СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ

Время сенсомоторной реакции (ВСМР). Испытуемый должен как можно быстрее отреагировать на появление стимула. Время этого ответа состоит из времени возбуждения рецепторов, передачи возбуждения в соответствующие участки коры, времени инициации моторной программы и моторного компонента самого ответа.

Время дифференциальной реакции простого выбора ($ВДР_1$). Субъект реагирует на стимулы определенного вида и игнорирует все остальные. Время этой реакции увеличивается из-за появления дополнительной стадии обработки информации.

Время когнитивных процессов (ВКП) – это центральная задержка при появлении дополнительной стадии обработки информации.

Время дифференцированного комплексного ответа с выбором (далее $ВДР_2$). Испытуемый отвечает на каждый тип стимула определенной двигательной реакцией. Время сложного выбора еще более увеличивается, в основном за счет увеличения продолжительности принятия решения о том, как реагировать на конкретный стимул [1].

Диаграмма вариантов использования (англ. *use case diagram)* в UML – диаграмма, отражающая отношения между акторами и прецедентами и являющаяся составной частью *модели прецедентов*, позволяющей описать систему на концептуальном уровне.

Прецедент – возможность моделируемой системы (часть её функциональности), благодаря которой пользователь может получить конкретный, измеримый и нужный ему результат. Прецедент соответствует отдельному сервису системы, определяет один из вариантов её использования и описывает типичный способ взаимодействия пользователя с системой. Варианты использования обычно применяются для спецификации внешних требований к системе.

ВВЕДЕНИЕ

Измерение времени реакции при экспериментальном изучении различных психических явлений имеет давнюю историю и традицию. Хронометрия - один из первых классических психофизиологических методов и, по словам голландского физиолога Франса Дондерса, основателя этого метода в 19 веке, один из самых важных для определения взаимосвязи между "особенностями каждого ощущения, каждого воображения, каждого волевого акта и определенными характеристиками мозговой деятельности".

Ф. Дондерс первым разработал принципиальную схему эксперимента, позволяющую определить временные параметры протекания психических процессов. Он предположил, что увеличение сложности экспериментального задания приведет к добавлению новых этапов и, следовательно, к увеличению времени реакции. Такая величина увеличения времени реакции соответствует продолжительности дополнительных этапов.

В настоящее время большинство исследователей заинтересованы не только в изучении средних значений времени реакции, но и в анализе распределения результатов. Действительно, повторное измерение времени реакции любого индивидуума в неизменных экспериментальных условиях показывает значительные вариации этого параметра, индивидуальные значения времени реакции могут отличаться от среднего значения, полученного у одного и того же индивидуума в одном и том же эксперименте, в 1,5-2 раза [2, 3, 4]. Однако форма распределения результатов времени реакции, зарегистрированных в разные моменты времени, относительно постоянна для каждого отдельного испытуемого в сопоставимых экспериментальных условиях [5].

Целью данного дипломного проекта является создание программного модуля для диагностики психического состояния человека на основе изучения простых реакций, используя клиент-серверную технологию и последующего предоставления рекомендаций пользователю.

Для достижения цели дипломного проекта необходимо было выполнить следующие задачи:

- изучить принцип клиент серверной технологии;
- спроектировать архитектуру программного средства;
- разработать программный модуль для изучения простых реакций;
- разработать программный модуль для серверной части проекта;
- выполнить технико-экономическое обоснование

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ОБЗОР СУЩЕСТВУЮЩИХ РЕШЕНИЙ

1.1 Анализ предметной области

Работа программного модуля, который будет реализован в рамках дипломного проекта, заключается в изучении простых реакций человека. Алгоритм модуля можно разбить на два этапа.

Метод регистрации времени реакции.

Второй этап заключается в исследовании результатов и формировании рекомендаций.

Рассмотрим подробнее методики регистрации времени реакции и варианты исследования.

1.1.1 Методики регистрации времени реакции

Появление персональных компьютеров значительно упростило организацию хронометрических исследований, автоматизировав процесс тестирования и одновременно расширив возможности экспериментатора. По этим причинам активно разрабатываются компьютерные методики регистрации времени сенсомоторных реакций [6, 7]. Каждая методика обычно состоит из нескольких серий, в которых регистрируются различные типы ответов, наиболее важными из которых являются:

1. Простая сенсомоторная реакция. Испытуемый должен как можно быстрее отреагировать на появление стимула. Время этого ответа (далее ВСМР) состоит из времени возбуждения рецепторов, передачи возбуждения в соответствующие участки коры, времени инициации моторной программы и моторного компонента самого ответа.

2. Дифференциальная реакция простого выбора. Субъект реагирует на стимулы определенного вида и игнорирует все остальные. Время этой реакции (далее $ВДР_1$) увеличивается из-за появления дополнительной стадии обработки информации (мы называем эту центральную задержку временем когнитивных процессов - ВКП).

3. Дифференцированный комплексный ответ с выбором. Испытуемый отвечает на каждый тип стимула определенной двигательной реакцией. Время сложного выбора (далее $ВДР_2$) еще более увеличивается, в основном за счет увеличения продолжительности принятия решения о том, как реагировать на конкретный стимул.

Задания в методиках подобраны таким образом, чтобы можно было максимально эффективно определить влияние различных факторов на общее время реакции и его составляющие. В качестве схемы интерпретации полученных результатов использованы идеи Ф. Дондерса о компонентном составе времени реакции и парадигма С. Стернберга, согласно которой каждый фактор влияет на длительность только одного, "своего" этапа

процесса решения когнитивной задачи и никак не может повлиять на длительность других этапов [8].

Учитывая, что приоритетной для разработки является тематика спортивной психофизиологии, многие методики разработаны специально для спортсменов, поэтому они удовлетворяют двум дополнительным требованиям:

1. Тематизированны и разнообразны по форме и содержанию, для покрытия большей аудитории, так, как только при наличии высокой мотивации пользователя к работе можно получить адекватные результаты.

2. При проведении занимают мало времени (вариант экспресс диагностики), чтобы не вызывать утомления, и в то же время позволяют получить статистически достоверные результаты.

Большинство результатов, представленных ниже, получено по четырем методикам. Методика «Первый-второй сигнал» была разработана одной из первых. Она позволяет регистрировать время простой сенсомоторной реакции и время дифференцированных реакций. В качестве стимулов используются цветные схематизированные изображения, рис. 1.1.

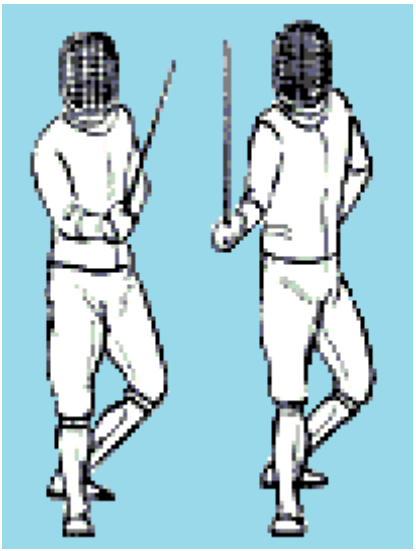

Рисунок 1.1 – Стимульные изображения, используемые в методике «Первый-второй сигнал»

Дифференцируемые в этой методике изображения отличаются сразу по многим признакам, поэтому стратегии дифференцировки картинок у разных испытуемых могут различаться. Вариант различения стимулов по многим ключевым признакам наиболее часто встречается в реальных жизненных ситуациях, поэтому результаты данной методики дают информацию о временных параметрах процесса дифференцировки в обобщенном виде [9].

Методика OZO позволяет сравнить результаты дифференцировки изображений по разным ключевым признакам. Для этого проводятся четыре аналогичных серии, в которых испытуемые дифференцируют изображения

соответственно по форме (квадрат – круг), по размеру (большой и маленький квадраты), по цвету (красный и зеленый квадраты), по ориентации (правильные треугольники, ориентированные вершиной вверх или вниз). В этой методике в каждой серии регистрируются все три основных типа сенсомоторных реакций. Такой экспериментальный план позволяет проанализировать компонентный состав реакций и роль двух факторов: 1) ключевого признака изображения, по которому осуществляется дифференцировка; 2) способа организации моторного ответа, а также проследить взаимное влияние этих факторов [10].

В *методике «Звезды»* акцент делается на выявлении механизмов дифференцировки зрительных стимулов. В качестве стимулов используются сконструированные изображения, напоминающие звезды, рис. 1.2.

Рисунок 1.2 – Изображения, используемые в методике «Звезды»

Эти изображения могут различаться по трем признакам:

1 – форме лучей (прямые или дугообразные);

2 – количеству лучей (четыре или шесть);

3 – расположению лучей (наличие или отсутствие вертикально ориентированных лучей).

В методике возможно сделать ключевым любой из признаков изображения (или любое их сочетание) и зарегистрировать временные параметры простого выбора. Во всех трех методиках изображение появляется в центре темного экрана монитора и исчезает при нажатии испытуемым на изображение (или по истечении определенного, заранее заданного временного интервала), определение времени реакции производится автоматически таймером компьютера с точностью 1 мс. У каждого испытуемого регистрируется по 5, или более, реакций каждого типа. Соотношение релевантных и нерелевантных стимулов во всех методиках при регистрации дифференцированных реакций составляет 1:1 [11].

В настоящее время накоплен значительный статистический материал, разработаны нормативы, что позволяет использовать многие методики в качестве тестов для определения сенсомоторного развития, биологического возраста, диагностики некоторых параметров зрительного восприятия и внимания.

1.1.2 Исследование результатов и формирование рекомендаций

При анализе результатов взрослых испытуемых мы неоднократно устанавливали, что изменение средней длительности реакций с усложнением задания происходит на вполне конкретную и предсказуемую величину, то есть наблюдается своеобразное временное квантование реакций. Выявлены две формы такого квантования.

1. Длительности простой и дифференцированной реакций соотносятся между собой как целые числа [12]. В качестве примера в табл. 1 приведены результаты выполнения теста OZO студентами (58 человек), усредненные по 4-м сериям.

Таблица 1.1 – Латентные периоды реакций разных типов и ВКП у студентов по тесту OZO, мс.

ВСМР	$ВДР_1$	$ВДР_2$	ВКП
302±14	434±19	450±17	140±10

Продолжительность дифференцированных ответов примерно в 1,5 раза больше, чем продолжительность простой сенсомоторной реакции. Если мы рассмотрим ВКП, то получим следующий набор коэффициентов: ВКП : ВСМР : ВДР = 0.15 : 0.30 : 0.45 с = 1:2:3. Один из возможных вариантов объяснения этого явления - циклическая обработка информации в нервной системе. Существование таких циклов обработки информации постулируется, в частности, А.М. Иваницким и др. в концепции синтеза информации [13].

2. длительность дифференцированных ответов увеличивается кратно 15-20 мс, когда задача несколько усложняется. Это явление наблюдается как в случае изменения моторного ответа (табл. 1, длительность $ВДР_2$ на 16 мс больше, чем $ВДР_1$), так и при усложнении процесса дифференцировки.

В тесте «Звезды», где испытуемые должны были нажимать на изображение в случае появления релевантной картинки, ориентируясь на 1 или 2 признака изображения, получены следующие результаты, табл. 2.

Таблица 1.2 – Среднее время реакции у студентов (120 человек) в 6-ти сериях теста «Звезды», мс.

Серия	Релевантные стимулы	Результаты
1	Только «морские» звезды	494±11
2	Любые звезды с 6 лучами	510±13
3	Любые звезды с присутствием вертикально расположенного луча	485±11
4	Только «морские» звезды с 6 лучами	495±12
5	Только «морские» звезды с присутствием вертикально расположенного луча	523±13
6	6-лучевые звезды с присутствием вертикально расположенного луча	525±12

Сопоставление результатов, полученных в разных сериях, показывает, что они разделяются на три группы, предположительно в зависимости от используемого механизма дифференцировки:

1 – наименьшее время в первой, третьей и четвертой сериях (около 490 мс), возможно здесь используются варианты детекторного опознания (детекторы ориентации линии, детекторы кривизны линий) [14], либо эталонное опознание (в четвертой серии одна и та же звезда, но по-разному ориентированная) [15].

2 – во второй серии, где в изображении должен быть выделен и проанализирован 1 признак изображения – количество лучей, время составляет 510 мс.

3 – наибольшее время реакции в пятой и шестой сериях, где требуется проанализировать два признака (525 мс).

Принципиально, что различие между сериями составляет примерно 15-20 мс. Подобные феномены отмечались и ранее с использованием иных методических приемов. По В. Д. Глезеру, 15-20 мс: предположительно столько протекает элементарный акт опознавательного процесса – принятие решения по какому-либо признаку. В тахистоскопических исследованиях с маскировкой Д. Саги и В. Джулеш показано, что на определение ориентации одной линии затрачивается 16,6±3,2 мс [16].

Эти и другие результаты свидетельствуют, что общее время реакции складывается из отдельных временных квантов, а сам процесс дифференцировки состоит из многочисленных операций, при этом набор этих операций всякий раз подбирается таким образом, чтобы суммарное время было минимальным, что говорит о том, что время реакции можно уменьшать с помощью решения специалистов.

1.2 Обзор существующих аналогов

Существует множество приложений диагностики психического состояния человека с клиент-серверной технологии.

1.2.1 CogniFit

Компания CogniFit («КогниФит») была основана в 1999 году профессором Шломо Брезницем [17]. Компания занимается медицинскими исследованиями и фокусируется на оценке и улучшении когнитивного здоровья. Проверяются все их инструменты для изучения и стимулирования работы мозга. Продукт доступен на 19 языках. Они сотрудничают с больницами, университетами, фондами и исследовательскими центрами по всему миру [18]. В настоящее время CogniFit («КогниФит») является ведущей программой, признанной и успешно используемой научным сообществом, и компаниями по всему миру.

Рисунок 1.3 – Компания CogniFit

Преимущества приложения CogniFit:
– мультиязычность (19 языков);
– протестированность инструментов для исследования;
Недостатки приложения CogniFit:
– дорогостоящая программа;
– недоступность в режиме оффлайн;
– сложность интерфейса;

1.2.2 Меморадо

Программа компании “Memorado Gmbh”, которая также имеет множество тестов и исследований.

Рисунок 1.4 – Меморадо: Тренировка памяти

Преимущества приложения Меморадо:
– Медитационное аудио сессий успокоения ума;
– Потрясающая графика в сочетании с легкостью использования;

– Персональные ежедневные задания, которые конфигурируются по желанию пользователя.

Недостатки приложения Меморадо:

– необходимость лицензии для использования;
– отсутствие кроссплатформенности (доступна только на Android);
– недоступность в режиме оффлайн;

1.2.3 Lumosity – тренировка мозга

Команда ученых и разработчиков, которые ищут новые способы бросить вызов мозгу и пытаются сделать новые открытия в сфере познания.

Рисунок 1.5 – Lumosity – тренировка мозга

Преимущества приложения Lumosity – тренировка мозга:

– гибкость разработки;
– дружелюбный интерфейс;
– отсутствие лицензии;

Недостатки приложения Lumosity – тренировка мозга:

– большое количество рекламы;
– недоступность в режиме оффлайн;

В заключение анализа, можно отметить, что в приложениях, для диагностики психического состояния человека пользователю необходимо быть в режиме онлайн. Также приложения, которые были проанализированы, являются самостоятельными продуктами. Не имеется возможность собственной интеграции их в сторонние разрабатываемые программные системы.

2 АНАЛИЗ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ

2.1 Описание функциональности программного средства

Для описания функциональности программного средства необходимо описать последовательность действий, которые может выполнять система в ответ на внешние воздействия пользователей или других программных систем. Данное описание поможет отразить функциональность системы с точки зрения получения результата, который значим для пользователя.

На основе технического задания, составленного в первом разделе, получена Use Case диаграмма (рисунок 2.1).

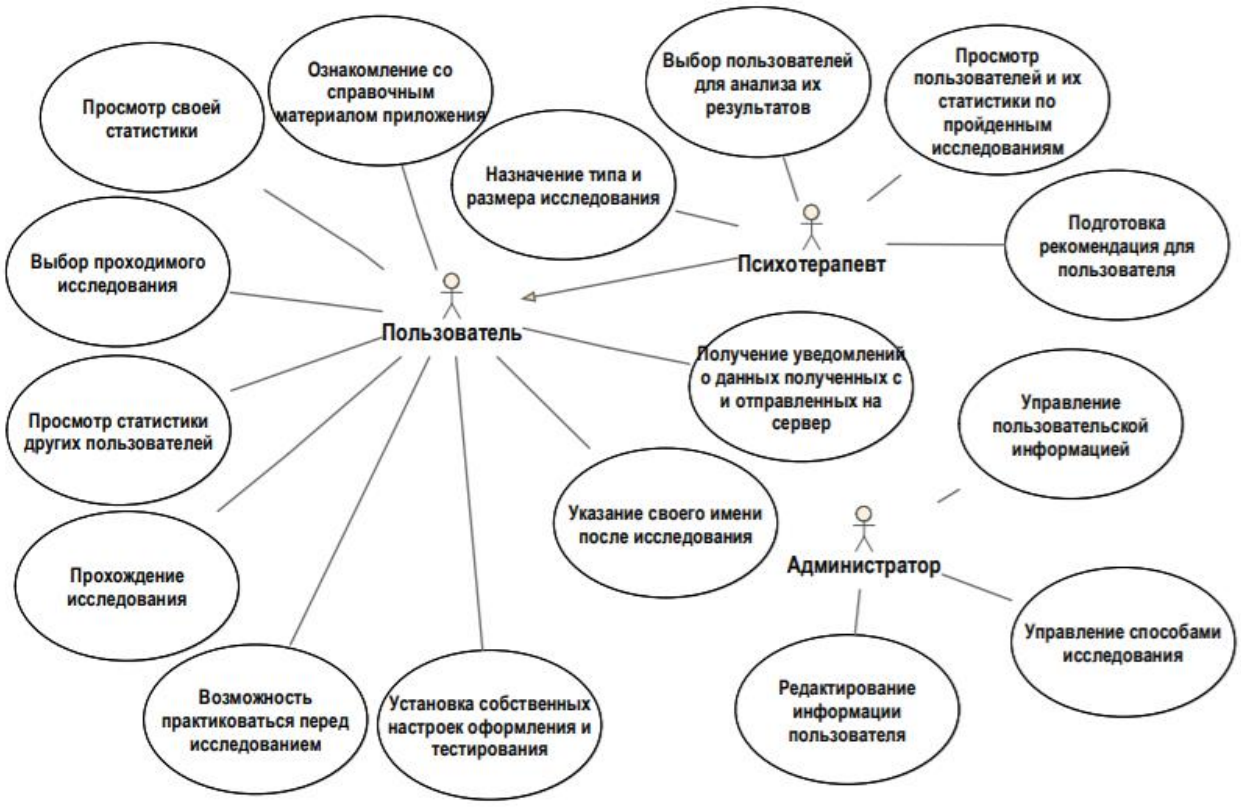

Рисунок 2.1 – Диаграмма вариантов использования

Данная диаграмма описывает всевозможные варианты использования программного средства с позиции пользователей относительно их ролей.

Как видно из диаграммы, в программном средстве пользователи могут выступать в качестве одной из трех ролей: обычный пользователь, психотерапевт и администратор.

Роль обычного пользователя является базовой ролью в системе. В целом, все основные функции программного средства доступны пользователям с этой ролью. Функции пользователя:

– Просмотр своей статистики. Информация о каждом пользователе хранится в базе данных. Каждый сотрудник может добавить информацию о себе в любой момент времени.

– Просмотр общей статистики. Каждый пользователь может просмотреть список результатов, которые получили другие пользователи. Список содержит всю историю от начала работы приложения.

– Просмотр справочной информации о прохождении исследования, навигации и приложении. Каждому пользователю доступна возможность установить свой набор настроек оформления и тестирования. Доступность данных настроек определяется Администратором, в настройках каждой опции настрое, путем выбора данного пользователя в качестве тестируемого по этому типу исследования.

– Прохождение исследования. Пользователь может пройти выбранное исследование. Тип исследования зависит от выбранного пользователем исследования.

– Написание имени пользователя к выолненному тесту. Помимо тестирования как такового предусмотрена возможность подписания пройденного теста и отправки имени вместе с результатами на сервер. Пользователь при отправке результатов может ввести свое имя, которое будет отправлено на сервер.

– Получение уведомлений о полученных с и отправленных на сервер данных. Каждому пользователю приходят уведомления в случае получения с или отправления на сервер данных. Эти данные хранятся в базе данных на стороне сервера, что соответственно позволит получить доступ к ним в любой момент.

Как видно из диаграммы Use Case, психотерапевт включает в себя все функции обычного сотрудника, однако имеет расширенные функции, такие как:

– Просмотр пользователей и их статистики по пройденнмым исследованиям. В данный список будут входитьпользователи, которые прошли исследования.

– Выбор пользователей для анализа их результатов. Психотерапевт выбирает, кого из пользователей следует анализировать.

– Назначение типа и размера исследования. Психотерапевт также указывает тип исследования и его размер, рекомендуемый к прохождению пользователем.

– Подготовка рекомендаций для пользователя. Психотерапевт, проанализировав результаты пользователей выносит индивидульные рекомендации, соответствующие его потребностям.

Администратор выполняет следующие функции:

– Управление пользовательской информацией. В эту функцию входит редактирование информации о пользователе, удаление пользователя. Данные функции позволяют гибко управлять пользовательской информацией.

– Управление способами исследования. Администратор может создать свой тип исследования в комплекте с оформлением и его размером. Администраторы настраивают приложение по мере необходимости, и сервер публикует эту программу любому пользователю.

– Редактирование информации пользователя. Так как в приложении будут размещены различные программы исследования, то для ускорения процесса их индивидуализации администраторы могут использовать данную функцию редактирования контекста приложения.

2.2 Разработка информационной модели базы данных

Прежде чем приступать к процессу разработки программного средства необходимо смоделировать базу данных, которая является основой будущего приложения. Для того чтобы база данных не вызывала трудностей при работе с ней, когда она будет заполнена информацией желательно, чтобы она соответствовала трем нормальным формам:

а) Первая нормальная форма. Переменная отношения находится в первой нормальной форме тогда и только тогда, когда в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов.

б) Вторая нормальная форма. Переменная отношения находится во второй нормальной форме тогда и только тогда, когда она находится в первой нормальной форме и каждый неключевой атрибут функционально полностью зависит от своего потенциального ключа.

в) Третья нормальная форма. Переменная отношения находится в третьей нормальной форме, только если она находится во второй нормальной форме и не существует переходных функциональных зависимостей неключевых атрибутов ключа.

Используя данные правила, была смоделирована схема базы данных программного средства. На рисунке 2.2 представлена полная версия схемы базы данных приложения, которая включает в себя все основные его функции.

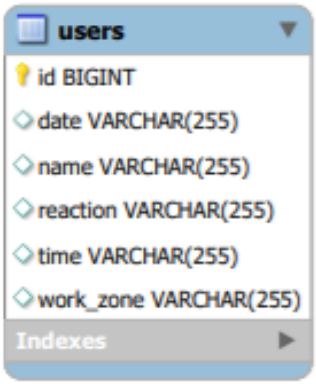

Рисунок 2.2 – Схема базы данных

2.2.1 Таблица информации о пользователях

Таблица “users” содержит подробную информацию о пользователях данного приложения. В таблице 2.1 рассмотрены атрибуты таблицы “users”.

Таблица 2.1 – Атрибуты таблицы “users”

Атрибут	**Описание**
id, PK	Идентификатор пользователя
date	Дата прохождения исследования
name	Имя пользователя
reaction	Тип реакции
time	Результаты прохождения исследования

2.3 Разработка спецификации функциональных требований к программному средству

Интерфейс

Данное программное средство должно иметь интуитивно понятный интерфейс. Пользовательский интерфейс должен быть выполнен в простом виде, понятный неподготовленному пользователю. Интерфейс программного средства будет включать в себя несколько страниц:

– главная страница приложения, на которой будет отображена информация о приложении, меню для пользователя;

– страница настроек;

– страница исследования;

– страница с информацией о результатах пользователей;

– страница с авторами приложения и логотипом.

Форматы входных (выходных) данных

Входными данными для исследования в данном приложении являются информация о выбранных для исследования сигналах, информация о программе тестривания, а также количество тестов и сопроводительное имя пользователя (опционально).

Выходные данные для процесса тестирования являются электронные статистические данные.

Для всех остальных процессов выходными данными будут служить Activity с необходимой информацией, зависящей от действия пользователя.

Функциональные требования

Разрабатываемое программное средство должно представлять собой мобильное приложение, предназначенное для эффективного тестирования пользователей с использованием клиент-серверной технологии. Основные возможности разрабатываемого программного средства заключаются в следующем:

– Возможность регистрации пользователей с помощью имени. Когда пользователь входит в систему или переходит на любую форму, ему предоставляется информация пользования приложением, с которыми ему следует ознакомиться. Данное ознакомление доступно во время входа в форму или при нажатии соответствующей кнопки «Помощь». После прохождения исследования пользователь вводит свое имя для подписания результатов.

– Возможность просмотра информации о себе. Данная возможность должна быть реализована в виде страницы результатов пользователя, на которой будут отображена информация о нм в виде текста. Поля, которые должны быть на этой странице, указаны в таблице “users”, которую можно найти в разделе информационной модели базы данных.

– Возможность просмотра общей статистики. Данная возможность должна быть реализована в виде отдельной страницы, на которой будет отображена информация обо всех результатах, которые пользователи приложения когда-либо получали. Также на данной странице должна быть справочная информация об отображаемых данных.

– Возможность прохождения исследования. Функция тестирования доступна на странице главного меню. Должна быть предусмотрена возможность прохождения любого выбранного исследования. На странице исследования должно быть описание этого исследования, того как она работает, а также поле для взаимодействия с тестами, результаты которых будут отправлены вместе с именем пользователя на сервер. Должна быть предоставлена возможность предпросмотра статуса результата отправленных данных.

– Возможность получения уведомлений о полученных с и отправленных на сервер данных. Уведомления представлены в виде сообщения. Пользователь может прочитать полученное сообщение на специальной странице результатов илипосле прохождения исследования, если он указал свое имя. Отправка уведомлений и их текст конфигурируются в настройках приложения.

– Возможность администрирования приложения. Предусмотрена панель администратора в виде отельной формы, на которой администратору будут доступен список пользователей, список исследований и список форм приложения. Администратор имеет возможность удалить пользователя, сбросить его данные, установить собственные. Администратор имеет функцию поиска исследований, создания новых исследований, настройки и редактирования существующих исследований, их активацию и деактивацию.

– Возможность отправки данных на сервер. После прохождения

исследования пользователь имеет возможность ввести свое имя и отправить результаты на сервер для дальнейшего изучения их психотерапевтом.

– Возможность получения данных с сервера. Пользователь имеет возможность получить данные с сервера с общей статистикой выполнения тестирований и ознакомиться с ними.

– Возможность настройки приложения. Пользователь может настроить приложения для персонализации. Он может изменить цвет фона приложения, изменить тип реакции исследования, размер рабочего поля, варианты первого и второго сигнала, задать отображение статистики после завершения тестирования, задать вариацию теститрования без учета пропущенных сигналов, задать количество успешных тестов для прохождения исследования, задать выделение пунктиром рабочей зоны.

В соответствии с возможностью прохождения исследований, в данном программном средстве представлены следующие типы исследований:

– Режим тренировки. Данное иследование позволяет пользователям практиковаться перед непосредственным прохождением теста. Пользователям рекомендуется выбирать один из специально определенных под компанию шаблонов-насроек, которые отправляются по электронной почте получателю.

– Режим прохожденя исследования. Данная функциональность позволяет пользователю пройти выбранное тестирование. В зависимости от выбранного пользователем типа исследования, изменяется количество успешных тестов для его прохождения.

– Режим настройки приложения. Данная функциональность позволяет пользователю выбрать и установить подоходящие ему настройки программного средства. В этом режиме пользователь изменить цвет фона приложения, изменить тип реакции исследования, размер рабочего поля, варианты первого и второго сигнала, задать отображение статистики после завершения тестирования, задать вариацию теститрования без учета пропущенных сигналов, задать количество успешных тестов для прохождения исследования, задать выделение пунктиром рабочей зоны. В зависимости от выбранного пользователем типа исследования, изменяется количество успешных тестов для его прохождения.

Приложение изменяет заголовок формы исследования для отражения информации, необходимой для процесса тестирования для выбранного его вида. Следовательно, как только пользователь выбирает тип тестирования, по которому он хочет пройти исследование, заголовок и подсказки настраиваются таким образом, чтобы была получена требуемая информация.

В большинстве форм приложения, будут появляться подсказки и информация, предназначенные для получения рекомендации по пользованию приложением.

3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

3.1 Проектирование диаграммы развертывания

Диаграмма развертывания представляет взаимосвязи между компонентами системы. Чтобы описать мобильное приложение с технологией клиент-сервер, диаграмма развертывания должна показать, какие аппаратные компоненты существуют, какие программные компоненты работают на каждом узле, и как различные части этого комплекса связаны друг с другом.

На рисунке 3.1 изображена диаграмма развертывания, разрабатываемого ПС.

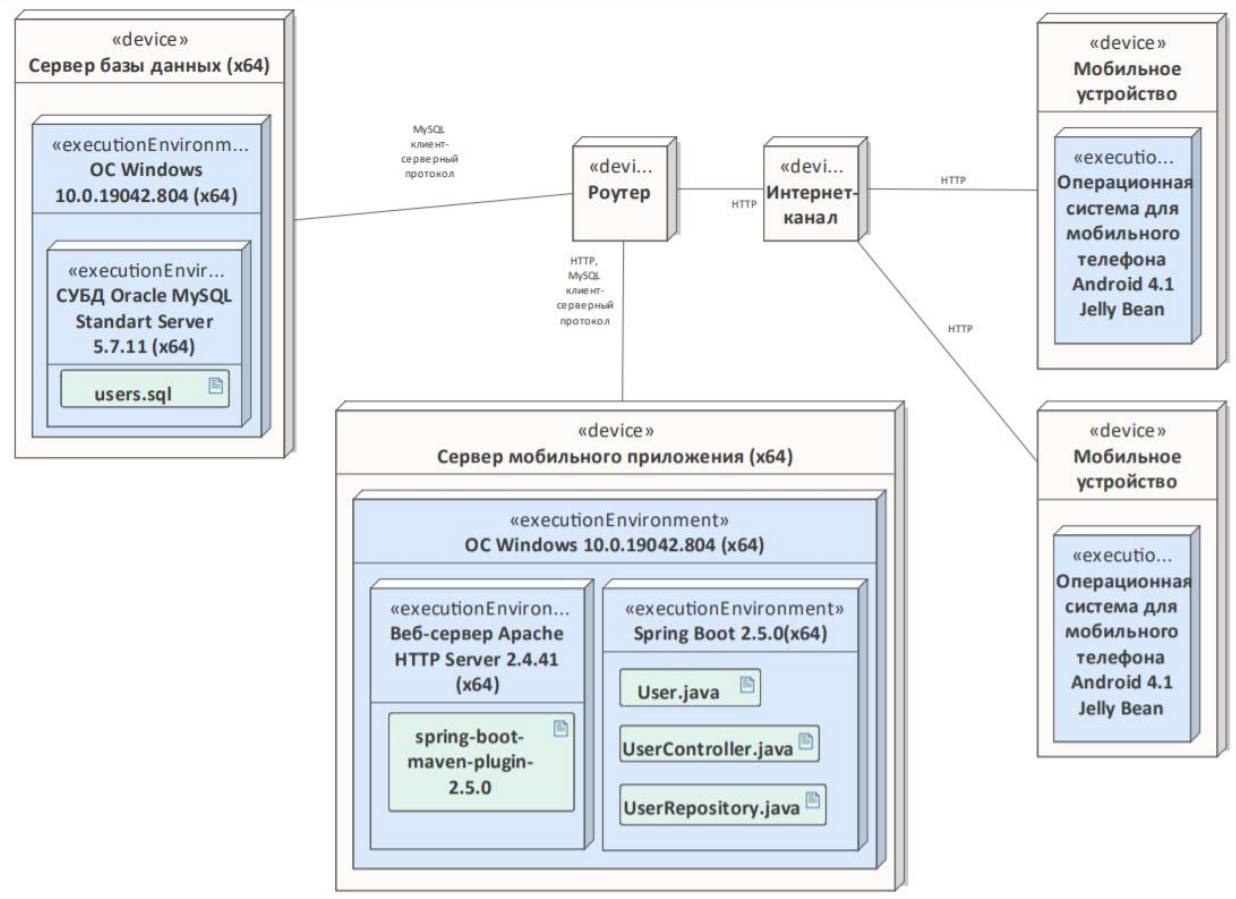

Рисунок 3.1 – Диаграмма развертывания

Коммуникации со всеми видами пользователей программного средства осуществляется через сеть интернет, что позволит управлять системой удаленно, что удобно для пользователей. Для работы программного средства потребуется веб сервер, на котором будет развернуто наше приложение, и сервер базы данных.

Веб сервер приложения отвечает за обработку и хранение данных. Обращение с приложением происходит по принцип “запрос-ответ”. Клиент,

которым является создаваемое программное средство, передает веб-серверу запросы на получение ресурсов, обозначенных URL-адресами. Ресурсы – это данные, которые необходимы клиенту. В ответ веб-сервер передает клиенту запрошенные данные. Этот обмен происходит по протоколу HTTP. Так как данное программное средство разрабатывается на платформе Java, с использование фреймворка Spring, то желательно, чтобы веб сервер в качестве операционной системы имел Windows 10.

Сервер базы данных также должен быть развернут на отдельной машине, а в качестве самого сервера выбран MySQL Server 2012, что опять-таки является основным выбором в качестве сервера баз данных для Java разработчиков. Основная задача данного сервера – хранение и запись данных. Взаимодействие с пользователем осуществляется через промежуточное звено – веб сервер.

3.2 Проектирование программной архитектуры

Архитектура программного обеспечения – это структура программы или вычислительной системы, которая включает программные компоненты, видимые снаружи свойства этих компонентов, а также отношения между ними. Для представления архитектуры данного программного обеспечения использована диаграмма компонентов (рисунок 3.2).

Диаграмма компонентов позволяет создать физическое отражение текущей модели. Она показывает организацию и взаимосвязи программных компонентов, представленных в различных модулях программы, а ее связи отражают зависимости одного компонента от другого.

Как видно из диаграммы, разрабатываемое программное средство имеет многоуровневую архитектуру. Если рассматривать физический уровень, то приложение разделено на три уровня: сервер базы данных, веб-приложение на веб-сервере и приложение на стройстве пользователя. С позиции логических уровней архитектуры приложения архитектура разрабатываемой программы представлена следующими уровнями:

– База данных. В нашем случае это MySQL Database. Единственный способ хранения информации в проекте.

– Data access layer (уровень доступа к данным). Хранит модели, которые описывают используемые сущности, также здесь размещаются специфичные классы для работы с разными технологиями доступа к данным (в нашем случае это класс контекста данных Entity Framework). Здесь также хранятся репозитории, через которые уровень бизнес-логики взаимодействует с базой данных. Как видно из диаграммы в качестве класса контекста данных используется класс Universe, который наследуется от ObjectContext, стандартного класса библиотеки EntityFramework. Данный класс был создан автоматически EntityFramework’ом, так как был использован метод создания связи между моделями приложения и базой данных, такой как Hibernate. Этот

метод позволяет EntityFramework самому позаботится о создании сущностей для приложения.

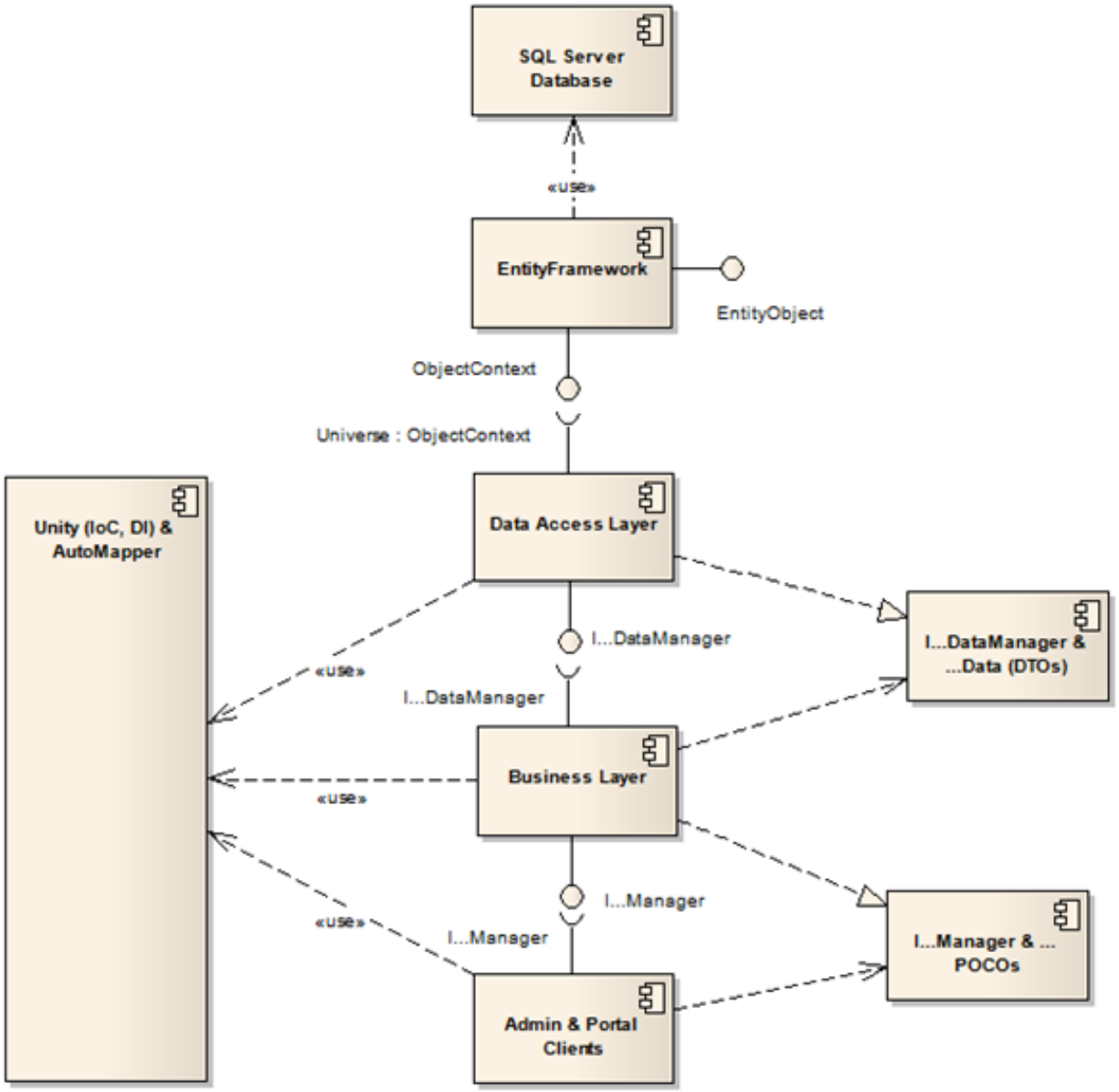

Рисунок 3.2 – Диаграмма компонентов

Также ручные изменения базы данных избавят от накатывания миграции для изменения схемы базы данных, так как EntityFramework сам обновит модели в коде программы в соответствии с изменением схемы базы данных.

– Business layer (уровень бизнес-логики): содержит набор компонентов, которые отвечают за обработку полученных от уровня представлений данных, реализует всю необходимую логику приложения, все вычисления, взаимодействует с базой данных и передает уровню представления результат обработки. Как видно из диаграммы, уровень бизнес-логики взаимодействует с слоями доступа к данным и представления с помощью интерфейсов

IDataManager и IManager соответственно. Для передачи данных между этими слоями приложения используются специальные классы, содержащие данные без какой-либо логики для работы с ними – DTO (Data Transfer Object). В приложении они рассматриваются как хранилища информации, единственная цель которых – передать информацию получателю.

– Presentation layer (уровень представления): это тот уровень, с которым непосредственно взаимодействует пользователь. Этот уровень включает компоненты пользовательского интерфейса, механизм получения ввода от пользователя. Применительно к Java на данном уровне расположены представления и все те компоненты, которые составляют пользовательский интерфейс, а также модели представлений, контроллеры, объекты контекста запроса.

Как видно из диаграммы, три слоя (представления, бизнес-логики и доступа к данным) использует в своей работе такие инструменты как Unity (IOC, DI).

Инструмент Unity является IOC-контейнером, который позволяет настроить и внедрить зависимости в приложение. Внедрение зависимостей дает возможность подменить некий сервис, не являющийся технологически нейтральным. Также благодаря внедрению зависимостей разработчики получают возможность автоматически тестировать различные модули программы независимо. К тому же появляется возможность повторно использовать написанный для зависимостей код с другими реализациями. В нашем случае объявлен ряд интерфейсов IDataManager и IManager, которые имеют свои реализации, которые применяются в процессе работы программы. Для настройки зависимостей используется класс UnityConfiguration.

3.3 Структура приложения

В данном разделе описывается карта приложения. Программное средство персонализируется в соответствии с действиями пользователей. Например, в случае некоторых настроек пользователей они будут видеть один цвет фона, а в другом случае другой. Пользователь также может настроить изменить тип реакции исследования, размер рабочего поля, варианты первого и второго сигнала, задать отображение статистики после завершения тестирования, задать вариацию теститрования без учета пропущенных сигналов, задать количество успешных тестов для прохождения исследования, задать выделение пунктиром рабочей зоны.

На рисунке 3.3 представлена основная карта приложения, которая соответствует обычному пользователю.

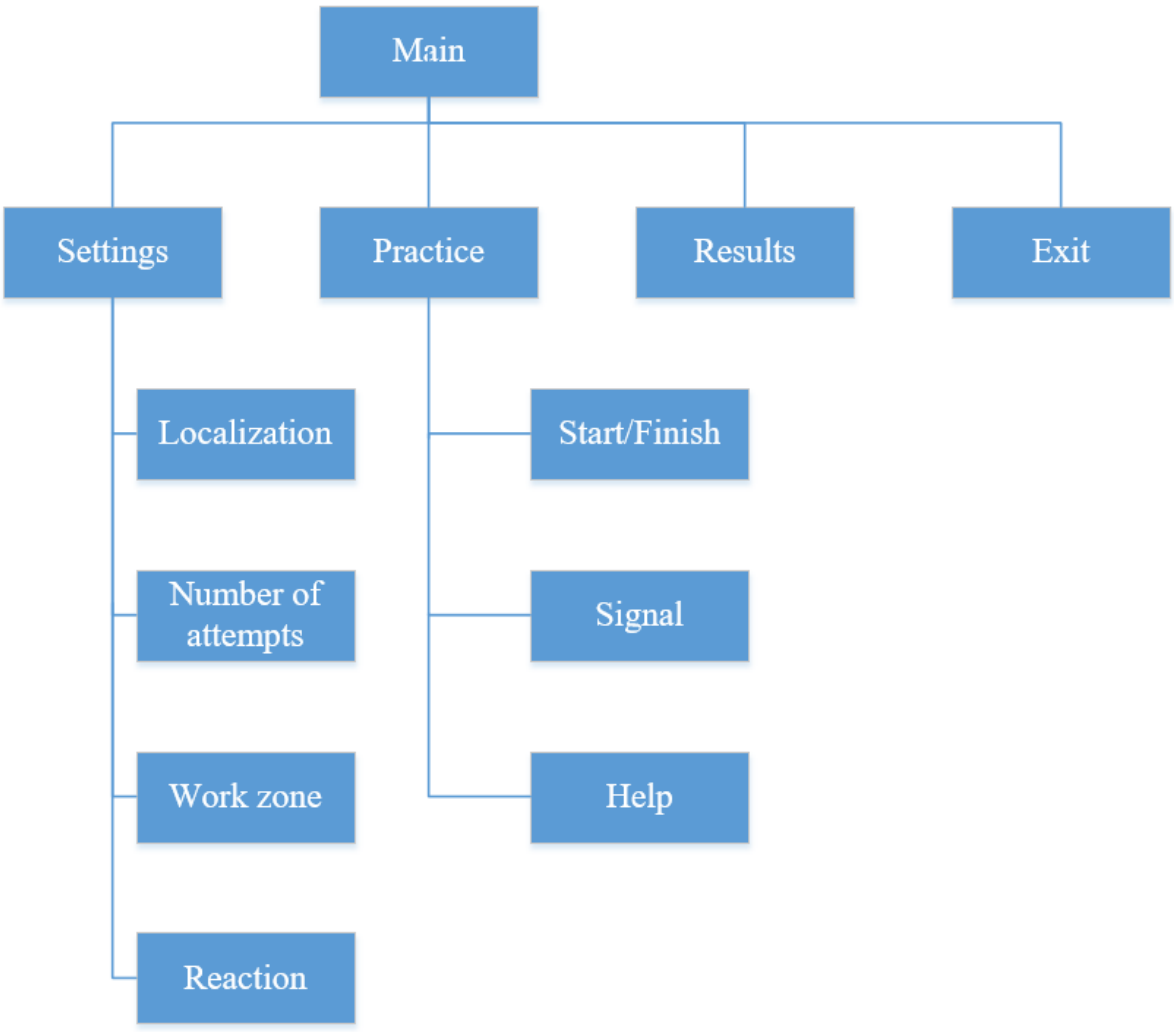

Рисунок 3.3 – Основная карта приложения

После входа пользователь перенаправляется на главную страницу, с которой он может попасть в один из трех подразделов: раздел “Настройки”, раздел “Практика” (исследования) и раздел “Результаты”.

В разделе настроек доступна страница с настройками приложения, форма со справочной информацией.

В разделе исследований пользователь может пройти тестирование, если он выбрал программу тестирования, либо же пройти практику, выбранную им.

В разделе “Результаты” пользователь может увидеть свою статистику и статистику других пользователей, прошедших исследование.

3.4 Разработка схемы работы программы

При проектировании была разработана функциональная модель программы. На основе этой модели были построены блок-схемы алгоритмов. Общая схема работы программы представлена на рисунке 3.4.

При запуске программного средства пользователю доступно несколько

направлений работы с данным приложением на текущем этапе.

При выборе такого действия, как просмотр и редактирование настроек, пользователь попадет на форму настроек своего профиля, на которой будет выведена информация об установках данного пользователя.

Также пользователь может просмотреть все свои результаты, которые он получил, на специальной форме «Результаты». На данной странице статистика отображаются в виде текста, которая содержит время прохождения исследования, его настройки и результаты.

При просмотре статистики пользователь перейдет на форму с результатами. Данные исследований будут отображаться в виде текста.

Действие по прохожднению тестирования пользователя включает в себя начало тестирования, непосредственное его прохождение, ознакомление со справкой и ввод своего имени при отправке конечных результатов на сервер. Данное действие разбивается на два варианта: данные отправлены или не отправлены.

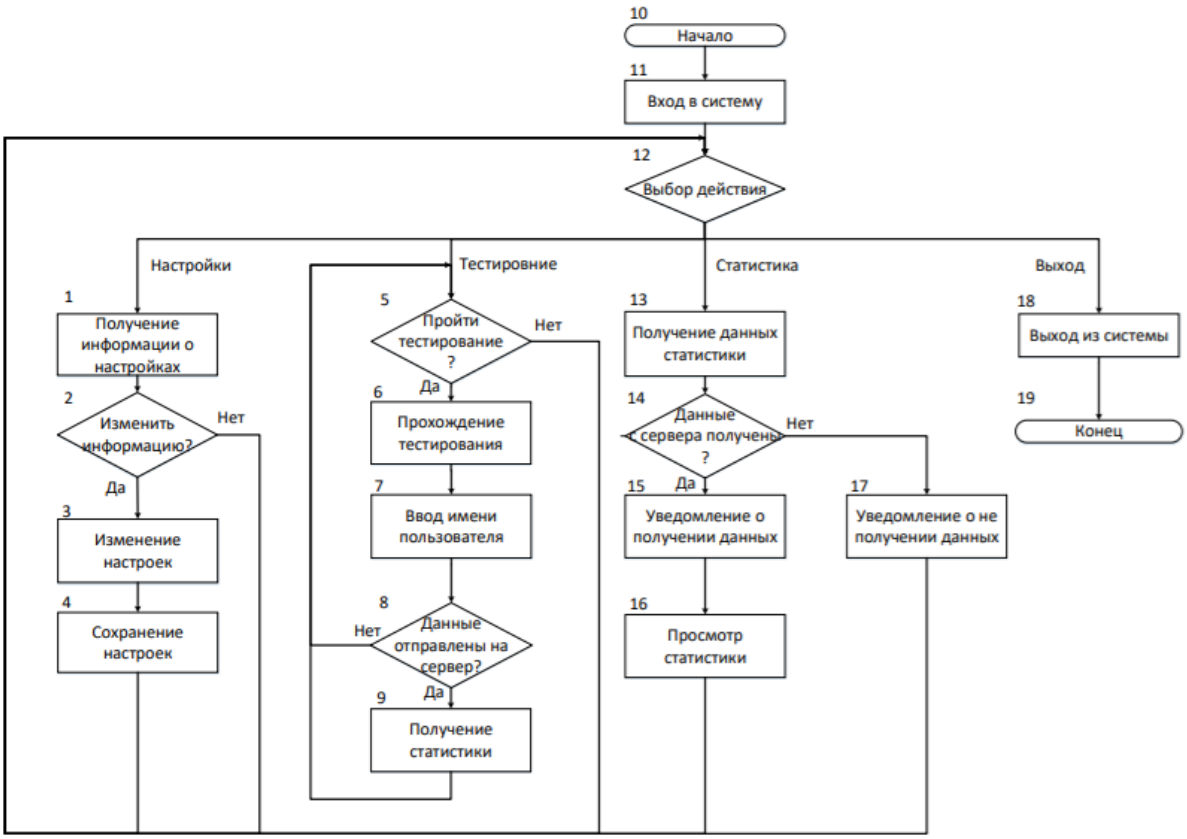

Рисунок 3.4 – Общая схема работы программы

Действие по подтверждению награды позволит пользователю выбрать из списка номинированных пользователей тех людей, которые по его решению получат награду. Выбор выполняется через процесс подтверждения или отказа номинации.

3.5 Разработка схемы установки настроек

Настройки позволяют пользователю выбрать тип исследования, которое он будет выполнять, персонализировать интерфейс приложения, задать формат рабочей зоны тестирования. Схема алгоритма работы настроек 3.5.

Как видно из блока 14 блок-схемы алгоритма, данная схема настроек подразумевает изменение локализации сразу всего приложения.

После выбора и установки настроек происходит сохранение их на устройстве.

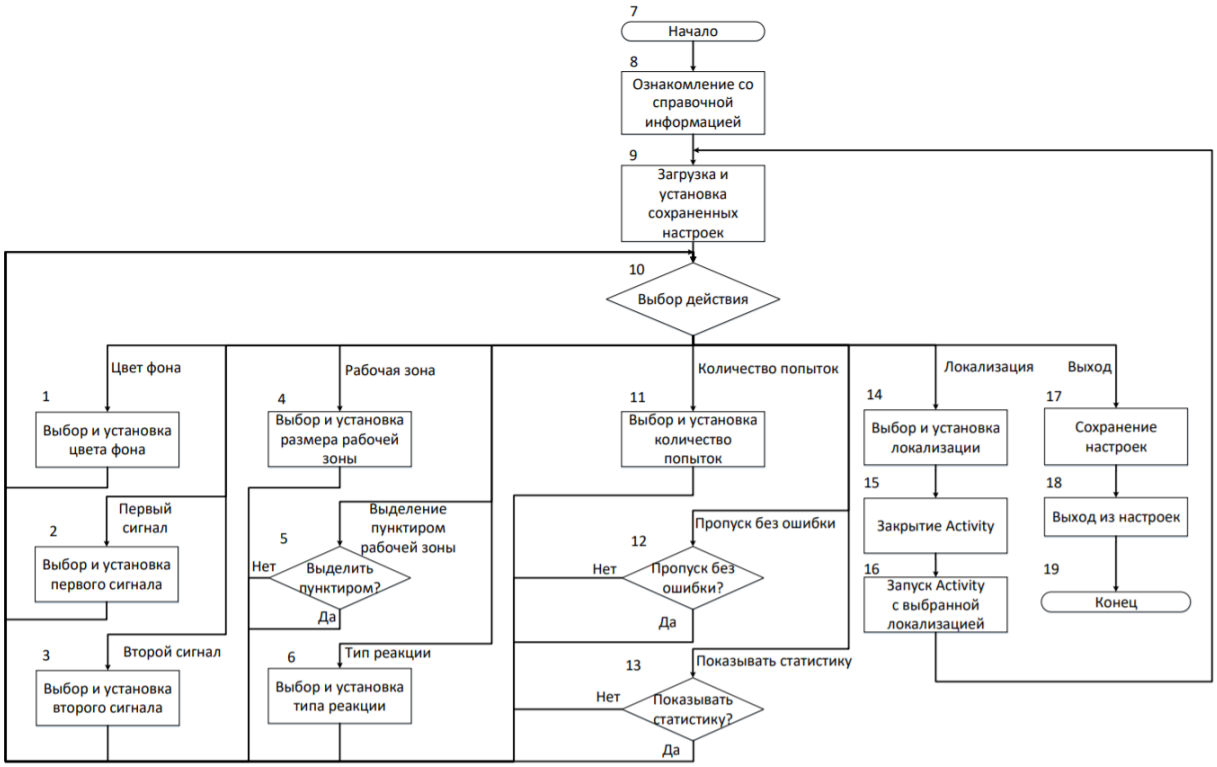

Рисунок 3.5 – Схема алгоритма работы установки настроек

В настройках должно быть указано два поля: первый сигнал, который будет отображен в начале теста, и второй сигнал, который будет появляться во второй половине теста. Если количество попыток больше 0, то пользователь будет выполнять тестирование, иначе, а именно при выборе «Тренировка», он будет практиковаться. Если второй сигнал, то первый сигнал будет исчезать. Если наоборот, то второй сигнал будет появляться.

Далее выполняется выбор размера рабочей зоны, выделение пунктиром и пропуск без ошибки. Также доступна возможность установки показа статистики после выполнения теста.

3.6 Разработка схемы прохождения тестирования

Для того чтобы психотерапевт проанализировал результаты исследования пользователь должен пройти тестирование. Блок-схема алгоритма данного процесса представлена на рисунке 3.6.

Пользователь должен перейти на форму тестирования. В зависимости от выбранных настроек тестирование будет. Пользователь может просмотреть справочную информацию о тестировании, которая включает пояснения по интерфейсу и сути исследования. После успешного прохождения всех тестов пользователь должен ввести свое имя и результаты исследования будут отправлены на сервер. При набранном количестве непройденных тестов большего чем половина общего их количества пользователю предложиться отдохнуть и приступить к исследованию в следующий раз.

Если выбран тип реакции «Касание» суть тестов будет заключаться в нажатии на сигнал после его смены. Если же выбран тип реакции «Отпускание» для начала теста необходимо нажать на сигнал и после удерживать на нем палец, пока сигнал не изменится, затем отпустить палец. В случае преждевременной или пропущенной реакции тест будет засчитан как непройденный и начнется заново.

Если выбран режим тренировки на экране будет изображен таймер отсчитывающий время реакции. Также после начала практики она будет идти непрекращаясь пока пользователь не выйдет в меню, либо же не остановит тестирование.

Если установлены количество тестов, то таймер с экрана будет убран. Также, после начала тестирования оно будет идти пока пользователь не пройдет успешно то количество тестов, что он указал в настройках. Либо же, пока количество непройденных тестов не станет больше половины общего количества тестов. В этом случае тестирование прекратиться и пользователю предложиться пройти исследование в другой раз.

При прохождении тестирования пользователь должен ожидать появления сигнала. Если он отрегагирует до этого момента тест не защитается, а станет непройденным. Для успешного прохождения теста пользователь должен отреагировать, когда появиться сигнал. Если пользователь опаздает и сигнал исчезнет до его реакции, тест также будет засчитан как провальный.

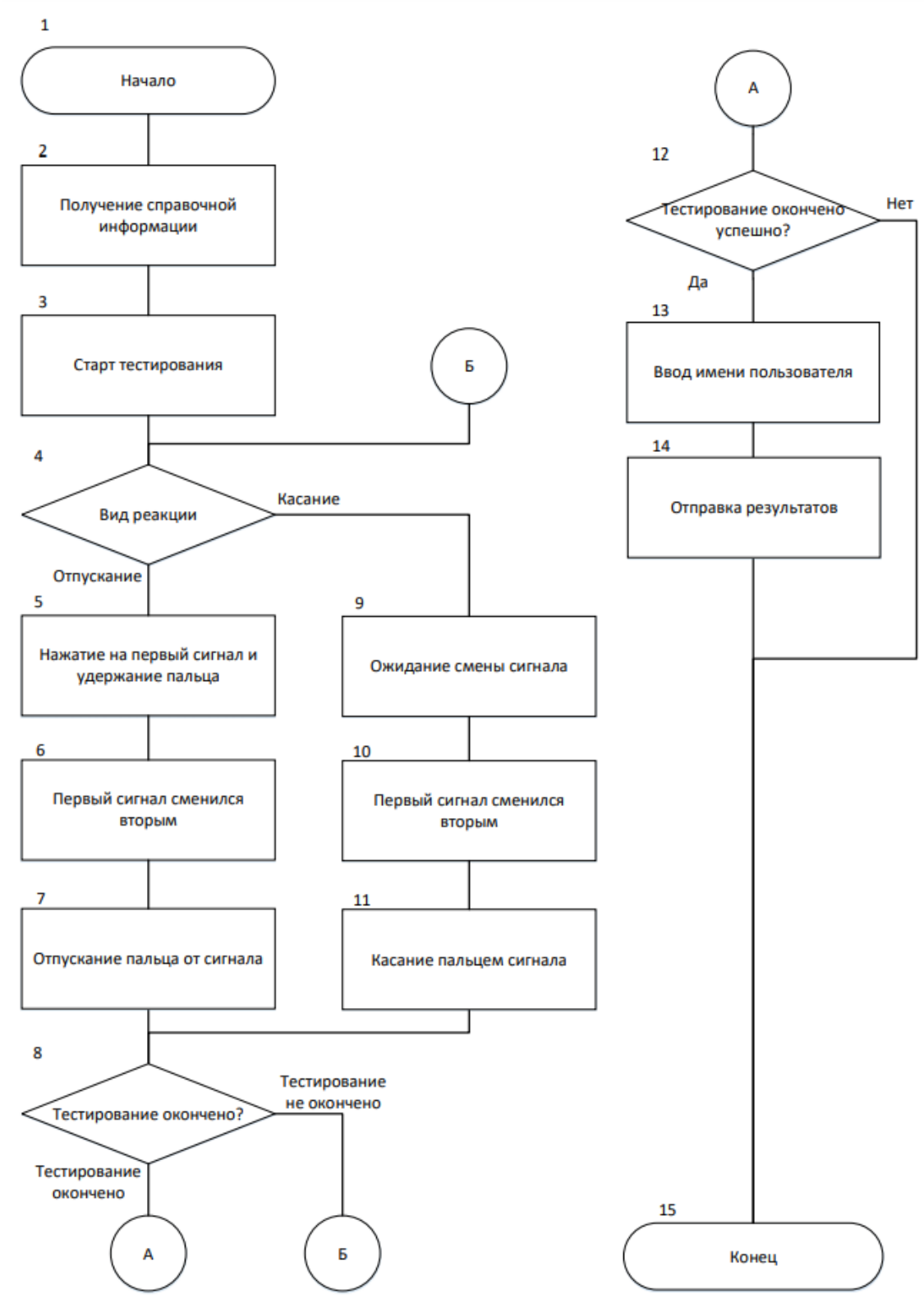

Рисунок 3.6 – Схема алгоритма прохождения тестирования

Если тестирование пройдено успешно и пользователь ввел свое имя, результаты будут отправлены на сервер, после придет уведомление о успешности этой операции, и, если данные успешно отправлены и в настройках не указано иного, приложение перейдет на форму результатов.

4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

Процесс тестирования заключается в составлении списка тестовых случаев с указанием ожидаемых и полученных результатов, которые дадут информацию о качестве продукта.

Тестирование готовой системы проведено на эмуляторе устройства Pixel 3a с версией андроида 4.0 установленном на персональном компьютере с характеристиками:

– процессор – Intel® Core™ i7-2700 CPU 2x3.10GHz;

– оперативная память – 16 ГБ;

– операционная система – Windows 10 Enterprise 64-bit.

Разрабатываемое программное средство предназначено для использования в мобильныхприложениях, что накладывает определенный отпечаток на особенности приложения. Мобильные приложения, запущенные на разных устройствах, могут вести себя неодинаково. Поэтому важно проводить тестирование одновременно на нескольких устройствах.

Для проверки работоспособности разработанного программного средства были составлены тест-кейсы, описание которых представлено в таблице 4.1. Суть тест-кейсов заключается в выполнении некоторого сценария с постоянной проверкой получаемых результатов. Данные тест-кейсы составлены так, что в одном тестовом сценарии может быть проверено сразу несколько компонентов. Успешное выполнение данных тестовых сценариев может гарантировать корректное поведение мобильного приложения.

Таблица 4.1 – Тест-кейсы

№ теста	**Название теста**	**Последовательность шагов**	**Ожидаемый результат**	**Полученный результат**
	Успешная установка настроек.	1.Открыть форму настроек; Внести изменения в настройки; Выйти из формы настроек.	Установка успешна. Происходит автоматическое сохранение настроек при выходи из формы.	Установка настроек в приложении выполнена успешно. Настройки автоматически сохранены и вступили в силу.
	Успешно выполненная тренировка.	1.Открыть форму тренировки; 2.Начать тестирование. Пройти несколько тестов;	Тренировка прошла успешно.	Тренировка в приложении выполнена успешно.

Продолжение таблицы 4.1

№ теста	Название теста	Последовательность шагов	Ожидаемый результат	Полученный результат
		Закончить тестирование; 5.Выйти в главное меню.		
	Просмотр статистики пользователей.	Перейти на форму результатов; Получить уведомления об успешности запроса на сервер; 3.Получить статистику пользователей; 4.Выйти в главное меню.	Система отображает статистику пользователей в которой размещена информация о данных пользователей	Страница результатов пользователей с информацией о них.
	Выполнение тестирования.	Перейти на форму тестирования; Начать тестирование; Успешно пройти тестирование; Ввести имя пользователя; 5.Получить уведомление об успешной отправке данных на сервер; 6.Получить статистику.	Результаты тестирования успешно отправлены на сервер. Система отображает статистику.	Страница статистики с уведомлением о том, что результаты тестирования были успешно отправлены.
	Невыполнение тестирования.	Перейти на форму тестирования; Начать тестирование; Не пройти тестирование; Получить уведомление о провале тестирования;	Отображение сообщения о провале тестирования.	Сообщение о провале тестирования

5 МЕТОДИКА ИСПОЛЬЗОВАНИЯ ПРОГРАММНОГО СРЕДСТВА

5.1 Вход в приложение

Задача данного модуля заключается в информировании пользователей о создателях программного средства, логотипе их компании и версии приложения. Информирующее окно пропадает само через несколько секунд (рисунок 5.1).

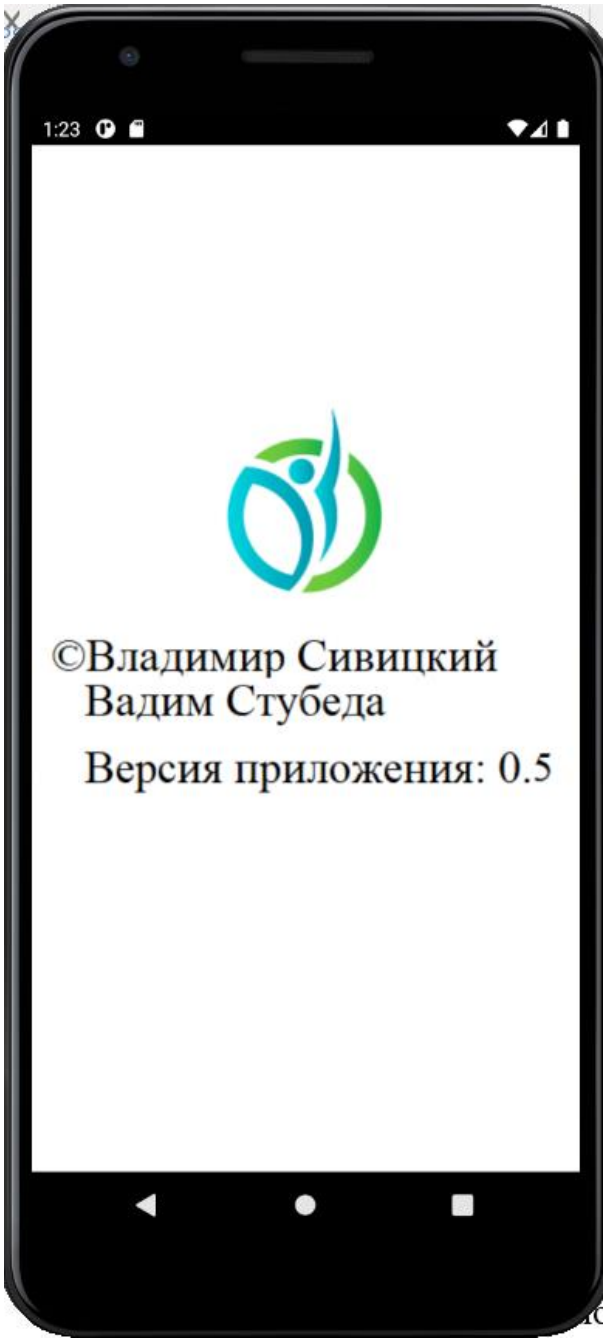

Рисунок 5.1 – Вход в приложение

После некоторого ожидание заглавное окно сменяется главным меню (рисунок 5.2).

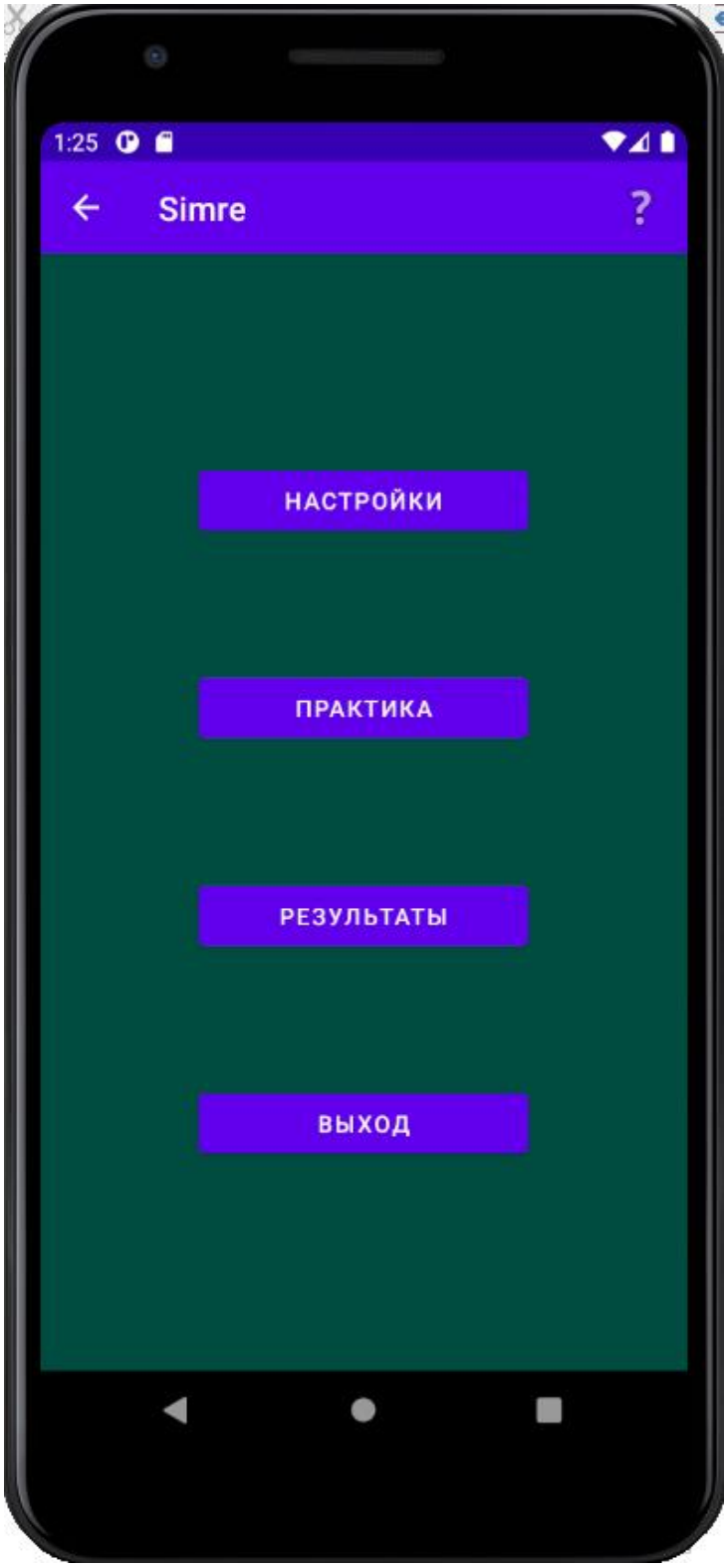

Рисунок 5.2 – Главное меню приложения

В главном меню пользователю доступны следующие разделы «Настройки», «Практика», «Результаты» и «Выход».

5.2 Раздел «Настройки»

В раздел «Настройки» входит конфигурация интерфейса приложения, тип тестирования и прочии опции. Форма раздела представлена на рисунке 5.3.

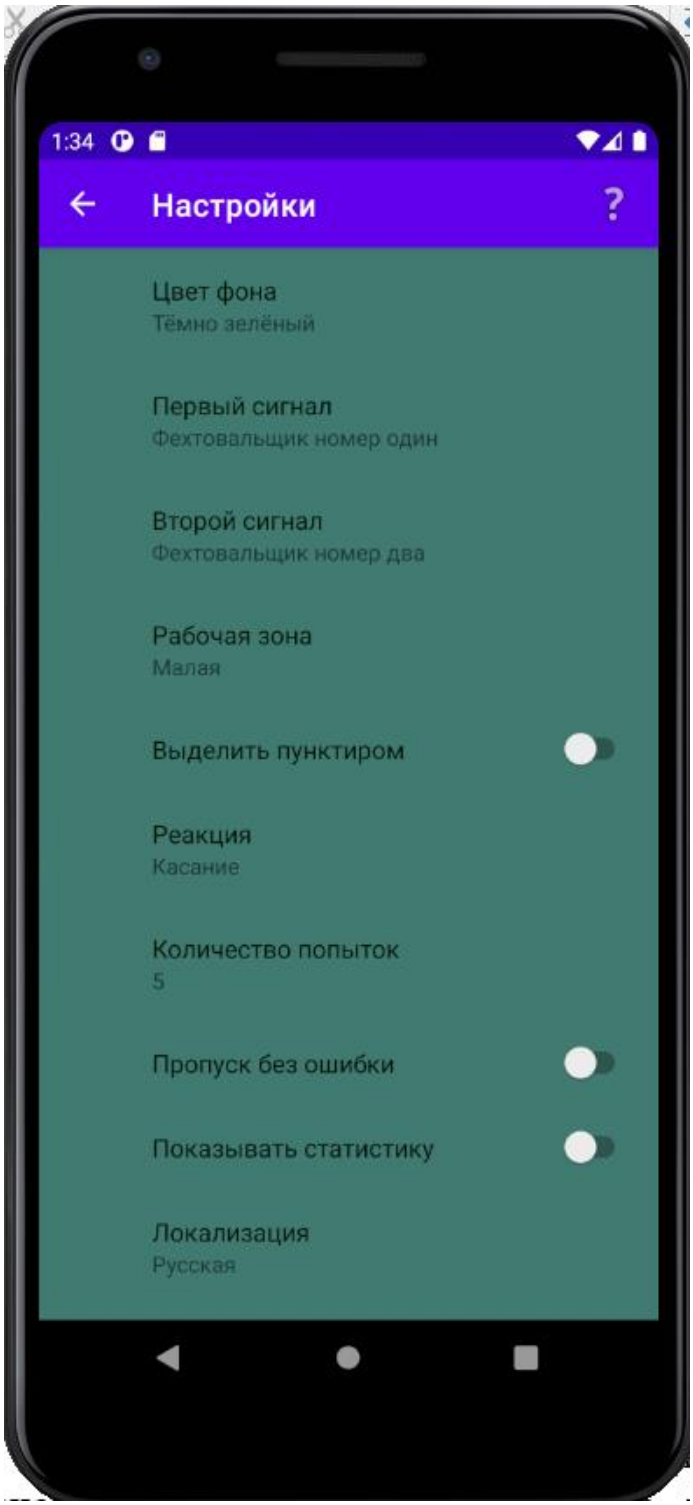

Рисунок 5.3 –Раздел «Настройки»

5.3 Тестирование

Для прохождения тестирования необходимо, чтобы было выбрано количество попыток. Список доступных количеств попыток пользователь может увидеть на форме «Настройки» (рисунок 5.3).

После перехода на форму тестирования (рисунок 5.4) пользователю предоставиться возможность ознакомиться со справочной информацией.

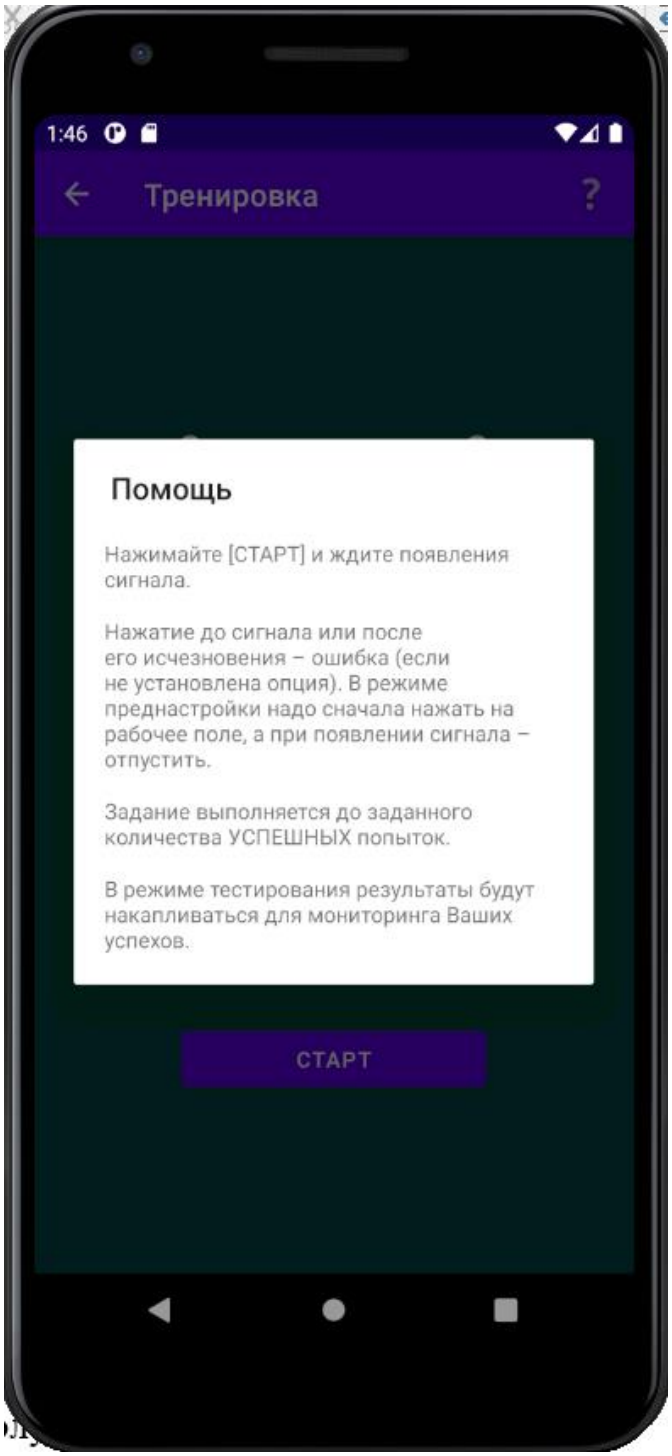

Рисунок 5.4 – Справочная информация в разделе «Практика»

Затем, при нажатии в любое место экрана, подсказка пропадет и на экране отобразится основной интерфейс (рисунок 5.5).

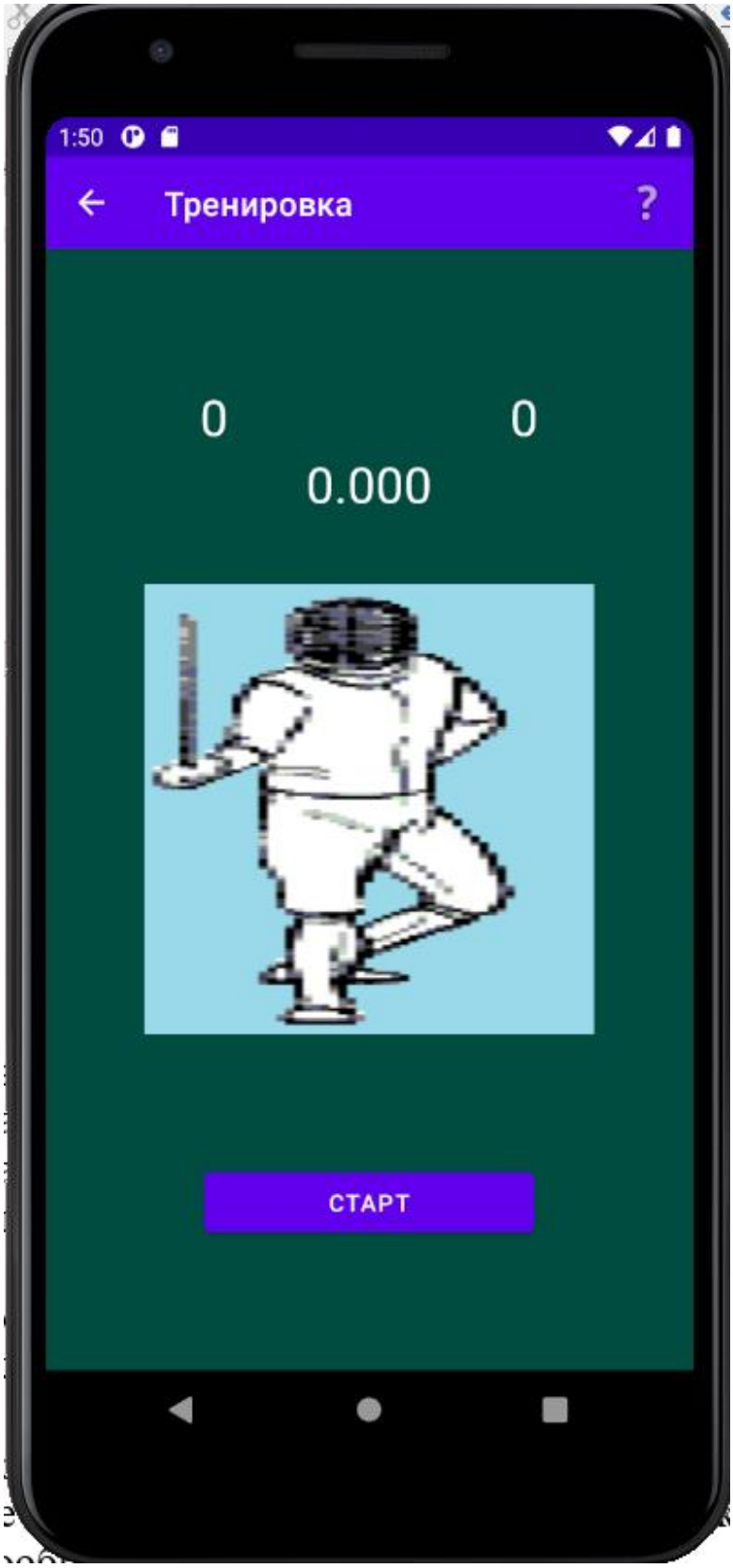

Рисунок 5.5 – Раздел «Практика»

В зависимости от выбранного количества попыток интерфейс раздела «Тестирование» будет незначительно отличаться: если выбран реджим тренировка – в заглавии будет «Тренировка» и будет отображен таймер (рисунок 5.5), если выбрано любое другое количество попыток – в заглавии будет слово «Тест» и выбранное количество удачных тестов (рисунок 5.6).

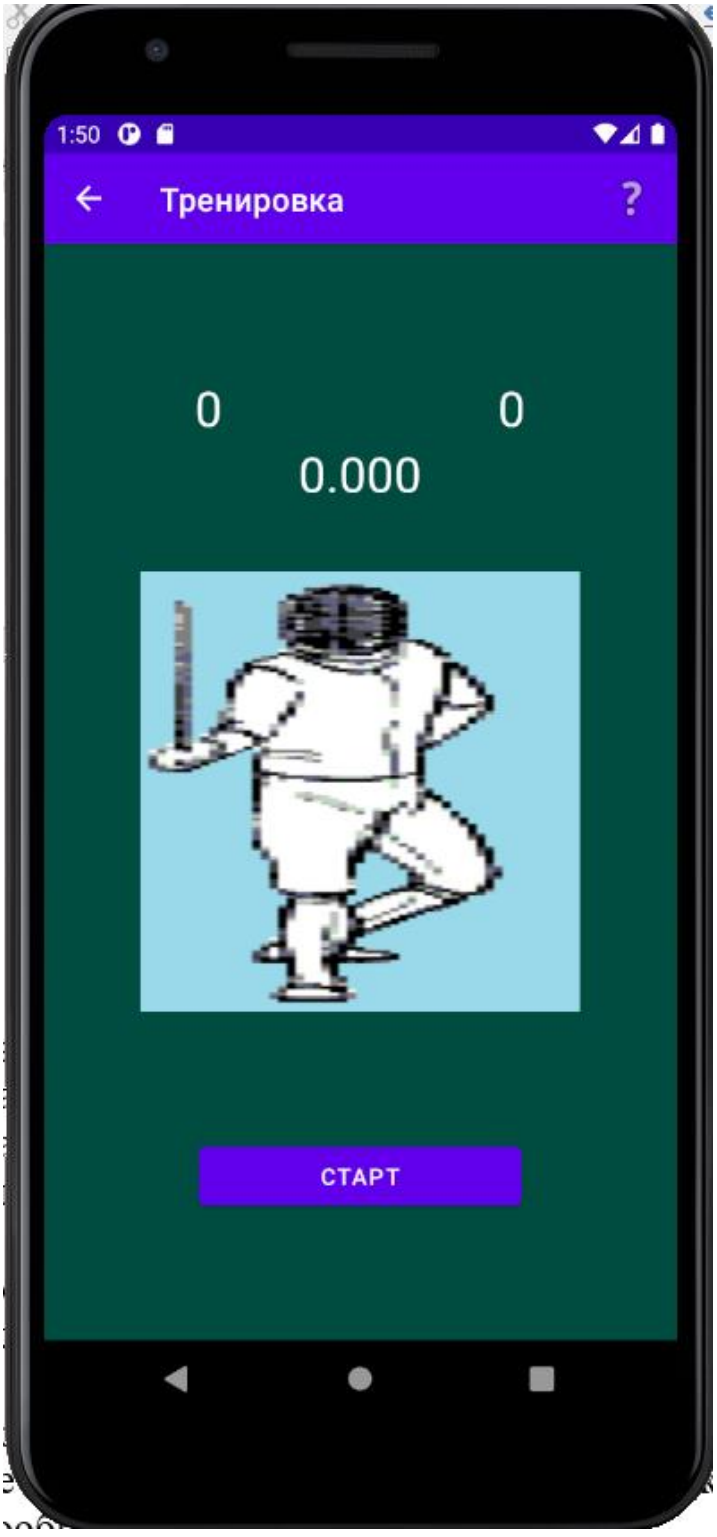

Рисунок 5.6 – Раздел «Тестирование»

После успешного прохождения теста будет отображена форма ввода имени пользователя (рисунок 5.7).

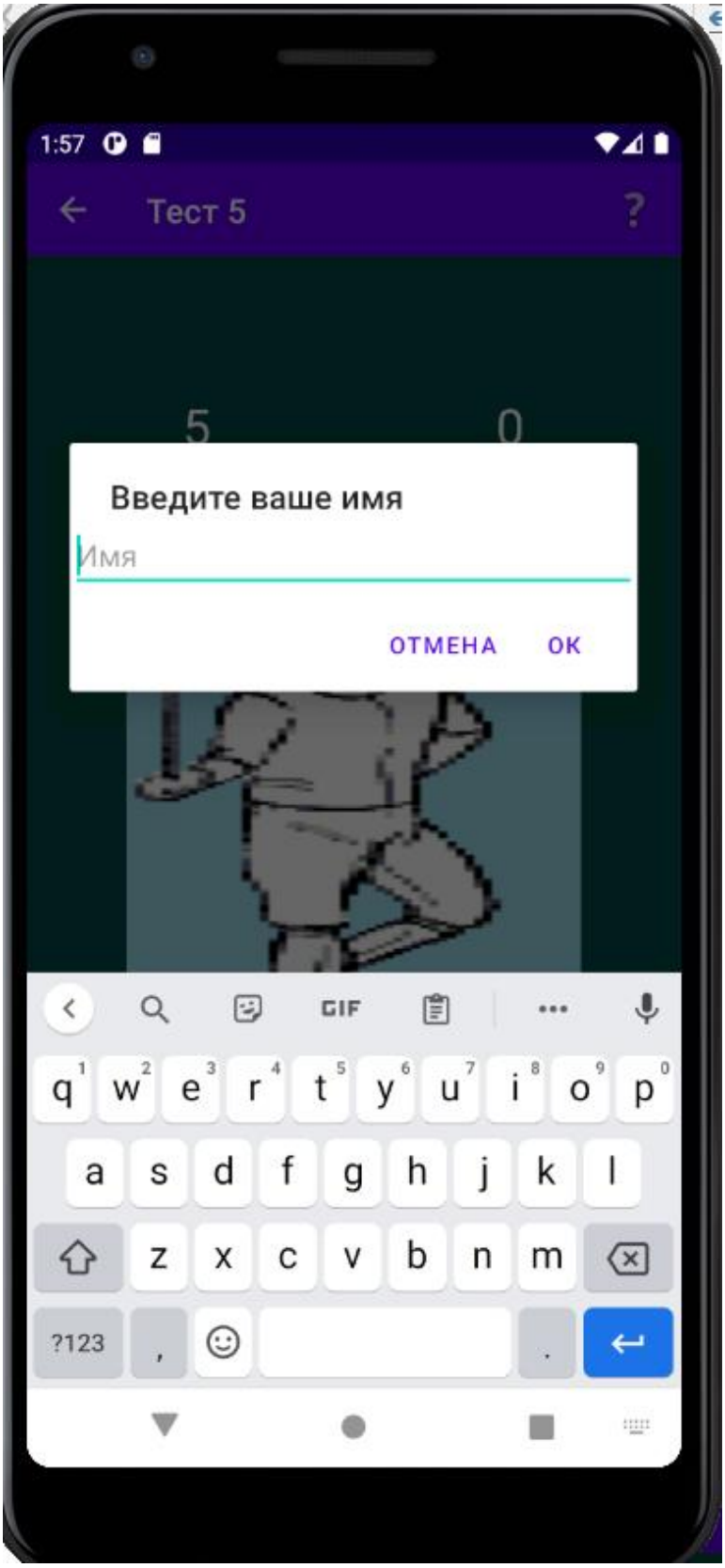

Рисунок 5.7 – Успешное завершение тестирования

Введенное имя вместе с результатами тестирования отправляются на сервер (рисунок 5.8).

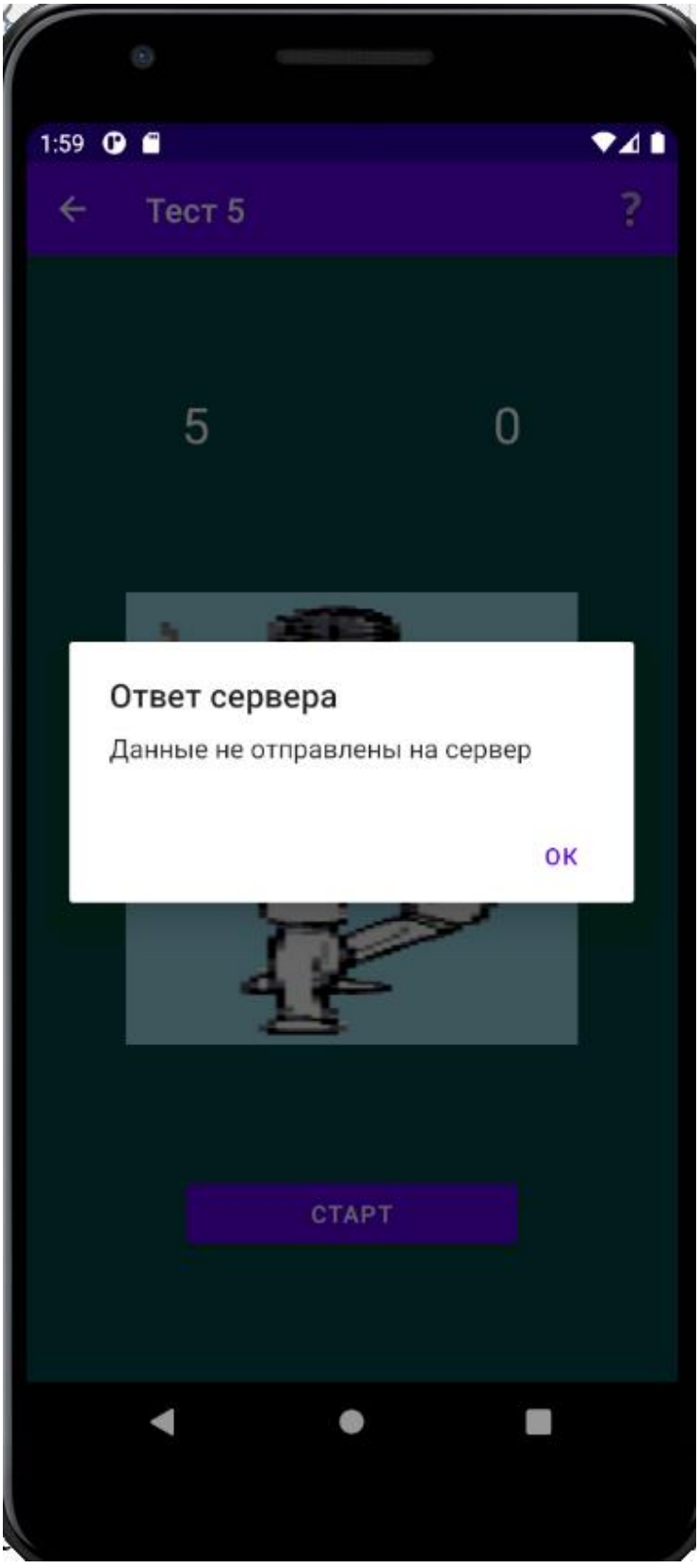

Рисунок 5.8 – Отправка данных на сервер

При успешной отправке данных открывается раздел статистики (рисунок 5.9).

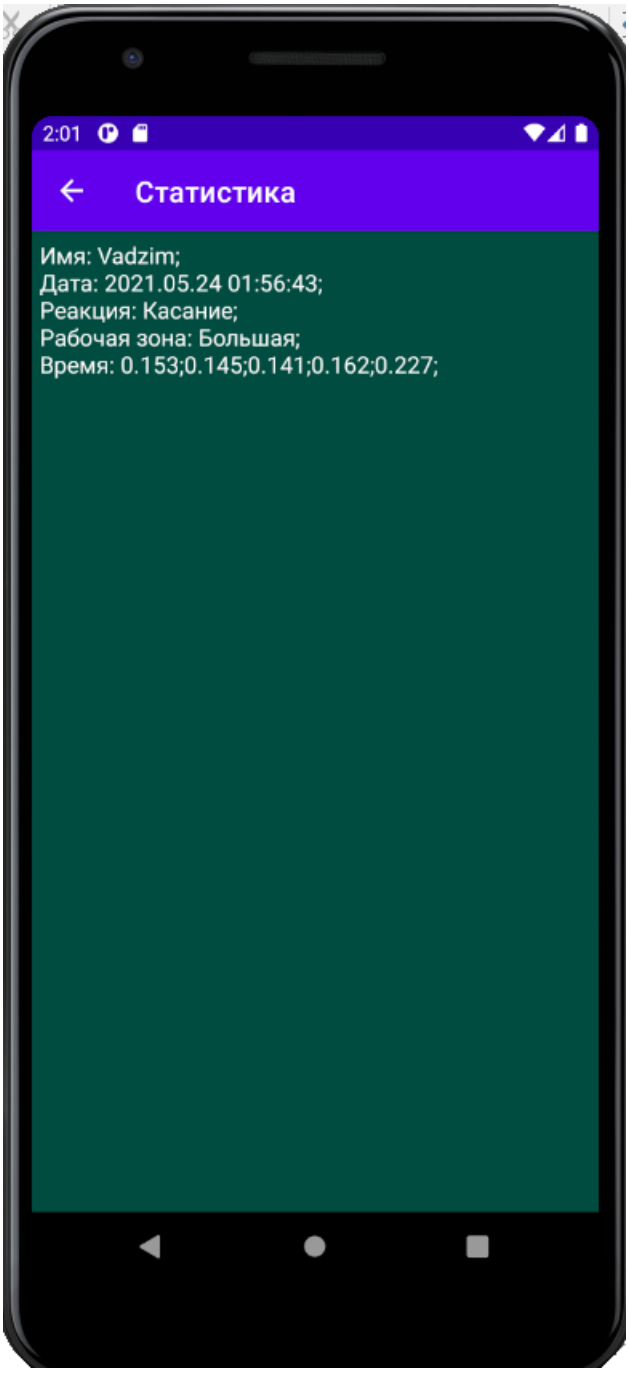

Рисунок 5.9 – Раздел «Результаты»

5.4 Раздел «Результаты»

Раздел «Результаты» начинается с загрузки данных сервера и последующего уведомления о успешности этой операции. Затем будет оображена статистика пользователей (рисунок 5.9).

6 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ И ВНЕДРЕНИЯ ПРОГРАММНОГО СРЕДСТВА

6.1 Описание функций, назначения и потенциальных пользователей ПО

Разрабатываемое приложение предназначено в первую очередь для использования клиентами научно-практического образовательного центра SportConsult [19] и последующего распространения разработки для проживающих в странах СНГ и странах постсоветского пространства.

Предоставляет возможность создания неограниченного числа аккаунтов, тестирования пользователей различными исследованиями простых реакция, а также возможность получения потребителем последующих рекомендаций и корректировок касательно его жизнедеятельности.

По данным [20] торговой площадки “Play Маркет”, число установок “CogniFit логические игры и задачи” на 20 апреля 2021 года превышает 100 тыс. Число установок другого популярного сервиса “Меморадо: Тренировка памяти” в ту же дату составило более 1 млн. Таким образом, можно с уверенностью говорить, что потенциальная целевая аудитория разрабатываемого приложения достаточно обширна.

6.2 Расчет затрат на разработку ПО

6.2.1 Расчет затрат на основную заработную плату разработчиков

Затраты на основную заработную плату команды разработчиков определяются исходя из состава и численности команды, размеров месячной заработной платы каждого из участников команды, а также общей трудоемкости разработки программного обеспечения.

Расчет основной заработной платы участников команды осуществляется по формуле

$$З_0 = К_{пр} \sum_{i=1}^{n} З_{ч.i} t_i, \quad (6.1)$$

где n – количество исполнителей, занятых разработкой конкретного ПО; $К_{пр}$ – коэффициент, учитывающий процент премий; $З_{ч.i}$ – часовая заработная плата i-го исполнителя, р.; t_i – трудоемкость работ, выполняемых i-м исполнителем, ч.

Месячная заработная плата была определена путем расчета средней заработной платы по конкретной профессии за последние три месяца по данным из открытых источников [21].

Часовая заработная плата была получена путем деления месячной заработной платы на количество рабочих часов в месяце (было принято равным 168 ч).

Расчет затрат на основную заработную плату представлен в таблице 6.1.

Таблица 6.1 – Расчет затрат на основную заработную плату разработчиков

Наименование должности разработчика	Вид выполняемой работы	Месячная заработная плата, р.	Часовая заработная плата, р.	Трудоемкость работ, ч	Зарплата по тарифу, р.
Бизнес-аналитик	- постановка задачи; - сбор исходных материалов; - выбор и обоснование критериев эффективности и качества разрабатываемой программы; - определение требований к программе; - определение стадий, этапов и сроков разработки программы и документации на нее.	1221	7,27	134	974
Системный архитектор	- определение структуры входных и выходных данных; - выбор методов решения задачи; - выбор средств программирования; - разработка алгоритма решения задачи, структуры компонентов, включая внешние интерфейсы, разработка пояснительной записки; - определение конфигурации технических средств, разработка плана мероприятий по разработке и внедрению программ.	907	5,4	904	4882

Продолжение таблицы 6.1

Наименование должности разработчика	Вид выполняемой работы	Месячная заработная плата, р.	Часовая заработная плата, р.	Трудоемкость работ, ч	Зарплата по тарифу, р.
	- определение формы представления входных и выходных данных; - уточнение логической структуры внешних интерфейсов; - разработка структуры программы, уточнение структуры компонентов на уровне программных модулей; - программирование и отладка программы; - изготовление программы-оригинала; - разработка программных документов в соответствии с требованиями; - корректировка программы и программной документации по результатам испытаний.				
Специалист по тестированию программного обеспечения	- определение требований к тестированию программных модулей; - разработка, согласование и утверждение порядка и методики тестирования; - проведение тестирования программных модулей, базы данных; - проведение приемосдаточных испытаний программы.	625	3,72	416	1548

Продолжение таблицы 6.1

Итого, р.	14597
Премии(30%), р.	4379
Всего основная заработная плата разработчиков, р.	18976

Размер премии взят условно равным 30% от размера основной заработной платы.

6.2.2 Расчет затрат на дополнительную заработную плату разработчиков

Затраты на дополнительную заработную плату разработчиков включают выплаты, предусмотренные законодательством о труде (оплата трудовых отпусков, льготных часов, времени выполнения государственных обязанностей и других выплат, не связанных с основной деятельностью исполнителей), и определяется по формуле

$$З_д = \frac{З_о \cdot Н_д}{100\%}, \quad (6.2)$$

где $З_о$ – затраты на основную заработную плату, р.; $Н_д$ – норматив дополнительной заработной платы, принятый в размере 20%.

$$З_д = \frac{18967 \cdot 20}{100} = 3793р.$$

6.2.3 Расчет отчислений на социальные нужды

Отчисления на социальные нужды (в фонд социальной защиты населения и на обязательное страхование) определяются в соответствии с действующими законодательными актами по формуле

$$Р_{соц} = \frac{(З_о + З_д) \cdot Н_{соц}}{100\%}, \quad (6.3)$$

где $Н_{соц}$ – норматив отчислений на социальные нужды (34% в ФСЗН и 0,6% на обязательное страхование)

$$Р_{соц} = \frac{(18976 + 3793) \cdot 34{,}6}{100\%} = 7875р.$$

6.2.4 Расчет прочих затрат

Прочие затраты включаются в себестоимость разработки ПО в процентах от затрат на основную заработную плату разработчиков по формуле

$$Р_{пз} = \frac{З_о \cdot Н_{пз}}{100\%}, \quad (6.4)$$

где $Н_{пз}$ – норматив прочих затрат, принятый равным 120%.

$$Р_{пз} = \frac{18976 \cdot 120}{100\%} = 22771р.$$

Полная сумма затрат на разработку программного обеспечения находится путем суммирования всех рассчитанных статей затрат (Таблица 6.2)

Таблица 6.2 – Затраты на разработку программного обеспечения

Наименование статьи затрат	Сумма, р.
1. Основная заработная плата разработчиков	18976
2. Дополнительная заработная плата разработчиков	3793
3. Отчисления на социальные нужды	7875
4. Прочие затраты	22771
Общая сумма затрат на разработку	53415

6.3 Оценка эффекта от продажи ПО

Экономический эффект от разрабатываемого приложения представляет собой прибыль его продажи множеству потребителей.

На основе имеющихся данных о количестве установок приложения “CogniFit логические игры и задачи” можно оценить ожидаемое количество копий ПО, которое будет приобретено пользователями в $N = 100000$.

Средняя стоимость подписки на 2 наиболее популярных [24] веб-сервисов диагностики психического состояния человека по состоянию на 20 апреля 2021 согласно информации, на официальных сайтах [22, 23] данных сервисов составляет 146,1р. в эквиваленте по курсу национального банка на ту же дату. Минимальная стоимость при этом равна 56,85р. в эквиваленте. В целях повышения конкурентоспособности приложения с гигантами индустрии, при неимении кроссплатформенности и большого количества дополнительных функций и при направленности на более широкую аудиторию, по сравнению с аналогами на первом этапе цену за единицу ПО можно принять равной

$$Ц = 1{,}99р$$

Столь высокое занижение цены объяснено также новизной компании соучредителей на рынке и потенциального недоверия со стороны пользователей. В будущем возможно повышение цены.

Поскольку предприятие является резидентом Парка высоких технологий, оно освобождено от уплаты НДС и налога на прибыль. Тогда чистая прибыль, полученная от реализации ПО на рынке, рассчитанная по формуле

$$П = Ц \cdot N - З_р \quad (6.5)$$

составит

$$П = 0{,}99 \cdot 100000 - 53415 = 45585.$$

6.4 Расчет показателей эффективности затрат в разработку ПО

Для оценки эффективности затрат в разработку ПО рассчитаем уровень рентабельности затрат по формуле

$$У_р = \frac{П}{З_р} \cdot 100\%, \quad (6.6)$$

Подставив рассчитанные ранее значения в формулу 6.6, получим

$$У_р = \frac{145585}{53415} \cdot 100\% = 272{,}55\%$$

Поскольку полученное значение намного превышает среднюю за последние три месяца процентную ставку по банковским депозитам для юридических лиц, равную 11,61%, проект может считаться экономически эффективным.

6.5 Вывод

Таким образом, в результате расчетов были получены следующие значения показателей экономической эффективности затрат в разработку ПО: чистая прибыль равна 145585 рублей и величина рентабельности затрат равна 272,55 %.

Годовая сумма продаж разработанного проекта значительно превышает все затраты на разработку, к тому же, уровень рентабельности намного превышает среднюю за последние три месяца процентную ставку по банковским депозитам для юридических лиц, равную 11,61%, следовательно, проект может считаться экономически эффективным.

ЗАКЛЮЧЕНИЕ

В ходе разработки программного средства был проведен анализ предметной области, в том числе были проанализированы существующие аналоги программного средства диагностики психического состояния человека с использованием клиент-серверной технологии. По результатам данного анализа были выявлены недостатки существующих прототипов, в устранении которых было решено отталкиваться в разрабатываемом программном средстве.

В программном средстве были реализованы функции тестирования, практики пользователей, а также взаимодействие с сервером и ведение статистики. Были предусмотрены различные роли пользователей, в том числе рядовой пользователь или психотерапевт.

Для реализации были выбраны оптимальные технологии для использования в ходе разработки, такие как язык и платформа разработки (Java и Spring Boot соответственно), способ хранения данных (MySQL).

Была смоделирована информационная модель разрабатываемого программного средства, а также разработана структура базы данных.

Было произведено проектирование программного средства, разработана программная архитектура, схема развертывания программного средства, диаграмма компонентов, а также разработаны алгоритмы основных функций программного средства, в том числе общий алгоритм работы приложения, алгоритм прохождения тестирования.

Было произведено тестирование программного средства, которое показало полное соответствие разработанного программного средства спецификации требований.

В завершение, были разработаны методики использования программного средства для пользователя.

Было рассмотрено технико-экономическое обоснование дипломного проекта, которое показало, что разработка экономически выгодна.

Итогом выполнения дипломного проекта является программное средство диагностики психического состояния человека с использованием клиент серверной технологии, которое полностью соответствует спецификации требований к программному средству. Данное программное средство может быть взята на вооружение в любой компании со средним или большим количеством сотрудников.

Нет предела совершенству, поэтому программное средство будет доработано. В новой версии планируется добавить обработкурезультатов тестирования на стороне клиента, возможность ведения статистики оффлайн и реализовать рассылку рекомендаций по электронной почте.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Бойко, Е. И. Время реакции человека / Е. И. Бойко – М. : Медицина, 1964. – 440с.

[2] Бовин, Б. Г. Нейрофизиологическая модель многоальтернативного выбора. Психофизиологические закономерности восприятия и памяти / Б. Г. Бовин – М. : Наука, 1985. – С. 55-86.

[3] Electrophysiological evidence of a perceptual precedence of global vs. local visual information / Cogn. Brain Research. – A. M. Proverbio, A. Minniti, A. Zani, 1998. V. 6. – P. 321–334.

[4] Лупандин, В. Я. Асимметрия распределения времени простой сенсомоторной реакции. Физиология человека. / В. Я. Лупандин, О. Е. Сурнина. 1988. – Т. 14. – №4. – С. 700–702.

[5] Лоскутова, Т. Д. Оценка функционального состояния центральной нервной системы человека по параметрам простой двигательной реакции / Т. Д. Лоскутова // Физиолог. СССР. 1975. – Т. – LXI. – № 1. – 3с.

[6] Киселев, С. Ю. Компьютерные методики изучения времени сенсомоторных реакций у детей дошкольного возраста. / С. Ю. Киселев, Ф. В. Гизуллина, В. А. Сурнин. // Журн. высш. нервн. деят. 1996. – Т. 46 – вып. 1. – С. 188–189.

[7] Зайцев, А. В. Возрастная динамика времени реакции на зрительные стимулы / А. В. Зайцев, В. Я. Лупандин, О. Е. Сурчина // Физиология человека. 1999. – Т. 25. – № 6. – С. 34–37.

[8] Sternberg S. The discovery of processing stages: extensions of Donders method / Acta Psychol. 1969. – V. 30. – P. 276–315.

[9] Зайцев, А. В. Оценка биологического возраста методом регистрации времени реакции. / А. В. Зайцев, В. Я. Лупандин, О.Е. Сурнина // Экология образования: актуальные проблемы. Архангельск: Изд-во Поморского госун-та. 1999. – С. 45–48.

[10] Зайцев, А. В. Диагностика задержки психического развития детей методом регистрации времени реакции. / А. В. Зайцев, В. Я. Лупандин // Медицинская техника. 2000 – №6.

[11] Зайцев, А. В. Возрастная динамика компонентного состава времени зрительно-моторных реакций. / А. В. Зайцев, В. Я. Лупандин // Север. Дети. Школа: Сб. науч. трудов. Под ред. А.В. Грибанова, Т.В. Волокитиной. Архангельск: Поморский госуниверситет, 2001. – Вып. 3. – С. 76–82.

[12] Лупандин, В. Я Исследование временных параметров когнитивных процессов путем измерения времени сенсомоторных реакций.

XXX Всеросс. совещание по проблемам высш. нервн. деят., посвящ. 150-летию И.П Павлова / А. В. Зайцев, Я. Я. Корякова [и р.]. – Тез. докл. в 2-х т. СПб. : Ин-т физиологии им. И. П. Павлова РАН, 2000. – Т. 1. – С. 257–259.

[13] Иваницкий, А. М. Информационные процессы мозга и психическая деятельность. / А. М. Иваницкий, В. Б. Стрелец, Я. А. Корсаков – М. : Наука, 1984. – 200с.

[14] Segregation of Form, Color, Movement and Depth. Anatomy, Physiology, and Perception. / Livingstone M., Rubel D. // Science. 1988. – V. 240. – P. 740–749.

[15] Глезер, В. Д. Зрение и мышление. – СПб.: Наука, 1993. – 284 с.

[16] «Where» and «What» in Vision. / Sagi D., Julesz B. // Science. 1985. – V. 228 – № 4704. – P. 1217–1219.

[17] Standardized memory scale for clinical use. The Journal of Psychology: Interdisciplinary and Applied / Wechsler, D. A, 1945. – P. 87–95.

[18] Manual for Conners' rating scales. North Tonawanda / Conners, C. K. – NY : Multi-Health Systems. 1989.

[19] Научно-практический образовательный центр SportConsult [Электронный ресурс]. – Режим доступа : https://sportconsult.by/. – Дата доступа: 20.04.2021.

[20] Play Маркет [Электронный ресурс]. – Режим доступа : https://play.google.com/. – Дата доступа: 20.04.2021.

[21] Trud.com [Электронный ресурс]. – Режим доступа : https://by.trud.com/salary/304581/3299.html. – Дата доступа: 12.04.2021.

[22] CogniFit, Play Маркет [Электронный ресурс]. – Режим доступа : https://www.cognifit.com/ru/pricing. – Дата доступа: 20.04.2021.

[23] Меморадо, Play Маркет [Электронный ресурс]. – Режим доступа : https://play.google.com/store/apps/details?id=com.memorado.brain.games&hl=ru&gl=US. – Дата доступа: 20.04.2021.

[24] Динамика ставок кредитно-депозитного рынка, Nbrb.by [Электронные данные]. – Режим доступа : https://www.nbrb.by/statistics/CreditDepositMarketRates. – Дата доступа: 12.04.2021.

ПРИЛОЖЕНИЕ А
(обязательное)
Исходный код

```
package com.stubeda.dips;

import android.content.Intent;
import android.content.SharedPreferences;
import android.content.res.Configuration;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.TextView;

import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import java.util.Locale;

public class MainActivity extends AppCompatActivity {
    private String curLocale;
    private SharedPreferences Settings;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        Settings = PreferenceManager.getDefaultSharedPreferences(getBaseContext());
        curLocale = Settings.getString("locale", "ru");

        Locale locale = new Locale(Settings.getString("locale", "ru"));
        Locale.setDefault(locale);
        Configuration config = getBaseContext().getResources().getConfiguration();
        config.locale = locale;
        getBaseContext().getResources().updateConfiguration(config,
                getBaseContext().getResources().getDisplayMetrics());

        setContentView(R.layout.activity_main);

        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null) {
            actionBar.setDisplayHomeAsUpEnabled(true);
            actionBar.setTitle(R.string.simre);
        }
    }

    public boolean onOptionsItemSelected(MenuItem menuItem) {
        finish();
        startActivity(new Intent(this, SplashActivity.class));
        return true;
```

```
}

public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.options_menu, menu);
    return true;
}

public void onHelpItemClick(MenuItem item) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    LayoutInflater inflater = getLayoutInflater();

    View customView = inflater.inflate(R.layout.help_dialog, null);
    TextView messageView1 = customView.findViewById(R.id.txtHelp1);
    TextView messageView2 = customView.findViewById(R.id.txtHelp2);
    TextView messageView3 = customView.findViewById(R.id.txtHelp3);
    TextView messageView4 = customView.findViewById(R.id.txtHelp4);

    builder.setView(customView)
            .setTitle(R.string.help);

    messageView1.setText(R.string.help_main1);
    messageView2.setText(R.string.help_main2);
    messageView3.setText(R.string.help_main3);
    messageView4.setText(R.string.help_main4);

    AlertDialog alert = builder.create();

    alert.show();
}

public void onClickExit(View view) {
    this.finish();
    System.exit(0);
}

public void onClickSetting(View view) {
    this.startActivity(new Intent(this, SettingsActivity.class));
}

public void onClickTrain(View view) {
    this.startActivity(new Intent(this, TestingActivity.class));
}
public void onClickResults(View view) {
    this.startActivity(new Intent(this, FastStatsActivity.class));
}

@Override
protected void onResume() {
    super.onResume();

    Settings = PreferenceManager.getDefaultSharedPreferences(getBaseContext());

    if (!curLocale.equals(Settings.getString("locale", "ru"))) {
        Intent intent = getIntent();
        finish();
```

```
            startActivity(intent);
        }

        Locale locale = new Locale(Settings.getString("locale", "ru"));
        Locale.setDefault(locale);
        Configuration config = getBaseContext().getResources().getConfiguration();
        config.locale = locale;
        getBaseContext().getResources().updateConfiguration(config,
                getBaseContext().getResources().getDisplayMetrics());


    }
}


package com.stubeda.dips;

import android.content.Intent;
import android.content.SharedPreferences;
import android.content.res.Configuration;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.TextView;

import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.Preference;
import androidx.preference.PreferenceFragmentCompat;
import androidx.preference.PreferenceManager;

import java.util.Locale;

public class SettingsActivity extends AppCompatActivity {
    private SettingsFragment curSettingsFragment;
    private SharedPreferences Settings;
    private String curLocale;

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

        if (savedInstanceState == null) {
            getSupportFragmentManager()
                    .beginTransaction()
                    .replace(R.id.ltSettings, curSettingsFragment = new
SettingsFragment())
                    .commit();
        }
```

```
        Settings = PreferenceManager.getDefaultSharedPreferences(getBaseContext());
        curLocale = Settings.getString("locale", "ru");



        Locale locale = new Locale(Settings.getString("locale", "ru"));
        Locale.setDefault(locale);
        Configuration config = getResources().getConfiguration();
        config.locale = locale;
        getResources().updateConfiguration(config,
getResources().getDisplayMetrics());

        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null) {
            actionBar.setDisplayHomeAsUpEnabled(true);
            actionBar.setTitle(R.string.setting);
        }
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.options_menu, menu);
        return true;
    }

    public void onHelpItemClick(MenuItem item) {
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        LayoutInflater inflater = getLayoutInflater();

        View customView = inflater.inflate(R.layout.help_dialog, null);
        TextView messageView1 = (TextView)customView.findViewById(R.id.txtHelp1);
        TextView messageView2 = (TextView)customView.findViewById(R.id.txtHelp2);
        TextView messageView3 = (TextView)customView.findViewById(R.id.txtHelp3);
        TextView messageView4 = (TextView)customView.findViewById(R.id.txtHelp4);

        builder.setView(customView)
                .setTitle(R.string.help);

        messageView1.setText(R.string.help_setting1);
        messageView2.setText(R.string.help_setting2);
        messageView3.setText(R.string.help_setting3);
        messageView4.setText(R.string.help_setting4);

        AlertDialog alert = builder.create();

        alert.show();
    }

    public boolean onOptionsItemSelected(MenuItem item){
        finish();
        return true;
    }

    public static class SettingsFragment extends PreferenceFragmentCompat {
        @Override
```

```
        public void onCreatePreferences(Bundle savedInstanceState, String rootKey) {
            setPreferencesFromResource(R.xml.root_preferences, rootKey);
        }
    }

    @Override
    protected void onResume() {
        super.onResume();

        Preference curPreference = curSettingsFragment.findPreference("locale");

        curPreference.setOnPreferenceChangeListener((preference, newValue) -> {
            if (!curLocale.equals(newValue.toString())) {
                Intent intent = getIntent();
                finish();
                startActivity(intent);
            }

            return true;
        });
    }
}
```

```
package com.stubeda.dips;

import android.content.Intent;
import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class SplashActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        try {
            Thread.sleep(1500);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        startActivity(new Intent(this, MainActivity.class));
        this.finish();
    }
}
```

```
package com.stubeda.dips;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.res.Configuration;
import android.graphics.Color;
```

```
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.SystemClock;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;
import java.util.Random;

public class TestingActivity extends AppCompatActivity {

    @SuppressLint("StaticFieldLeak")
    private class UserStatsPOSTTask extends AsyncTask<String, String, String> {

        String answerHTTP;
        String name, date, reaction, work_zone, time;
        JSONObject postDataParams;

        String serverURL = "http://10.0.2.2:8080/api/v1/users";

        @Override
        protected void onPreExecute() {
            name = curName;
            date = curDate;
            reaction = curReaction;
            work_zone = curWorkZone;
            time = curTimeOfAttempts;

            super.onPreExecute();
        }
```

```
        @Override
        protected String doInBackground(String... params) {
            postDataParams = new JSONObject();

            try {
                postDataParams.put("name", name);
                postDataParams.put("date", date);
                postDataParams.put("reaction", reaction);
                postDataParams.put("work_zone", work_zone);
                postDataParams.put("time", time);
            } catch (JSONException e) {
                e.printStackTrace();
            }

            answerHTTP = performPostCall(serverURL, postDataParams);

            return null;
        }

        @Override
        protected void onPostExecute(String result) {
            AlertDialog.Builder builder = new AlertDialog.Builder(curActivity);
            builder.setTitle(R.string.server_answer)
                    .setMessage(answerHTTP.equals("") ?
R.string.data_not_sent_to_server : R.string.data_sent_to_server)
                    .setPositiveButton(R.string.ok, (dialog, id) -> {
                        dialog.cancel();

                        if (Settings.getBoolean("show_statistics", true)) {
                            Intent intent = new Intent(curActivity,
FastStatsActivity.class);
                            startActivity(intent);
                        }
                    });

            AlertDialog alert = builder.create();

            alert.show();

            super.onPostExecute(result);
        }

        public String performPostCall(String requestURL,
                                      JSONObject postDataParams) {
            URL url;
            StringBuilder response = new StringBuilder();
            try {
                url = new URL(requestURL);

                HttpURLConnection conn = (HttpURLConnection) url.openConnection();
                conn.setReadTimeout(200);
                conn.setConnectTimeout(200);
                conn.setRequestMethod("POST");
                conn.setDoInput(true);
                conn.setDoOutput(true);
```

```
                conn.addRequestProperty("Content-Type", "application/json");

                OutputStream os = conn.getOutputStream();
                os.write(postDataParams.toString().getBytes());
                os.close();

                int responseCode = conn.getResponseCode();

                if (responseCode == HttpURLConnection.HTTP_OK) {
                    String line;
                    BufferedReader br = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
                    while ((line = br.readLine()) != null) {
                        response.append(line);
                    }
                } else {
                    response = new StringBuilder();

                }
            } catch (Exception e) {
                e.printStackTrace();
            }

            return response.toString();
        }
    }

    private int curStopwatch = 0;
    private int curCountRight = 0;
    private int curCountFall = 0;
    private int curSecondWaveform;
    private int curFirstWaveform;
    private int curLimitingAttempts;
    private String curName = "";
    private String curDate = "";
    private String curReaction = "";
    private String curWorkZone = "";
    private String curTimeOfAttempts = "";
    private final Random curRandom = new Random();
    private boolean flgTooEarlyMistake = false;
    private boolean flgTooLateMistake = false;
    private boolean flgTestingInProgress = false;
    private boolean flgPreparationForTheTest = false;
    private boolean flgTestInProgress = false;
    private boolean flgPassWithoutError = false;
    private boolean flgReactionClickHold;

    SharedPreferences Settings;

    private View ltLed;
    private Button btnLed;
    private Button btnStartFinish;
    private TextView txtTimer;
    private TextView txtCounterRight;
    private TextView txtCounterFall;
```

```
    private final UserStatsPOSTTask userStatsPOSTTask = new UserStatsPOSTTask();
    private Thread Test;
    private final Activity curActivity = this;

    @SuppressLint({"ClickableViewAccessibility", "SimpleDateFormat"})
    protected void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_testing);

        ltLed = findViewById(R.id.ltSignal);
        btnLed = findViewById(R.id.btnSignal);
        btnStartFinish = findViewById(R.id.btnStartFinish);
        txtTimer = findViewById(R.id.txtTimer);
        txtCounterRight = findViewById(R.id.txtCounterRight);
        txtCounterFall = findViewById(R.id.txtCounterFall);

        Settings = PreferenceManager.getDefaultSharedPreferences(getBaseContext());

findViewById(R.id.ltTrain).setBackgroundColor(Color.parseColor(Settings.getString("ba
ckground_color", "#004D40")));
        flgPassWithoutError = Settings.getBoolean("pass_without_error", false);
        curFirstWaveform = curChooseSignal(Settings.getString("first_signal",
"nothing"));
        curSecondWaveform = curChooseSignal(Settings.getString("second_signal",
"cyan"));
        if (!Settings.getBoolean("highlight_with_dotted_line", true)){
            if (curFirstWaveform == R.drawable.dotted_border){
                curFirstWaveform = R.drawable.color_icon_transparent;
            }
            if (curSecondWaveform == R.drawable.dotted_border){
                curSecondWaveform = R.drawable.color_icon_transparent;
            }
        }
        if (Settings.getString("workzone", "big").compareTo("big") == 0){
            ViewGroup.LayoutParams params = btnLed.getLayoutParams();
            params.width = params.height = 750;
            btnLed.setLayoutParams(params);
        }
        else {
            ViewGroup.LayoutParams params = btnLed.getLayoutParams();
            params.width = params.height = 325;
            btnLed.setLayoutParams(params);
        }
        flgReactionClickHold = Settings.getString("reaction",
"click").compareTo("click") != 0;
        try {
            curLimitingAttempts =
Integer.parseInt(Settings.getString("number_of_attempts", "20"));
        }
        catch (Exception e) {
            curLimitingAttempts = 0;
        }
        Log.d("1", String.valueOf(curLimitingAttempts));

        btnLed.setOnTouchListener((v, event) -> {
```

```
    if (flgTestingInProgress) {
        if (event.getAction() == MotionEvent.ACTION_DOWN) {
            if (!flgReactionClickHold) {
                if (flgTestInProgress) {
                    ltLed.setBackgroundResource(curFirstWaveform);
                    txtCounterRight.setText(String.valueOf(++curCountRight));

                    curTimeOfAttempts += (float) curStopwatch / 1000 + ";";


                } else {
                    flgTooEarlyMistake = true;

                    txtTimer.setText(R.string.too_early);
                    setCurCountFall();
                }
                txtCounterFall.setText(String.valueOf(curCountFall));


                flgTestInProgress = false;
            } else {
                flgPreparationForTheTest = true;
            }

        } else if (event.getAction() == MotionEvent.ACTION_UP) {
            if (flgReactionClickHold) {
                if (flgTestInProgress) {
                    ltLed.setBackgroundResource(curFirstWaveform);
                    txtCounterRight.setText(String.valueOf(++curCountRight));

                    curTimeOfAttempts += (float) curStopwatch / 1000 + ";";

                } else if (!flgTooLateMistake & flgPreparationForTheTest) {
                    flgTooEarlyMistake = true;

                    txtTimer.setText(R.string.too_early);

                    setCurCountFall();
                    txtCounterFall.setText(String.valueOf(curCountFall));
                } else flgTooLateMistake = false;
                flgTestInProgress = false;
            }
        }
    }
    return false;
});

ActionBar actionBar = getSupportActionBar();
if (actionBar != null) {
    actionBar.setDisplayHomeAsUpEnabled(true);
    if (curLimitingAttempts == 0) {
        actionBar.setTitle(R.string.train);
    }
    else {
        actionBar.setTitle(R.string.test);
        actionBar.setTitle(actionBar.getTitle() + " " + curLimitingAttempts);
```

```
        }
    }

    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    LayoutInflater inflater = getLayoutInflater();

    View customView = inflater.inflate(R.layout.help_dialog, null);
    TextView messageView1 = customView.findViewById(R.id.txtHelp1);
    TextView messageView2 = customView.findViewById(R.id.txtHelp2);
    TextView messageView3 = customView.findViewById(R.id.txtHelp3);
    TextView messageView4 = customView.findViewById(R.id.txtHelp4);

    builder.setView(customView)
            .setTitle(R.string.help);

    messageView1.setText(R.string.help_testing1);
    messageView2.setText(R.string.help_testing2);
    messageView3.setText(R.string.help_testing3);
    messageView4.setText(R.string.help_testing4);


    AlertDialog alert = builder.create();

    alert.show();

}

public void setCurCountFall() {
    curCountFall++;
    curTimeOfAttempts += "-;";
    if (curLimitingAttempts != 0 && curLimitingAttempts / 2 < curCountFall) {
        txtTimer.setText(R.string.try_again_later);

        flgTestInProgress = false;
        flgTestingInProgress = false;

        btnLed.setVisibility(View.INVISIBLE);
        btnStartFinish.setText(R.string.start);
        ltLed.setBackgroundResource(curFirstWaveform);

        try {
            Test.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        btnStartFinish.setEnabled(false);
    }
}

public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.options_menu, menu);
    return true;
}

@SuppressLint("InflateParams")
```

```
public void onHelpItemClick(MenuItem item) {

    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    LayoutInflater inflater = getLayoutInflater();

    View customView = inflater.inflate(R.layout.help_dialog, null);
    TextView messageView1 = customView.findViewById(R.id.txtHelp1);
    TextView messageView2 = customView.findViewById(R.id.txtHelp2);
    TextView messageView3 = customView.findViewById(R.id.txtHelp3);
    TextView messageView4 = customView.findViewById(R.id.txtHelp4);

    builder.setView(customView)
            .setTitle(R.string.help);

    messageView1.setText(R.string.help_testing1);
    messageView2.setText(R.string.help_testing2);
    messageView3.setText(R.string.help_testing3);
    messageView4.setText(R.string.help_testing4);


    AlertDialog alert = builder.create();

    alert.show();
}

protected void onResume() {
    super.onResume();


    Locale locale = new Locale(Settings.getString("locale", "ru"));
    Locale.setDefault(locale);
    Configuration config = getBaseContext().getResources().getConfiguration();
    config.locale = locale;
    getBaseContext().getResources().updateConfiguration(config,
            getBaseContext().getResources().getDisplayMetrics());
    if (curLimitingAttempts != 0) {
        txtTimer.setText("");
    }
    else {
        if (curStopwatch != 0) {
            txtTimer.setText(String.valueOf((float) curStopwatch / 1000));
        } else {
            txtTimer.setText(R.string._0_0000);
        }
    }
    txtCounterRight.setText(String.valueOf(curCountRight));
    txtCounterFall.setText(String.valueOf(curCountFall));

    ltLed.setBackgroundResource(curFirstWaveform);

    if (flgTestingInProgress) {
        btnLed.setVisibility(View.VISIBLE);
        btnStartFinish.setText(R.string.finish);

        Test = new Thread(this::curTest);
        Test.start();
```

```
        }
    }

    @Override
    protected void onPause() {
        super.onPause();

        Settings.edit().putString("curName", curName).apply();
        Settings.edit().putString("curDate", curDate).apply();
        Settings.edit().putString("curReaction", curReaction).apply();
        Settings.edit().putString("curWorkZone", curWorkZone).apply();
        Settings.edit().putString("curTimeOfAttempts", curTimeOfAttempts).apply();
    }

    public boolean onOptionsItemSelected(MenuItem menuItem) {
        finish();
        return true;
    }

    @SuppressLint("SimpleDateFormat")
    public void onClickStarFinish(View view) throws InterruptedException {
        if (!flgTestingInProgress) {
            curCountRight = 0;
            curCountFall = 0;
            flgTestingInProgress = true;
            if(!flgReactionClickHold) {
                flgPreparationForTheTest = true;
            }

            btnLed.setVisibility(View.VISIBLE);
            btnStartFinish.setText(R.string.finish);

            if (curLimitingAttempts == 0){
                txtTimer.setText(R.string._0_0000);
            }
            else {
                txtTimer.setText("");
            }
            txtCounterRight.setText(R.string._0);
            txtCounterFall.setText(R.string._0);

            curDate = new SimpleDateFormat("yyyy.MM.dd HH:mm:ss").format(new
Date(System.currentTimeMillis()));
            curReaction = (flgReactionClickHold ? getString(R.string.hold) :
getString(R.string.click));
            curWorkZone = (Settings.getString("workzone", "big").compareTo("big") ==
0 ? getString(R.string.big) : getString(R.string.small));

            Test = new Thread(this::curTest);
            Test.start();
        }
        else {
            flgTestInProgress = false;
            flgTestingInProgress = false;

            btnLed.setVisibility(View.INVISIBLE);
```

```
            btnStartFinish.setText(R.string.start);
            ltLed.setBackgroundResource(curFirstWaveform);

            Test.join();
        }
    }

    protected void onSaveInstanceState(@NonNull Bundle outState) {
        super.onSaveInstanceState(outState);
        outState.putInt("curStopwatch", curStopwatch);
        outState.putInt("curCountRight", curCountRight);
        outState.putInt("curCountFall", curCountFall);

        outState.putBoolean("flgTestingInProgress", flgTestingInProgress);
        outState.putBoolean("flgTestInProgress", flgTestInProgress);

        outState.putString("curName", curName);
        outState.putString("curDate", curDate);
        outState.putString("curReaction", curReaction);
        outState.putString("curWorkZone", curWorkZone);
        outState.putString("curTimeOfAttempts", curTimeOfAttempts);
    }

    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
        curStopwatch = savedInstanceState.getInt("curStopwatch");
        curCountRight = savedInstanceState.getInt("curCountRight");
        curCountFall = savedInstanceState.getInt("curCountFall");

        flgTestingInProgress = savedInstanceState.getBoolean("flgTestingInProgress");
        flgTestInProgress = savedInstanceState.getBoolean("flgTestInProgress");

        curName = savedInstanceState.getString("curName", curName);
        curDate = savedInstanceState.getString("curDate", curDate);
        curReaction = savedInstanceState.getString("curReaction", curReaction);
        curWorkZone = savedInstanceState.getString("curWorkZone", curWorkZone);
        curTimeOfAttempts = savedInstanceState.getString("curTimeOfAttempts",
curTimeOfAttempts);
    }

    public int curChooseSignal(String curCondition) {
        switch (curCondition) {
            case "serios_smile":
                return R.drawable.pic1;

            case "smile_with_tongue":
                return R.drawable.pic2;

            case "fencer_number_one":
                return R.drawable.fenc_1;

            case "fencer_number_two":
                return R.drawable.fenc_2;

            case "black":
                return R.drawable.color_icon_black;
```

```
            case "red":
                return R.drawable.color_icon_red;

            case "yellow":
                return R.drawable.color_icon_yellow;

            case "gray":
                return R.drawable.color_icon_gray;

            case "#004D40":
                return R.drawable.color_icon_green;

            case "cyan":
                return R.drawable.color_icon_cyan;

            default:
                return R.drawable.dotted_border;
        }
    }

    @SuppressLint("InflateParams")
    public void curTest() {
        while (flgTestingInProgress & ((curLimitingAttempts != 0 & curCountRight <
curLimitingAttempts) | (curLimitingAttempts == 0))) {
            if (flgPreparationForTheTest) {
                if (!flgTestInProgress) {
                    int curTimer = curRandom.nextInt(2000) + 1000;
                    while (flgTestingInProgress & curTimer-- > 0 &
!flgTooEarlyMistake) {
                        try {
                            Thread.sleep(1);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                    }

                    curStopwatch = 0;
                    if (flgTestingInProgress & !flgTooEarlyMistake) {
                        flgTestInProgress = true;
                        runOnUiThread(() ->
ltLed.setBackgroundResource(curSecondWaveform));
                    }
                } else {
                    runOnUiThread(() ->
ltLed.setBackgroundResource(curSecondWaveform));
                }

                if (flgPreparationForTheTest) {
                    if (flgReactionClickHold) {
                        flgPreparationForTheTest = false;
                    }
                    if (flgTestingInProgress & flgTestInProgress &
!flgTooEarlyMistake){
                        runOnUiThread(() -> txtTimer.setText(""));
                    }
```

```
                    while (flgTestingInProgress & flgTestInProgress & curStopwatch <
500 & !flgTooEarlyMistake) {
                        try {
                            Thread.sleep(1);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                        if (flgTestingInProgress & flgTestInProgress &
!flgTooEarlyMistake) {
                            curStopwatch++;
                            if (curLimitingAttempts == 0) {
                                runOnUiThread(() ->
txtTimer.setText(String.valueOf((float) curStopwatch / 1000)));
                            }
                        }
                    }
                }

                if (curStopwatch == 500) {
                    flgTestInProgress = false;
                    flgTooLateMistake = true;
                    runOnUiThread(() -> {
                        if (!flgPassWithoutError) {
                            txtTimer.setText(R.string.too_late);

                            setCurCountFall();
                            txtCounterFall.setText(String.valueOf(curCountFall));
                        }
                        ltLed.setBackgroundResource(curFirstWaveform);

btnLed.dispatchTouchEvent(MotionEvent.obtain(SystemClock.uptimeMillis(),
SystemClock.uptimeMillis(), MotionEvent.ACTION_UP,  0, 0, 0));
                    });
                }

                if (flgTooEarlyMistake) {
                    flgTooEarlyMistake = false;
                }
            }
            else {
                try {
                    Thread.sleep(1);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
        if (curLimitingAttempts != 0 & curCountRight == curLimitingAttempts) {
            runOnUiThread(() -> {
                btnLed.setVisibility(View.INVISIBLE);
                btnStartFinish.setText(R.string.start);
                ltLed.setBackgroundResource(curFirstWaveform);


                AlertDialog.Builder builder = new AlertDialog.Builder(this);
```

```
                final EditText input = new EditText(TestingActivity.this);
                LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
                        LinearLayout.LayoutParams.MATCH_PARENT,
                        LinearLayout.LayoutParams.MATCH_PARENT);
                input.setLayoutParams(lp);
                input.setHint(R.string.name);
                builder.setView(input)
                        .setTitle(R.string.enter_your_name)
                        .setPositiveButton(R.string.ok, (dialog, id) -> {

                            curName = input.getText().toString();

                            userStatsPOSTTask.execute();


                        })
                        .setNegativeButton(R.string.cancel, (dialog, id) ->
dialog.cancel());

                AlertDialog alert = builder.create();

                alert.show();
            });
            flgTestInProgress = false;
            flgTestingInProgress = false;
        }
    }
}


package com.stubeda.dips;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.SharedPreferences;
import android.graphics.Color;
import android.os.AsyncTask;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.view.MenuItem;
import android.widget.TextView;

import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.preference.PreferenceManager;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
```

```
import java.net.HttpURLConnection;
import java.net.URL;

public class FastStatsActivity extends AppCompatActivity {

    private final Activity curActivity = this;
    private final UserStatsGETTask userStatsGETTask = new UserStatsGETTask();
    @SuppressLint("StaticFieldLeak")
    private class UserStatsGETTask extends AsyncTask<String, String, String> {

        String answerHTTP = "";
        JSONArray postDataParams = new JSONArray();

        String serverURL = "http://10.0.2.2:8080/api/v1/users";

        @Override
        protected String doInBackground(String... params) {
            URL url;
            HttpURLConnection urlConnection = null;
            try {
                url = new URL(serverURL);

                urlConnection = (HttpURLConnection) url.openConnection();

                int responseCode = urlConnection.getResponseCode();
                if(responseCode == HttpURLConnection.HTTP_OK){
                    answerHTTP =
convertInputStreamToString(urlConnection.getInputStream());
                }
            } catch (Exception e) {
                e.printStackTrace();
            } finally {
                if (urlConnection != null) {
                    urlConnection.disconnect();
                }
            }

            try {
                postDataParams = new JSONArray(answerHTTP);
            } catch (JSONException e) {
                e.printStackTrace();
            }

            return null;
        }

        @Override
        protected void onPostExecute(String result) {
            if (answerHTTP.length() != 0) {
                StringBuilder answer = new StringBuilder();
                for (int i = 0; i < postDataParams.length(); i++) {
                    try {
                        answer
                                .append(getString(R.string.name)).append(": ")
                                .append(((JSONObject)
postDataParams.get(i)).getString("name"))
```

```
                                .append("\n")

                                .append(getString(R.string.date)).append(": ")
                                .append(((JSONObject)
postDataParams.get(i)).getString("date"))
                                .append("\n")

                                .append(getString(R.string.reaction)).append(": ")
                                .append(((JSONObject)
postDataParams.get(i)).getString("reaction"))
                                .append("\n")

                                .append(getString(R.string.work_zone)).append(": ")
                                .append(((JSONObject)
postDataParams.get(i)).getString("work_zone"))
                                .append("\n")

                                .append(getString(R.string.time)).append(": ")
                                .append("\n")
                                .append(((JSONObject)
postDataParams.get(i)).getString("time")
                                        .replaceAll(";", "\n"))
                                .append("\n")
                                .append("\n");
                    } catch (JSONException e) {
                        e.printStackTrace();
                    }
                }
                textView.setText(answer.toString());
            }
            else {
                AlertDialog.Builder builder = new AlertDialog.Builder(curActivity);
                builder.setTitle("Server answer")
                        .setMessage(R.string.failed_to_connect_to_server)
                        .setPositiveButton(R.string.ok, (dialog, id) ->
dialog.cancel());

                AlertDialog alert = builder.create();

                alert.show();
            }

            super.onPostExecute(result);
        }

        private String convertInputStreamToString(InputStream in) {
            BufferedReader reader = null;
            StringBuilder response = new StringBuilder();
            try {
                reader = new BufferedReader(new InputStreamReader(in));
                String line;
                while ((line = reader.readLine()) != null) {
                    response.append(line);
                }
            } catch (IOException e) {
                e.printStackTrace();
```

```
            } finally {
                if (reader != null) {
                    try {
                        reader.close();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
            return response.toString();
        }
    }

    SharedPreferences Settings;

    private String curName = "";
    private String curDate = "";
    private String curReaction = "";
    private String curWorkZone = "";
    private String curTimeOfAttempts = "";

    private TextView textView;

    @SuppressLint("SetTextI18n")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_fast_stats);

        Settings = PreferenceManager.getDefaultSharedPreferences(getBaseContext());

        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null) {
            actionBar.setDisplayHomeAsUpEnabled(true);
            actionBar.setTitle(R.string.fast_stats);
        }
        textView = new TextView(this);
        textView.setTextSize(16);
        textView.setPadding(16, 16, 16, 16);

textView.setBackgroundColor(Color.parseColor(Settings.getString("background_color",
"#004D40")));
        textView.setMovementMethod(new ScrollingMovementMethod());
        textView.setTextColor(Color.WHITE);

        curName = Settings.getString("curName", curName);
        curDate = Settings.getString("curDate", curDate);
        curReaction = Settings.getString("curReaction", curReaction);
        curWorkZone = Settings.getString("curWorkZone", curWorkZone);
        curTimeOfAttempts = Settings.getString("curTimeOfAttempts",
curTimeOfAttempts);

        userStatsGETTask.execute();

        textView.setText(
                getString(R.string.name) + ": " + curName + ";\n" +
```

```
                getString(R.string.date) + ": " + curDate + ";\n" +
                getString(R.string.reaction) + ": " + curReaction + ";\n" +
                getString(R.string.work_zone) + ": " + curWorkZone + ";\n" +
                getString(R.string.time) + ": " + curTimeOfAttempts
        );

        setContentView(textView);
    }

    public boolean onOptionsItemSelected(MenuItem menuItem) {
        finish();
        return true;
    }
}
```

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

</androidx.constraintlayout.widget.ConstraintLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#004D40"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/btnSetting"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:onClick="onClickSetting"
        android:text="@string/setting"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.2" />

    <Button
        android:id="@+id/btnStartFinish"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:onClick="onClickTrain"
```

```
        android:text="@string/testing"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.4"
        tools:ignore="OnClick" />

    <Button
        android:id="@+id/btnResults"
        android:layout_width="200dp"
        android:layout_height="48dp"
        android:enabled="true"
        android:onClick="onClickResults"
        android:text="@string/results"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.6" />

    <Button
        android:id="@+id/btnExit"
        android:layout_width="200dp"
        android:layout_height="48dp"
        android:onClick="onClickExit"
        android:text="@string/exit"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.8" />

</androidx.constraintlayout.widget.ConstraintLayout>


<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#004D40 "
    tools:context=".MainActivity">

    <Button
        android:id="@+id/btnResults"
        android:layout_width="200dp"
        android:layout_height="48dp"
        android:enabled="true"
        android:onClick="onClickResults"
```

```
        android:text="@string/results"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.6"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias=".6" />

    <Button
        android:id="@+id/btnSetting"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:onClick="onClickSetting"
        android:text="@string/setting"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.2"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.2" />

    <Button
        android:id="@+id/btnExit"
        android:layout_width="200dp"
        android:layout_height="48dp"
        android:onClick="onClickExit"
        android:text="@string/exit"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.8"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.8" />

    <Button
        android:id="@+id/btnStartFinish"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:onClick="onClickTrain"
        android:text="@string/testing"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.4"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias=".4"
        tools:ignore="OnClick" />

</androidx.constraintlayout.widget.ConstraintLayout>


<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
```

```
    <FrameLayout
        android:id="@+id/ltSettings"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#BF004D40" />
</LinearLayout>


<PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/idPreference">

    <ListPreference
        app:defaultValue="#004D40"
        app:entries="@array/color_entries"
        app:entryValues="@array/color_values"
        app:key="background_color"
        app:title="@string/background_color"
        app:useSimpleSummaryProvider="true" />

    <ListPreference
        app:defaultValue="nothing"
        app:entries="@array/signal_entries"
        app:entryValues="@array/signal_values"
        app:key="first_signal"
        app:title="@string/first_signal"
        app:useSimpleSummaryProvider="true" />

    <ListPreference
        app:defaultValue="cyan"
        app:entries="@array/signal_entries"
        app:entryValues="@array/signal_values"
        app:key="second_signal"
        app:title="@string/second_signal"
        app:useSimpleSummaryProvider="true" />

    <ListPreference
        app:defaultValue="big"
        app:entries="@array/workzone_entries"
        app:entryValues="@array/workzone_values"
        app:key="workzone"
        app:title="@string/work_zone"
        app:useSimpleSummaryProvider="true" />

    <SwitchPreferenceCompat
        app:defaultValue="true"
        app:key="highlight_with_dotted_line"
        app:title="@string/highlight_with_dotted_line" />

    <ListPreference
        app:defaultValue="click"
        app:entries="@array/reaction_entries"
        app:entryValues="@array/reaction_values"
        app:key="reaction"
        app:title="@string/reaction"
```

```
        app:useSimpleSummaryProvider="true" />

    <ListPreference
        app:defaultValue="0"
        app:entries="@array/number_of_attempts_entries"
        app:entryValues="@array/number_of_attempts_values"
        app:key="number_of_attempts"
        app:title="@string/number_of_attempts"
        app:useSimpleSummaryProvider="true" />

    <SwitchPreferenceCompat
        app:key="pass_without_error"
        app:title="@string/pass_without_error" />

    <SwitchPreferenceCompat
        app:key="show_statistics"
        app:title="@string/show_statistics" />

    <ListPreference
        app:defaultValue="ru"
        app:entries="@array/locale_entries"
        app:entryValues="@array/locale_values"
        app:key="locale"
        app:title="@string/locale"
        app:useSimpleSummaryProvider="true" />

    <!--
    <SwitchPreferenceCompat
        app:key="picture_signal"
        app:title="@string/picture_signal" />

    <ListPreference
        app:dependency="picture_signal"
        app:defaultValue="serios_smile"
        app:entries="@array/picture_entries"
        app:entryValues="@array/picture_values"
        app:key="signal_picture"
        app:title="@string/signal_picture"
        app:useSimpleSummaryProvider="true" />

        -->

    <!--
        app:dependency="sync"
        app:summaryOff="@string/attachment_summary_off"
        app:summaryOn="@string/attachment_summary_on"
        -->


</PreferenceScreen>


<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/ltTrain"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="#004D40"
tools:context=".TestingActivity">

<TextView
    android:id="@+id/txtCounterFall"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/_0"
    android:textColor="#FFFFFF"
    android:textSize="30sp"
    android:visibility="visible"
    app:layout_constraintBottom_toTopOf="@+id/ltSignal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.75"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.5" />

<TextView
    android:id="@+id/txtCounterRight"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/_0"
    android:textColor="#FFFFFF"
    android:textSize="30sp"
    android:visibility="visible"
    app:layout_constraintBottom_toTopOf="@+id/ltSignal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.25"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.5" />

<Button
    android:id="@+id/btnStartFinish"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:onClick="onClickStarFinish"
    android:text="@string/start"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ltSignal"
    app:layout_constraintVertical_bias="0.5" />

<TextView
    android:id="@+id/txtTimer"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/_0_0000"
```

```
        android:textColor="#FFFFFF"
        android:textSize="30sp"
        android:visibility="visible"
        app:layout_constraintBottom_toTopOf="@+id/ltSignal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.75" />

    <androidx.constraintlayout.widget.ConstraintLayout
        android:id="@+id/ltSignal"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="@drawable/dotted_border"
        app:layout_constraintBottom_toBottomOf="@+id/btnSignal"
        app:layout_constraintEnd_toEndOf="@+id/btnSignal"
        app:layout_constraintHorizontal_bias="0"
        app:layout_constraintStart_toStartOf="@+id/btnSignal"
        app:layout_constraintTop_toTopOf="@+id/btnSignal"
        app:layout_constraintVertical_bias="0">

    </androidx.constraintlayout.widget.ConstraintLayout>

    <Button
        android:id="@+id/btnSignal"
        style="?attr/borderlessButtonStyle"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:visibility="invisible"
        app:backgroundTint="@android:color/transparent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>


<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/ltTrain"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#004D40"
    tools:context=".TestingActivity">

    <Button
        android:id="@+id/btnStartFinish"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
```

```
    android:onClick="onClickStarFinish"
    android:text="@string/start"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toEndOf="@+id/ltSignal"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.5" />

<TextView
    android:id="@+id/txtTimer"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/_0_0000"
    android:textColor="#FFFFFF"
    android:textSize="30sp"
    android:visibility="visible"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/ltSignal"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.75" />

<TextView
    android:id="@+id/txtCounterRight"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/_0"
    android:textColor="#FFFFFF"
    android:textSize="30sp"
    android:visibility="visible"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/ltSignal"
    app:layout_constraintHorizontal_bias="0.33"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.25" />

<TextView
    android:id="@+id/txtCounterFall"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/_0"
    android:textColor="#FFFFFF"
    android:textSize="30sp"
    android:visibility="visible"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/ltSignal"
    app:layout_constraintHorizontal_bias="0.66"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.25" />

<androidx.constraintlayout.widget.ConstraintLayout
    android:id="@+id/ltSignal"
```

```
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="@drawable/dotted_border"
        app:layout_constraintBottom_toBottomOf="@+id/btnSignal"
        app:layout_constraintEnd_toEndOf="@+id/btnSignal"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="@+id/btnSignal"
        app:layout_constraintTop_toTopOf="@+id/btnSignal"
        app:layout_constraintVertical_bias="0.5">

    </androidx.constraintlayout.widget.ConstraintLayout>

    <Button
        android:id="@+id/btnSignal"
        style="?attr/borderlessButtonStyle"
        android:layout_width="250dp"
        android:layout_height="250dp"
        android:visibility="invisible"
        app:backgroundTint="@android:color/transparent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.5" />

</androidx.constraintlayout.widget.ConstraintLayout>


<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:padding="20dp"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/txtHelp1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="15dp"/>
    <TextView
        android:id="@+id/txtHelp2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="15dp"/>
    <TextView
        android:id="@+id/txtHelp3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="15dp"/>
    <TextView
        android:id="@+id/txtHelp4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

```
package com.example.dipsserver;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DipsServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(DipsServerApplication.class, args);
    }

}

package com.example.dipsserver.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    @Column(name = "name")
    private String name;
```

```
    @Column(name = "date")
    private String date;

    @Column(name = "reaction")
    private String reaction;

    @Column(name = "work_zone")
    private String work_zone;

    @Column(name = "time")
    private String time;

    public User() {
    }

    public User(String name, String date, String reaction, String work_zone, String t
ime) {
        this.name = name;
        this.date = date;
        this.reaction = reaction;
        this.work_zone = work_zone;
        this.time = time;
    }

    public long getId() {
        return id;
    }

    public void setId(long id) {
        this.id = id;
    }

    public String getName() {
        return name;
```

```
}

public void setName(String name) {
    this.name = name;
}

public String getDate() {
    return date;
}

public void setDate(String date) {
    this.date = date;
}

public String getReaction() {
    return reaction;
}

public void setReaction(String reaction) {
    this.reaction = reaction;
}

public String getWork_zone() {
    return work_zone;
}

public void setWork_zone(String work_zone) {
    this.work_zone = work_zone;
}

public String getTime() {
    return time;
}
```

```
    public void setTime(String time) {
        this.time = time;
    }
}

package com.example.dipsserver.repository;

import com.example.dipsserver.model.User;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface UserRepository extends JpaRepository<User, Long>{
}

package com.example.dipsserver.controller;

import com.example.dipsserver.exception.ResourceNotFoundException;
import com.example.dipsserver.model.User;
import com.example.dipsserver.repository.UserRepository;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
```

```
import java.util.HashMap;
import java.util.List;
import java.util.Map;

@RestController
@CrossOrigin
@RequestMapping("/api/v1/")
public class UserController {

    @Autowired
    private UserRepository userRepository;

    // get all users
    @GetMapping("/users")
    public List<User> findAllUsers(){
        return userRepository.findAll();
    }

    // create user rest api
    @PostMapping("/users")
    public User createUser(@RequestBody User user) {
        return userRepository.save(user);
    }

    // get user by id rest api
    @GetMapping("/users/{id}")
    public ResponseEntity<User> getUserById(@PathVariable Long id) {
        User user = new User();
        user = userRepository.findById(id)
            .orElseThrow(() -
> new ResourceNotFoundException("User not exist with id: " + id));
        return ResponseEntity.ok(user);
    }

    // update user rest api
```

```
    @PutMapping("/users/{id}")
    public ResponseEntity<User> updateUser(@PathVariable Long id, @RequestBody User u
serDetails){
        User user = userRepository.findById(id)
            .orElseThrow(() -
> new ResourceNotFoundException("User not exist with id :" + id));

        user.setName(userDetails.getName());

        // !!!
        // parameters
        // !!!

        User updateUser = userRepository.save(user);

        return ResponseEntity.ok(updateUser);
    }

    // delete user rest api
    @DeleteMapping("/users/{id}")
    public ResponseEntity<Map<String, Boolean>> deleteUser(@PathVariable Long id){
        User user = userRepository.findById(id)
            .orElseThrow(() -
> new ResourceNotFoundException("User not exist with id :" + id));

        userRepository.delete(user);
        Map<String, Boolean> response = new HashMap<>();
        response.put("deleted", Boolean.TRUE);
        return ResponseEntity.ok(response);
    }
}

package com.example.dipsserver.exception;

import org.springframework.http.HttpStatus;
```

```
import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(value = HttpStatus.NOT_FOUND)
public class ResourceNotFoundException extends RuntimeException{

   private static final long serialVersionUID = 1L;

   public ResourceNotFoundException(String message) {
       super(message);
   }
}

spring.datasource.url=jdbc:mysql://localhost:3306/dips?useSSL=false
spring.datasource.username=root
spring.datasource.password=123

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.hibernate.ddl-auto = update
```