Оглавление

ВВЕДЕНИЕ.....6

1. АНАЛИЗ ЗАДАЧИ.....7

1.1. Актуальность разработки.....7

1.2. Роль приложений и платформы Android в современном обществе 8

1.3. Технологии построения API для взаимодействия с ИС.....11

1.4. Словесное описание задачи.....12

1.5. Структура программы.....13

1.6. Расчет FP-метрик и LOC-метрик.....16

1.7. Расчет COCOMO-МЕТРИК.....22

1.8. Модель прецедентов.....25

2. ПРОЕКТИРОВАНИЕ.....27

2.1. Проектирование базы данных API сервера.....27

2.2. Построение диаграммы классов хранения и обработки данных Android приложения.....31

2.3. Построение диаграммы последовательности Android приложения.....34

3. РЕАЛИЗАЦИЯ.....37

3.1. Настройка рабочего пространства.....37

3.1.1. *Android Studio.....37*

3.1.2. *Java Development Kit.....39*

3.1.3. *Debian.....39*

3.1.4. *Панель управления Webmin.....40*

3.1.5. *Протокол L2TP.....40*

3.1.6. *Файловый сервер Samba.....41*

3.2. Использование симметричного шифрования.....41
3.3. Разработка интерфейса приложения.....42
3.4. Разработка API сервера.....51
ЗАКЛЮЧЕНИЕ.....53
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....54
ПРИЛОЖЕНИЕ А.....56
ПРИЛОЖЕНИЕ Б.....58
ПРИЛОЖЕНИЕ В.....85
ПРИЛОЖЕНИЕ Г.....93
ПРИЛОЖЕНИЕ Д.....105
ПРИЛОЖЕНИЕ Е.....115
ПРИЛОЖЕНИЕ Ж.....125

Введение

Важной проблемой в учебном процессе в высшем учебном заведении является задача автоматизации учебного расписания. Использование информационных систем в учебном секторе, в наше время, не является редкостью. В современном вузе достаточно много процессов подлежат автоматизации, начиная от совсем мелких задач, заканчивая полной автоматизацией ВУЗа.

Системы автоматизации в ВУЗах внедряются в первую очередь для повышения качества образования, объектами автоматизации могут быть как преподавательский состав так и администрация университета.

В разных ВУЗах процесс автоматизации происходит по разному, несмотря на то что понятие образовательной деятельности едино для всех образовательных учреждений. Значительное влияние оказывает финансирование автоматизации и готовность сотрудников ВУЗа использовать предлагаемых рынком информационных услуг и программных продуктов.

Разработка мобильного android-приложения "Кабинет РГУПС" обеспечит легкость и доступность пользования расписанием, позволит преподавателям и студентам, как очникам, так и заочникам быстро ориентироваться в нагрузке своего рабочего дня, а также планировать ближайшие дни заранее.

1. Анализ задачи

1.1. Актуальность разработки

Изменение средств и форм обучения, а так же стремительный прогресс средств вычислительной техники, диктуют необходимость внедрения информационных технологий в образование.

Следует начать с определения понятия «автоматизация». Автоматизация это применение технических и программных средств, освобождающих человека от непосредственного участия в процессах получения, преобразования, передачи и использования информации. Формализация предшествует процессу автоматизации, то есть необходимо получить полный набор однозначно трактуемых инструкций, исходя из которых достигается результат реализации процесса.

Одно из главных преимуществ автоматизации - при ускорении выполнения операций снижается количество ошибок при их выполнении, снижаются издержки на реализацию операций и повышение качества. Автоматизация может считаться успешной, в результате внедрения и использования которой удалось возвратить инвестированные в нее средства.

Этапы автоматизации, применяемые в целом к деятельности того или иного учреждения. Все этапы требует осмысленного и последовательного выполнения.

Этап 1: постановка проблемы, оценка необходимости автоматизации и возможностей предприятия;

Этап 2: формирование требований к программно-аппаратному комплексу, выбор или реализация программного продукта и технического обеспечения;

Этап 3: внедрение программного продукта;

Этап 4: послегарантийное обслуживание программно-аппаратного комплекса.

В начале процесса автоматизации необходимо ясно и четко сформулировать свои требования к ней. Требуется определить какие именно функции необходимо автоматизировать. Так же необходимо учитывать, что при

внедрении автоматизированных систем снижается степень влияния человеческого фактора на выполнение тех или иных операций.

Необходимо выделить сотрудника из числа руководителей высшего звена, ответственного за проведение данного процесса и уполномоченного принимать решения по вопросам автоматизации.

Обязательным условием является выделение квалифицированных специалистов, отвечающих за разработку или внедрение программного продукта. Каждый отдельный специалист в рамках своих обязанностей должен уметь оперативно устранять возникающие в системе отказы, выявлять и устранять нарушения условий эксплуатации, решать те или иные проблемы пользователей.

Выбор среды разработки программного продукта не следует отделять от выбора технического обеспечения, на котором предстоит в дальнейшем работать. Не следует забывать об интеграции в единое информационное пространство, что позволяет использовать ранее реализованные компоненты и единую базу данных. Это ведет к повышению гибкости информационной системы в целом и снижает дублирование уже имеющейся информации.

Пред проектным обследованием учреждения обычно называют описанный процесс анализа деятельности ВУЗа. Модель организации стоится ходе обследованием организации, и описывает не только взаимодействие структурных единиц, но и информационные потоки, а так же операции.

После определения требований к программному продукту, необходимо определиться с будущей функциональностью полнотой будущей программы.

1.2. Роль приложений и платформы Android в современном обществе

XXI век – это век информационных технологий. Технологии прочно вошли в нашу жизнь: они окружают нас повсюду. Компания Google и ее инженеры, создали операционную систему, позволяющую разрабатывать

сторонние приложения, начиная от самых простых в виде фонарика и заканчивая распределенным расчетом вычислений.

Компания Google создала открытую операционную систему Android. Android (Андроид) – операционная система для множества устройств основанная на ядре Linux. Изначально разрабатывалась компанией Android Inc., которую купила Google.

Более 75 % рынка смартфонов, оснащены операционной системой Android. Каждый разработчик электронного устройства имеет возможность переделать Android под свое устройство, гарантируя совместимость своего оборудования со сторонними приложениями для этой ОС. Это оказалось выигрышной тактикой. Раньше, до выхода Android каждый производитель самостоятельно писал или покупал у кого-то операционную систему, и не мог обеспечить совместимость массы полезных программ, созданных программистами всего мира, то после выхода ОС Android перед производителями чаще встает вопрос, какую версию Android им нужно поддерживать.

Установка Android на телефоны дала возможность с разрабатывать новые модели мобильных устройств, расширяя функционал – как телефонов, так и самой операционной системы при этом, не теряя совместимости с массой полезных приложений. Появление программ, предназначенных для помощи пользователю в самых разных ситуациях, привело к тому, что человек, скорее всего, найдет приложение для практически любого случая в жизни. Пользователю предоставляются приложения для покупки билетов, социальных сетей, картографических сервисов. При использовании SIP телефонии связываться с людьми в любой части планеты практически бесплатно.

Все это, наряду с легкостью разработки приложений, делает рассматриваемую платформу одной из наиболее востребованных в современном обществе.

Раньше рынок мобильных платформ делили множество платформ мобильных ОС, то на текущий момент можно выделить только 2 основные платформы: iOS и Google Android, которые делят около 90% всего рынка. Доля Google Android достигает 75%, т.е. каждый третий смартфонов, работают под управлением мобильной платформы от Google.

В таблице 1 представлены данные по различным версиям Android, которые мы также используем для анализа, чтобы после подвести общий итог и сделать выбор. Статистика собрана за двухнедельный период на основе обращений к сервису Google Play.

Таблица 1. Распределение версий ОС Android, на 1-е августа 2016

Название версии	Уровень API	Распределение
1.5 Cupcake	3	0.2%
1.6 Donut	4	0.5%
2.1 Eclair	7	4.2%
2.2 Froyo	8	15.5%
2.3 – 2.3.2 Gingerbread	9	0.3%
2.3.3 – 2.3.7	10	60.3%
3.1 Honeycomb	12	0.5%
3.2	13	1.8%
4.0 – 4.0.2 Ice Cream Sandwich	14	0.1%
4.0.3 – 4.0.4	15	15.8%
4.1 Jelly Bean	16	0.8%

Разработка для платформы Android ведется преимущественно на языке Java. Для создания необходимо специальное программное обеспечение. Самые последние версии этого ПО можно загрузить на официальном сайте разработчика. К этому программному комплексу относятся такие инструменты как JRE (Java Runtime Environment) и JDK (Java Development Kit). JRE

представляет собой среду выполнения – минимальную реализацию виртуальной машины, в которой запускается и выполняется программный код. JDK – это в свою очередь целый набор инструментов, комплект разработчика приложений на языке Java. На самом деле, JRE также входит в состав JDK, равно как и различные стандартные библиотеки классов Java, компилятор javac, документация, примеры кода и разнообразные служебные утилиты. Весь набор ПО распространяется свободно и имеет реализацию для различных ОС. В состав JDK не входит среда разработки, разработчик должен установить ее отдельно. При разработке под ОС Android рекомендуется использовать среду разработки Android Studio от Google.

Невозможно отрицать, что в сфере информационных технологий вчерашние инновации – это сегодняшние привычные вещи. Чтобы идти «в ногу со временем», специалистам необходимо использовать концепцию постоянного обучения, и главным является своевременное освоение наиболее значительных, распространенных и перспективных технологий. Технологии разработки под данную ОС относятся именно к таковым. Можно сделать вывод о целесообразности разработки клиентского приложения на базе ОС Android.

1.3. Технологии построения API для взаимодействия с ИС.

API(интерфейс программирования приложений) - набор готовых классов, процедур, функций, структур, констант для использования во внешних программных продуктах.

API часто используется при разработке веб-приложений и сервисов. Как правило API определяет определенный набор HTTP-запросов, а для ответа используются такие структуры, как XML или JSON. Web API является синонимом веб-службы, основными технологиями работы которой являются SOAP и REST.

1.3.1. SOAP

SOAP –(Simple Object Access Protocol) – протокол основанный на XML, определяющий правила передачи данных между прикладными системами через интернет. SOAP устанавливает стандарт взаимосвязи клиента и сервера, в

основе заложен принцип вызова удаленных процедур, которые принимают различные параметры, в зависимости от которых возвращают значения. Для представления любой информации, передаваемой от клиента к серверу и наоборот, используется XML.

Разрабатывался с расчетом использования по протоколу HTTP , однако может задействовать иные транспортные протоколы, например SMTP.

1.3.2. REST

REST(Representational State Transfer) – это архитектурный стиль построения распределенных систем, описывающий принципы разработки программного обеспечения. REST использует протокол HTTP, данные передаются в виде распространенных стандартных форматов (HTML,XML,JSON). Основными достоинствами REST является его простота, а также кэширование, что увеличивает скорость и масштабируемость системы, функциональная совместимость. Из этих данных можно сделать выводы о том, что SOAP это целое семейство протоколов и стандартов, что говорит о его избыточности и тяжеловесности по сравнению с REST. SOAP использует HTTP как транспортный протокол, а REST основан на нем, из этого следует что все технологии такие как, кэширование на уровне сервера, масштабируемость, используются в архитектуре REST ,в то время как в SOAP необходимо искать другие средства. Так же преимуществом REST над SOAP является то, что SOAP привязан к XML, в то время как REST может использовать JSON и другие форматы данных.

1.4. Словесное описание задачи

В рамках данной работы рассматривается процесс систематизации информационных баз данных университета, которые используются при автоматизации расписания занятий факультетов высшего учебного заведения на сайте РГУПС. Происходит выгрузка расписание из информационной системы РГУПС, которая будет применима для android-приложения «Кабинет РГУПС» в качестве информационной базы.

Данная программа обеспечивает:

– оперативное получение и обновление расписания очного и заочного отделения, а также посещаемости, успеваемости и результатах тестирования студента;
– возможность поиска преподавателя и просмотра его пар;

– добавление неограниченного количества пользователей в приложение;
– добавление заметок к любым парам а так же прикреплении любых файлов к этим заметкам;
– возможность связаться с разработчиком через встроенный чат технической поддержки;

1.5. Структура программы

Для упрощения разработки и возможности дальнейшего портирования приложения под другие операционные системы было принято решение о разделении графической части приложение и части хранения и обработки данных.

Функциональная структура разрабатываемого нами программного обеспечения состоит из следующих модулей:

Графическая часть:

– модуль отображения расписания очной формы обучения;
– модуль отображения расписания заочной формы обучения;
– модуль отображения посещаемости;
– модуль отображения успеваемости;
– модуль отображения результатов тестирования;
– модуль настроек приложения;
– модуль виджета;
– модуль отображения текущей пары на панели уведомлений;
– модуль отображения и поиска преподавателей;
– модуль чата технической поддержки;

Часть хранения и обработки данных:

– модуль хранения и обработки посещаемости;
– модуль хранения и обработки расписания очной формы обучения;
– модуль хранения и обработки расписания заочной формы обучения;
– модуль хранения и обработки учетных данных;

– модуль хранения и обработки успеваемости;

– модуль хранения и обработки результатов тестирования;

– модуль синхронизации данных;

– сервисный модуль.

Модуль отображения расписания очной формы обучения обеспечивает отображение расписания, навигацию между днями и неделями, верхними и нижними неделями, просмотр произвольной даты с помощью календаря. В модуле так же выводится наглядная информация о посещаемости студента в виде цветовой подкраски каждой пары в зависимости от того был студент на паре или нет, в конце каждой недели имеется краткая сводка о посещаемости студента за эту неделю. Также модуль отображает расписание преподавателя если в дне присутствуют пару у заочников то в расписание вставляется пара заочной формы обучение с пометкой.

Модуль отображения расписания заочной формы обучения обеспечивает отображение расписания, навигацию между днями, неделями обучения.

Модуль отображения посещаемости обеспечивает отображение посещаемости студента в виде суммарной сводки за весь семестр и детальной, понедельной сводке с отображением информации за каждую неделю семестра с возможностью перехода в модуль отображения расписания для наглядного просмотра посещаемости студента.

Модуль отображения успеваемости обеспечивает отображение успеваемости студента за весь срок обучения с возможностью выбора семестра для просмотра успеваемости, а так же с группировкой результатов по типу аттестации.

Модуль отображения результатов тестирования обеспечивает отображение результатов тестирования за весь срок обучения с возможностью выбора семестра для просмотра результатов, а так же с группировкой в зависимости от вида тестирования.

Модуль настроек приложения позволяет детально настроить интерфейс приложения и выполнить экстренное обновление базы данных приложения в случае ее сбоя.

Модуль виджета позволяет на рабочем столе мобильного устройства отобразить информацию о текущей паре.

Модуль отображения текущей пары на панели уведомлений позволяет отобразить информацию о текущей паре в панели уведомлений.

Модуль отображения и поиска преподавателей позволяет произвести поиск по преподавателям университета с возможностью сортировки преподавателей по критериям.

Модуль чата технической поддержки позволяет через интерфейс приложения связаться с разработчиком.

Модуль хранения и обработки посещаемости выполняет функцию доступа к базе данных, выборку данных за семестр, выборку данных за неделю и за конкретный день.

Модуль хранения и обработки расписания очной формы обучения выполняет функцию доступа к базе данных, выборку расписания определенной группы, а так же расписание преподавателя

Модуль хранения и обработки расписания заочной формы обучения выполняет функцию доступа к базе данных, выборку расписания для группы, выборку расписания для преподавателя, выборку пары преподавателя в конкретный день.

Модуль хранения и обработки учетных данных выполняет функцию доступа к базе данных, выборку и удаление учетных записей зарегистрированных в приложении

Модуль хранения и обработки успеваемости выполняет функцию доступа к базе данных, выборку списка семестров, выборку успеваемости по конкретному семестру

Модуль хранения и обработки результатов тестирования выполняет функцию доступа к базе данных, выборку списка семестров, выборку результатов тестирования за конкретный семестр

Модуль синхронизации данных выполняет одностороннюю синхронизацию с сервером, обновляет расписание, посещаемость, успеваемость, результаты тестирования

Сервисный модуль выполняет внутренние задания, такие как запуск синхронизации по расписанию, обновление виджета.

1.6. Расчет FP-метрик и LOC-метрик

Метрика программного обеспечения – это мера, позволяющая получить численное значение некоторого свойства программного обеспечения или его спецификаций. Поскольку количественные методы хорошо зарекомендовали себя в других областях, многие теоретики и практики информатики пытались перенести данный подход и в разработку программного обеспечения.

Измерения помогают оценить, как продукт, так и сам процесс его разработки. В результате измерений определяется мера – количественная характеристика какого-либо свойства объекта. Некоторые измерения позволяют сразу определить свойства объекта. А остальные можно получить лишь за счет вычисления от значений опорных характеристик. Результаты подобных вычислений называют метриками. Зачастую понятие мера и метрика рассматривают как равноценные определения.

Цель этой деятельности – сформировать предварительные оценки, которые позволят:

– предъявить заказчику корректные требования по стоимости и затратам на разработку программного продукта;

– составить план программного проекта.

Размерно-ориентированные метрики прямо измеряют программный продукт и процесс его разработки. Основываются размерно-ориентированные метрики на LOC-оценках (Lines Of Code). LOC-оценка – это количество строк в программном продукте.

Достоинства размерно-ориентированных метрик:

– широко распространены;

– просты и легко вычисляются.

Недостатки размерно-ориентированных метрик:

– зависимы от языка программирования;
– требуют исходных данных, которые трудно получить на начальной стадии проекта;
– не приспособлены к непроцедурным языкам программирования.

Функционально-ориентированные метрики косвенно измеряют программный продукт и процесс его разработки. Вместо подсчета LOC – оценки при этом рассматривается не размер, а функциональность или полезность продукта.

Используется 5 информационных характеристик:

– количество внешних вводов. Подсчитываются все вводы пользователя, по которым поступают разные прикладные данные. Вводы должны быть отделены от запросов, которые подсчитываются отдельно;
– количество внешних выводов. Подсчитываются все выводы, по которым к пользователю поступают результаты, вычисленные программным приложением. В этом контексте выводы означают отчеты, экраны, распечатки, сообщения об ошибках. Индивидуальные единицы данных отчета отдельно не подсчитываются;
– количество внешних запросов. Под запросами понимают диалоговый ввод, который приводит к немедленному программному ответу в форме диалогового вывода. При этом диалоговый ввод в приложении не сохраняется, а диалоговый вывод не требует вычислений. Подсчитываются все запросы – каждый учитывается отдельно;
– количество внутренних логических файлов. Подсчитываются все логические файлы (т.е. логические группы данных, которые могут быть частью базы данных или отдельным файлом);
– количество внешних интерфейсных файлов. Подсчитываются все логические файлы, на которые ссылается данное приложение.

Каждой из выявленных характеристик ставится в соответствие сложность. Для этого характеристике назначается низкий, средний или высокий ранг, а затем формируется числовая оценка ранга.

Достоинства функционально-ориентированных метрик:

– не зависят от языка программирования;

– вычисляются на любой стадии проекта;

– FP – оценки легко пересчитать в LOC-оценки.

Недостатком функционально-ориентированных метрик является то, что результаты основаны на субъективных данных, используются не прямые, а косвенные измерения. Определение рангов сложности для внешних вводов, выводов, запросов и логических файлов приведены в таблице 1.1.

- количество внутренних логических файлов;

Транзакции: Внешние вводы						
Название ввода	Поля ввода и элементы данных	Кол-во элементов данных	Ссылки на файлы	Ранг	Кол-во вводов	Общая сложность (общ. ранг)
Авторизация	Кнопки: Вход; Поле ввода: Фамилия, Номер зачетной книжки	3	0-1	Средний = 4	3	3*4=12
Добавление заметки	Кнопка: Добавить; Поля: Файл, Заголовок, Текст	3	0-1	Средний = 4	3	3*4=12

Транзакции: Внешние выводы						
Название вывода	Поля вывода и элементы	Кол-во элемент	Ссылки	Ранг	Кол-во	Общая сложнос

	данных	ов данных	на файлы		вывод ов	ть (общ. ранг)
Расписание очников	Поле: ListView1 Кнопки: Календарь, Сегодня	3	0-1	Средний = 5	3	5*3=15
Расписание заочников	Поле: ListView1 Кнопки: Календарь, Сегодня	3	0-1	Средний = 5	3	5*3=15
Расписание преподавателей	Поле: ListView1 Кнопки: Календарь, Сегодня	3	0-1	Средний =5	3	5*3=15
Результаты тестирования	Поле: ListView1 Выборка: По семестрам	2	0-1	Средний =5	2	5*2=10
Успеваемость	Поле: ListView1 Выборка: По семестрам	2	0-1	Средний =5	2	5*2=10

Посещаемость	Поле: ListView1 Выборка: По семестрам Кнопки: Переход к неделе	3	0-1	Средний = 5	3	3*5=15

	Внутренние логические файлы					
Название файла	Поля ввода и элементы данных	Кол-во элементов данных	Кол-во элементов данных-записей	Ранг	Кол-во файлов	Общая сложность (общ. ранг)
Database.db	Очная форма обучения, заочная форма обучения, Пользователи, заметки, успеваемость, посещаемость, результаты тестирования		1	Низкий (7)	1	7*7=49

Внешних интерфейсных файлов нет.

Исходные данные для расчета сводятся в табл.:

Имя характеристики	Ранг, сложность, количество			
	Низкий	Средний	Высокий	Итого
Внешние вводы	0x3 = 0	12x2 = 24	0x6 = 0	= 24
Внешние выводы	80x4 = 0	0x7 =0	0x7 = 0	= 320
Внешние запросы	0x3 = 0	0x4 =0	0x6 =0	= 0
Внутренние логические файлы Внешние интерфейсные файлы	49x7 = 343 0x5 = 0	0x 10= 0 0x7 = 0	0x15 = 0 0x10 =0	= 343 = 0
			Общее количество S =	687

Исходные данные для расчета FP-метрик

Определение системных параметров приложения

Системный параметр	Описание	F_i
Передача данных		$F_1=3$
Обработка данных		$F_2=3$
Производительность		$F_3=3$
Распространенность		$F_4=0$
Скорость транзакций		$F_5=5$
Оперативный ввод данных		$F_6=5$
Эффективность работы		$F_7=3$

Оперативное обновление		$F_8=3$
Сложность обработки		$F_9=0$
Повторная используемость		$F_{10}=3$
Легкость инсталляции		$F_{11}=2$
Легкость эксплуатации		$F_{12}=3$
Разнообразные условия размещения		$F_{13}=3$
Простота изменений		$F_{14}=3$
	$\sum_{i=1}^{14} F_i$ =39	

После сбора всей необходимой информации приступаем к расчету FP-метрики.

$$FP = S\times (0{,}65+ 0{,}01 \times \sum_{i=1}^{14} F_i)= 687\times 1 = 687$$

Пересчет FP-оценок в LOC-оценки

LOC= FP×23=687*23= 15801

1.7. Расчет COCOMO-МЕТРИК

На сегодняшний день для оценки стоимости ПО используется несколько различных моделей. Одной из самых популярных, открытых и хорошо документированных моделей оценки стоимости ПО является COCOMO (Constructive Cost Model – конструктивная модель стоимости), разработанная Барри Боэмом. Эволюция COCOMO отражает то, каким образом эволюционировала экономика ПО последние 20 лет.

История модели COCOMO позволяет увидеть эволюцию приоритетов экономики программного обеспечения. COCOMO состоит из иерархии трех последовательно детализируемых и уточняемых форм. Первый уровень, базовый, подходит для быстрых ранних оценок стоимости разработки ПО и

обладает неточностью вследствие некоторых факторов, которые невозможно учесть на ранних стадиях разработки. Средний уровень COCOMO учитывает эти факторы, тогда как детальный уровень дополнительно учитывает влияние отдельных фаз проекта на его общую стоимость.

Общая характеристика масштабных факторов W_i приведена в таблице 1.5.

Таблица 1.5 – Характеристика масштабных факторов W_i

Масштабный фактор (W_i)	Пояснение	W_i
1 Предсказуемость, наличие прецедентов	Отражает предыдущий опыт организации в реализации проектов этого типа. Очень низкий (=5) означает отсутствие опыта. Сверхвысокий (=0) означает, что организация полностью знакома с этой прикладной областью	5 нет опыта

Продолжение таблицы 1.5

2 Гибкость разработки	Отражает степень гибкости процесса разработки. Очень низкий означает, что используется заданный процесс. Сверхвысокий означает, что клиент установил только общие цели	3 среднее
3 Разрешение рисков в архитектуре	Отражает степень выполняемого анализа риска. Очень низкий (=5) означает малый анализ. Сверхвысокий (=0) означает полный и сквозной анализ риска	3 среднее
4 Связность группы	Отражает, насколько хорошо разработчики группы знают друг друга и насколько удачно они совместно работают. Очень низкий (=5) означает очень трудные взаимодействия. Сверхвысокий, (=0) означает интегрированную группу, без проблем взаимодействия	0 – один в группе, сам студент выполняет
5 Зрелость процесса	Означает зрелость процесса в организации. Вычисление этого фактора может выполняться по вопроснику CMM	2 – низкое
–	$\sum_{i=1}^{5} W_i =$	13

Для расчета показателя степени B отражающего нелинейную зависимость затрат от размера проекта (от длины кода LOC), воспользуемся формулой:

$$B = 1{,}01 + 0{,}01* \qquad (1.3)$$

$$B = 1{,}01 + 0{,}01*13 = 1{,}14$$

Множитель поправки M_e зависит от набора формирователей затрат EM_i, перечисленных в таблице 1.6.

Таблица 1.6 – Формирователи затрат EM_i для раннего этапа проектирования

Обозначение	Название	EM_i
PERS	Возможности (способности) персонала	Среднее = 1
RCPX	Надежность и сложность продукта	Среднее = 1
RUSE	Требуемое повторное использование	Среднее = 1
PDIF	Трудность (сложность) платформы	Средняя = 1
PREX	Опытность персонала	Среднее = 1
FCIL	Средства поддержки	Среднее = 1
SCED	Сроки	Низкие= 0.5

Для расчета множителя поправки, зависящего от 7 формирователей затрат, характеризующих продукт, процесс и персонал, воспользуемся формулой:

Me = (1.4) M_e = *1*1*1*1*1*1*0,5 = 0,5*

Для расчета затрат на разрабатываемый продукт, воспользуемся формулой, составные части которой мы уже подсчитали:

$$ЗАТРАТЫ = A * M_e * РАЗМЕР^e, \quad (1.5)$$

где A – масштабный коэффициент, который равен 2,5;

РАЗМЕР – размер ПО выраженный в тысячах LOC.

$ЗАТРАТЫ = 2,5 \times 0,5 \times 15^{1,14} = 27,3$

Исходя из расчетов, один человек может выполнить поставленную задачу за 27 месяцев.

1.8. Модель прецедентов

Диаграмма прецедентов (диаграмма вариантов использования) в UML – диаграмма, отражающая отношения между актёрами 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. Основное назначение диаграммы – описание функциональности и поведения, позволяющее заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему.

При моделировании системы с помощью диаграммы прецедентов системный аналитик стремится:

– чётко отделить систему от её окружения;
– определить действующих лиц (актёров), их взаимодействие с системой и ожидаемую функциональность системы;
– определить в глоссарии предметной области понятия, относящиеся к детальному описанию функциональности системы (то есть, прецедентов).

Развернутые описания прецедентов структурированы и содержат большое количество деталей. При итеративной и эволюционной разработке на стадии анализа требований в этом формате представляют порядка 10% жизненно важных прецедентов. Обычно такое описание формируется на первом семинаре по анализу требований, после чего сразу же начинается реализация этих прецедентов в системе.

Для развернутого описания прецедентов существуют различные шаблоны форматирования. Однако, начиная с 1990-х годов прошлого столетия, чаще

всего используется шаблон, созданный автором популярной книги по моделированию на основе прецедентов Алистером Коберном (Alistair Cockburn). На основе этого стиля и представлен прецедент, расположенный ниже.

Диаграмма прецедентов, изображенная рисунке 1.2 демонстрирует основные прецеденты в рамках данной программы.

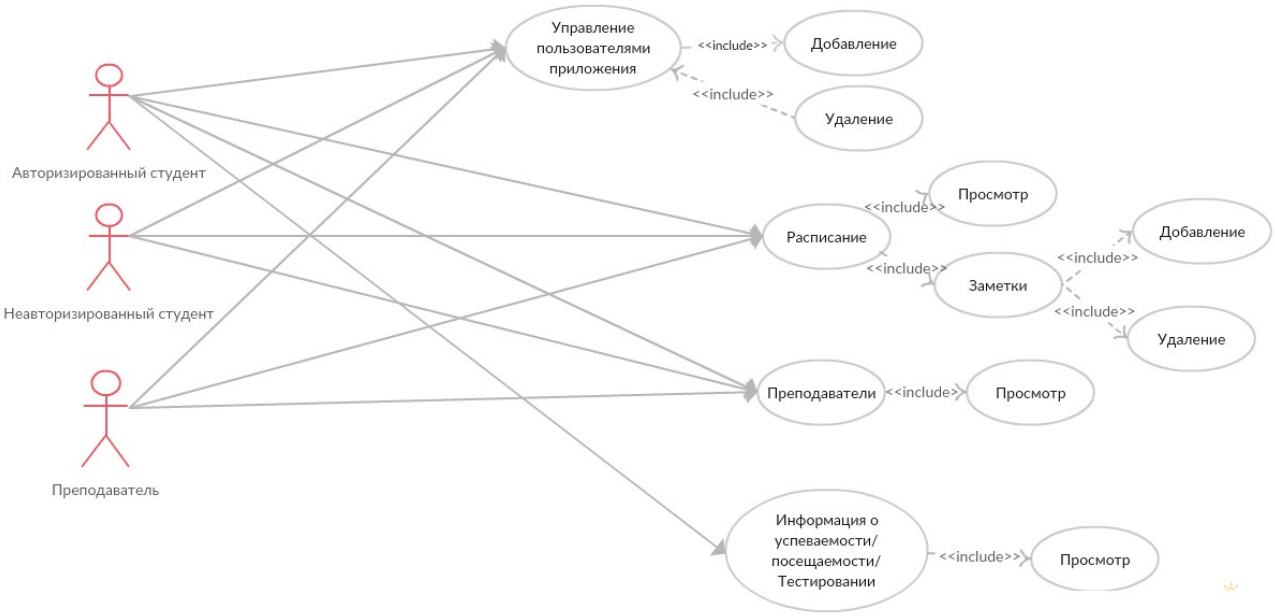


Рисунок 1.2 – Модель прецедентов

2. Проектирование

2.1. Проектирование базы данных api сервера

Основной составляющей объектно-ориентированного анализа при разработке программных продуктов является декомпозиция на отдельные объекты. Объектно-ориентированный анализ и проектирование принципиально отличаются от традиционных подходов структурного проектирования.

Модель предметной области – это визуальное представление концептуальных классов или объектов реального мира в терминах предметной области. Моделирование предметной области – один из начальных этапов

проектирования системы, необходимый для выявления, классификации и формализации сведений обо всех аспектах предметной области, определяющих свойства разрабатываемой системы.

В данной работе в соответствии с заданием автоматизируется доступ к расписанию студентом или преподавателем.

Для нормального функционирования сервер должен отдавать данные о расписании, если студент авторизирован то данные о посещаемости, успеваемости и результатах тестирования, также для нормальной функциональности api сервера должны быть служебные таблицы.

В результате проведенного анализа предметной области можно перечислить основные сущности этой БД. Так как на физическом уровне сущности соответствует таблица, то просто перечислим основные таблицы базы данных.

В модель проектированной БД (таблица 2.1) будут входить следующие таблицы (сущности): посещаемость, пользователи, тестирование, преподаватели, успеваемость, google токен, филиал, группы, конфигурация, заблокированные аккаунты.

Таблица 2.1 – Список сущностей

Название	Назначение
Посещаемость	Информация о посещаемости студентов
Пользователи	Информация о пользователе
Тестирование	Информация о тестировании
Преподаватели	Список преподавателей
Успеваемость	Информация о успеваемости
google токен	Информация о токене выдаваемая приложению облачной платформой google

Филиал	Информация о филиалах
Группы	Информация о группах
Конфигурация	Настройки выдаваемые приложению
Заблокированные аккаунты	Информация о заблокированных аккаунтах

Каждая сущность обладает своей именованной характеристикой или атрибутом. Атрибуты сущностей в нашей реляционной базе данных представлены в таблице 2.2.

Таблица 2.2 – Список атрибутов сущностей

Список атрибутов таблицы «Посещаемость»		
Ключевое поле	Название	Назначение
ПК	Идентификатор	Идентификатор записи
ВК	Номер зачетной книжки	Номер зачетной книжки студента
–	Год	Год
–	Номер недели	Номер недели
–	Заголовок недели	Заголовок который отображается студенту
–	День недели	День недели которому соответствует запись расписания
–	Номер пары	Номер пары которой соответствует запись
–	Статус	Статус посещения студента

Продолжение таблицы 2.2

Список атрибутов таблицы «Пользователи»		
ПК	Идентификатор	Идентификатор записи
ВК	Филиал	Идентификатор филиала
–	Фамилия	Фамилия студента для авторизации
ВК	Номер зачетной книжки	Номер зачетной книжки студента
–	Имя	Имя студента для вывода
–	Отчество	Отчество студента для вывода
–	Группа	Название группы для вывода
ВК	Идентификатор группы	Цифровое значение группы

–	Тип группы	Тип группы – очная или заочная
Список атрибутов таблицы «Тестирование»		
ПК	Идентификатор	Идентификатор записи
ВК	Филиал	Идентификатор филиала
ВК	Номер зачетной книжки	Номер зачетной книжки студента
–	Год	Год в котором произведено тестирование
–	Номер семестра	Номер семестра в котором произведено тестирование
–	Тип	Тип тестирования
–	Название предмета	Название предмета
–	Статус	Статус сдачи
Список атрибутов таблицы «Преподаватели»		
ПК	Идентификатор	Идентификатор записи
–	Идентификатор РГУПС	Идентификатор в системе РГУПС
–	Короткое имя преподавателя	Короткое имя преподавателя
–	Полное имя преподавателя	Полное имя преподавателя
–	Имя поиска	Служебное поле для поиска по хэшу имени
Список атрибутов таблицы «Успеваемость»		
ПК	Идентификатор	Идентификатор записи
–	Филиал	Идентификатор филиала
ВК	Номер зачетной книжки	Номер зачетной книжки студента
–	Год	Год в котором выставлена оценка
–	Тип	Тип сдаваемой работы
–	Название предмета	Название предмета
–	Статус	Статус сдачи
Список атрибутов таблицы «google токен»		
ПК	Идентификатор	Идентификатор записи
–	Филиал	Идетификатор филиала
ВК	Номер зачетной книжки	Номер зачетной книжки студента
–	Фамилия	Фамилия студента
–	Токен	Токен выданный облачной

		платформой
–	Последнее обновление	Дата последнего обновление
Список атрибутов таблицы «Филиал»		
ПК	Идентификатор записи	Идентификатор записи
–	Название	Название филиала
–	Статус	Технический статус работает система или нет
–	Расписание	Флаг доступности сервиса расписания
–	Расписание заочников	Флаг доступности расписания заочников
–	Авторизация студента	Флаг доступности авторизации студента
–	Тест	Флаг доступности информации о тестировании
–	Тех работы	Флаг технических работ на сервисе
–	Успеваемость	Флаг доступности информации о успеваемости
Список атрибутов таблицы «Группы»		
ПК	Идентификатор	Идентификатор записи
ВК	Филиал	Идентификатор филиала
–	Название группы	Название группы
–	Тип группы	Тип группы
Список атрибутов таблицы «Конфигурация»		
–	Начальная неделя	Информация о том с какой недели начинается учебный год
–	Минимальная версия приложения	Минимальная версия приложения
Список атрибутов таблицы «Заблокированные аккаунты»		
ПК	Идентификатор	Идентификатор записи
–	Фамилия	Фамилия студента
ВК	Номер зачетной книжки	Номер зачетной книжки
–	Время	Время блокировки

2.2. Построение диаграммы классов хранения и обработки данных Android приложения

Диаграммы классов используются при моделировании ПС наиболее часто. Они являются одной из форм статического описания системы с точки зрения ее проектирования, показывая ее структуру. Диаграмма классов не отображает динамическое поведение объектов, изображенных на ней классов. На диаграммах классов показываются классы, интерфейсы и отношения между ними.

Класс – это основной строительный блок ПС. Это понятие присутствует и в объектно-ориентированных языках программирования, то есть между классами UML и программными классами есть соответствие, являющееся основой для автоматической генерации программных кодов или для выполнения реинжиниринга. Каждый класс имеет название, атрибуты и операции. Класс на диаграмме показывается в виде прямоугольника, разделенного на 3 области. В верхней содержится название класса, в средней – описание атрибутов (свойств), в нижней – названия операций и услуг, предоставляемых объектами этого класса.

На основе технического задания, спроектированной уже базе данных и с учетом всей функциональной составляющей, были спроектированы и созданы основные классы. Описание классов приведено в таблице 2.3, а диаграмма классов изображена на рисунке 2.1, рисунке 2.2.

Таблица 2.3 – Описание классов Android приложения

Наименование	Описание
RaspisanieData	Модуль работы с расписанием
RaspisanieObj	Описание полей расписания
RaspisanieZaochnikData	Модуль работы с расписанием заочников
RaspisanieZaochnikObj	Описание полей расписания заочников
ProgressData	Модуль работы с успеваемость
ProgressObj	Описание полей расписания
TestData	Модуль работы с результатами тестирования
TestObj	Описание полей результатов тестирования
VisitsData	Модуль работы с посещаемостью

VisitsObj	Описание полей посещаемости
ZametkaData	Модуль работы с заметками
ZametkaFileObj	Описание полей файлов заметок
ZametkaObj	Описание полей заметок

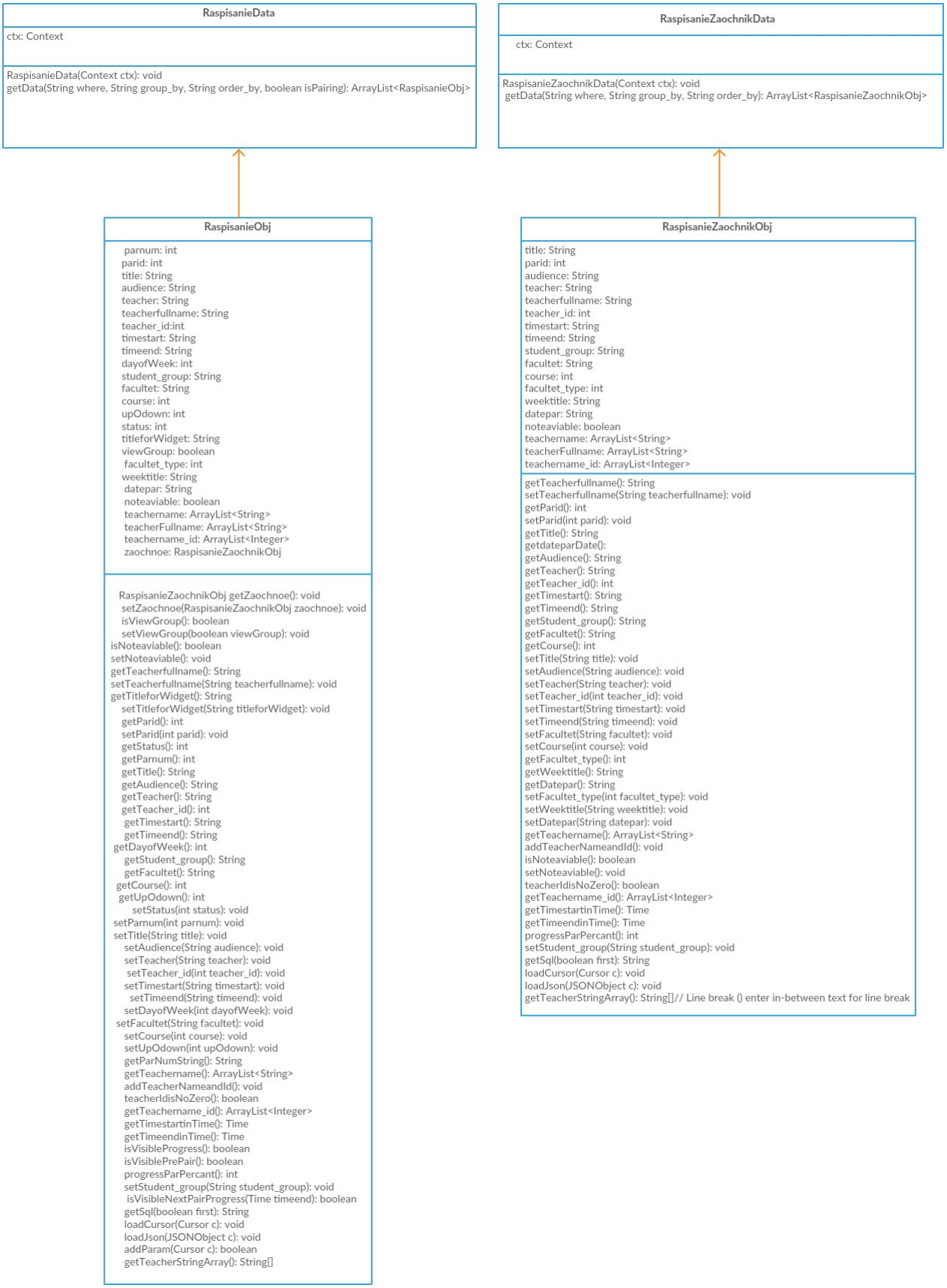


Рисунок 2.1 – Диаграмма классов Android приложения

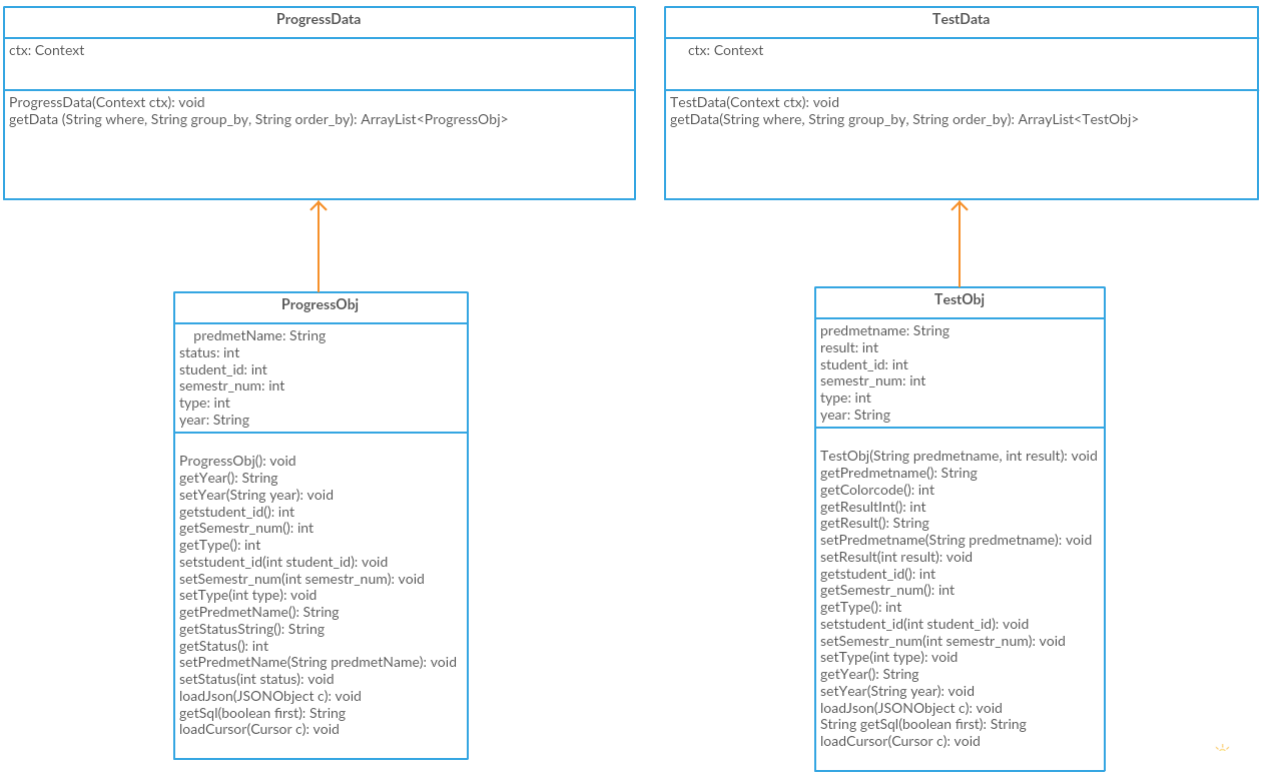


Рисунок 2.2 диаграмма классов Android приложения

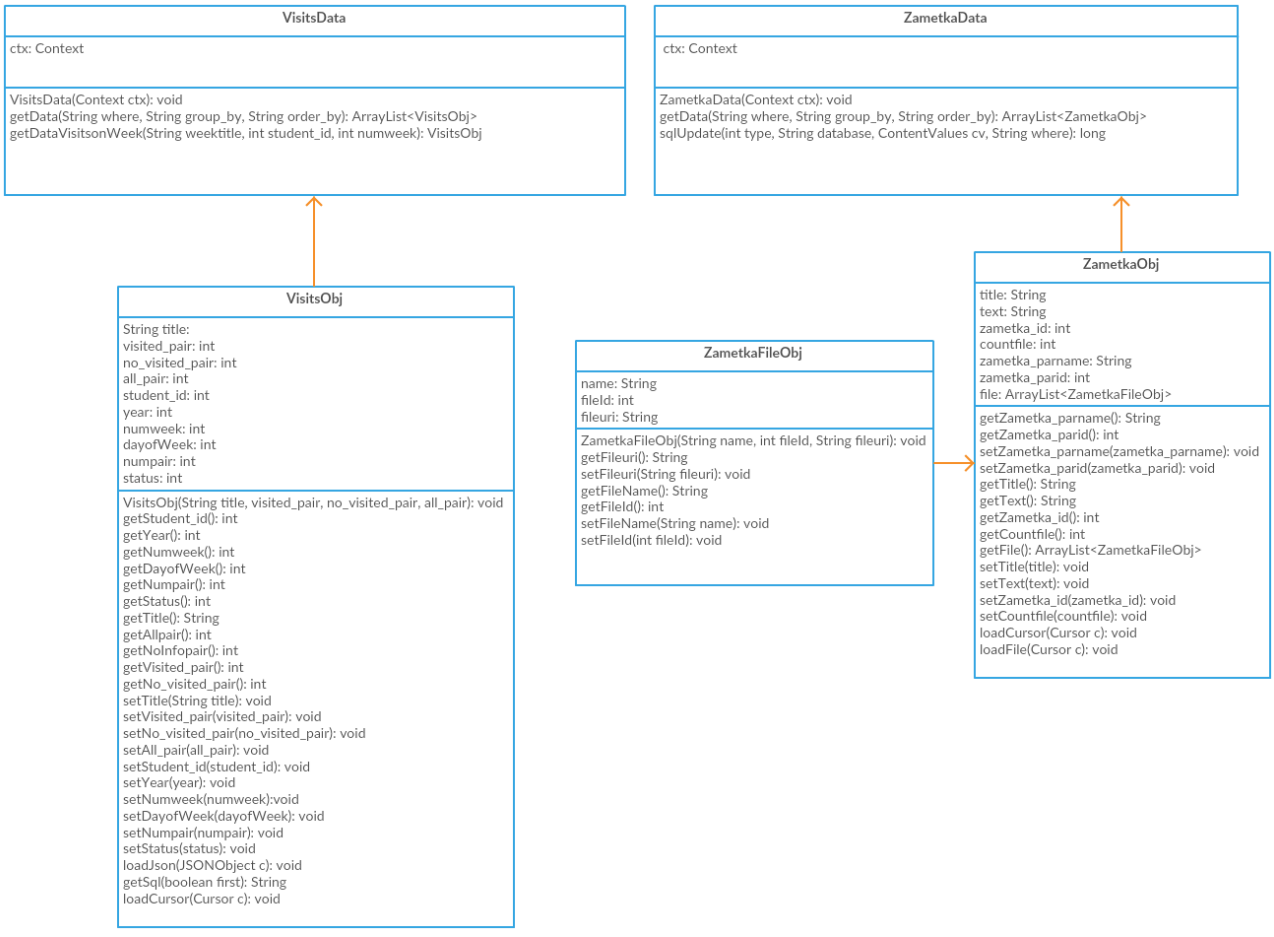


Рисунок 2.3 диаграмма классов Android приложения

2.3. Построение диаграммы последовательности Android приложения

Диаграмма последовательности – диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актёров (действующих лиц) информационной системы в рамках какого-либо определённого прецедента (отправка запросов и получение ответов).

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные «линии жизни», отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелки, показывающие обмен сигналами или сообщениями между объектами.

Выделив классы и отношения между ними, представим еще один вид UML диаграммы – диаграммы последовательностей (рисунок 2.2). На этой диаграмме изображены объекты классов, времена их жизни и посылаемые или возвращаемые сообщения между ними.

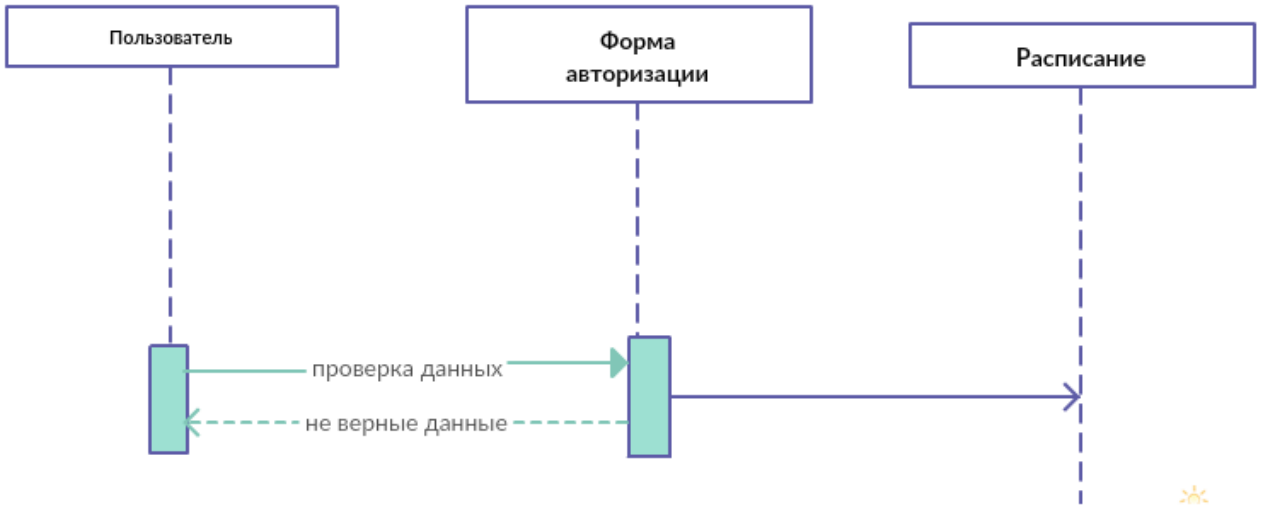


Рисунок 2.2 – «Диаграмма последовательности»

Как видно из диаграммы последовательности при запуске программы пользователь взаимодействует с окном авторизации, если данные для авторизации правильные то пользователь получает доступ к функционалу

приложения, если данные не верные то пользователю прийдется авторизироваться заново.

2.4. Диограмма деятельности

Диограмма деятельности это UML-диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов — вложенных видов деятельности и отдельных действий, соединённых между собой потоками, которые идут от выходов одного узла ко входам другого.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Диограмма деятельности разрабатываемого приложения изображена на Рисунке 2.3

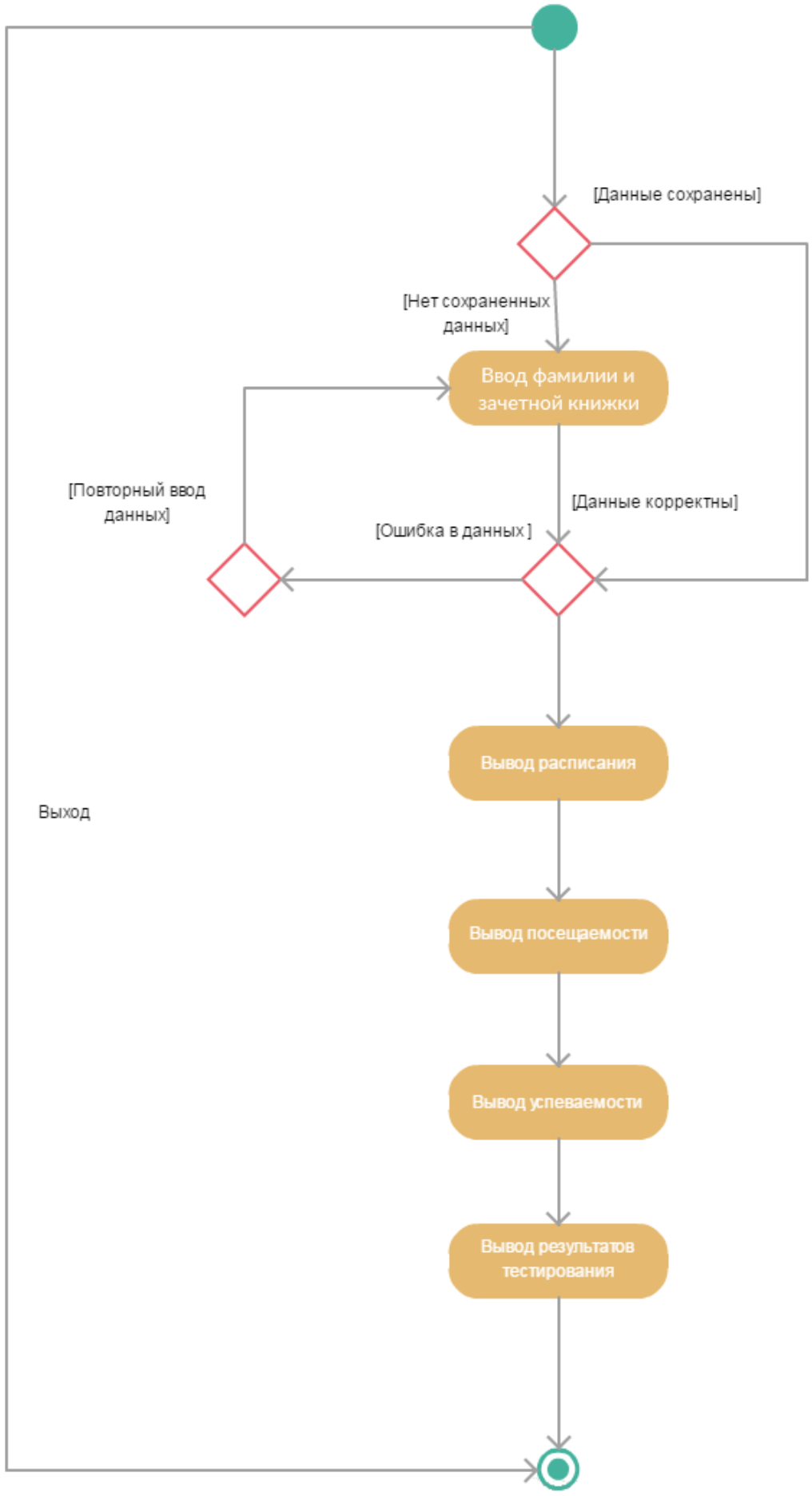


Рисунок 2.3 – диограмма деятельности

3. Реализация

3.1. Настройка рабочего пространства

Для разработки приложения под ОС Android нам понадобится настроить рабочее пространство под ОС Windows. Нам понадобится установить Java JDK и Android Studio.

Для разработки API нам необходимо настроить сервер который будет обрабатывать запросы, так как в университете обучается около 5000 человек и около 1000 человек преподавателей, и как мы выяснили в 1 главе, у 75% людей телефон на платформе Android, то около 4500 человек будет пользоваться приложением, это достаточно серьезная нагрузка на сервер. Для того чтобы проект выдержал такую нагрузку мы выберем виртуальный приватный сервер с такими параметрами:

- 2 ядра процессора Intel® Xeon® Processor E5450
- 2 Гб оперативной памяти
- 100 Гб SSD дискового пространства
- 1 Гбит порт в интернет
- Операционная система Debian 7.0
- Панель управления сервером Vesta и Webmin
- Для защиты административной части настроим VPN по протоколу L2TP
- Для удобства доступа к файлам сервера настроим Samba
- Для резервной копии настроим ежесуточное резервное копирование в облачное хранилище Yandex Disk объемом 200 Гб
- В качестве защиты от Ddos атак и распределенного CDN выберем сервис cloudflare

3.1.1. Debian

операционная система, состоящая из свободного ПО с открытым исходным кодом. В настоящее время Debian GNU/Linux — один из

самых популярных и важных дистрибутивов GNU/Linux, в первичной форме оказавший значительное влияние на развитие этого типа ОС в целом. Debian может использоваться в качестве операционной системы как для серверов, так и для рабочих станций. Debian отличается широким спектром возможностей. В текущую стабильную версию включено свыше тридцати семи тысяч пакетов программ для десяти архитектур на основе ядра Linux и также двух архитектур на основе ядра FreeBSD.

Отличительными чертами Debian являются: система управления пакетами Advanced Packaging Tool (APT), жёсткая политика по отношению к пакетам, репозитории с огромным их количеством, а также высокое качество выпускаемых версий. Это сделало возможным простое обновление между версиями, а также автоматическую установку и удаление пакетов.

3.1.2. Панель управления Webmin

Webmin — это программный комплекс, позволяющий администрировать операционную систему через веб-интерфейс, в большинстве случаев, позволяя обойтись без использования командной строки и запоминания системных команд и их параметров. Используя любой браузер, администратор сервера может создавать новые учётные записи пользователей, почтовые ящики, изменять настройки служб и сервисов, например: веб-сервера Apache, DNS. Однако, в некоторых случаях необходимо знание операционной системы и редактирование конфигурационных файлов вручную.

3.1.3. Протокол L2TP

L2TP — в компьютерных сетях туннельный протокол, использующийся для поддержки виртуальных частных сетей. Главное достоинство L2TP состоит в том, что этот протокол

позволяет создавать туннель не только в сетях IP, но и в таких, как ATM, X.25 и Frame Relay.

Несмотря на то, что L2TP действует наподобие протокола Канального уровня модели OSI, на самом деле он является протоколом Сеансового уровня и использует зарегистрированный UDP-порт 1701.

3.1.4. Файловый сервер Samba

Samba — пакет программ, которые позволяют обращаться к сетевым дискам и принтерам на различных операционных системах по протоколу SMB/CIFS. Имеет клиентскую и серверную части. Является свободным программным обеспечением, выпущена под лицензией GPL.

Samba работает на большинстве Unix-подобных систем, таких как Linux, POSIX-совместимых Solaris и Mac OS X Server, на различных вариантах BSD; в OS/2 портирован Samba-клиент, являющийся плагином к виртуальной файловой системе NetDrive. Samba включена практически во все дистрибутивы Linux.

Главное его преимущество, что он работает гораздо быстрее ftp

3.2. Использование симметричного шифрования

Advanced Encryption Standard — симметричный алгоритм блочного шифрования, принятый в качестве стандарта шифрования правительством США по результатам конкурса AES. Этот алгоритм хорошо проанализирован и сейчас широко используется, как это было с его предшественником DES. Национальный институт стандартов и технологий США (англ. National Institute of Standards and Technology, NIST) опубликовал спецификацию AES 26 ноября 2001 года после пятилетнего периода, в ходе которого были созданы и оценены 15 кандидатур. 26 мая 2002 года AES был объявлен стандартом шифрования. По состоянию на 2009 год AES является одним из самых распространённых алгоритмов симметричного шифрования.

В данной работе шифрование AES позволяет передавать чувствительные данные даже по открытым сетям и при этом работает прозрачно для пользователя. На рисунке 3.2 показан код который на стороне сервера шифрует данные, а на рисунке 3.1 фрагмент когда на стороне клиента расшифровывает данные

```
    protected static String decrypt(String input, String key) {
        byte[] output = null;
        try {
            SecretKeySpec skey = new SecretKeySpec(key.getBytes(), "AES");
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
            cipher.init(Cipher.DECRYPT_MODE, skey);
            output = cipher.doFinal(Base64.decode(input, Base64.DEFAULT));

        } catch (Exception e) {

            e.printStackTrace();
        }
        return new String(output);
    }
```

Рисунок 3.1 – Фрагмент кода расшифровки на стороне Android

```
class Security {
    public static function encrypt($input, $key) {
        $size = mcrypt_get_block_size(MCRYPT_RIJNDAEL_128, MCRYPT_MODE_ECB);
        $input = Security::pkcs5_pad($input, $size);
        $td = mcrypt_module_open(MCRYPT_RIJNDAEL_128, '', MCRYPT_MODE_ECB, '');
        $iv = mcrypt_create_iv (mcrypt_enc_get_iv_size($td), MCRYPT_RAND);
        mcrypt_generic_init($td, $key, $iv);
        $data = mcrypt_generic($td, $input);
        mcrypt_generic_deinit($td);
        mcrypt_module_close($td);
        $data = base64_encode($data);
        return $data;
    }

    private static function pkcs5_pad ($text, $blocksize) {
        $pad = $blocksize - (strlen($text) % $blocksize);
        return $text . str_repeat(chr($pad), $pad);
    }
```

Рисунок 3.2 – фрагмент когда шифрования на стороне сервера

3.3. Разработка интерфейса приложения

Для создания графического интерфейса в IDE Android Studio предусмотрен графический редактор с возможность перетаскивания элементов интерфейса на макет экрана, что в свою очередь очень упрощает разработку,

элементы расположенные на макете конфигурируются с помощью классов, к примеру с помощью класса мы можем не отображать часть элементов на макете экрана. Для тонкой настройки элементов макета возможно переключить графический редактор в режим редактирования xml, в данном режиме все объекты представлены в виде xml объектов, в этом режиме возможно произвести тонкую подстройку элементов макета, так же xml редактор проверяется правильность описания элементов, если он заметит ошибку он обязательно подсветит ее красным. Пример интерфейса редактора и макета экрана LaunchActivity который пользовать увидит при первом запуске приложения изображен на рисунке 3.2, данный макет в xml редакторе представлен на рисунке 3.3

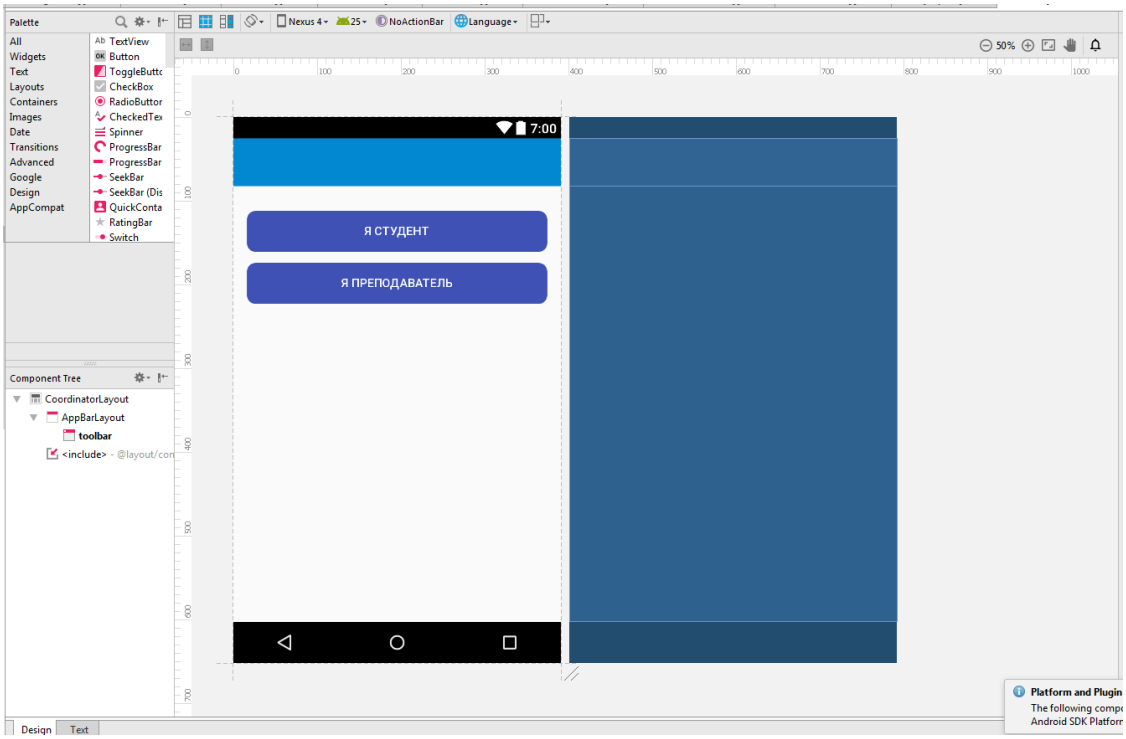


Рисунок 3.2 – графический редактор макетов

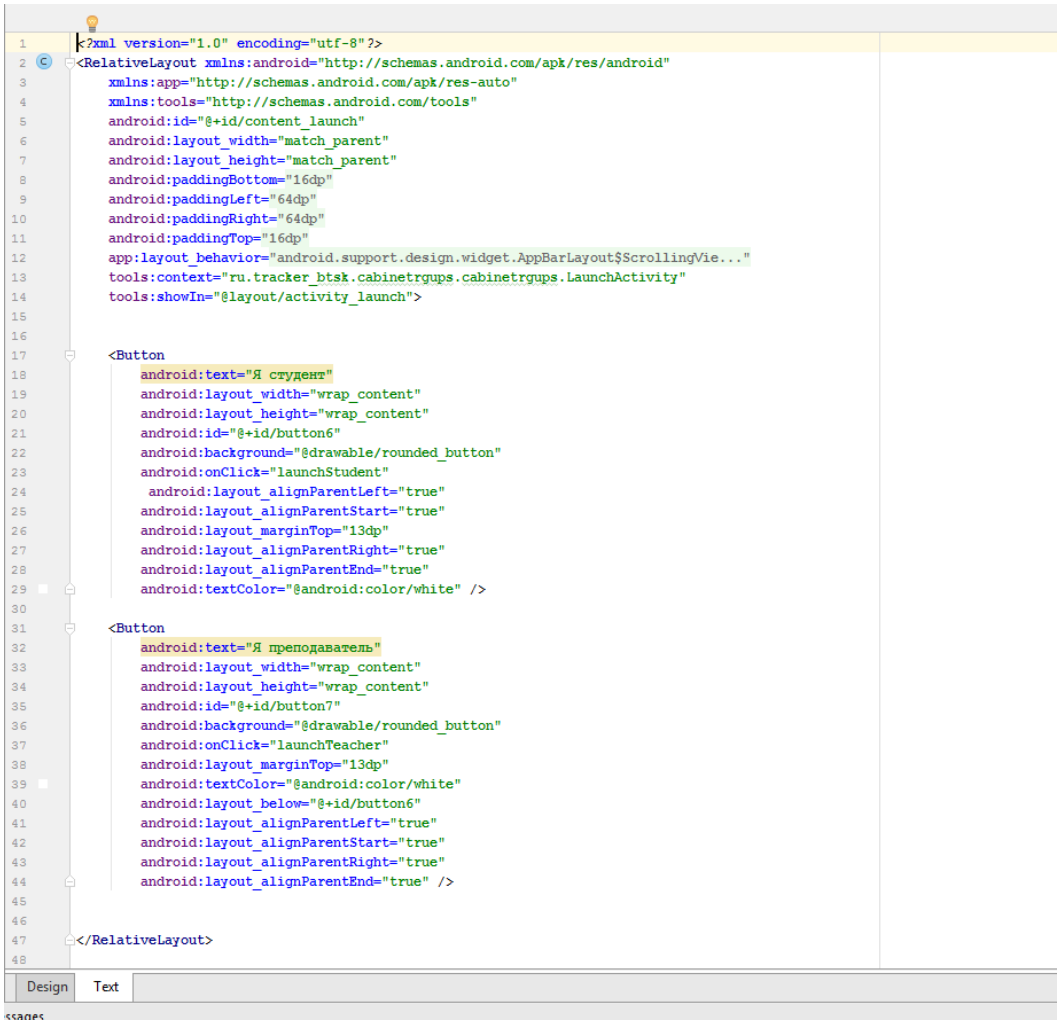
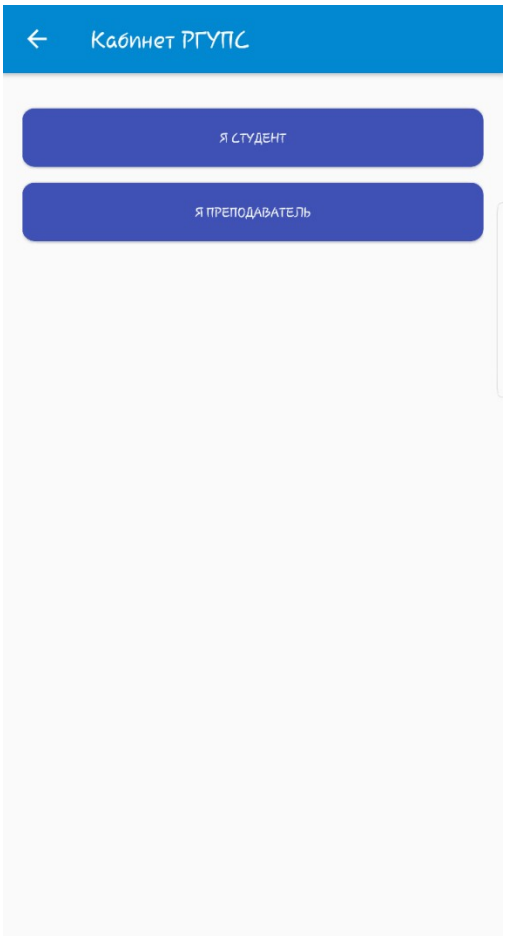


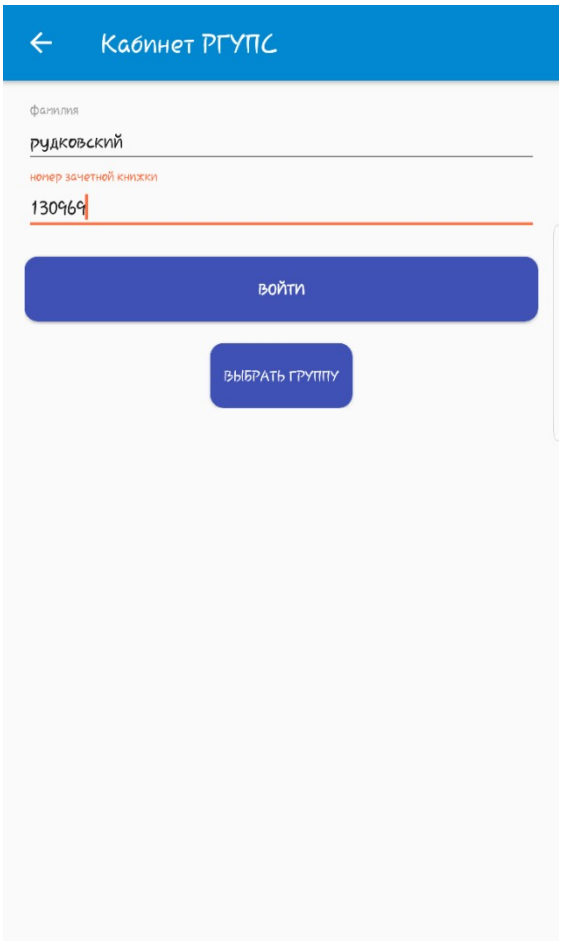
Рисунок 3.3 – xml редактор макета

При входе в приложение пользователю предлагается выбрать кем он является, сейчас мы рассмотрим использование приложение в качестве студента, который знает свой номер зачетной книжки и фамилию. Главный экран отображен на рисунку 3.4. Код реализации класса LaunchActivity приведен в приложении А.

Каждому студенту при поступлении в университет присваивается номер зачетной книжки, с помощью него и своей фамилии он может авторизоваться в приложении. Если по каким-то причинам студент не может ввести номер личного дела или фамилию то он может выбрать свою группу и так же пользоваться приложением только с ограниченным функционалом.



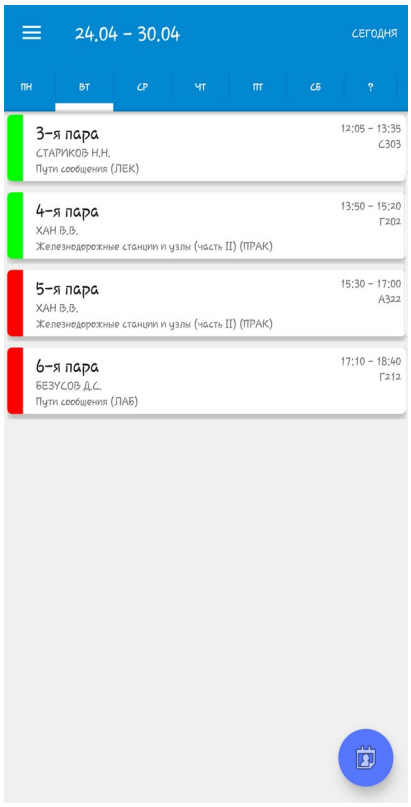
Окно выбора кем является пользователь

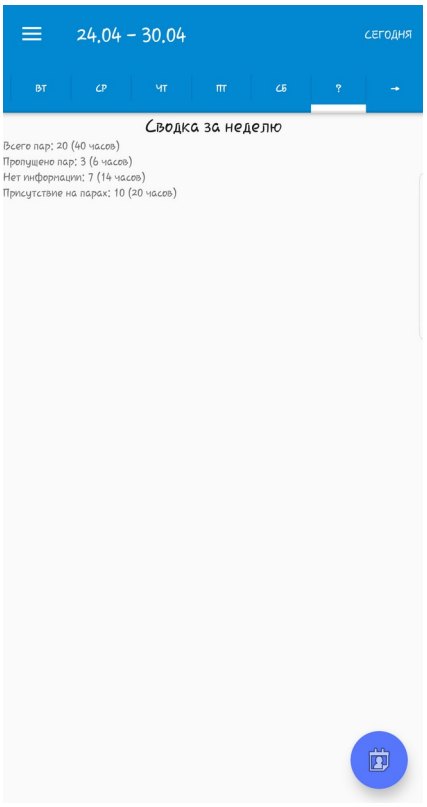


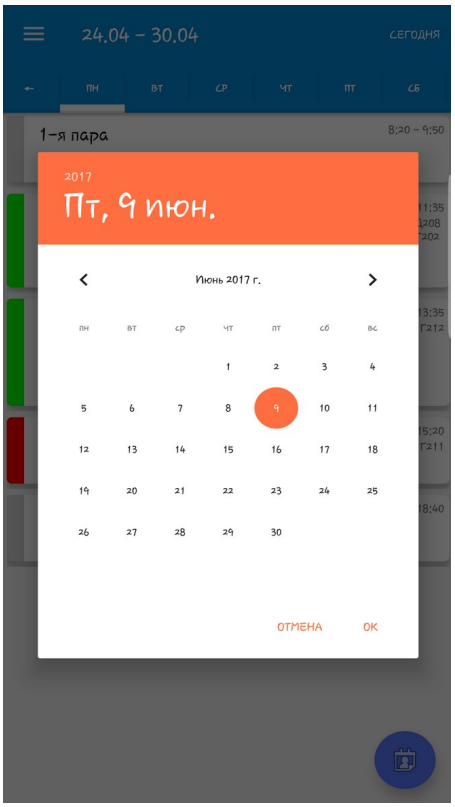
Ввод учетных данных пользователя

Рисунок 3.4 – Интерфейс входа в приложение

После авторизации приложение позволяет просматривать расписание, с возможностью выбора нужного дня недели, а так же с помощью календаря возможно посмотреть свое расписание в нужную дату. Также в конце каждой недели есть сводка о посещаемости за эту неделю. Взаимодействие отображено на рисунке 3.5. Класс RaspisanieActivity – отвечающий за отображение расписания приведен в приложении Б.







Расписание студента с метками о посещении им пары

Сводка за неделю

Выбор нужного дня с помощью календаря

Рисунок 3.5 – работа с расписанием

Также пользователю доступно меню навигации оно отображено на рисунке 3.6, меню вызывается движением слева на право от левого края экрана, меню это самый удобный метод навигации в приложении.

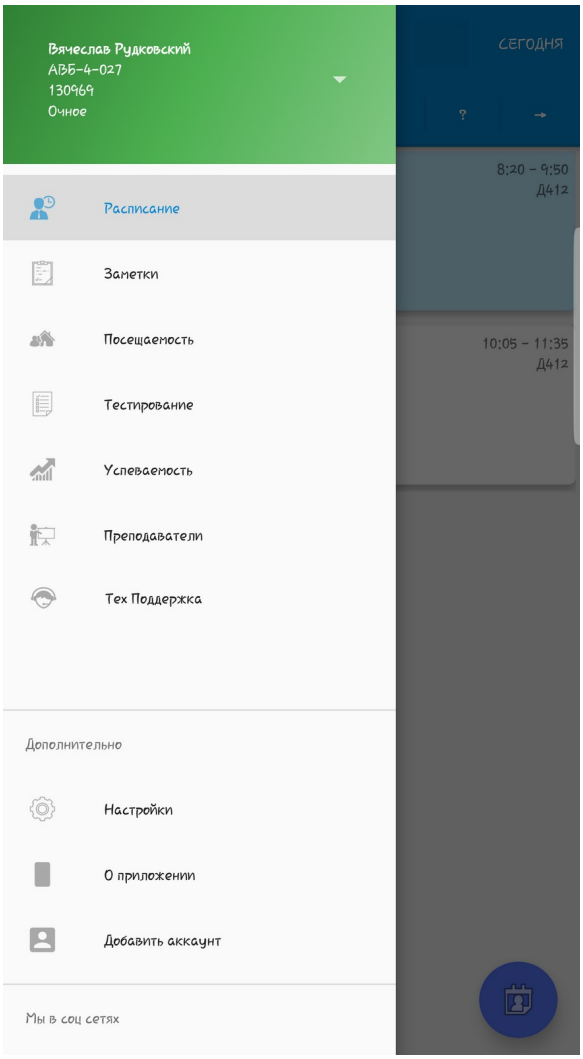


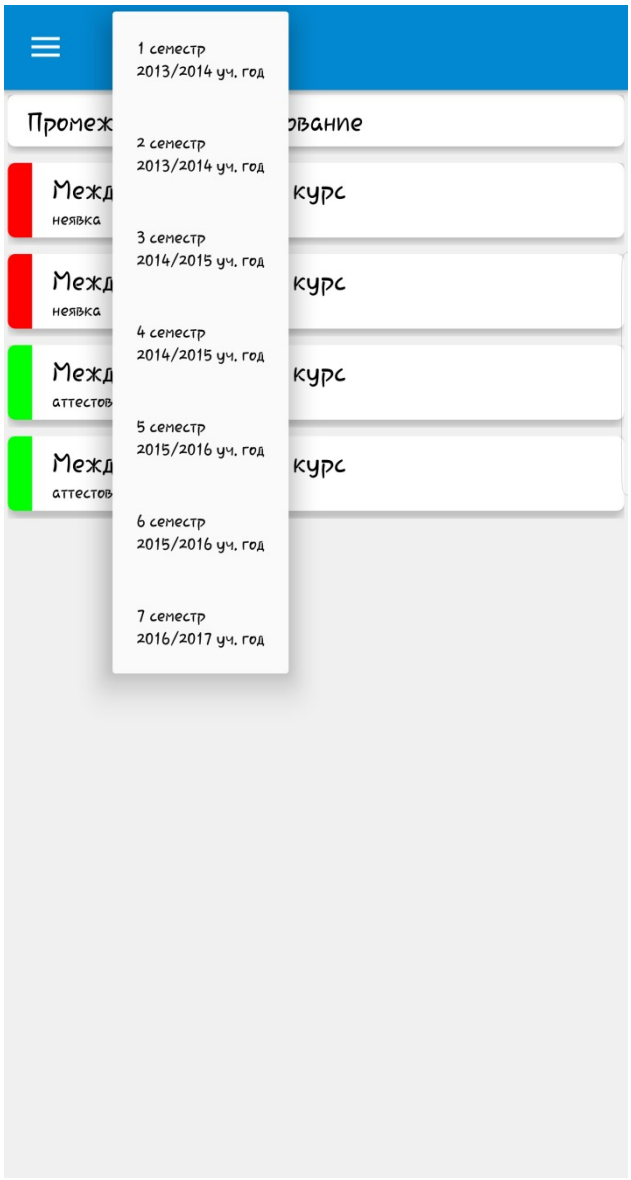
Рисунок 3.6 – меню навигации

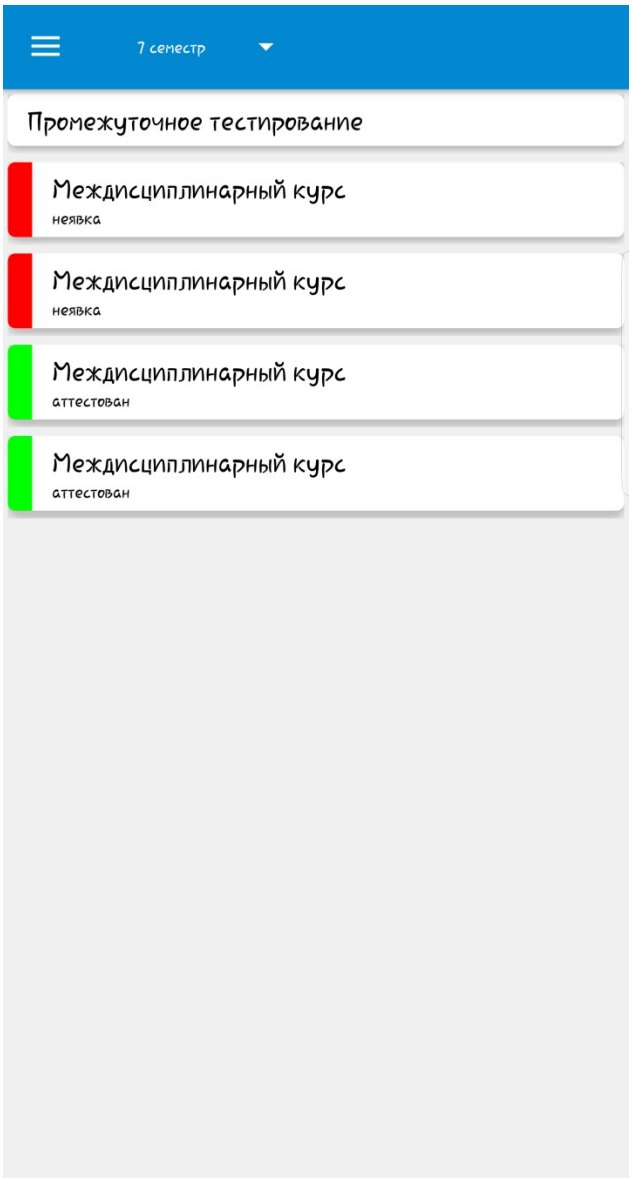
С помощью меню мы переходим в раздел посещаемость оно изображено на рисунке 3.7, там мы видим посещаемость студента, в виде краткой сводки за весь семестр и с понедельной информацией, класс VisitsActivity отвечающий за отображение посещаемости приведен в приложении В.



Рисунок 3.7 – посещаемость студента

С помощью меню изображенного на рисунке 3.6 мы переходим в раздел тестирование. В нем отображается информация по всем семестрам. Переключение семестров происходит с помощью меню в верхней части экрана. Результаты тестирования группируются в зависимости от типа тестирования, пробное тестирование либо это тест сессия. Данный экран отображен на рисунке 3.7, код класса TestActivity находится в приложении Г.



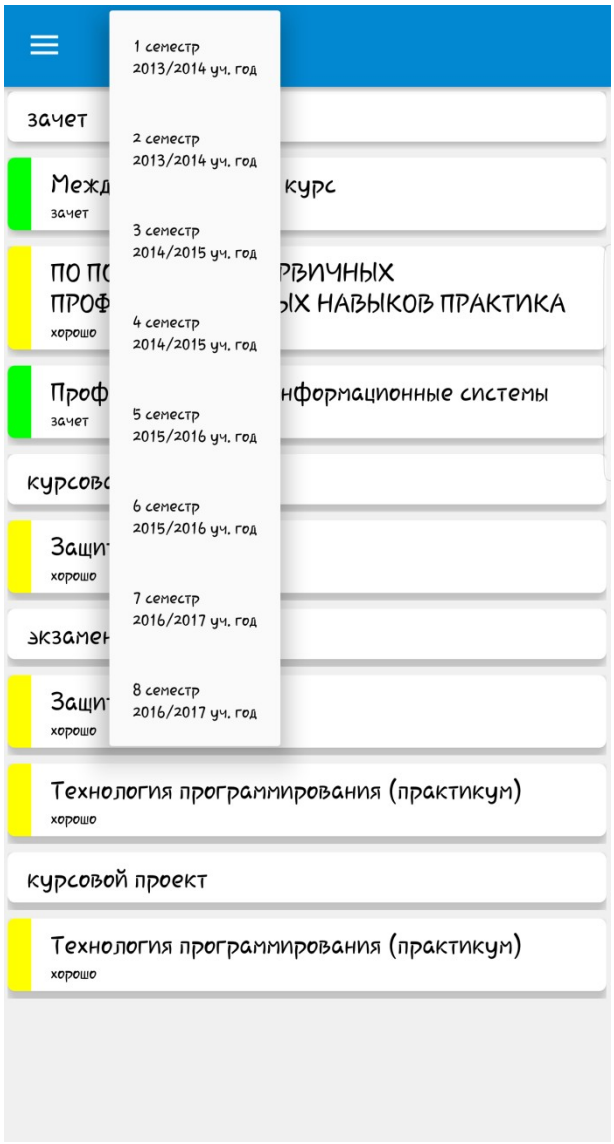


Выпадающее меню с выбором семестра

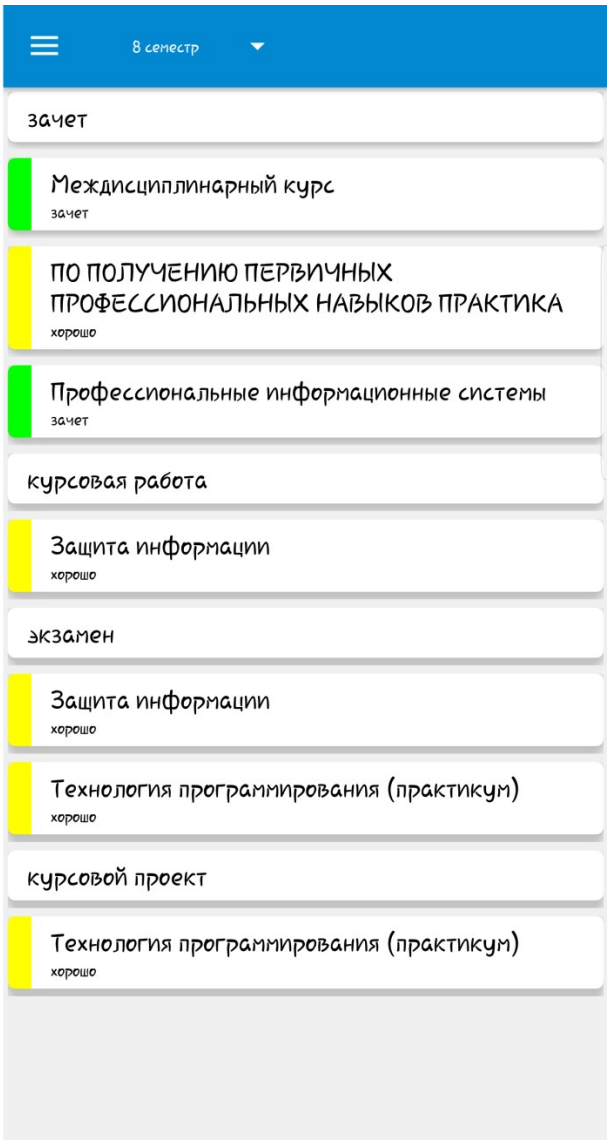
Результаты тестирования

Рисунок 3.7 – просмотр результатов тестирования

С помощью меню изображенного на рисунке 3.6 мы переходим в раздел успеваемость. В нем отображается информация по всем семестрам. Переключение семестров происходит с помощью меню в верхней части экрана. Результаты успеваемости студента группируются в зависимости от типа контроля предмета. Данный экран отображен на рисунке 3.8, код класса ProgressActivity находится в приложении Д.



Выпадающее меню с выбором семестра

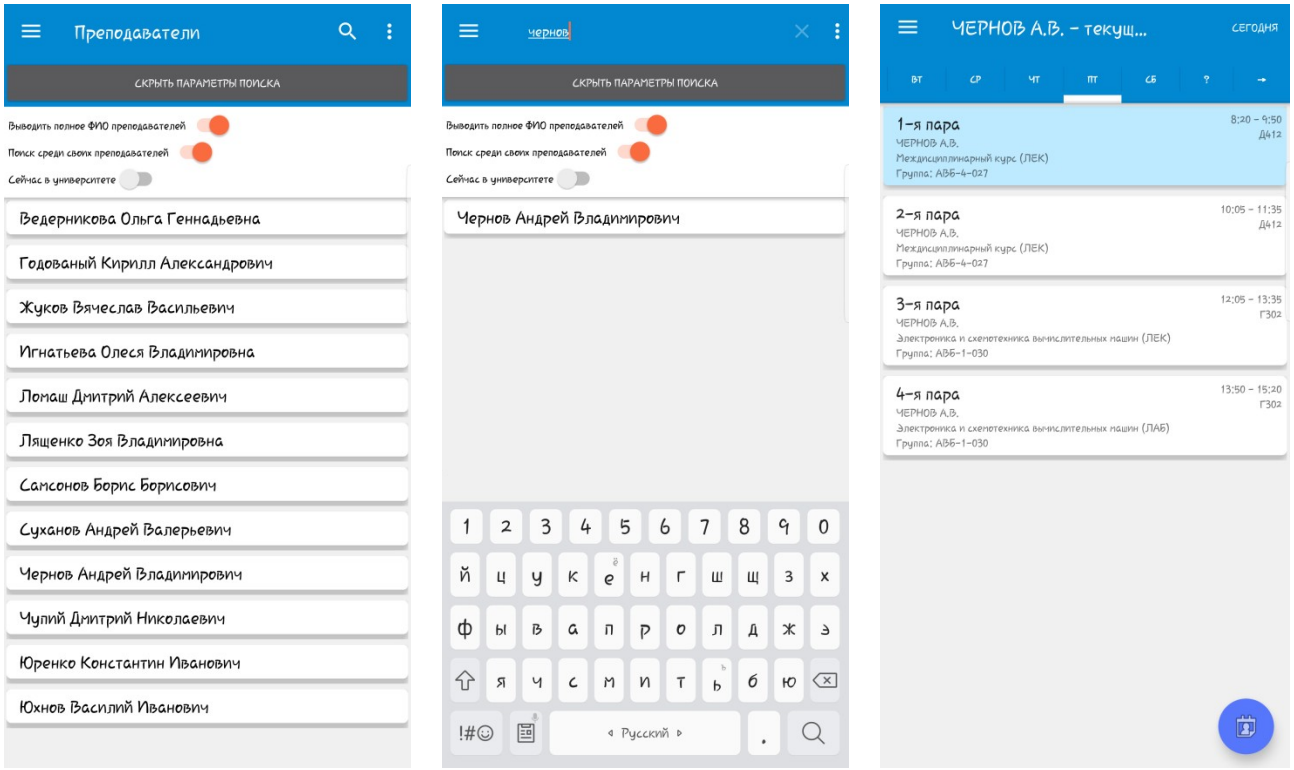


Успеваемость студента

Рисунок 3.8 – просмотр успеваемости

С помощью меню изображенного на рисунке 3.6 мы переходим в раздел преподаватели. С помощью этого раздела мы можешь найти нужного нам преподавателя, а с помощью дополнительных опций мы можем уточнить критерии поиска, к примеру мы можем отобразить только тех преподавателей которые ведут пару у конкретной группы, отобразить полное фамилию имя отчество преподавателя, а также отобразить только тех преподавателей которые по расписанию сейчас в университете. Пункты можно комбинировать между собой к примеру посмотреть только своих преподавателей которые сейчас в университете. Когда мы нашли нужного преподавателя мы можем нажать на

его фамилию имя отчество и мы перейдем в его расписание, навигация по расписанию преподавателя аналогична навигации по расписанию группы. Данный экран отображен на рисунке 3.9, код класса TeacherActivity находится в приложении Е.



Список преподавателей с критериями | Поиск среди преподавателей | Просмотр расписания преподавателя

Рисунок 3.9 – поиск среди преподавателей

В ходе выполнения выпускной квалификационной работы было создано Android приложение.

3.4. Разработка API сервера

Для нормальной работы приложения требуется постоянное обновление данных, расписание может быть изменено в любой момент, посещаемость студента может обновиться и главная задача приложения как можно быстрей отобразить эту информацию пользователю.

Взаимодействие с Android приложения с информационной системой РГУПС схематически изображено на рисунке 3.10

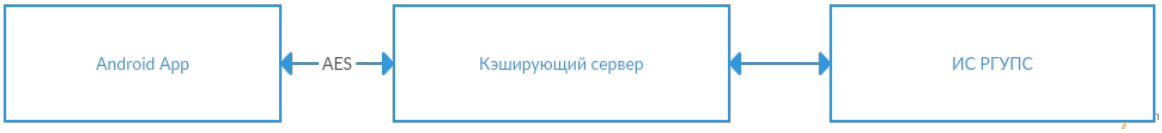


Рисунок 3.10 – взаимодействие с информационной системой РГУПС

Для хранения данных используется несколько подходов, данные о расписании хранятся в сжатом с помощью gzip файле, так как записей очень много и они не часто меняются целесообразно хранить их в таком виде, чем каждый раз запрашивать их из базы данных. Данные студентов напротив хранятся в базе данных, так как они меняются чаще и хранить их в файлах не целесообразно.

Кеширующий сервер раз в сутки запускает процесс, который запрашивает в информационной системе РГУПС обновление расписания, обновление посещаемости, успеваемости, результатов тестирования пользователей и сохраняет в свою базу данных. Все части кеширующего сервера реализованы на языке высокого уровня php с использованием базы данных mysql.

Пример кода обновление данных в приложение Ж

Заключение

В процессе выполнения работы было рассмотрено взаимодействие отделов при составлении расписания. Ключевым моментом работы стала разработка программы для использования на мобильном устройстве. Было реализовано приложение, для операционной системы Android, позволяющее студенту ознакомится со своим расписанием, посещаемостью и успеваемостью. При разработке указанного приложений сочетались лаконичность интерфейса с достаточно сложной логикой задач. Современный смартфон или планшетный компьютер может полностью заменить все функции домашнего компьютера. В связи с тем, что имеет важное значение разработка приложений не только для персональных компьютеров, так и для планшетов и смартфонов. Подводя итоги проведенного исследования, можно сказать, что все поставленные задачи решены, следовательно, главная цель, поставленная в выпускной квалификационной работе, достигнута.

Список использованных источников

1. Голицына, О.Л. Базы данных. - М.: Форум, 2014. - 352 с.

2. Голощапов, А.Л. Google Android. Создание приложений для смартфонов и планшетных ПК. – Питер, 2014.

2. Джеффри, Д.У. Основы реляционных баз данных. – Лори, 2016. - 382 с

3. Захаров, В.П. Информационные системы: учеб. пособие / В.П. Захаров. - СПб., 2014. - 187 с.

4. Голицына, О.Л. Информационные технологии: Учебник / О.Л. Голицына, Н.В. Максимов, Т.Л. Партыка, И.И. Попов. - М.: Форум, ИНФРА-М, 2013. - 608 с.

5. Гультяев А.К. Проектирование и дизайн пользовательского интерфейса – С- Пб: Корона-принт, 2010 г. - 349 с.

6. Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. - М.: ИД ФОРУМ, 2013. - 352 с.

7. ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов.

8. .ГОСТ 34.201-89. Информационная технология. Комплекс стандартов на ав- томатизированные системы. Виды, комплектность и обозначение докумен- тов при создании автоматизированных систем.

9. .Цветкова М.С. Информатика и ИКТ. / М.С.Цветкова, Л.С.Великович 3-е изд., стер. - М.: ИНФРА, 2012. – 352 с.

10. Хлебников, А.А. Информационные технологии: Учебник / А.А. Хлебников. - М.: КноРус, 2014. - 472 с

11. Федотова, Е.Л. Информационные технологии и системы: Учебное пособие / Е.Л. Федотова. - М.: ИД ФОРУМ, 2013. - 352 с

12. Васильев А.Н. Java. – СПБ, Питер, 2011

13. Варакин М.В. Разработка мобильных приложений под Android. УЦ «Специалист» при МГТУ им. Н. Э. Баумана, 2012.

14. Котляров В.П. Основы тестирования программного обеспечения. Издательство Бином, 2009

15. Дейтел П. Android для программистов: создаем приложения. Издательство Питер, 2012.

16. Амелин К. С., Граничин О. Н., Кияев В. И., Корявко А. В.. Введение в разработку приложений для мобильных платформ. Издательство ВВМ, 2011.

17. Пайлон Д. UML 2 для программистов. Издательство Питер, 2012.

18. Харди Б., Филлипс Б., Стюарт К., Марсикано К.Android программирование для профессионалов. 2 издание. Издательство Питер, 2016.

Приложение А

(обязательное)

Класс LaunchActivity – отвечающий за главный экран

```
public class LaunchActivity extends AppCompatActivity {
    private boolean newacc;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_launch);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        HotlineConfig hlConfig=new HotlineConfig("","");
        hlConfig.setVoiceMessagingEnabled(true);
        hlConfig.setCameraCaptureEnabled(true);
        hlConfig.setPictureMessagingEnabled(true);
        Hotline.getInstance(getApplicationContext()).init(hlConfig);
FirebaseAnalytics.getInstance(LaunchActivity.this).logEvent(FirebaseAnalytics.Eve
nt.APP_OPEN, new Bundle());
        try {
            Intent intent = getIntent();
            newacc = intent.getBooleanExtra("newacc", false);
        }catch (Exception e){}
        if (newacc || !new AccountData(getApplication()).isAviableAccounts()) {
            setupActionBar();
        }else{
                Intent raspintent = Help.RaspisanieController(getApplicationContext(),
"1");
                if (raspintent != null) {
                    try {
                        startActivity(raspintent);
                    } catch (Exception e) {
                        startActivity(new Intent(this, RaspisanieActivity.class));
                    }
                    finish();
                    return;
                } else {
                    startActivity(new Intent(this, RaspisanieActivity.class));
                    finish();
                    return;
                }
            }
        }
```

```
    public void launchAllAccount(View view){
    startActivity(new Intent(this, LoginAllAccountActivity.class));
        if(newacc) finish();
    }
    public void launchStudent(View view){
        startActivity(new Intent(this, LoginActivity.class));
        if(newacc) finish();
    }
    public void launchTeacher(View view){
        startActivity(new Intent(this, SelectTeacherActivity.class));
        if(newacc) finish();
    }
}
```

Приложение Б

(обязательное)

Класс RaspisanieActivity – отвечающий за отображение расписания

```
    public static class PlaceholderFragment extends Fragment {
        private static final String ARG_SECTION_NUMBER = "section_number";
        private ArrayList<RaspisanieAdapter> raspisanieadapter = new
ArrayList<RaspisanieAdapter>();
        private ArrayList<Timer> myTimer = new ArrayList<Timer>();
        private ArrayList<BroadcastReceiver> br = new
ArrayList<BroadcastReceiver>();
        private View rootView;
        public static PlaceholderFragment newInstance(int position) {
            PlaceholderFragment fragment = new PlaceholderFragment();
            Bundle args = new Bundle();
            args.putInt(ARG_SECTION_NUMBER, position);

            fragment.setArguments(args);
            return fragment;
        }
        class loadWeekInfoAsync extends AsyncTask<Object, Object, VisitsObj> {
            @Override
            protected void onPreExecute() {
                super.onPreExecute();
}
            @Override
            protected VisitsObj doInBackground(Object... params) {
```

```
            if(RaspisanieWeekInfo.getNumweeku() <=
mycalendar.get(Calendar.WEEK_OF_YEAR)) {
                return new VisitsData(getActivity()).getDataVisitsonWeek("",
account.getaccount_id(), RaspisanieWeekInfo.getNumweeku());
            }else return null;
        }
        @Override
        protected void onPostExecute(VisitsObj result) {
            super.onPostExecute(result);
if (result == null){
                viewallpair.setText("Нет информации");
                viewallpair.setVisibility(View.VISIBLE);
                viewunaviablepair.setVisibility(View.GONE);
                viewaviablepair.setVisibility(View.GONE);
                viewnoinfopair.setVisibility(View.GONE);
                return;
            }
            try {
                progressBarSvodka.setVisibility(View.GONE);
                if (result.getAllpair() == 0 && result.getNo_visited_pair() == 0 &&
result.getVisited_pair() == 0) {
                    viewallpair.setText("Нет информации");
                    viewallpair.setVisibility(View.VISIBLE);
                    viewunaviablepair.setVisibility(View.GONE);
                    viewaviablepair.setVisibility(View.GONE);
                    viewnoinfopair.setVisibility(View.GONE);
                } else {
                    viewallpair.setVisibility(View.VISIBLE);
                    viewunaviablepair.setVisibility(View.VISIBLE);
                    viewaviablepair.setVisibility(View.VISIBLE);
                    viewnoinfopair.setVisibility(View.VISIBLE);
                    viewallpair.setText(Html.fromHtml("Всего пар: " + result.getAllpair()
+ " (" + (result.getAllpair() * 2) + " часов)"));
                    viewunaviablepair.setText(Html.fromHtml("Пропущено пар: " +
result.getNo_visited_pair() + " (" + (result.getNo_visited_pair() * 2) + " часов)"));
                    viewaviablepair.setText(Html.fromHtml("Присутствие на парах: " +
result.getVisited_pair()) + " (" + (result.getVisited_pair() * 2) + " часов)");
                    viewnoinfopair.setText(Html.fromHtml("Нет информации: " +
(result.getNoInfopair()) + " (" + (result.getNoInfopair()) * 2) + " часов) ");
                }
            }catch (Exception e){}
        }
    }
    public class SectionsPagerAdapter extends FragmentPagerAdapter {
        public SectionsPagerAdapter(FragmentManager fm) {
```

```
            super(fm);
        }
        @Override
        public CharSequence getPageTitle(int position) {
            SharedPreferences sharedPref =
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
            boolean fullday = sharedPref.getBoolean("fullday", false);
            boolean visibledateday = sharedPref.getBoolean("visibledateday", false);
            String day = null;
            switch (position) {
                case 0:
                    return "" + (char) 8592;
                case 1:
                    if (fullday)
                        day = "понедельник";
                    else
                        day = "пн";
                    break;
                case 2:
                    if (fullday)
                        day = "вторник";
                    else
                        day = "вт";
                    break;
                case 3:
                    if (fullday)
                        day = "среда";
                    else
                        day = "ср";
                    break;
                case 4:
                    if (fullday)
                        day = "четверг";
                    else
                        day = "чт";
                    break;
                case 5:
                    if (fullday)
                        day = "пятница";
                    else
                        day = "пт";
                    break;
                case 6:
                    if (fullday)
                        day = "суббота";
```

```
                else
                    day = "сб";
                break;
            case 7:
                if (fullday) {
                    return "сводка за неделю";
                } else {
                    return "?";
                }
            case 8:
                return "" + (char) 8594;
        }
        if (visibledateday)
            day = day + " (" +
HelpFull.getDayInWeek(RaspisanieWeekInfo.getNumweeku(), position, true) + ")";
        return day;
    }
  }
}
```

Приложение В

(обязательное)

Класс VisitsActivity отвечающий за отображение посещаемости

```
package ru.tracker_btsk.cabinetrgups.cabinetrgups;
public class VisitsActivity extends AppCompatActivity
    implements NavigationView.OnNavigationItemSelectedListener,
AccountUpdateInterface {
  @Override
  public void onResume() {
    super.onResume();
    if (!new AccountData(getApplicationContext()).isAviableAccounts()) {
      startActivity(new Intent(this, LoginActivity.class));
      finish();
      return;
    }
    account = new AccountData(getApplicationContext()).getCurrentAccount();
    br = new BroadcastReceiver() {
      public void onReceive(Context context, Intent intent) {
        int status = 0;
        int refresh = 0;
        try{
          status =
intent.getIntExtra(Config.BROADCAST_PARAM_STATUS_REFRESH, 0);
```

```
                refresh =
intent.getIntExtra(Config.BROADCAST_PARAM_STATUS_SPINNER_SELECT, 0);
            }catch (Exception e){}
            if (status == Config.BROADCAST_REFRESH_CODE) {
            }
            if (refresh ==
Config.BROADCAST_REFRESH_CODE_ON_SPINNER_SELECT) {
                if (account.getType() != Config.ACCOUNT_TYPE_STUDENT) {
                    startActivity(new Intent(getApplicationContext(),
RaspisanieActivity.class));
                    finishAffinity();
                    finish();
                    return;
                }
            }
        }
    };
    IntentFilter intFilt = new IntentFilter(Config.BROADCAST_ACTION);
    registerReceiver(br, intFilt);
    Help.updateAccountSpin(getApplicationContext());
    try {
        new loadViewVisits().execute();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
class loadViewVisits extends AsyncTask<Void, Void, ArrayList<VisitsObj>> {

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

    }
    @Override
    protected ArrayList<VisitsObj> doInBackground(Void... params) {
        ArrayList<VisitsObj> visitsObjtmp = new
VisitsData(getApplicationContext()).getData("student_id=" +
account.getaccount_id(), "week_title", "numweek DESC");
        ArrayList<VisitsObj> visits = new ArrayList<>();
        for (VisitsObj visitstmp : visitsObjtmp) {
            VisitsObj visitsdata = new
VisitsData(getApplicationContext()).getDataVisitsonWeek(visitstmp.getTitle(),
account.getaccount_id(), visitstmp.getNumweek());
            visits.add(visitsdata);
        }
```

```
            for(VisitsObj visitstmp:visits){
                allaviable = allaviable+visitstmp.getVisited_pair();
                allunaviable = allunaviable + visitstmp.getNo_visited_pair();
                allpar = allpar+visitstmp.getAllpair();
            }
            if(visits.size()>0)
            visits.add(0, new VisitsObj("За весь семестр", allaviable, allunaviable,
allpar));
            return visits;
        }
        @Override
        protected void onPostExecute(ArrayList<VisitsObj> result) {
            super.onPostExecute(result);
            progressBarVisits.setVisibility(View.GONE);
            if(result.size()==0){
                textnoinfo.setVisibility(View.VISIBLE);
            }else textnoinfo.setVisibility(View.GONE);
            VisitsAdapter visitedadapter = new VisitsAdapter(getApplicationContext(),
result);
            listViewVisits.setAdapter(visitedadapter);
                }
            });
        }
    }
```

Приложение Г

(обязательное)

Класс TestActivity отвечающий за результаты тестирования

```
public class TestActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener,
AccountUpdateInterface {
    @Override
    public void onResume() {
        super.onResume();
        if (!new AccountData(getApplicationContext()).isAviableAccounts()) {
            startActivity(new Intent(this, LaunchActivity.class));
            finish();
            return;
        }
        account = new AccountData(getApplicationContext()).getCurrentAccount();
        Help.updateAccountSpin(getApplicationContext());
```

```
        br = new BroadcastReceiver() {
            public void onReceive(Context context, Intent intent) {
                int status = 0;
                int refresh = 0;
                try{
                    status =
intent.getIntExtra(Config.BROADCAST_PARAM_STATUS_REFRESH, 0);
                    refresh =
intent.getIntExtra(Config.BROADCAST_PARAM_STATUS_SPINNER_SELECT, 0);
                }catch (Exception e){}
                if (status == Config.BROADCAST_REFRESH_CODE) {
                }
                if (refresh ==
Config.BROADCAST_REFRESH_CODE_ON_SPINNER_SELECT) {
                    if (account.getType() != Config.ACCOUNT_TYPE_STUDENT) {
                        startActivity(new Intent(getApplicationContext(),
RaspisanieActivity.class));
                        finishAffinity();
                        finish();
                        return;
                    }
                }
            }
        };
        IntentFilter intFilt = new IntentFilter(Config.BROADCAST_ACTION);
        registerReceiver(br, intFilt);
        try {
            new loadTestSpinner().execute();
        } catch (Exception e) {
            e.printStackTrace();
        }
            notifyUpdate = new NotifyUpdate(this);
    }
    class loadTestSpinner extends AsyncTask<Void, Void,
ArrayList<ProgressSpinnerObj>> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            try{
                spinner.setAdapter(spinnerProgressAdapter);
                spinner.setSelection(spinner.getAdapter().getCount() - 1);
            }catch (Exception e){
                    ((TextView)
findViewById(R.id.testnofound)).setVisibility(View.GONE);
                    ((ProgressBar)
```

```
findViewById(R.id.progressBarTest)).setVisibility(View.VISIBLE);
                spinner.setVisibility(View.GONE);
                setTitle("Тестирование");
            }
        }
        @Override
        protected ArrayList<ProgressSpinnerObj> doInBackground(Void... params) {
            ArrayList<TestObj> testData = new
TestData(getApplicationContext()).getData("student_id=" + account.getaccount_id()
+ "", "semestr_num", "semestr_num");
            ArrayList<ProgressSpinnerObj> spinnerList = new
ArrayList<ProgressSpinnerObj>();
            for(TestObj testtmp:testData) {
                spinnerList.add(new ProgressSpinnerObj(testtmp.getSemestr_num(),
testtmp.getYear()));
            }
            return spinnerList;
        }
        @Override
        protected void onCancelled() {
            super.onCancelled();
            try {
            }catch (Exception e){
            }
        }
        @Override
        protected void onPostExecute(ArrayList<ProgressSpinnerObj> result) {
            super.onPostExecute(result);
            if(result==null) {
                spinner.setVisibility(View.GONE);
                ((TextView)
findViewById(R.id.testnofound)).setVisibility(View.VISIBLE);
                setTitle("Тестирование");
                return;
            }
 ((ProgressBar)findViewById(R.id.progressBarTest)).setVisibility(View.GONE);
            spinnerProgressAdapter = new
ProgressSpinnerAdapter(getApplicationContext(), result);
            spinner.setAdapter(spinnerProgressAdapter);
            spinner.setSelection(result.size() - 1);
            if (result.size() == 0) {
                spinner.setVisibility(View.GONE);
                setTitle("Тестирование");
                ((TextView)
findViewById(R.id.testnofound)).setVisibility(View.VISIBLE);
```

```
            } else {
                spinner.setVisibility(View.VISIBLE);
              setTitle("");
            }
            spinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
                @Override
                public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {
                    ProgressSpinnerObj testspinobj = (ProgressSpinnerObj)
spinner.getAdapter().getItem(position);
                    try {
                        new loadViewTest(testspinobj.getSemestr_num()).execute();
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
    class loadViewTest extends AsyncTask<Void, Void, ArrayList<TestObj>> {
        private int position;
        ProgressBar progressBarTest;
        TextView testnofound;
        public loadViewTest(int position)throws Exception {
            this.position = position;
        }
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            progressBarTest = (ProgressBar) findViewById(R.id.progressBarTest);
            progressBarTest.setVisibility(View.VISIBLE);

            testnofound = (TextView) findViewById(R.id.testnofound);
            testnofound.setVisibility(View.GONE);
        }
        @Override
        protected ArrayList<TestObj> doInBackground(Void... params) {
            ArrayList<TestObj> testData = new
TestData(getApplicationContext()).getData("student_id=" + account.getaccount_id()
+ " and semestr_num=" + (position), null, "type");
            ArrayList<TestObj> test = new ArrayList<TestObj>();
            int lasttype = -1;
            for (TestObj testtmp : testData) {
                if (lasttype == -1) {
                    test.add(new TestObj(HelpFull.getTypeTest(testtmp.getType()), -1));
                    lasttype = testtmp.getType();
                } else if (lasttype != testtmp.getType()) {
```

```
                test.add(new TestObj(HelpFull.getTypeTest(testtmp.getType()), -1));
                lasttype = testtmp.getType();
            }
            test.add(testtmp);
        }
        return test;
    }
    @Override
    protected void onCancelled() {
        super.onCancelled();
    }
    @Override
    protected void onPostExecute(ArrayList<TestObj> result) {
        super.onPostExecute(result);
        progressBarTest.setVisibility(View.GONE);
        if(result==null) return;
        if (result.size() == 0) {
        } else {
            testnofound.setVisibility(View.GONE);
            TestAdapter testadapter = new TestAdapter(getApplicationContext(),
result);
            listViewTest.setAdapter(testadapter);
        }
    }
}
```

Приложение Д

(обязательное)

Класс ProgressActivity отвечающий за успеваемость

```
public class ProgressActivity extends AppCompatActivity
    implements NavigationView.OnNavigationItemSelectedListener,    @Override
  public void onResume() {
    super.onResume();
    if (!new AccountData(getApplicationContext()).isAviableAccounts()) {
        startActivity(new Intent(this, LaunchActivity.class));
        finish();
        return;
    }
    try {
        new loadProgressSpinner().execute();
```

```
        } catch (Exception e) {
            e.printStackTrace();
        }
        br = new BroadcastReceiver() {
            public void onReceive(Context context, Intent intent) {
                int status =0;
                int refresh = 0;
                try{
                    status =
intent.getIntExtra(Config.BROADCAST_PARAM_STATUS_REFRESH, 0);
                    refresh =
intent.getIntExtra(Config.BROADCAST_PARAM_STATUS_SPINNER_SELECT, 0);
                }catch (Exception e){}
                Log.d("broadcast", "onReceive: status = " + status);
                if (status == Config.BROADCAST_REFRESH_CODE) {
                }
                if (refresh ==
Config.BROADCAST_REFRESH_CODE_ON_SPINNER_SELECT) {
                    AccountObj account = new
AccountData(getApplicationContext()).getCurrentAccount();
                    if (account.getType() != Config.ACCOUNT_TYPE_STUDENT) {
                        startActivity(new Intent(getApplicationContext(),
RaspisanieActivity.class));
                        finishAffinity();
                        finish();
                        return;
                    }
                }
            }
        };
        IntentFilter intFilt = new IntentFilter(Config.BROADCAST_ACTION);
        registerReceiver(br, intFilt);
Help.updateAccountSpin(getApplicationContext());
        notifyUpdate = new NotifyUpdate(this);
    }
    class loadProgressSpinner extends AsyncTask<Void, Void,
ArrayList<ProgressSpinnerObj>> {
        Spinner spinner = (Spinner) findViewById(R.id.spinner);
    public loadProgressSpinner() throws Exception{
    }
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            if(spinner.getAdapter()==null) {
                ((ProgressBar)
```

```
findViewById(R.id.ProgressBarProgress)).setVisibility(View.VISIBLE);
                findViewById(R.id.progressnofound).setVisibility(View.GONE);
                spinner.setVisibility(View.GONE);
                 setTitle("Успеваемость");
            }
        }
        @Override
        protected ArrayList<ProgressSpinnerObj> doInBackground(Void... params) {
                    ArrayList<ProgressObj> progress = new
ProgressData(getApplicationContext()).getData("student_id=" + new
AccountData(getApplicationContext()).getCurrentAccount().getaccount_id(),
"semestr_num", "semestr_num");
            ArrayList<ProgressSpinnerObj> spinnetobj = new
ArrayList<ProgressSpinnerObj>();
                for(ProgressObj progresstmp:progress){
                    spinnetobj.add(new
ProgressSpinnerObj(progresstmp.getSemestr_num(), progresstmp.getYear()));
                }
            return spinnetobj;
        }
        @Override
        protected void onCancelled() {
        }
        @Override
        protected void onPostExecute(ArrayList<ProgressSpinnerObj> result) {
            if(result==null) {
                spinner.setVisibility(View.GONE);
                setTitle("Успеваемость");
                findViewById(R.id.progressnofound).setVisibility(View.VISIBLE);
                return;
            }
((ProgressBar)findViewById(R.id.ProgressBarProgress)).setVisibility(View.GONE)
;
            spinner.setAdapter(new ProgressSpinnerAdapter(getApplicationContext(),
result));
            spinner.setSelection(result.size() - 1);
            if (result.size() == 0) {
                spinner.setVisibility(View.GONE);
                setTitle("Успеваемость");
                findViewById(R.id.progressnofound).setVisibility(View.VISIBLE);
            } else {
                spinner.setVisibility(View.VISIBLE);
                setTitle("");
                new loadViewProgress(result.size()-1);
            }
```

```
            spinner.setOnItemSelectedListener(ProgressActivity.this);
            super.onPostExecute(result);
        }
    }
    class loadViewProgress extends AsyncTask<Void, Void,
ArrayList<ProgressObj>> {
        private int position;
        public loadViewProgress(int position) {
            this.position = position;
        }
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
((ProgressBar)findViewById(R.id.ProgressBarProgress)).setVisibility(View.VISIBL
E);
            findViewById(R.id.progressnofound).setVisibility(View.GONE);
        }
        @Override
        protected ArrayList<ProgressObj> doInBackground(Void... params) {
            ArrayList<ProgressObj> progressData = new
ProgressData(getApplicationContext()).getData("student_id=" + new
AccountData(getApplicationContext()).getCurrentAccount().getaccount_id() + " and
semestr_num=" + (position), null, "type");
            ArrayList<ProgressObj> progress = new ArrayList<ProgressObj>();
            int lasttype = -1;
            for(ProgressObj progresstmp:progressData) {
                if (lasttype == -1) {
                    progress.add(new
ProgressObj(HelpFull.getTypeProgress(progresstmp.getType()), -1));
                    lasttype = progresstmp.getType();
                } else if (lasttype != progresstmp.getType()) {
                    progress.add(new
ProgressObj(HelpFull.getTypeProgress(progresstmp.getType()), -1));
                    lasttype = progresstmp.getType();
                }
                progress.add(new ProgressObj(progresstmp.getPredmetName(),
progresstmp.getStatus()));
            }
            return progress;
        }
        @Override
        protected void onCancelled() {
            super.onCancelled();
        }
        @Override
```

```
        protected void onPostExecute(ArrayList<ProgressObj> result) {
            super.onPostExecute(result);
((ProgressBar)findViewById(R.id.ProgressBarProgress)).setVisibility(View.GONE)
;
            if(result==null) return;
            if (result.size() == 0) {
                findViewById(R.id.progressnofound).setVisibility(View.VISIBLE);
            } else {
                listViewProgress.setVisibility(View.VISIBLE);
                findViewById(R.id.progressnofound).setVisibility(View.GONE);
                ProgressAdapter progressadapter = new
ProgressAdapter(getApplicationContext(), result);
                listViewProgress.setAdapter(progressadapter);
            }
        }
    }
    @Override
    public void onItemSelected(AdapterView<?> adapterView, View view, int i, long
l) {
        ProgressSpinnerObj spinnerobj = (ProgressSpinnerObj)
adapterView.getAdapter().getItem(i);
        new loadViewProgress(spinnerobj.getSemestr_num()).execute();
    }
    @Override
    public void onAccountChange() {
        try {
            new loadProgressSpinner().execute();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

Приложение Е

(обязательное)

Класс TeacherActivity отвечающий за вывод и поиск преподавателей

```
public class TeacherActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener,
TeacherSearchFragment.OnItemSelectedListener {
    class loadTeacherInfoStart extends AsyncTask<Void, Void,
ArrayList<TeacherObj>> {
        private RaspisanieObj lastrasp;
```

```
        private ListView listViewTeacher;
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            listViewTeacher = (ListView) findViewById(R.id.listViewteacher);
            listViewTeacher.setVisibility(View.GONE);
            lastrasp = null;
            ((ProgressBar)
findViewById(R.id.teacherlistprogressBar)).setVisibility(View.VISIBLE);
        }

        @Override
        protected ArrayList<TeacherObj> doInBackground(Void... params) {
            ArrayList<TeacherObj> teacherres = new ArrayList<TeacherObj>();
            ArrayList<RaspisanieObj> raspisanietmp;
            if (onlyMyTeacher) {
                raspisanietmp = new
RaspisanieData(getApplicationContext()).getData("teacher_id!=0 and student_group
= '" + new
AccountData(getApplicationContext()).getCurrentAccount().getGroupname() + "'",
"teacher", null, false); //true
            } else {
                raspisanietmp = new
RaspisanieData(getApplicationContext()).getData("teacher_id!=0", "teacher", null,
false);
            }
            for (RaspisanieObj raspisanie : raspisanietmp) {
                if (isCancelled()) {
                    Log.d("async", "cancel");
                    return null;
                }
                    ArrayList<RaspisanieObj> parinfo = new
RaspisanieData(getApplicationContext()).getData("teacher_id = '" +
raspisanie.getTeacher_id() + "' and dayofWeek=" + HelpFull.getDayofWeek() + "
and upOdown=" + HelpFull.getUporLowWeek(), null, null, false);
                    boolean teacherisaviable = false;
                    for (RaspisanieObj parinfotmp : parinfo) {
                        if (isCancelled()) {
                            Log.d("async", "cancel");
                            return null;
                        }
                        try {
                            if (parinfotmp.isVisibleProgress())
                                teacherisaviable = true;
                            else if (lastrasp != null)
```

```
                        if
(lastrasp.isVisibleNextPairProgress(parinfotmp.getTimestartinTime()))
                            teacherisaviable = true;
                        if (teacherisaviable) break;
                        lastrasp = parinfotmp;
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            if (onlyOnlineTeacher) {
                if (teacherisaviable)
                    teacherres.add(new TeacherObj(raspisanie.getTeacher(),
raspisanie.getTeacherfullname(), raspisanie.getTeacher_id(), teacherisaviable));
            } else
                teacherres.add(new TeacherObj(raspisanie.getTeacher(),
raspisanie.getTeacherfullname(), raspisanie.getTeacher_id(), teacherisaviable));
        }
        return teacherres;
    }
    @Override
    protected void onCancelled() {
        super.onCancelled();
        try {
            teacherTask = new loadTeacherInfoStart();
            teacherTask.execute();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    @Override
    protected void onPostExecute(ArrayList<TeacherObj> result) {
        super.onPostExecute(result);
        ((ProgressBar)
findViewById(R.id.teacherlistprogressBar)).setVisibility(View.GONE);
        teachereadapter = new TeacherAdapter(getApplicationContext(), result,
fullFIO);
        listViewTeacher.setVisibility(View.VISIBLE);
        listViewTeacher.setAdapter(teachereadapter);
        teachereadapter.notifyDataSetChanged();
        listViewTeacher.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View view,
                                    int position, long id) {
                Intent intent = new Intent(TeacherActivity.this,
```

```
RaspisanieActivity.class);
                intent.putExtra("teachersename",
teachereadapter.getItem(position).getTeachername());
                intent.putExtra("teacherid",
teachereadapter.getItem(position).getTeacherid());
                startActivity(intent);
            }
        });
        listViewTeacher.setOnItemLongClickListener(new
AdapterView.OnItemLongClickListener() {
            @Override
            public boolean onItemLongClick(AdapterView<?> parent, View view, int
position, long id) {
                Intent intent = new Intent(TeacherActivity.this,
AboutTeacherActivity.class);
                intent.putExtra("teacher_id",
teachereadapter.getItem(position).getTeacherid());
                startActivity(intent);
                return true;
            }
        });
        teacherTask = null;
    }
}


@Override
public void onResume() {
    super.onResume();
    if (!new AccountData(getApplicationContext()).isAviableAccounts()) {
        startActivity(new Intent(this, LaunchActivity.class));
        finish();
        return;
    }
    Help.updateAccountSpin(getApplicationContext());
    restartTaskLoadTeacher();
}
```

Приложение Ж

(обязательное)

Код обновления данных на сервере Ж

```
#!/usr/bin/php

<?php

//send telegram msg

	$msg = 'load timetable: start';

	$token='null';

	$private='null';

	$hash2 = md5($private.'|'.md5($msg.'|'.$token));

	file_get_contents("http://api.localhost.ru/token/token2/send?
message=".urlencode($msg)."&token=".urlencode($token)."&hash=".$hash2);


if (strtoupper(substr(PHP_OS, 0, 3)) === 'WIN') {

$docdir='E:\Dropbox/share/demon/cabinetrgupsv2';

} else {

$docdir='/home/demon/cabinetrgupsv2';

}

		copy("http://databaseSRV/databaseController.php?faculty=-1",
$docdir."/cache/facultet.xml");

$allfac = file_get_contents($docdir."/cache/facultet.xml");

$allfac = simplexml_load_string($allfac);

$allfac = json_decode(json_encode($allfac, JSON_UNESCAPED_UNICODE));

for($i=0; $i<count($allfac->ba); $i++){

	copy("http://databaseSRV/databaseController.php?faculty=" . $allfac->ba[$i]-
>z, $docdir."/cache/course/". $allfac->ba[$i]->z.'.xml');

	$curfac = file_get_contents($docdir."/cache/course/". $allfac->ba[$i]-
>z.'.xml');

	$curfac = simplexml_load_string($curfac);

	$curfac = json_decode(json_encode($curfac,
JSON_UNESCAPED_UNICODE));
```

```
for($j=0; $j<count($curfac->ba); $j++){
        copy("http://databaseSRV/databaseController.php?group=" .$curfac->ba[$j]->z, $docdir."/cache/raspisanie/".$curfac->ba[$j]->z.'.xml');
        echo "Load: cache/raspisanie/".$curfac->ba[$j]->z.'.xml
';
        }
        }
        echo'end load internal fac
';
        echo'start load external fac
';
        copy("http://databaseSRV/databaseController.php?facultyz=-1", $docdir."/cache/facultet_z.xml");
$allfac = simplexml_load_file($docdir."/cache/facultet_z.xml");
        foreach($allfac -> ba as $fac) {
copy("http://databaseSRV/databaseController.php?facultyz=" . extract_json(json_decode(json_encode($fac -> z))), $docdir."/cache/course_z/". extract_json(json_decode(json_encode($fac -> z))).'.xml');
$groupfile = simplexml_load_file($docdir."/cache/course_z/". extract_json(json_decode(json_encode($fac -> z))).'.xml');
foreach($groupfile -> ba as $group) {
copy("http://databaseSRV/databaseController.php? comingz=" .extract_json(json_decode(json_encode( $group -> z))), $docdir."/cache/coming_z/".extract_json(json_decode(json_encode( $group -> z))).'.xml');
echo "Load: cache/coming_z/".extract_json(json_decode(json_encode( $group -> z))).'.xml
';
```

```
$coming = simplexml_load_file($docdir."/cache/coming_z/".
extract_json(json_decode(json_encode( $group -> z))).'.xml');

        foreach($coming -> ba as $coming) {

        copy("http://databaseSRV/databaseController.php?
groupz=" .extract_json(json_decode(json_encode( $group ->
z))."&comingz=".extract_json(json_decode(json_encode($coming -> z))),
$docdir."/cache/raspisanie_z/".extract_json(json_decode(json_encode($coming ->
z))).".".extract_json(json_decode(json_encode( $group -> z))).'.xml');

echo" Load: /cache/raspisanie_z/".extract_json(json_decode(json_encode($coming ->
z))).".".extract_json(json_decode(json_encode( $group -> z))).'.xml

';

        }

}

        }

        function extract_json($json){

        if(is_object($json)){

                return extract_json($json->{'0'});

        }else return $json;

}

$msg = 'load timetable: success';


$hash2 = md5($private.'|'.md5($msg.'|'.$token));

echo file_get_contents("http://api.localhost.ru/token/token2/send?
message=".urlencode($msg)."&token=".urlencode($token)."&hash=".$hash2);
```