указанной сложности

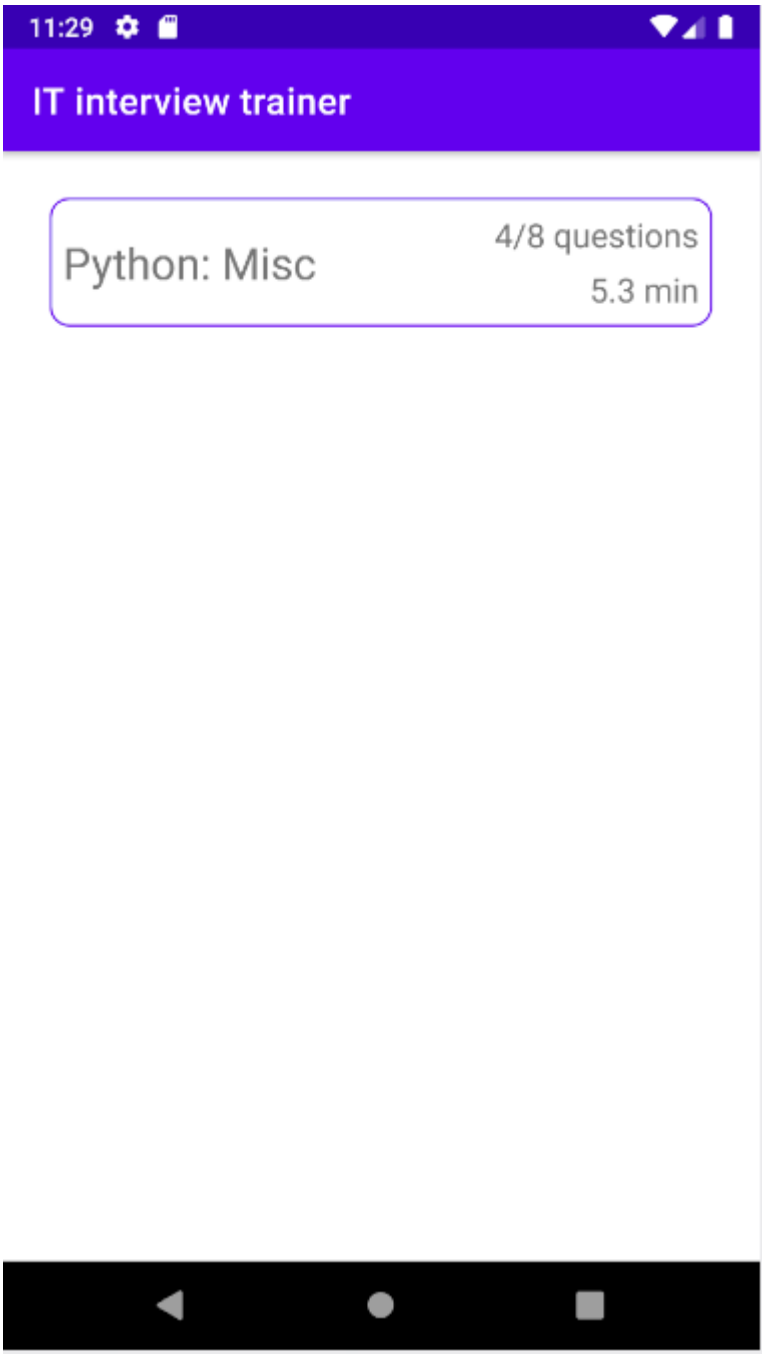


Рисунок 12 – Статистика (экран после клика на кнопку “Statistics”)