

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

**РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ МНОГОЗВЕННОЙ АРХИТЕКТУРЫ НА БАЗЕ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID**

Работу выполнил Д. С. Сырчин

СОДЕРЖАНИЕ

Введение…………………………………………………………………… 3

[1 Понятие развития человека и его личности 5](#_Toc32797)

[1.1 Роль приложений в развитии личности человека 7](#_Toc32798)

[1.2 Обзор существующих решений 8](#_Toc32799)

[2 Модель многозвенной архитектуры приложения 11](#_Toc32800)

[2.1 Разделение приложения по уровням 12](#_Toc32801)

[2.1.1 Уровень пользовательского интерфейса 13](#_Toc32802)

[2.1.2 Уровень обработки запросов 13](#_Toc32803)

[2.1.3 Уровень данных 14](#_Toc32804)

[2.2 Сравнение с другими видами архитектур 15](#_Toc32805)

[3 Мобильное приложение для ОС Android 19](#_Toc32806)

[3.1 Архитектура и реализация 21](#_Toc32807)

[3.2 Активности приложения 22](#_Toc32808)

[3.2.1 Вход/регистрация 25](#_Toc32809)

[3.2.2 Выбор уровня развития 29](#_Toc32810)

[3.2.3 Главный экран уровня развития 30](#_Toc32811)

[3.2.4 Чат с куратором и интерфейс Pusher 36](#_Toc32812)

[3.2.5 Информация о задании 41](#_Toc32813)

[3.2.6 Таблица рекордов 43](#_Toc32814)

[3.2.7 Настройки 45](#_Toc32815)

[3.3 Локальная база данных SQLite 46](#_Toc32816)

Заключение………………………………………………………………… 49

Список использованных источников…………………………………….. 50

ВВЕДЕНИЕ

Мобильные телефоны уже долго время являются основным инструментом коммуникации между людьми, находящимися на расстоянии друг от друга. Помимо основной функции мобильные телефоны обладают и многими другими, например, функцией фотосъемки, плеера, доступа в сеть Интернет.

Вместе с развитием мобильных технологий и доступностью мобильных телефонов многие люди немалую часть своего свободного времени проводят за смартфонами или планшетами. При этом широкое внедрение скоростного мобильного Интернета имеет как положительные, так и отрицательные стороны. Люди, которые постоянно подключены к сети Интернет, отключаются при этом от реальной жизни. Вследствие этого страдает формирование личностных качеств человека, например, социальных качеств. Человеку становится намного проще контактировать с людьми в сети, нежели в реальной жизни, и в приоритете у него всегда будет виртуальное общение.

В рамках данной квалификационной работы будет описано создание мобильного приложения для развития человека и его личных качеств по четырем уровням развития: физическому, материальному, социальному и духовному. Идея продвижения по каждому из уровней развития состоит в выполнении соответствующих каждому уровню заданий, которые представляют собой набор определенных действий, выполнение которых преследуют какую-либо единую цель развития навыков на выбранном уровне развития. При этом каждому пользователю назначается куратор, который наставляет его при выполнении каждого задания и оценивает успехи пользователя по его завершению.

Подавляющее большинство мобильных устройств в наши дни выпускаются на базе операционной системы Android. Это связано с тем, что платформу Android поддерживает большое количество устройств самых разных производителей, а также с тем, что разработка приложений под ОС Android доступна каждому желающему разработчику, при этом имеется бесплатный большой и удобный инструментарий для разработки. Поэтому для создания обозначенного мобильного приложения использовалась именно операционная система Android и среда разработки Android Studio, которая является наиболее удобной и понятной для разработки мобильных приложений на базе ОС Android.

Исходя из концепции приложения, оно предполагает собой одновременное использование многими пользователями, хранение структур данных и подключение к сети Интернет. Поэтому разработка приложения опиралась на многозвенную архитектуру создания приложений, где клиентские приложения четко отделены от серверной части, которая подразделяется на сервер приложений, где описана основная логика обработки запросов, исходящих с клиентских приложений, и сервер баз данных, где хранятся все необходимые для использования приложением данные.

# Понятие развития человека и его личности

Концепции развития личности – это конкретные способы понимания и объяснения развития личности человека. Сегодня существуют различные альтернативные концепции развития, которые описывают личность индивида как объединяющее единое целое и объясняют различия между субъектами.

Понятие личностного развития намного шире, чем простое вырабатывание возможностей и способностей. Знание психологии развития личности позволяет понять саму суть натуры человека и его индивидуальности. Однако современная наука на данный момент не может предложить единой концепции развития личности индивида. Силами, которые способствуют развитию, подталкивая его, являются внутренние противоречия, присущие процессу развития. Противоречия состоят из противоположных конфликтующих начал.

Постепенное становление личности субъекта – это не простое совпадение различных случайностей, а процесс, обусловленный закономерностью развития психики индивидов. Под понятием развитие понимают процесс качественных и количественных изменений в психике, духовной и интеллектуальной сфере индивида, в организме в целом, которое определяется воздействием внутренних и внешних обстоятельств, неуправляемых и управляемых условий.

В психологии издавна существовали два направления теорий о движущих силах развития личности и ее становления: социогенетическая и биологизаторская концепция развития личности.

Первая концепция показывает развитие личности вследствие прямого воздействия социальными окружающими факторами среды. Эта теория игнорирует собственную активность прогрессирующего индивида. В этой концепции человеку отводят пассивную роль существа, которое всего лишь адаптируется к среде и обстановке. Если следовать этой концепции, то остается необъяснимым тот факт, что в одних и тех же социальных условиях вырастают абсолютно разные индивиды.

Вторая теория исходит из личностного развития, обуславливаемого, главным образом, наследственными факторами. Именно поэтому процесс личностного развития носит самопроизвольный (спонтанный) характер. Основываясь на этой теории, предполагали, что человек с рождения предрасположен к определенным особенностям эмоциональных проявлений, темпу проявлений действий и к конкретному комплексу мотивов. Например, одни от рождения имеют склонность к преступлениям, другие – к успешной административной деятельности. Согласно этой теории изначально в индивиде заложено природой формы и содержание его психической деятельности, определены стадии психического развития, порядок их появления.

Безусловно, на формирование личности субъекта влияют биологические и социальные факторы, такие как: окружающие обстоятельства и условия, наследственность, образ жизни. Это все сопутствующие факторы, так как доказано многими психологами, что личностью не рождаются, а становятся в процессе ее развития.

Однако и по сей день существует множество различных взглядов на развитие личности.

Психоаналитическая концепция относится к развитию, как к адаптации биологического естества субъекта к социальной жизни, выработке у него конкретных средств удовлетворения потребностей и защитных функций.

Концепция черт основывается на том, что абсолютно все черты личности вырабатываются прижизненно. Данная теория настаивает на том, что процесс зарождения, преобразования, стабилизации черт личности подчиняется другим, небиологическим факторам и законам.

Биосоциальная концепция развития личности представляет человека как биологическое и социальное существо. Все его психические процессы, такие как: ощущение, мышление, восприятие и другие, обусловлены биологическим происхождением. А интересы, направленность, способности индивида образуются в результате воздействия социальной среды. Биосоциальная концепция развития личности рассматривает проблему соотношения социального и биологического в личностном развитии.

Гуманистическая концепция развития личности трактует личностное развитие как непосредственно становление «Я» субъекта, утверждения его значимости.

## Роль приложений в развитии личности человека

В век технологий люди каждый день пользуются десятками приложений на различных устройствах. Их количество превышает сотни тысяч штук и с каждым днем их становится все больше и больше.

Все эти приложения могут сделать нашу жизнь легче и веселее. Их различают по нескольким категориям, исходя из того, для кого это приложение разрабатывается, какие цели преследует и как оно будет реализовано. К ним относятся приложения для развлечения, путешествий, бизнеса, социальные сети, заказ и доставка еды, спорт, игры, новости и т.д. Приложения охватывают практически все сферы быта, начиная от работы и заканчивая личной жизнью.

В современном мире у подавляющего количества людей с собой всегда под рукой смартфон, с помощью которого они используют различные приложения, необходимые им для комфортной жизни. Мобильные приложения могут использоваться в качестве помощника для развития человека и его личности. Это может быть как приложение, где можно прочитать статьи или прослушать лекции для самостоятельного обучения, так и приложение, где нужно выполнять различные задания, выполнение которых в какой-то степени отражается на жизни человека, в том числе и для развития его личности.

## Обзор существующих решений

В магазине приложений Google Play операционной системы Android можно найти готовые реализации мобильных приложений на затронутую тематику развития человека и его личности. Почти все из них представляют собой готовые статьи, которые предлагается изучить пользователю.

Рассмотрим некоторые варианты из них.

Приложение «Развитие личности». Как указано в описании к нему, данное приложение – проводник на самом достойном, и, пожалуй, довольно трудном пути к развитию. В нём будут описаны технологии и теоретические знания, которые могут помочь вам в области самосовершенствования, развития личности и достижения успеха. Пользовательский интерфейс приложения представлен на рисунке 1.

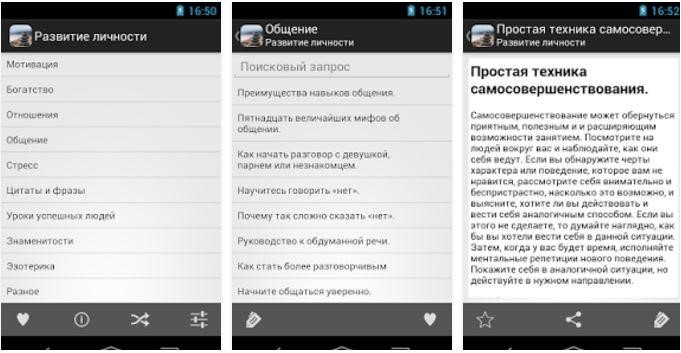


Рисунок 1 – Приложение «Развитие личности»

Данное приложение посвящено самосовершенствованию, развитию личности и достижению успеха в жизни. Все самые полезные знания о том, как обрести власть над разумом и телом, стать более здоровым, научиться получать от жизни то, что вы действительно желаете получить, стать более спокойным, достичь внутренней гармонии и многое другое, вы найдёте здесь. Вы можете прожить этот день так, как вы делаете это всегда, порою просто осознавая, что двигаясь небольшими шагами, в таинственную неизвестность, полную непредсказуемых и не совсем ожидаемых событий, вы мало что можете изменить. Вы можете просто лежать на диване, мечтая о прекрасных вещах, но это вряд ли что изменит в вашей жизни. Но, вы так же можете взять на вооружение что-то, чего вы ещё не знали, и удивительным образом преобразить свою жизнь, сделать её лучше.

Приложение представляет собой широкий список разделов, по которым пользователь может почитать различные статьи и начать развиваться в выбранном направлении. Имеет довольно высокие оценки и высокую популярность, хотя реализация является максимально простой.

Приложение «Психологион». Предоставляет более широкий спектр возможностей. Как указано в описании, данное приложение – онлайнмастерская развития личности. Здесь можно найти практически все, что нужно человеку для саморазвития: психологические упражнения, бесплатные вопросы психологу, общение с психологами и другими людьми.

Пользовательский интерфейс приложения представлен на рисунке 2.

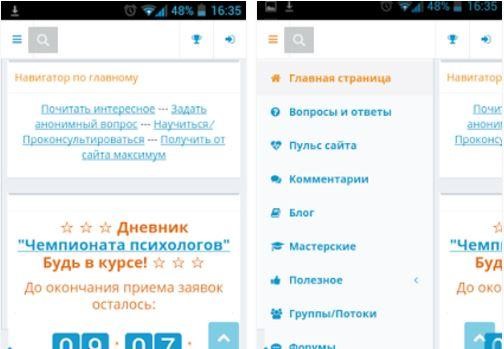


Рисунок 2 – Приложение «Психологион»

Приложение «TED». Достаточно популярное приложение, которое предоставляет доступ к более чем 2000 видео лекций, множеству статей, в том числа и для развития человека и его личности. Интерфейс приложения представлен на рисунке 3.

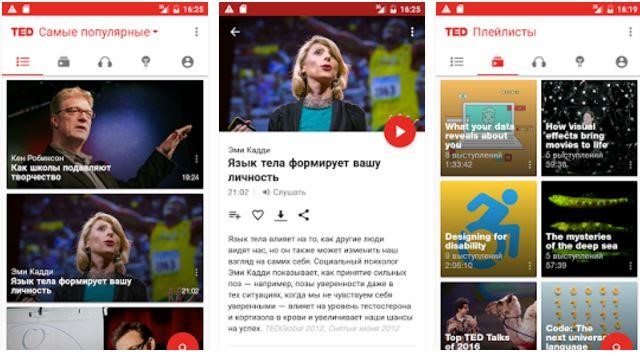


Рисунок 3 – Приложение «TED»

Приложение имеет довольно приятный и понятный интерфейс и позволяет пользователю расширять свои знания и навыки во многих сферах жизни человека. Лекции и статьи доступны на нескольких языках. Можно просматривать зарубежные лекции и включать субтитры на русском языке.

Здесь можно найти семинары на любую тему: музыка, психология, бизнес, спорт и т.д. Лекции достаточно интересные и познавательные, которые расширяют кругозор пользователя, рассказывая про что-то необычное, но повседневное. Все лекции можно скачивать в память устройства и делиться с друзьями в социальных сетях.

# Модель многозвенной архитектуры приложения

Многозвенная архитектура приложения – обобщение трехзвенной архитектурной модели комплекса программ, которая содержит в себе три основных компонента:

а) клиентское приложение;

б) сервер приложений (к которому идет подключение от клиентского

приложения);

в) сервер баз данных (с которым взаимодействует сервер приложений).

Многозвенная архитектура позволяет приложению обеспечить более оптимальную загрузку технических средств, чем другие виды архитектур программ, и обеспечивает возможность плавного масштабирования информационной системы. Схема многозвенной архитектуры приложения приведена на рисунке 4.



Рисунок 4 – Многозвенная архитектура приложения

## Разделение приложения по уровням

Долгое время стоял вопрос о том, как в модели клиент-сервер однозначно разделить клиента от сервера. Очевидно, что четкого различия между ними нет. К примеру, сам сервер базы данных может выступать в качестве клиента, которые посылает запросы на серверы программной логики, отвечающие за обработку данных, содержащихся в этой базе данных. В таком случае сервер баз данных выполняет только функцию обработки запросов.

В ходе рассмотрения множества приложений типа клиент-сервер, которые подразумевали под собой обращение пользователей к серверам баз данных, было принято разделять приложение на три уровня (рис. 5):

а) уровень пользовательского интерфейса (клиент);

б) уровень обработки запросов (сервер приложений);

в) уровень данных (сервер баз данных).

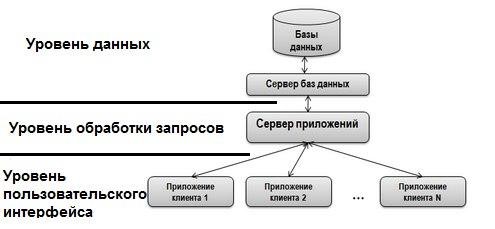


Рисунок 5 – Разделение приложения на 3 уровня

Уровень пользовательского интерфейса содержит в себе все необходимые компоненты для непосредственного взаимодействия с пользователем. Уровень обработки данных обычно содержит приложения, обрабатывающие запросы, поступающие от пользователей, а уровень данных содержит данные, с которыми взаимодействует приложение.

### Уровень пользовательского интерфейса

Уровень пользовательского интерфейса реализуется на клиентах. Он обычно содержит программы, с помощью которых пользователи взаимодействуют с приложением. Уровень пользовательского интерфейса не дожжен иметь прямых связей с сервером базы данных из-за требований безопасности и масштабируемости, быть оснащен основной бизнес-логикой по требованиям масштабируемости и хранить состояние приложения по требованиям надежности.

На уровень пользовательского интерфейса обычно выносится только самая простая бизнес-логика, например:

* интерфейс авторизации или регистрации;
* различные алгоритмы шифрования;
* проверка вводимых пользователем значений на соответствие и допустимость формату;
* простые операции с данными, которые уже находятся на устройстве.

### Уровень обработки запросов

Уровень обработки запросов является средним уровнем, где сосредоточена большая часть бизнес-логики. Вне этого уровня находятся только фрагменты, которые передаются клиенту, а также элементы логики, находящие в базе данных, например, процедуры и триггеры.

Реализация этого уровня обеспечивается связующим программным обеспечением. Серверы приложений на уровне обработке запросов создаются таким образом, чтобы добавление к ним дополнительных компонентов программной логики обеспечивало горизонтальное масштабирование производительности всего программного комплекса и не требовало корректировки программного кода клиентского приложения.

### Уровень данных

Уровень данных в модели клиент-сервер содержит программную логику, которая представляет данных, использующиеся при работе приложения. Отличительным свойством этого уровня является требование сохранности, т.е. когда приложение не используется пользователями. данные должны сохраняться в определенном месте для их дальнейшего использования. В самом простом варианте уровень данных реализуется с помощью файловой системы, но чаще всего для его реализации задействуется полномасштабная база данных, которая находится на стороне сервера.

Кроме обычного хранения данных, с которыми взаимодействуют приложения пользователей, уровень данных также отвечает за поддержание целостности данных для этих приложений. Для базы данных это означает, что метаданные, например, описания таблиц, ограничения и специальные метаданные приложений, также хранятся на уровне хранения данных.

В общем случае уровень данных организуется в виде реляционной базы данных. Ключевым моментом здесь является независимость данных, которые организуются независимо от приложений так, чтобы изменения в структуре данных не сказывались на работоспособности клиентского приложения, а клиентские приложения не оказывали влияние на структуру данных. Использование реляционных баз данных в модели клиент-сервер обеспечивает отделения уровня сервера приложений от уровня данных, что позволяет проводить обработку данных и их хранение отдельно друг от друга.

Подключение к уровню данных реализуется только с уровня обработки запросов (сервера приложений).

## Сравнение с другими видами архитектур

Архитектура приложения характеризует все важные проектные решения в плане структур программы и взаимодействия между этими структурами. Проектные решения раскрывают необходимые свойства для приложения, которые используются приложением для его успешного функционирования. Также они формируют основу для разработки приложения, его поддержки и обслуживания.

Архитектура приложения включает в себя не только структурные и поведенческие особенности приложения, но и правила взаимодействия приложения с другими системами, а также производительность, гибкость, функциональность, надежность, экономические и технологические ограничения, возможность повторного использования, а также представление пользовательского интерфейса.

Выделяют 4 основных вида архитектур приложений, которые отличаются количеством входящих в них звеньев:

а) однозвенная архитектура;

б) двухзвенная архитектура;

в) трехзвенная архитектура;

г) многозвенная архитектура.

Однозвенная архитектура приложения подразумевает собой то, что все три звена или уровня (уровень пользовательского интерфейса, уровень обработки запросов и уровень данных) являются частью одной и той же программы, то есть находятся в одном звене. Здесь в качестве клиента выступает обычный пользовательский интерфейс, представляющий собой терминал, для которого доступен только ввод команд оператора и экран для отображения ответов от вычислителя. Схема однозвенной архитектуры представлена на рисунке 6.

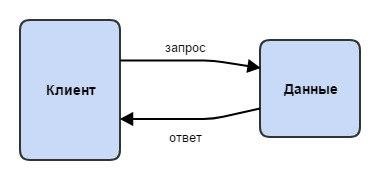


Рисунок 6 – Однозвенная архитектура приложения

К достоинствам однозвенной архитектуры можно отнести то, что объединение уровней может привести к повышению эффективности приложения. Обычно в таких приложениях расходы на переключение контекста и переходы между компонентами сведены к минимуму. Так же для приложений однозвенной архитектуры не требуется специально разрабатывать клиентскую часть, что снижает затраты на внедрение и поддержку.

К недостаткам однозвенной архитектуры приложений относится то, что их сопровождение достаточно трудно и затратно.

Двухзвенные архитектуры имеют популярность в представлении в виде архитектур клиент/сервер. В зависимости от сложности клиентской программы ее называют тонким клиентом, если она выполняет минимальный набор функций, и толстым клиентом, если набор функций достаточно широк.

Схема двухзвенной архитектуры представлена на рисунке 7.

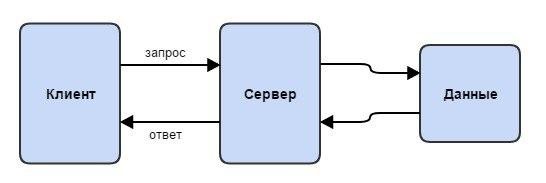


Рисунок 7 – Двухзвенная архитектура приложения

В скором времени был предложен некий компромисс между тонким и толстым клиентом, которые носит название «rich»-клиента. Как и тонкий клиент, rich-клиент также реализует графический интерфейс, который описывает средствами XML и включает в себя некоторый набор функций, присущих толстому клиенту. Прикладная логика rich-клиента реализована на сервере и данные при этом передаются в стандартном формате на основе XML и принимаются клиентом.

К достоинствам двухзвенных архитектур относят объединение прикладной логики и управление ресурсами, что позволяет выполнять главные вычисления достаточно быстро, так как переключение контекста не происходит. Также двухзвенные системы намного более мобильны, так как уровень сервера в них отделен от уровня пользовательского интерфейса.

Недостатком двухзвенной архитектуры приложения является то, что существует ограничение на взаимодействие сервера с несколькими клиентами одновременно.

Трехзвенные архитектуры приложений представляют собой некое расширение архитектур клиент/сервер. В трехзвенной архитектуре приложения все уровни архитектуры четко разделены между собой. Уровень пользовательского интерфейса размещается на клиенте, как и в двухзвенной архитектуре. Прикладная логика размещается в среднем звене на сервере приложений. Уровень данных располагается в третьем звене и состоит из всех серверов, которые взаимодействуют с приложением. Схема трехзвенной архитектуры приложения обозначена на рисунке 8.

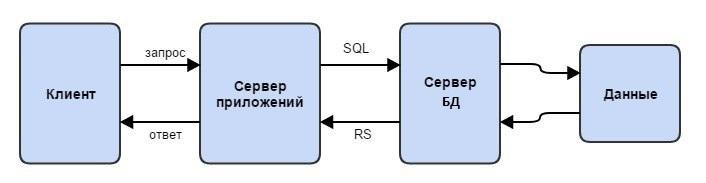


Рисунок 8 – Трехзвенная архитектура приложения

К достоинствам трехзвенной архитектуры относят:

а) возросшие возможности по масштабированию (каждый уровень

приложения может работать на отдельной вычислительной машине);

б) не требуется определять интерфейсы уровня обработки запросов, так

как в трехзвенной архитектуре уровень данных стандартизирован;

в) потери в производительности компенсируются распространением

модели уровня обработки запросов на разные сетевые узлы.

К недостаткам трезвенных архитектур относят ограниченность совместного использования нескольких трехзвенных систем.

Многозвенные архитектуры не сильно отличаются от трехзвенных архитектур приложений. В многозвенной архитектуре важное место имеет доступ к данным через сеть Интернет.

Многозвенные архитектуры разрабатываются в основном для двух ситуаций:

- объединение разнородных систем; - подключение к Интернету.

Отдельные звенья многозвенной системы сами являются двух- или

трехзвенными системами

# Мобильное приложение для ОС Android

Для реализации мобильного приложения, нацеленное на развитие человека и его личности, была выбрана операционная система Android, как одна из ведущих операционных систем на рынке мобильных устройств. Операционная система Android и инструментарий к ней дают возможность разработчику довольно просто и удобно реализовывать различные идеи в виде пользовательских мобильных приложений.

Система Android разработана компанией Google и является открытой для разработки системой, основанной на ядре Linux, для смартфонов, планшетных компьютеров, электронных книжек и многого другого. По статистике 80,7% смартфонов, которыми пользуются люди по всему миру, оснащены операционной системой Android, 17,7% приходится на продукцию Apple с операционной системой iOS.

Разработка приложения для платформы Android ведется на языке Java. В качестве инструментария для создания приложения была использована среда разработки Android Studio.

Android Studio – интегрированная среда разработки (IDE) для работы с платформой Android. Она основана на программном обеспечении IntelliJ IDEA от компании JetBrains и является официальным средством разработки Android приложений. Данная среда является максимально дружелюбной и понятной в использовании для разработчика. Поэтому именно она использовалась при разработке приложения.

Разработанное приложение по развитию человека и его личности представляет собой некий формат игры, где пользователи выполняют задания, набирают очки опыта и уровни прогресса по завершению заданий и соревнуются друг с другом. Основная идея приложения состоит в том, что пользователь развивает себя как личность по четырем основным уровням развития, а именно:

* физический уровень;
* материальный уровень;
* социальный уровень; - духовный уровень.

На каждом уровне предлагается свой набор заданий, характерный для продвижения по определенному уровню развития. Задание предполагает выполнение каких-либо действий, достижения определенных жизненных целей. Пользователь имеет возможность просмотреть доступные ему для выполнения задания, что они в себя включают, какие цели достигаются и какие навыки приобретаются по его завершению. Единовременно пользователь может выполнять только по одному заданию на каждом уровне развития.

В ходе выполнения задания пользователь ведет диалог с куратором, который наставляет его, помогает, подсказывает по выполняемому заданию на текущем уровне развития. Пользователь также время от времени делится успехами по продвижению в задании. По завершению задания пользователь представляет отчет куратору на выбранном уровне развития, вид которого описан в информации о задании. Исходя из этого отчета, куратор оценивает успехи в продвижении пользователя по выбранному уровню развития и выносит решение о количестве очков опыта, присуждаемых пользователю, учитывая рекомендуемое количество очков опыта, указанное в информации о задании.

Перед тем как приступить к продвижению по основным четырем уровню развития, новому пользователю доступен только базовый уровень развития и задания на нем. Базовый уровень развития позволяет оценить возможности и имеющиеся навыки пользователя. Здесь также происходит общение с куратором. Как только куратор решит, что пользователь достаточно развит для начала продвижения по основным уровням развития, куратор открывает пользователю доступ к ним, предварительно оценив его начальный уровень прогресса на каждом из уровней развития.

На каждом уровне развития доступна таблица рекордов, которая отображает всех пользователей, продвигающихся по данному уровню развития, в порядке убывания их уровней прогресса.

## Архитектура и реализация

Разработанное приложение подразумевает использование несколькими пользователями одновременно, взаимодействие между ними и взаимодействие между клиентской и серверной частями. Поэтому важным фактором в использовании разработанного приложения является подключение к сети Интернет. Исходя из этого, для реализации приложения была выбрана многозвенная архитектура приложения, где имеется клиентская часть (мобильное приложение на устройстве пользователя), сервер приложений (прикладная логика, необходимая для связи мобильного приложения и сервера баз данных) и сервер баз данных (который используется для хранения всей необходимой при использовании приложения информации).

Клиентской частью является мобильное приложение для платформы Android. Именно с него пользователь совершает действия, которые обрабатываются сервером приложений.

Для реализации сервера приложений необходимо выбрать хостинг, куда будут посылаться запросы с клиентского приложения, и где будут храниться файлы, содержащие логику обработки посылаемых с приложения запросов. Хостинг – услуга по предоставлению ресурсов для размещения информации на сервере, постоянно находящемся в сети Интернет. Был выбран хостинг на домене jino.ru, так как требовался недорогой и надежный хостинг, предоставляющий просто и понятный интерфейс для работы с данными на сервере и для работы с базой.

Сервер баз данных на выбранном хостинге использует язык структурированных запросов (SQL), так как он удобен для описания всех логических подмножеств базы данных. Язык SQL предназначен для:

* создания базы данных и таблиц в ней с полным описанием их структуры;
* выполнения основных операций для работы с данными (вставка, обновление, удаление);
* выполнения простых и сложных запросов.

Данные, передаваемые между клиентом и сервером, представлены в формате JSON, где данные представляют собой набор пар «ключ: значение», что является очень удобным для их обработки как на клиентском приложении, так и на сервере приложений.

## Активности приложения

Активности в приложении под операционной системой Android – одни из фундаментальных составляющих любого приложения. Они служат точкой входа для взаимодействия пользователя с приложением, а также играют ключевую роль в навигации пользователя в приложении. Умение правильно управлять активностями является гарантией стабильной и удобной для пользователя работы приложения, например, необходимо, чтобы смена ориентации экрана в приложении происходило плавно, не нарушая работу пользователя, чтобы пользовательские данные не терялись во время переходов между активностями и так далее.

Каждая активность в приложении имеет свое предназначения и определенные функции, с которыми пользователь может работать. Основными методами любой активности являются:

* onCreate(Bundle) – метод, который инициализирует активность. Самый важный метод, в котором обычно вызывается подметод setContentView(int), которому в качестве параметра подается id макета или слоя для представления визуальной структуры пользовательского интерфейса текущей активности. Также немало важен метод findViewById(int), который служит для инициализации отдельного компонента, описанного в файле макета текущей активности;
* onPause() – метод, который срабатывает, когда пользователь переключает одну активность на другую, и первая из них ставится «на паузу». Любые действия, которые пользователь совершил в активности, перед уходом из нее, сохраняются и отображаются после её возобновления.

Все активности между собой организуют стек активностей. Когда новая активность начинает работу, она помещается на верхушку стека и становится главной активностью. Предыдущая активность запоминает свое положение в стеке и не возьмет на себя управление до тех пор, пока не произойдет выход из активности, которая находится непосредственно над ним в стеке.

Жизненный цикл активности представлен на рисунке 9.

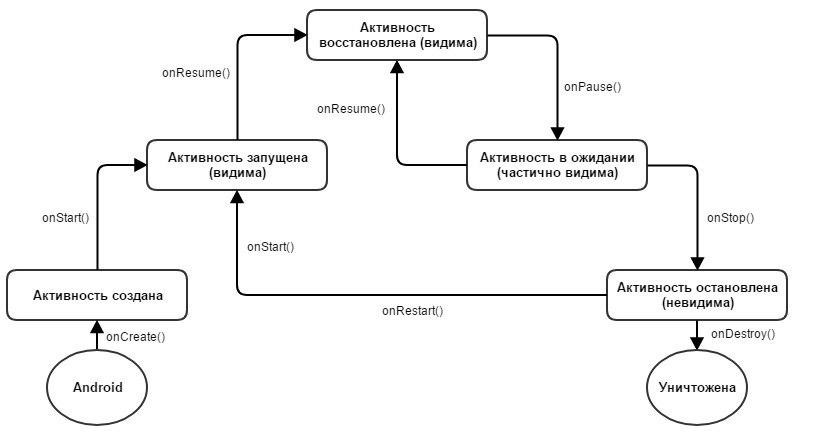


Рисунок 9 – Жизненный цикл активности

Активность имеет 4 состояния:

а) если активность находится на верхушке стека активностей, то она в

данный момент находится в активном состоянии;

б) если активность частично скрыта другой активностью, которая не полностью занимает экран устройства, то первая активность в данный момент находится в состоянии ожидания. Активности в состоянии ожидания полностью работоспособны, но могут быть приостановлены в ситуациях нехватки оперативной памяти;

в) если активность полностью скрыта другой активностью, то он находится в приостановленном состоянии. Она по-прежнему хранит в себе информацию о своем состоянии, но в том случае, когда устройству будет не хватать оперативной памяти, активность будет полностью остановлена или уничтожена;

г) если активность находится в состоянии ожидания или приостановленном состоянии, система может уничтожить ее в случае нехватки устройству оперативной памяти. Когда активность вновь переходит в активное состояние, она полностью перезапускается и возвращает предыдущее состояние.

Чтобы стандартный Java-класс был активностью приложения, он должен унаследовать библиотеку AppCompatActivity.

Все классы активности должны быть прописаны в главном манифесте приложения AnroidManifest.xml с тегом <activity>, в котором обязательно указывается имя класса активности и в общем случае указываются заголовок активности и ее тема.

В разработанном приложении реализованы следующие активности:

* активность входа в учетную запись пользователя LoginActivity.java;
* активность регистрации пользователя RegisterActivity.java;
* активность выбора уровня развития LevelSelectionActivity.java;
* активность главного экрана уровня развития LevelActivity.java;
* активность чата пользователя с куратором ChatActivity.java;
* активность предоставления дополнительных материалов к заданию TaskStuffActivity.java;
* активность таблицы рекордов по тому или иному уровню развития RecordsActivity.java;
* активность личных настроек пользователя SettingsActivity.java.

Также реализована вспомогательная стартовая активность Start.java. Здесь происходит инициализация объекта prefs стандартного класса SharedPreferences, который служит для хранения и доступа к некоторым

сохраненным настройкам, необходимым для дальнейшей работы приложения, например, имя авторизированного пользователя или статус авторизированного пользователя. Также в активности Start.java инициализируется объект dbHelper созданного класса DBHelper, необходимый для работы с встроенной базой данных на устройстве.

### Вход/регистрация

Для того чтобы начать использовать приложение, пользователю необходимо авторизоваться, либо зарегистрироваться, если пользователь ещё не создавал учетную запись в разработанном приложении. Активность входа является стартовой активностью, если пользователь не был авторизован ранее. Также переход на активность входа осуществляется, когда пользователь выходит из своего профиля из контекстного меню активности главного экрана уровня развития. Внешний вид активности входа в учетную запись пользователя изображен на рисунке 10.

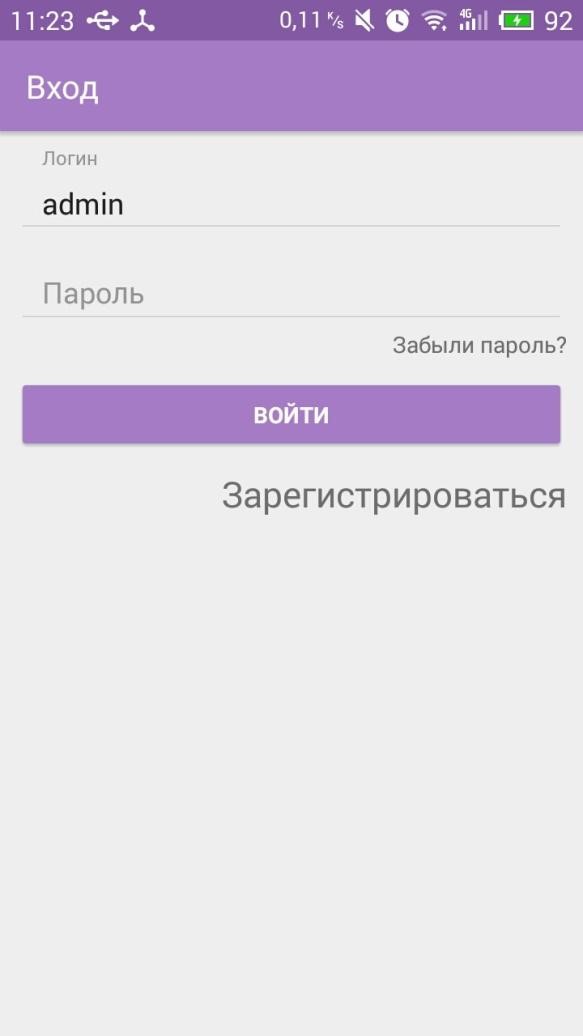


Рисунок 10 – Активность входа в учетную запись пользователя

Активность входа пользователя описана в файле LoginActivity.java и является стартовой активностью, которую видит пользователь при первом запуске приложения. На экране данной активности имеется 2 поля для ввода (EditText), кнопка для входа (Button) и две ссылки для перехода на активность регистрации и запроса пользователя на восстановление пароля.

Макет активности входа описан в файле activity\_login.xml.

В случае запроса на восстановление пароля пользователю во всплывающем диалоговом окне предлагается ввести электронный почтовый ящик, который указывался при регистрации. На этот почтовый ящик ему будет отправлено письмо с новым случайно сгенерированным паролем. Поля для ввода в активности входа предлагают пользователю ввести его логин, который он указывал при регистрации и пароль. Как только пользователь нажимает кнопку «войти», форма авторизации скрывается и на ее место помещается полоса загрузки авторизации. На сервер базы данных отправляется запрос на вход пользователя, обработка которого описана в файле серверной части login.php. Здесь проверяется соответствие введенного логина и пароля с содержащимися логинами и паролями в базе данных. Если проверка прошла успешно, то приложение продолжает свою работу, осуществив переход на активность выбора уровня развития.

Активность регистрации пользователя описана в файле

RegisterActivity.java. На экране данной активности имеется 6 полей для ввода текста (EditText), метка, указывающая на обязательные поля для ввода (TextView) и кнопка регистрации (Button). Макет активности регистрации описан в файле activity\_register.xml. Внешний вид активности регистрации учетной записи пользователя изображен на рисунке 11.

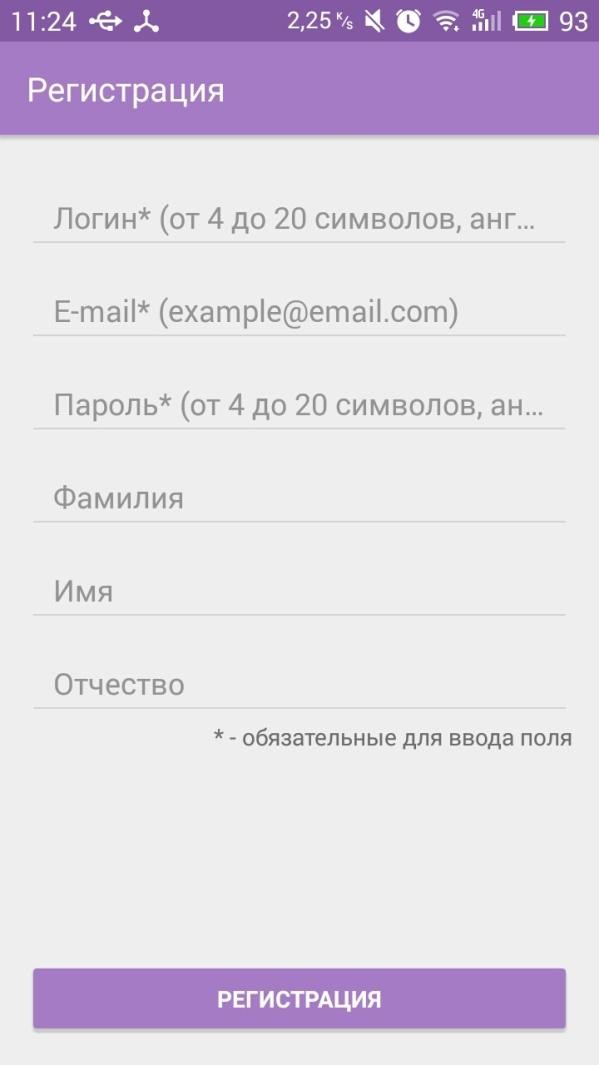


Рисунок 11– Активность регистрации учетной записи пользователя Процедура регистрации предполагает собой ввод пользователем уникального логина для авторизации в приложении (от 4 до 20 символов латинского алфавита без пробелов), электронного почтового ящика пользователя, пароля (от 4 до 20 символов латинского алфавита без пробелов) и имени пользователя для обращения к нему. Имя пользователя состоит из его фамилии, имени и отчества. Все поля при регистрации являются обязательными, за исключением полного имени пользователя, его можно указать при желании.

Как только пользователь нажимает кнопку «регистрация», форма регистрации скрывается и на ее место помещается полоса загрузки регистрации. На сервер базы данных отправляется запрос на регистрацию пользователя, обработка которого описана в файле серверной части registration.php. Логин и почта каждого пользователя должны быть уникальными, поэтому перед тем, как одобрить процедуру регистрации, проверяется отсутствие введенного логина и почтового ящика в базе данных. Если введенный логин или почта уже существует в базе данных, то пользователю возвращается сообщение о соответствующей ошибке во всплывающем сообщении. Если же такого логина и почты нет в базе данных, введенные пользователей данные заносятся в базу данных, при этом пароль шифруется функцией PHP password\_hash() для его защищенности, и приложение продолжает свою работу, осуществив переход на активность выбора уровня развития.

По завершении процедуры авторизации или регистрации в объект сохраненных настроек prefs класса SharedPreferences заносятся флаг авторизации, id пользователя, имя пользователя и статус доступа к основным четырем уровням развития, который при регистрации пользователя помечается как запрещенный. Далее осуществляется переход на активность выбора уровня развития пользователя.

### Выбор уровня развития

Активность выбора уровня развития предоставляет пользователю возможность перехода на активность желаемого уровня развития. Переход на активность выбора уровня развития пользователя осуществляется после того как пользователь прошел процедуру авторизации или регистрации. Также она является стартовой активностью после перезапуска приложения, как только пользователь стал авторизован. Внешний вид активности выбора уровня развития изображен на рисунке 12.



Рисунок 12 – Активность выбора уровня развития

Активность описана в файле LevelSelectionActivity.java. Представляет собой простую и понятную навигацию по уровням развития в виде пяти кнопок (Button), каждая из которых соответствует своему уровню развития.

Макет активности выбора уровня развития описан в файле activity\_level\_selection.xml.

Всего 5 уровней развития: базовый уровень, физический уровень, материальный уровень, социальный уровень и духовный уровень.

Изначально, непосредственно после регистрации пользователя, ему доступен для прохождения только базовый уровень. Базовый уровень необходим для оценки сил и возможностей пользователя на остальных четырех уровнях и для оценки начального уровня прогресса на каждом уровне развития. Оценка происходит в ходе выполнения пользователем заданий на базовом уровне и в ходе общения с его куратором, который и выносит решение об уровне прогресса пользователя на каждом из уровней развития.

Как только пользователь проходит основной этап базового уровня, выполнив необходимые задания, ему становятся доступны остальные четыре уровня развития. Но перед этим пользователю на каждом уровне развития куратор присваивает определенный уровень прогресса, исходя из продвижения пользователя на базовом уровне развития. В дальнейшем базовый уровень остается активным для пользователя, и он может продолжить развиваться на этом уровне.

При нажатии на кнопку, соответствующую какому-либо уровню развития, осуществляется переход на активность главного экрана выбранного уровня развития.

### Главный экран уровня развития

Активность экрана уровня развития представляет собой основную информацию о текущем прогрессе авторизированного пользователя на выбранном уровне развития. На активность экрана уровня развития пользователь переходит из активности выбора уровня развития. Содержание активности зависит от того, какой именно уровень развития выбрал пользователь.

Активность описана в файле LevelActivity.java. Включает в себя фото профиля пользователя (ImageView), имя и статус пользователя (TextView), блок текущего задания, блок доступных в данных момент для выполнения заданий. Снизу экрана активности располагается шкала текущего прогресса пользователя на выбранном уровне развития (ProgressBar). Макет активности главного экрана уровня развития описан в файле activity\_level.xml. Внешний вид активности главного экрана уровня развития на примере материального уровня изображен на рисунке 13.

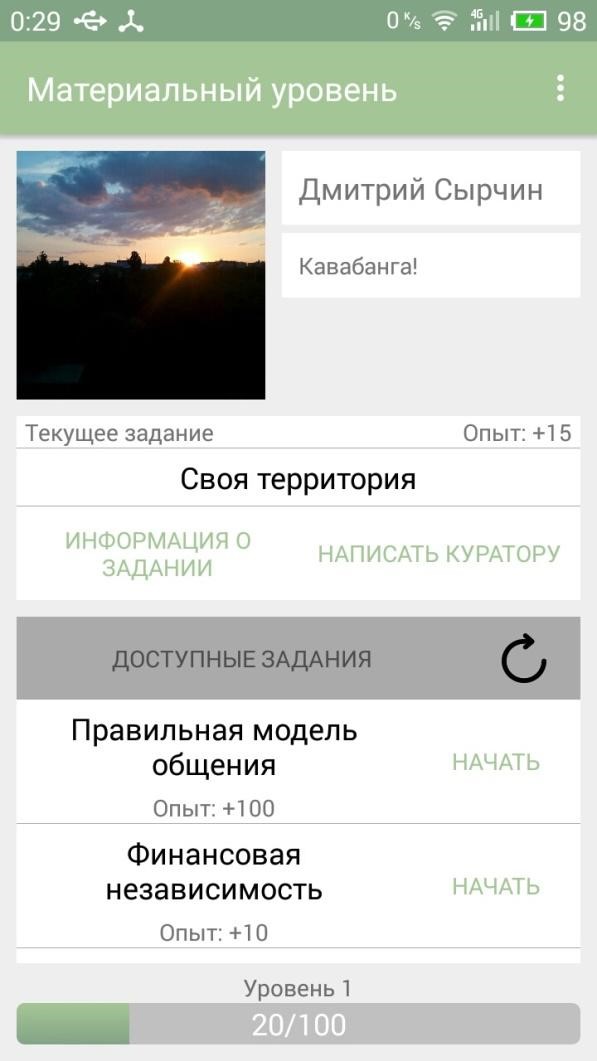


Рисунок 13 – Активность главного экрана уровня развития на примере материального уровня развития

Фото пользователя устанавливается им по желанию. Изначально для каждого пользователя фото отсутствует. При нажатии на стандартную иконку фото пользователю предлагается диалоговое окно, представляющее собой выбор одного из трех действий: открыть фото, сменить фото (галерея), сменить фото (камера).

При нажатии на действие «открыть фото» фото разворачивается на весь экран для его детального рассмотрения.

При нажатии на действие «сменить фото (галерея)» производится переход на вспомогательную активность, которая осуществляет переход в галерею фотографий на устройстве пользователя для выбора одной фотографии и установки ее в качестве фото своего профиля. Реализация перехода на активность выбора фото из галереи фотографий выглядит следующим образом:

intent = new Intent (Intent.ACTION\_PICK,

android.provider.MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI); startActivityForResult(intent, PICK\_IMAGE\_REQUEST);

Как только фото было выбрано, происходит переход на дополнительную активность загрузки изображения на сервер, описанную в файле ImageUploadActivity.java. Данная активность состоит из предварительного просмотра загружаемого фото, которое помещается в ImageView, и кнопки «загрузить» для подтверждения и отправки фото на сервер (рисунок 14). При нажатии на кнопку «загрузить» загружаемое изображение кодируется с помощью Java функции Base64.encodeToString() и посылается на сервер в виде зашифрованной строки. Обработка изображения в виде зашифрованной строки на сервере описана в файле серверной части image\_upload.php.

Сначала строка декодируется с помощью PHP функции base64\_decode($image), а затем изображение с уникальным названием, которое имеет вид user\_+ <имя пользователя> + .jpg, помещается в папку на сервере, где хранятся все фотографии пользователей, загрузившись фото своего профиля. По завершению загрузки сервер посылает клиенту ответ об успешной загрузке, после чего активность загрузки изображений завершает свою работу. Загруженное фото в виде зашифрованной строки помещается в

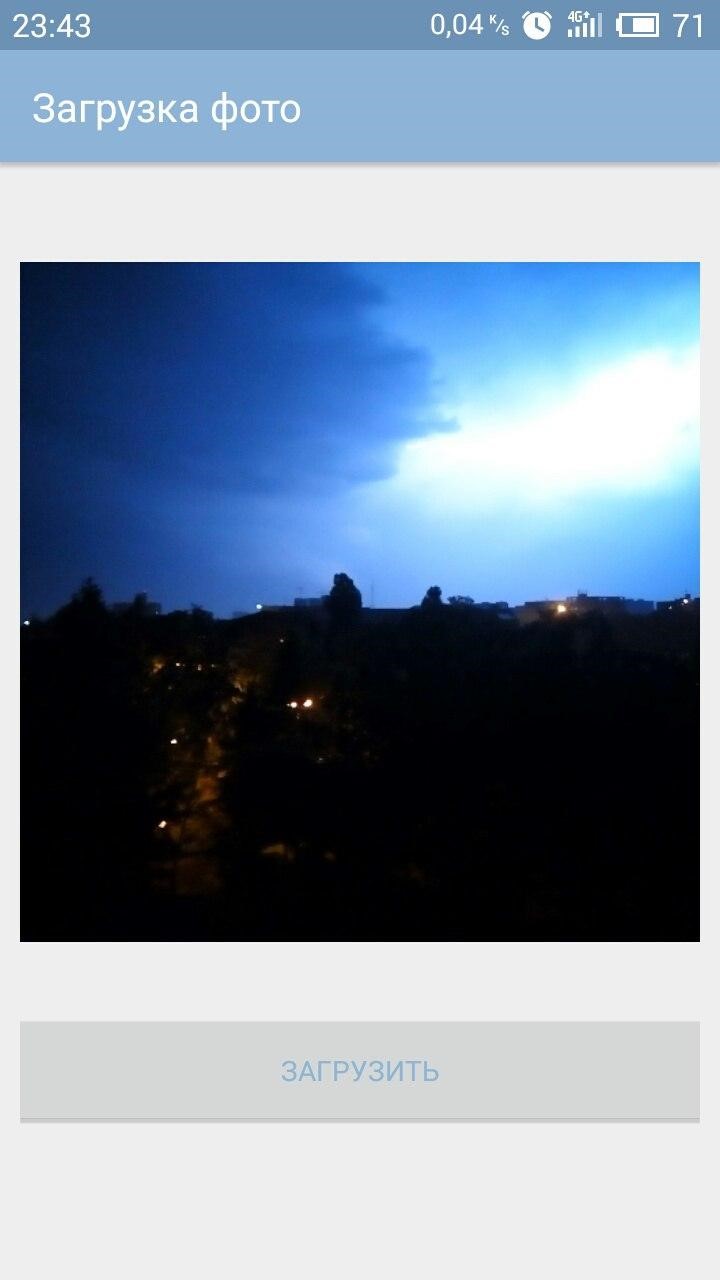


Рисунок 14 – Активность загрузки изображения на сервер

локальную базу данных SQLite. Это необходимо для того, чтобы при повторном переходе на активность уровня развития не происходило дополнительного обращения к серверу базы данных и фото профиля пользователя не загружалось повторно. В итоге управление снова переходит на активность главного экрана уровня развития.

При нажатии на действие «сменить фото (камера)» производится переход на вспомогательную активность, которая запускает стандартное приложение камеры на устройстве пользователя, чтобы сделать фото.

Реализация перехода на данную активность выглядит следующим образом:

intent = new Intent(MediaStore.ACTION\_IMAGE\_CAPTURE); photoUri = Uri.fromFile(getOutputMediaFile()); intent.putExtra(MediaStore.EXTRA\_OUTPUT, photoUri); startActivityForResult(intent, PHOTO\_IMAGE\_REQUEST); После того, как пользователь сделает фото, оно, как и в случае с выбором фото из галереи, переносится и отображается на вспомогательной активности, описанной в файле ImageUploadActivity.java и предлагается загрузить сделанное фото на сервер. По завершению загрузки фото также в виде зашифрованной строки помещается в локальную базу данных SQLite, и управление переходит активности главного экрана уровня развития.

Рядом с фото профиля пользователя находятся два компонента TextView. В первом представлено имя авторизированного пользователя в виде связки имени и фамилии, а во втором компоненте находится его текущий статус, который пользователь может изменить в настройках своего профиля, переход в которые осуществляется из контекстного меню в правом верхнем углу экрана текущей активности.

Под фото профиля пользователя располагается блок текущего задания, который несет в себе информацию о задании, которое пользователь выбрал для выполнения. Информация состоит из названия текущего задания, количества рекомендуемого опыта, присуждаемого пользователю по завершению текущего задания. Под описанием задания располагаются 2 кнопки: «о задании» для перехода на активность, содержащую полную информацию по заданию, и «написать куратору» для перехода на активность личного чата с куратором.

Под блоком текущего задания пользователя располагается блок доступных для выполнения пользователем заданий, исходя из его текущего прогресса на выбранном уровне развития. Для пользователя отображается три случайно выбранных задания из списка доступных ему заданий, хранящихся в базе данных. Каждое доступное задание содержит в себе его название и рекомендуемое количество опыта, присуждаемое пользователю в ходе завершения выполнения задания. При нажатии на доступное задание происходит переход на активность полной информации о нем. Пользователь в любой момент может обновить список доступных ему заданий, если его не устраивает текущий список. Новые доступные задания также подгружаются из базы данных, учитывая текущий уровень прогресса пользователя на выбранном уровне развития. Задания, которые пользователь уже выполнил или выполняет в текущий момент, не загружаются. Во время загрузки отображается полоса загрузки для визуального отображения посланного запроса на сервер и ожидания с ответа с сервера. Также напротив каждого доступного для выполнения задания имеется кнопка «начать», по нажатию которой пользователь опрашивается, уверен ли он, что хочет начать выполнять данное задание. Если пользователь соглашается и в данный момент он выполняет одно из заданий и хочет его заменить, то по согласию пользователя у него отнимается количество опыта, равное половине рекомендуемого количества опыта в заменяемом текущем задании. Выполняется запрос на сервер, подразумевающий начало задания пользователем. Обработка данного запроса описана в файле серверной части start\_task.php. Если запрос выполнен успешно, то выбранное пользователем

задание становится текущим для выполнения.

Блок доступных заданий по умолчанию находится в свернутом состоянии, если пользователь в данный момент выполняет одно из заданий. Если же пользователь не выполняет ни одно из заданий в данный момент, блок доступных для выполнения заданий находится по умолчанию в развернутом состоянии.

Блоки текущего задания и доступных для выполнения заданий объединены в один общий прокручиваемый вниз или вверх блок. Данных ход является полезным в том случае, когда информация, содержащаяся в блоке текущего задания или блоке доступных заданий, не помещается на экране активности и выходит за ее пределы.

Шкала текущего прогресса пользователя на выбранном уровне привязана к нижней части активности. Шкала предоставляет информацию о текущем уровне прогресса пользователя, текущем количестве опыта и максимальном количестве опыта, которое может получить пользователь на текущем уровне прогресса. Шкала меняет свое значение каждый раз, когда пользователь завершает задание и куратор выносит решение о количестве начисляемых пользователю очков опыта. Как только шкала достигнет максимального на данном уровне прогресса значения, пользователь получает новый уровень прогресса, где максимальный уровень прогресса будет выше, чем на предыдущем уровне прогресса.

Также на активности главного экрана уровня развития присутствует контекстное меню, которое находится в правом верхнем углу экрана активности. Контекстное меню включает в себя три ссылки:

а) Таблица рекордов (при выборе осуществляет переход на активность таблицы рекордов, где пользователь может узнать текущий прогресс развития всех пользователей на данном уровне развития);

б) Настройки (при выборе осуществляет переход на активность

настроек, где пользователь может изменить информацию о себе);

в) Выйти (при выборе осуществляет выход из профиля и переход на активность авторизации, при следующем запуске приложения будет запускаться активность авторизации).

### Чат с куратором и интерфейс Pusher

Активность чата с куратором представляет собой окно диалога авторизированного в приложении пользователя с назначенным ему куратором на выбранном уровне развития. На активность чата пользователь переходит из активности главного экрана уровня развития с помощью нажатия на кнопку «написать куратору», которая находится в блоке текущего задания пользователя.

Активность описана в файле ChatActivity.java. Состоит из прокручиваемого вертикально блока (ListView), содержащего отправленные

сообщения между собеседниками, поля для ввода сообщения для отправления (EditText) и кнопки отправления введенного сообщения. Макет активности чата с куратором описан в файле activity\_chat.xml. Внешний вид активности чата с куратором изображен на рисунке 15.

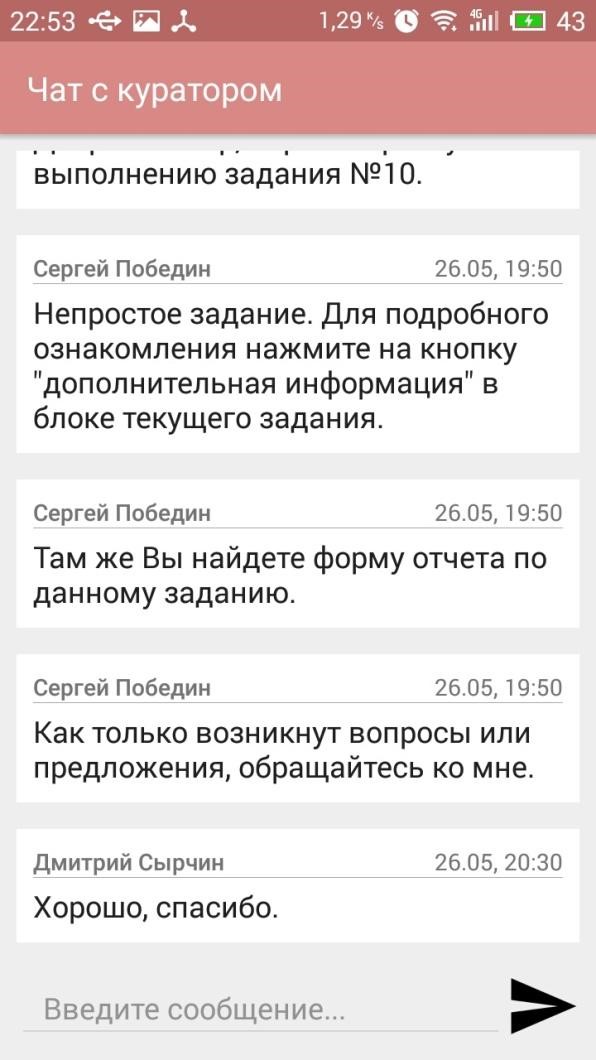


Рисунок 15 – Активность чата с куратором

Каждое сообщение блока отправленных сообщений является объектом вспомогательного созданного класса ChatMessage.java и представляет собой отдельный блок, содержащий в себе информацию об имени отправителя сообщения, времени и дате отправления сообщения и сам текст сообщения.

Сам блок отправленных сообщений принимает в качестве параметра адаптер сообщений, описанный во вспомогательном созданном классе ChatArrayAdaper.java. Адаптер берет предоставленные ему сообщения и размещает их в списке по порядку. При этом адаптер в своем теле создает необходимые для отображения компоненты, а именно всю информацию о сообщении, и помещает в них подготовленный текст.

Для удобства использования чата необходимо, чтобы активность чата с куратором обновлялась и добавляла новые передаваемые сообщения в реальном времени. Для этого использовался интерфейс Pusher.

Pusher – простой и удобный программный интерфейс приложения для быстрой, легкой и надежной интеграции возможностей работы в реальном времени с помощью веб-сокетов в веб-приложения, мобильные приложения или любое другое устройство, имеющее доступ в сеть Интернет. Веб-сокеты, используемые в работе данного программного интерфейса, уникально идентифицируют приложение и обеспечивают двунаправленную связь с приложением с минимальным использованием данных.

Интерфейс Pusher располагает богатым набор библиотек, которые можно использовать при разработке продукта. Здесь используется абстракция, основанная на мгновенных событиях, которая упрощает привязку пользовательского интерфейса к событиям, которые исполняются с любого клиента или сервера.

Главная идея программного интерфейса Pusher – модель

публикаций/подписок с использованием каналов, что позволяет фильтровать и управлять отправляемыми запросами. Также поддерживаются механизмы аутентификации для частных каналов и отслеживания тех пользователей, кто сейчас эти каналы прослушивает.

Таким образом, интерфейс Pusher является крайне удобным для реализации чата в реальном времени, так как пользователи при общении могут мгновенно получать сообщения друг от друга, не обновляя ничего вручную и используя при этом минимальное количество данных.

Для начала работы с интерфейсом Pusher необходимо

зарегистрироваться на официальном сайте данного программного интерфейса и далее зарегистрировать приложение в личном кабинете. После этого выдаются реквизиты или токены, необходимые для подключения к каналу прослушивания интерфейса Pusher. В токены входят уникальный идентификатор зарегистрированного приложения, уникальный ключ доступа, секретный ключ и кластер, то есть регион, куда будут посылаться запросы на обработку событий.

Для операционной системы Android инициализация канала прослушивания в приложении с помощью интерфейса Pusher выглядит следующим образом:

PusherOptions options = new PusherOptions(); options.setCluster("eu");

Pusher pusher = new Pusher(/\*уникальный ключ доступа\*/, options); Channel channel = pusher.subscribe(/\*канал прослушивания\*/); channel.bind(/\*событие\*/, new SubscriptionEventListener() {

@Override

public void onEvent(String channelName,

String eventName, final String data) {

System.out.println(data);

} });

pusher.connect();

Сначала указываются опции, необходимые для подключения. В данном случае указывается регион eu – Европа. Создается экземпляр класса Pusher, который принимает в качестве параметров уникальный ключ доступа и объявленные опции. Затем создается экземпляр класса Channel, которые характеризуется подпиской на определенный канал прослушивания, и, наконец, регистрируется событие для прослушивания на обозначенном канале прослушивания.

В разработанном приложении чат представляет собой индивидуальное общение пользователя с назначенным ему куратором. По каждому уровню развития свой отдельный чат. Поэтому для уникального обозначения канала прослушивания событий используются два параметра: логин пользователя и уровень развития. Например, логин пользователя – user1, уровень развития – материальный, следовательно, канал прослушивания будет носить название, messages\_user1\_material. Тогда к этому каналу будет иметь доступ этот пользователь и куратор, прикрепленный к пользователю по данному уровню развития.

Каждый раз при переходе на определенном уровне развития на активность чата с куратором будет производиться подключение к каналу прослушивания с уникальным обозначением, характерным для этого пользователя и уровня. Далее на сервер базы данных отправляется запрос, в результате которого подгружаются последние сообщения, отправленные между текущим пользователем и его куратором. В каждом сообщении указывается информация об имени отправителя сообщения, времени и дате отправления сообщения и сам текст сообщения.

Главное событие, которое обрабатывается интерфейсом Pusher в разработанном приложении, является событие new\_message, обозначающее отправление нового сообщения со стороны пользователя или куратора в чат между ними. Как только сообщение было отправлено с любой стороны, оно переправляется на сервер для обработки, в ходе которой сообщение заносится в базу данных с параметрами:

* текущий уровень развития (physical, material, social, spiritual);
* id пользователя;
* id прикрепленного к пользователю куратора по данному уровню развития;
* id отправителя (пользователь или куратор отправил сообщение);
* текст сообщения;
* время и дата отправления сообщения.

Далее на сервере создается объект класса Pusher для обозначения канала прослушивания, на который отправляется основная информация о сообщении, а именно имя отправителя сообщения, текст сообщения и время и дата отправления сообщения. После чего создается событие интерфейса Pusher, которое принимает в качестве параметров имя канала прослушивания, имя события и пересылаемые данные.

$options = array(

'cluster' => 'eu',

'encrypted' => true

);

$pusher = new Pusher(

/\*ключ доступа\*/,

/\*секретный ключ\*/,

/\*идентификатор приложения\*/,

$options

);

$data = "";

$data["message\_user"] = $message\_user\_name;

$data["message\_text"] = $message\_text;

$data["message\_time"] = $message\_time;

$pusher->trigger($channel\_name, 'new\_message', $data);

Интерфейс принимает эту информацию и мгновенно сообщает пользователю, который в данный момент прослушивает канал, о новом событии и подгружает новое сообщение на экран активности чата.

Для уменьшения объема расходуемых данных канал прослушивается только при открытой активности чата.

### Информация о задании

Активность информации о задании представляет собой указание подробного описания выбранного задания и вспомогательной информации, которая может быть полезна пользователю при выполнении этого задания.

Активность описана в файле TaskStuffActivity.java. Состоит из прокручиваемого вертикально блока, содержащего в себе несколько полей, которые подробно описывают задание, выбранное пользователем. Макет активности информации о задании описан в файле activity\_task\_stuff.xml.

Главным полем активности информации о задании является поле подробного описания задания. Здесь указывается вся необходимая информация, которая раскрывает суть выбранного задания, а именно действия и шаги, которые необходимо сделать пользователю для выполнения этого задания.

Перед полем подробного описания задания находится поле, содержащее название задания, которое было отображено на активности главного экрана уровня развития. Далее указывается цель выбранного задания, то есть чего пользователь должен достичь в ходе завершения задания и какие навыки он может приобрести в ходе его выполнения.

Под полем подробного описания задания располагается поля отчета о задании. В этом поле указывается, в какой форме пользователь должен предоставлять свои успехи в ходе выполнения своему куратору на выбранном уровне развития. По этому отчету куратор выносит решение о продвижении пользователя на выбранном уровне развития и присуждении ему количества очков опыта для повышения уровня прогресса на этом уровне развития.

Также для некоторых заданий доступны ссылки на книги и статьи в сети Интернет, которые либо могут помочь пользователю в выполнении выбранного задания, либо являются необходимыми материалами для его выполнения.

Последним идет поле минимального уровня пользователя, с которого доступно выполнение текущего задания. Задания, имеющие больший уровень прогресса, после которого становится доступно его выполнение, чем уровень прогресса, который имеет авторизированный пользователь, не будут отображаться в списке доступных для выполнения заданий.

### Таблица рекордов

Активность таблицы рекордов представляет собой отображение уровня прогресса каждого зарегистрированного пользователя на выбранном уровне развития в порядке убывания уровня прогресса пользователей. На активность таблицы рекордов пользователь переходит из контекстного меню, которое находится на активности главного экрана уровня развития в правом верхнем углу экрана. Внешний вид активности таблицы рекордов изображен на рисунке 16.

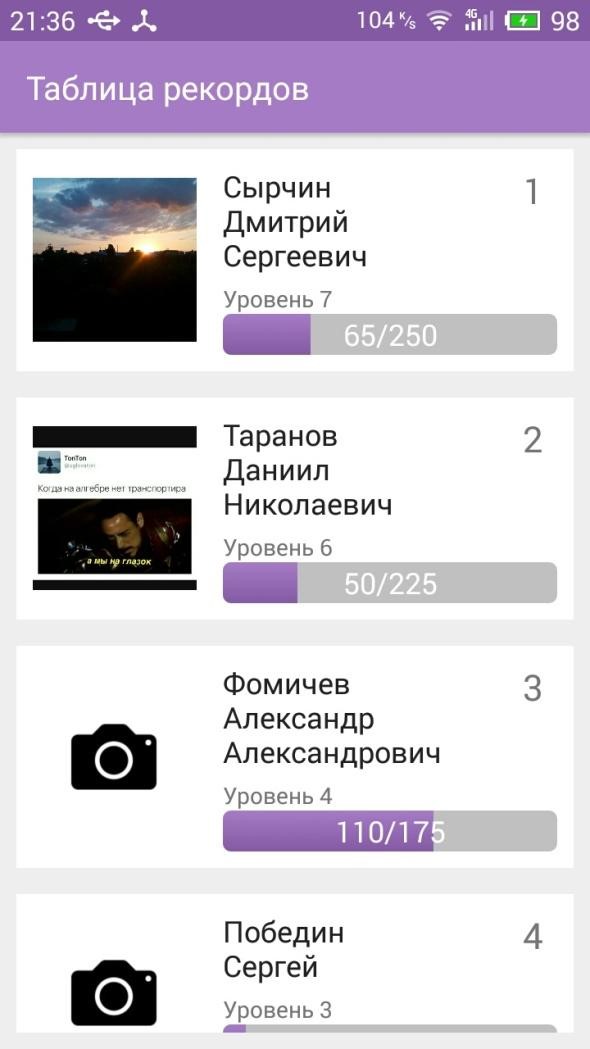


Рисунок 16 – Активность таблицы рекордов

Активность описана в файле RecordsActivity.java. Состоит из прокручиваемого вертикально блока (ListView), содержащего уровни прогресса всех пользователей на выбранном уровне развития. Макет активности таблицы рекордов описан в файле activity\_records.xml.

Каждый элемент прокручиваемого блока является отдельным блоком, который содержит в себе информацию о текущем уровне прогресса конкретного пользователя на выбранном уровне развития. Слева блока расположено фото пользователя, которое он установил в качестве фото своего профиля. Справа вверху указано число, которое означает положение пользователя в таблице рекордов по выбранному уровню развития. Непосредственно справа от фото находится основная информация о каждом пользователе в таблице рекордов. Указывается имя пользователя и его степень прогресса на данном уровне развития, а именно его уровень развития, полученный опыт на текущем уровне развития и максимальный опыт, который может получить пользователь на текущем уровне развития.

Всякий раз при переходе на активность таблицы рекордов происходит обращение к серверу с запросом на получение данных для наполнения таблицы рекордов. При отправке запроса на сервер и ожидания ответа с него отображается полоса загрузки. Обработка запроса описана в файле серверной части get\_records.php. В ответ на запрос пользователь получает массив уровней прогресса каждого пользователя на уровне развития, в котором пользователь перешел на таблицу рекордов. В этот массив входят фото каждого пользователя (если он имеется), имя пользователя, его уровень прогресса на данном уровне развития вместе с текущим и максимальным количеством опыта. Для получения ответа с сервера необходимо подключение устройства к интернету. Если оно отсутствует, то пользователь получит сообщение об ошибке и для повторного запроса ему необходимо отправить запрос повторно.

### Настройки

Активность настроек представляет собой отображения личных установок пользователя, которые он может изменить по своему усмотрению. Переход на активность настроек осуществляется из контекстного меню, которое находится на активности главного экрана уровня развития в правом верхнем углу экрана. Внешний вид активности настроек изображен на рисунке 17.

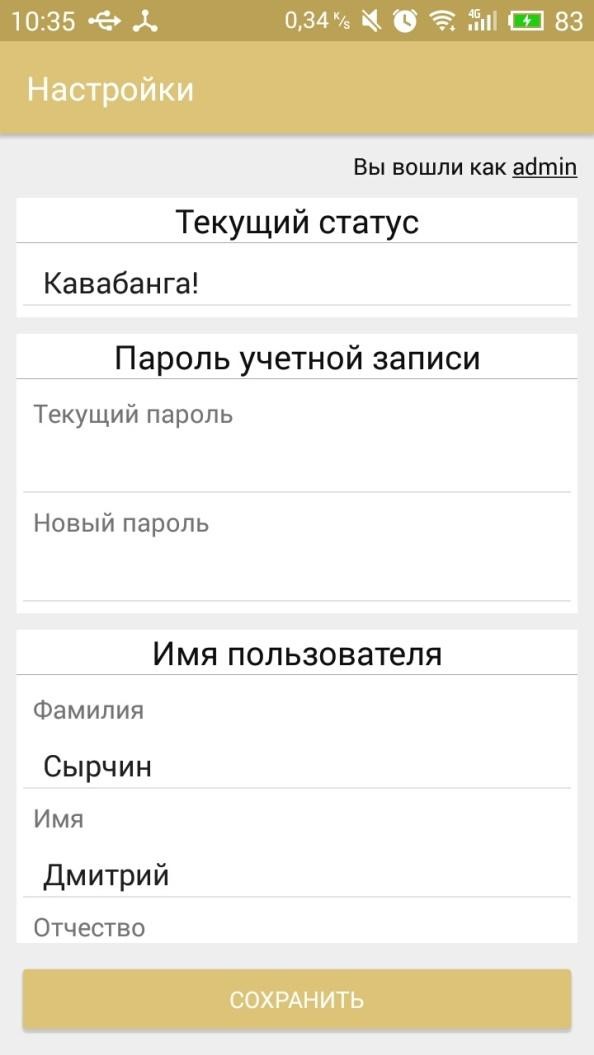


Рисунок 17 – Активность настроек

Активность описана в файле Settings.java. Состоит из нескольких компонентов редактируемого текста (EditText), идущих друг за другом. Среди них присутствует редактирование пароля учетной записи пользователя, личного имени пользователя (имени, фамилии и отчества) и статуса пользователя. Внизу экрана активности находится кнопка «сохранить» для сохранения изменений, внесенных пользователем. Макет активности таблицы рекордов описан в файле activity\_settings.xml.

Пользователь способен изменить пароль своей учетной записи на любой другой по своему усмотрению. Длина пароля должны быть в пределах от 4 до 20 символов, пароль дожжен содержать латинские буквы и не содержать пробелов. Для смены пароля пользователю необходимо ввести свой текущий пароль, установленный для авторизированной учетной записи.

По нажатию на кнопку «сохранить» внизу экрана активности настроек пользователю предлагается согласиться с его выбором, после которого отправляется запрос на сервер для изменения личных настроек пользователя. Обработка запроса изменения настроек описана в файле серверной части accept\_settings.php, Перед тем, как занести новые настройки, проверяется соответствие введенного текущего пароля (если таков был введен) с паролем, хранящимся на сервере базы данных. Если соответствие установлено, то новые настройки утверждаются и пользователю возвращается ответ об успешном их изменении. Во время обработки запроса появляется полоса загрузки, которая активна до тех пор, пока пользователь не получит ответ с сервера.

По возвращению с активности настроек на активность главного экрана уровня развития пользователь видит уже установленные им новые настройки, а при следующей его авторизации пользователю необходимо ввести в поле пароля новый установленный пароль.

## Локальная база данных SQLite

Локальная база данных необходима приложению для хранения большого количества структурированных данных и представляет собой обычную базу с таблицами и запросами, хранящуюся на устройстве. Она облегчает работу с приложением в плане того, что нет необходимости делать лишние запросы на сервер для получения данных, а достаточно лишь обратиться к файлу, хранящемуся на устройстве. Вследствие этого значительно уменьшается расход интернет трафика.

Локальная база данных описана в файле DBHelper.java, где

инициализируется класс DBHelper, который является наследником класса SQLiteOpenHelper. Этот класс позволяет работать с методами для создания или обновления базы данных. Содержит два основных метода:

* onCreate() – метод, который вызывается, если базы данных, к которой происходит обращение, не существует;
* onUpgrade() – метод, который вызывается, если происходит обращение к базе данных, которая имеет более новую версию, чем существующая.

Основными таблицами локальной базой данных в классе DBHelper

являются:

а) таблица levels – хранит уровни прогресса, доступные для получения пользователем и максимальные значения опыта, которых может достичь пользователь на соответствующем уровне развития. Данные в эту таблицу загружаются при первой авторизации пользователя и проверяются на наличие обновлений при запуске приложения;

б) таблица tasks – хранит информацию о заданиях всех уровней развития пользователя. Каждое задание несет в себе информацию о принадлежности к определенному уровню развития, id задания в базе данных на сервере, основной информации о задании, минимальном уровне пользователя, необходимом для выполнения данного задания и рекомендуемом количестве опыта, которые присуждается пользователю по завершению задания. Данные в эту таблицу загружаются при первой авторизации пользователя и проверяются на наличие обновлений при повторном запуске приложения;

в) таблица user\_level\_stats – хранит информацию о текущем прогрессе пользователя на уровнях развития, а именно название уровня развития, уровень прогресса пользователя, текущий опыт пользователя и текущее выполняемое пользователем задание на данном уровне развития. Данные в эту таблицу загружаются при первой авторизации пользователя и обновляются по мере продвижения пользователя на одном из уровней развития;

г) таблица messages – хранит историю переписки авторизированного пользователя с кураторами на каждом уровне развития. Данные в эту таблицу загружаются при первом обращении к активности чата с куратором на каждом уровне развития и обновляются при отправлении сообщения или при получении сообщения;

д) таблица user\_photo – хранит информацию о текущем фото профиля авторизированного пользователя. Фото хранится в зашифрованном виде. Также присутствует поле, которое ссылается на наличие фото профиля у текущего пользователя. Если пользователь не устанавливал фото своего профиля, то на место фото профиля ставится значок отсутствия фото.

ЗАКЛЮЧЕНИЕ

Проанализировав уже имеющиеся решения по проблеме развития человека и его личности, можно сказать, что разработанное приложение является уникальным в своем роде и, несомненно, найдет своих потребителей среди разных слоев населения.

В рамках данной квалификационной работы были достигнуты следующие результаты:

а) Был проведен анализ реализованных решений по выдвинутой проблеме, находящихся в свободном доступе в магазин приложений Google

Play.

б) Было создано мобильное приложение, представляющее следующий

функционал:

1. Возможность продвижения в развитии человека и его личности по четырем уровням развития: физический, материальный, социальный и духовный;
2. Возможность общения с куратором по каждому из уровней развития для получения помощи и успешного выполнения поставленных заданий;
3. Получение пользователями очков опыта и уровней прогресса на каждом уровне развития для формирования игрового формата приложения.

в) Мобильное приложение было протестировано и показало свою работоспособность как на эмуляторах, запущенных на компьютере, так и на реальных устройствах на базе операционной системы Android.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Таненбаум Э. Распределенные системы. Принципы и парадигмы – СПб.: Питер, 2003. – 880с.
2. Медникс З., Дорнин Л. Программирование под Android – СПб.: Питер, 2012. – 496с.
3. Голощапов А. Л. Google Android. Программирование для мобильных устройств – СПб.: Питер, 2012 – 448с.
4. Котляров В. П. Основы тестирования программного обеспечения – Издательство Бином, 2000 – 288с.
5. Дейтел П. Android для разработчиков – СПб.: Питер, 2012 – 512с.
6. Activity | Android Developers [Электронный ресурс]. Статья, URL: https://developer.android.com/reference/android/app/Activity.html (дата обращения: 10 февраля 2017)
7. Android working with Camera [Электронный ресурс]. Статья, URL: http://www.androidhive.info/2013/09/android-working-with-camera-api/ (дата обращения: 21 февраля 2017)
8. Build an Android Chat App with Pusher [Электронный ресурс]. Статья, URL: https://blog.pusher.com/build-an-android-chat-app-with-pusher/ (дата обращения: 19 марта 2017)
9. Хранение данных. SQLite [Электронный ресурс]. Статья, URL: http://startandroid.ru/ru/uroki/vse-uroki-spiskom/74-urok-34-hranenie-dannyhsqlite.html (дата обращения: 22 марта 2017)
10. Концепции развития личности [Электронный ресурс]. Статья, URL: http://psihomed.com/kontseptsii-razvitiya-lichnosti/ (дата обращения 10 апреля 2017)