Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ А. И. Легалов
подпись инициалы, фамилия
«____» _____ 20 __ г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Разработка мобильного приложения электронный журнал
тема

Руководитель	_____ подпись, дата	ст. преп. должность, ученая степень	В. С. Васильев инициалы, фамилия
Студент	_____ подпись, дата		А. В. Пушкарев инициалы, фамилия
Нормоконтролер	_____ подпись, дата		В.И. Иванов инициалы, фамилия
Консультант	_____ подпись, дата		Г.А. Доррер

Красноярск 2017

Федеральное государственное автономное
образовательное учреждение
высшего профессионального образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ А.И. Легалов
подпись инициалы, фамилия

«__»_____2017 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы
бакалаврской работы, дипломного проекта, дипломной работы, магистерской диссертации

Студенту Пушкареву Александру Владимировичу
фамилия, имя, отчество

Группа КИ13-10Б Направление (специальность) 09.03.01.09
номер код

«Информатика и вычислительная техника»
наименование

Тема выпускной квалификационной работы Разработка мобильного приложения электронный журнал

Утверждена приказом по университету № 7366/с от 5.06.2017

Руководитель ВКР В.С. Васильев, старший преподаватель кафедры ВТ
инициалы, фамилия, должность, учебное звание и место работы

Исходные данные для ВКР: задание на ВКР

Перечень разделов для ВКР: введение, анализ технического задания, разработка приложения, тестирование, список использованных источников, приложение А

Перечень графического материала: презентация Power Point

Руководитель ВКР _____ В.С.Васильев
подпись инициалы и фамилия

Задание принял к исполнению _____ А.В.Пушкарев
подпись, инициалы и фамилия студента

«____» _____ 2017 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Разработка мобильного приложения: электронный журнал» содержит 33 страницы текстового документа, 10 использованных источников, 1 приложение, 24 иллюстрации.

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, ANDROID, ЭЛЕКТРОННЫЙ ЖУРНАЛ, РАСПИСАНИЕ, ПРЕДМЕТ, ГРУППА, СТУДЕНТ, ПРЕПОДАВАТЕЛЬ.

Цель: разработать мобильное приложение, позволяющее преподавателю просматривать расписание, вести учет посещаемости и успеваемости студентов.

Задачи: изучить предметную область, изучить аналоги приложения, разработать и протестировать приложение.

В результате работы было разработано приложение, позволяющее преподавателю просматривать свое расписание и вести учет посещаемости и успеваемости студентов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С развитием и повсеместным внедрением информационных технологий современные электронные носители постепенно заменяют альтернативные бумажные. Электронные носители обладают рядом преимуществ перед бумажными, такими как удельная стоимость хранения, размер носителя, удобство в изменении, копировании и передаче информации, возможность представления информации в удобном для конечного пользователя виде, хранение большого количества информации на относительно небольшом по размеру носителе, защита информации от несанкционированного доступа.

Благодаря этим преимуществам электронные устройства успешно выполняют функции тех или иных бумажных аналогов. Например, приложение для записи заметок легко заменит ежедневник. При этом записи можно просматривать на других устройствах, к примеру, на компьютере или ноутбуке. Информацию можно хранить в облачном хранилище, при этом даже потеряв свое устройство, данные можно восстановить.

Журнал, используемый преподавателями для учета посещаемости студентов и выполнения ими необходимых задач и работ, также не является исключением и может быть реализован на электронных устройствах.

В качестве платформы для реализации приложения были выбраны мобильные устройства на базе Android. Android является самой популярной мобильной системой и позволяет покрыть большинство целевой аудитории.

Целью выпускной квалификационной работы является разработка мобильного приложения электронный журнал, позволяющего преподавателям вести учет посещаемости студентов и учет выполненных ими лабораторных, курсовых и иных работ.

Задачи: изучить предметную область, спроектировать структуру приложения, разработать и протестировать приложение.

1 Анализ технического задания

1.1 Техническое задание

1.1.1 Наименование системы

Полное наименование системы:

Приложение электронный журнал для учета посещаемости и выполнения студентами лабораторных работ для мобильных устройств на базе Android.

Краткое наименование системы:

Приложение электронный журнал

1.1.2 Назначение и цели создания системы

Назначение:

Электронный журнал предназначен для:

- просмотра текущего расписания преподавателя;
- учета посещаемости студентов;
- учета выполненных работ студентом.

Цель создания системы:

Целью разработки является создание приложения для того, чтобы сделать удобнее ведение журнала преподавателем, автоматизации заполнения журнала, упрощения ведения учета выполнения студентами работ. Также целью является замещение классических бумажных журналов электронными.

1.1.3 Требования к системе

Требование к функционалу

Система должна обладать следующим функционалом:

- возможность войти в систему и выйти;

- просмотр расписания преподавателя в зависимости от четности недели;
- учет посещаемости студентов для каждого предмета;
- учет выполненных лабораторных и иных работ;
- добавление и удаление предметов по которым ведется посещаемость;
- добавление и удаление групп, обучающихся на определенном предмете;
- добавление и удаление студентов в группе;
- добавление и удаление лабораторных для предмета;
- возможность переименовать предмет, группу, студента или лабораторную.

Требование к особенностям программы

Система должна обладать следующими особенностями:

- автоматическое добавление журналов по предметам, основываясь на расписание преподавателя;
- автоматическое добавление групп, занимающихся на каждом предмете;
- автоматическое добавление дат, в которые проводится предмет;
- приложение запоминает последнего вошедшего пользователя.

Требование к аппаратной части

Приложение должно корректно выполняться на различных устройствах на базе Android версии 4.4 и выше. Вне зависимости от размеров и разрешения экрана интерфейс должен отображаться корректно и должны быть доступны все функциональные возможности.

Для работы приложения, мобильное устройство должно иметь доступ в интернет.

1.2 Обзор существующих решений

На данный момент существует несколько решений для удобного учета посещаемости студентов на мобильном устройстве. Проанализированные приложения являются универсальными, т.е. для пользования функционалом необходимо самостоятельно заполнить необходимые данные, такие как

дисциплина, группы обучающиеся на данной дисциплине, студенты, дни по которым проводится соответствующая дисциплина.

Первое проанализированное приложение называется Журнал преподавателя. Автор позиционирует это приложение как инструмент для ведения учета посещаемости и результативности работы студентов на занятиях. Бесплатная версия имеет ограничения по количеству создаваемых групп – не более 3, и количеству подгрупп – не более 2 для каждой группы. Кроме того, отключена функция импорта списка групп, подгрупп и студентов из CSV-файла. Неактивна также опция экспорта журнала в файл формата XLS (Excel). В платной версии приложения эти ограничения снимаются. Платную версию можно приобрести за 60 рублей. Интерфейс приложения изображен на рисунке 1.2.1.

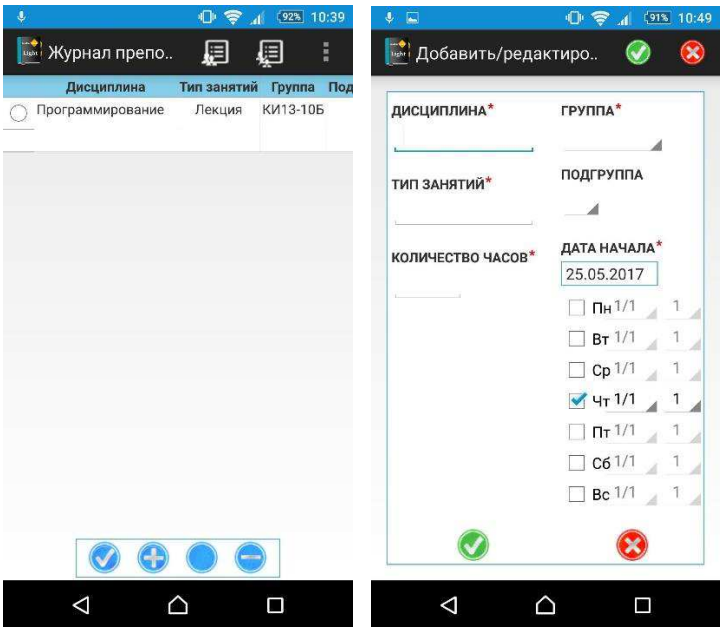

Рисунок 1.2.1 – Интерфейс приложения Журнал преподавателя

При пользовании этим приложением возникли трудности с заполнением журнала. Данные в приложении должны быть заполнены в определённом порядке. Журнал не может быть создан без предварительного создания группы, после создания группы приложение сообщает, что журнал не может быть создан

без студентов, обучающихся в группе, и наконец, после создания студентов журнал успешно добавляется в приложение. Несколько раз приложение останавливало свою работу. А также не удалось выяснить каким образом отмечается успеваемость. По итогу пользования приложение не оправдало своей заявленной функциональности.

Вторым проанализированным приложением стал Журнал учителя. Данное приложение подходит как для преподавателей вузов, так и для учителей школ. Интерфейс приложения не адаптирован под разные разрешения экранов и отображается некорректно (рисунок 1.2.2).

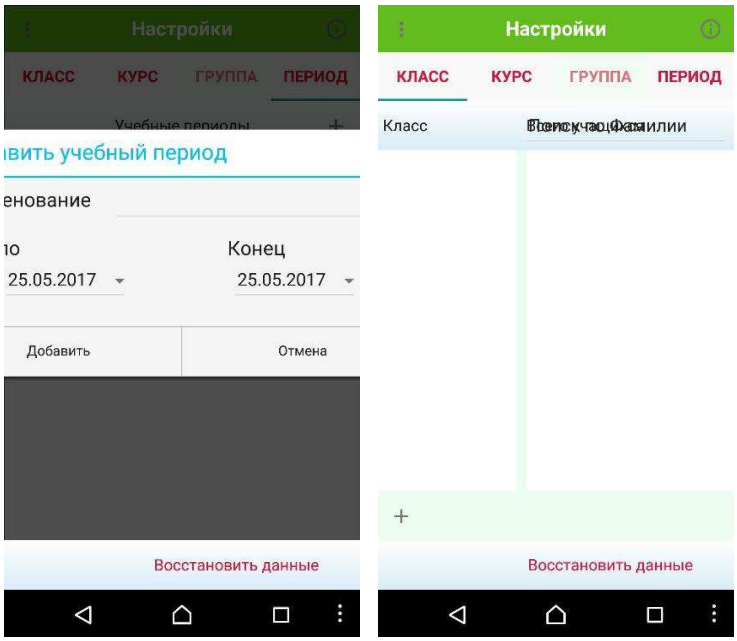

Рисунок 1.2.2 – Интерфейс приложения Журнал учителя

Сложный интерфейс, не понятный на интуитивном уровне, усложняет пользование этим приложением.

Последним приложением стало приложение Study Journal. Данная программа предназначена для преподавателей ВУЗов. Предоставляет возможность вести журналы посещаемости и успеваемости учащихся. Дополнительные возможности: настройка графического представления журнала (таблицы), импорт записей о студентах/группах из файлов,

сохранение/восстановление базы данных (SD-card). Интерфейс представлен на рисунке 1.2.3.

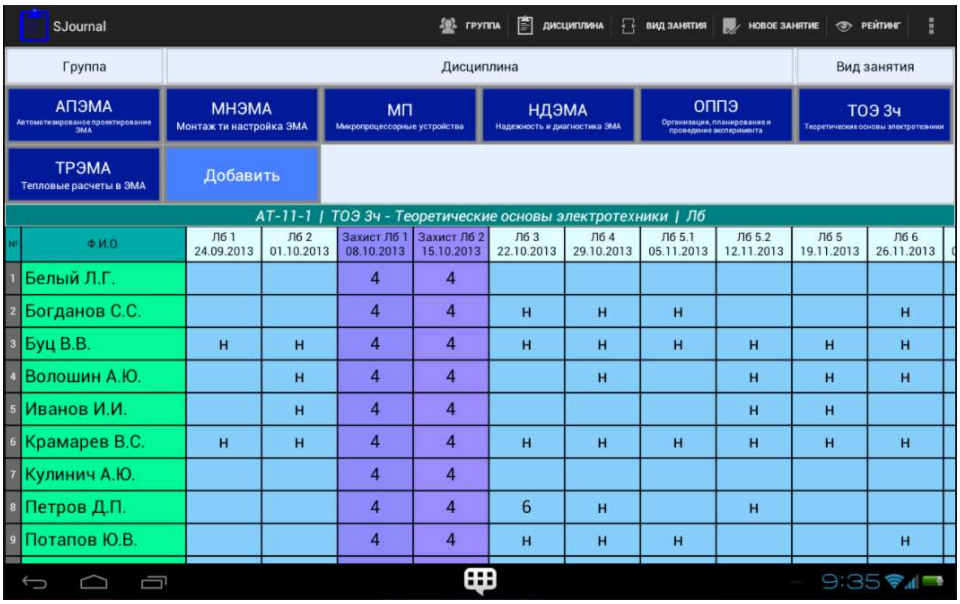

Рисунок 1.2.3 – Интерфейс приложения Study Journal

Недостатком данного приложения, как заявляет сам разработчик, является то, что оно подходит только для планшетов. На смартфонах приложение запускается, но часть интерфейса может оказаться недоступной, что сделает невозможным полноценную работу с приложением.

2 Разработка приложения

2.1 Инструмент разработки

Средой разработки была выбрана программа Android Studio. Эта программа является официальной средой для разработки Android приложений. Разработка в данной среде ведется с помощью языка Java.

2.2 Прецеденты

На рисунке 2.2.1 представлена use-case диаграмма системы. На ней изображены варианты использования системы.

Рисунок 2.2.1 – use-case диаграмма системы

Название прецедента: вход в систему.

Цель: войти в систему для получения доступа к основным функциям приложения.

Предусловия: ранее не было произведено входа в систему.

Главная последовательность:

- преподаватель вводит свою фамилию и инициалы, нажимает кнопку войти;

- система успешно запускается и выводит расписание преподавателя на текущую неделю.

Альтернативная последовательность (отсутствует интернет соединение):

- преподаватель вводит фамилию и инициалы, нажимает кнопку войти;

- выводится сообщение о том, что отсутствует интернет соединение.

Альтернативная последовательность (в базе расписания СФУ не существует данный преподаватель):

- преподаватель вводит фамилию и инициалы, нажимает кнопку войти;

- выводится сообщение о том, что такого преподавателя не существует.

Название прецедента: добавление или удаление предмета

Цель: добавление предметов преподавателя, которых нет в расписании на сайте sfu-kras.ru (таких как расписание заочных групп). Удаление предметов, которые не актуальны для преподавателя.

Предусловия: в навигационном меню выбран раздел «Журнал».

Главная последовательность (добавление нового предмета)**:**

- на экране отображен список предметов, которые ведет преподаватель;

- пользователь нажимает кнопку «Добавить»;

- открывается диалоговое окно с полем для ввода названия добавляемого предмета и кнопкой «Добавить»;

- пользователь вводит название нового предмета и нажимает кнопку «Добавить»;

- новый предмет добавлен и отображен в списке предметов.

Главная последовательность (удаление предмета):

- на экране отображен список предметов, которые ведет преподаватель;

- пользователь долгим нажатием на предмет вызывает контекстное меню и нажимает кнопку «Удалить»;

- происходит удаление предмета и появляется окно с обновленным списком предметов преподавателя.

Название прецедента: добавление или удаление группы

Цель: добавление новой группы или удаление неактуальной группы для конкретного предмета

Предусловия: был совершен выбор одного из предметов в списке предметов преподавателя.

Главная последовательность (добавление группы):

- на экране отображен список групп, которые обучаются на конкретном предмете;

- пользователь нажимает кнопку «Добавить»;

- открывается диалоговое окно с полем для ввода названия добавляемой группы и кнопкой «Добавить»;

- пользователь вводит название новой группы и нажимает кнопку «Добавить»;

- новая группа добавлена и отображена в списке групп конкретного предмета.

Главная последовательность (удаление группы):

- на экране отображен список групп для конкретного предмета;

- пользователь долгим нажатием на группу вызывает контекстное меню и нажимает кнопку «Удалить»;

- происходит удаление группы и появляется окно с обновленным списком групп.

Название прецедента: добавление или удаление студентов

Цель: добавление студентов в группы, удаление студентов, которые были отчислены, переведены в другую группу или по другим причинам.

Предусловия: была выбрана конкретная группа

Главная последовательность (добавление студента):

- на экране отображен список студентов, выбранной ранее группы

- пользователь нажимает кнопку «Добавить»;

- открывается диалоговое окно с полем для ввода имени и фамилии студента и кнопкой «Добавить»;

- пользователь вводит имя студента и нажимает кнопку «Добавить»;

- новый студент добавлен и отображен в списке студентов выбранной группы.

Главная последовательность (удаление студента):

- на экране отображен список студентов, выбранной ранее группы;

- пользователь долгим нажатием на студента вызывает контекстное меню и нажимает кнопку «Удалить»;

- происходит удаление студента и появляется окно с обновленным списком студентов.

Название прецедента: отметить посещаемость

Цель: отметить посещаемость того или иного студента, на конкретном предмете

Предусловия: был выбран предмет, группа и студент

Главная последовательность:

- на экране отображен список дат, по которым проводится выбранный предмет;

- при долгом нажатии на дату, открывается контекстное меню с выбором отметить присутствие или отсутствие студента;

- пользователь выбирает кнопку «Присутствовал»;

- выбранная дата помечается зеленым цветом.

Альтернативная последовательность (выбранная дата отмечена зеленым цветом):

- на экране отображен список дат, по которым проводится выбранный предмет;

- при долгом нажатии на дату, открывается контекстное меню с выбором отметить присутствие или отсутствие студента;

- пользователь выбирает кнопку «Отсутствовал»;

- выбранная дата помечается белым цветом.

Название прецедента: выход

Цель: выйти из системы, для возможности зайти другому пользователю

Предусловие: был осуществлен вход в приложение

Главная последовательность:

- в меню навигации пользователь выбирает пункт «Выйти»;

- появляется окно входа в систему, предоставляется возможность войти новому пользователю.

2.3 Исходные данные

Данные о расписании, предметах и группах преподавателя берутся с официального сайта СФУ по адресу http://edu.sfu-kras.ru/timetable#teachers. Подставляя в ссылке edu.sfu-kras.ru/timetable?teacher=Фамилия+И.+О. фамилию и инициалы преподавателя можно получить его расписание.

С помощью библиотеки jsoup происходит парсинг страницы с расписанием конкретного преподавателя, откуда берутся данные о предметах, группах, четности недели, связях предмет с группами. Все эти данные заносятся в базу данных SQLite при входе пользователя в систему.

Парсинг происходит при попытке входа пользователя в систему, если пользователь ввел фамилию и инициалы преподавателя, расписания которого не существует, при парсинге страницы обнаружится пустая таблица с расписанием, и приложение выведет сообщение о том, что данного преподавателя не существует.

2.4 Структура базы данных

В Android для локального хранения и доступа к данным используется SQLite.

В базе данных хранятся данные о предметах(Subjects), группах(Grousp), связях предметов и групп(SubjectGroup), датах по каким дням проводится предмет(Dates), студентах(Students), посещаемости студентов(Attendance) и их успеваемости(Labs). Диаграмма базы данных представлена на рисунке 2.3.1.

При входе в приложение в таблицу Subjects автоматически добавляются предметы, которые ведет преподаватель, основываясь на расписании. Затем в таблицу Groups добавляются группы, которые занимаются у преподавателя, а также в SubjectGroup добавляются связи между группами и предметами, то есть какие группы занимаются на том или ином предмете. Также заполняется таблица Dates датами по каким дням проводится определенный предмет. Таблицы Students, Labs и Attendance изначально пустые и заполняются по мере внесения данных пользователем.

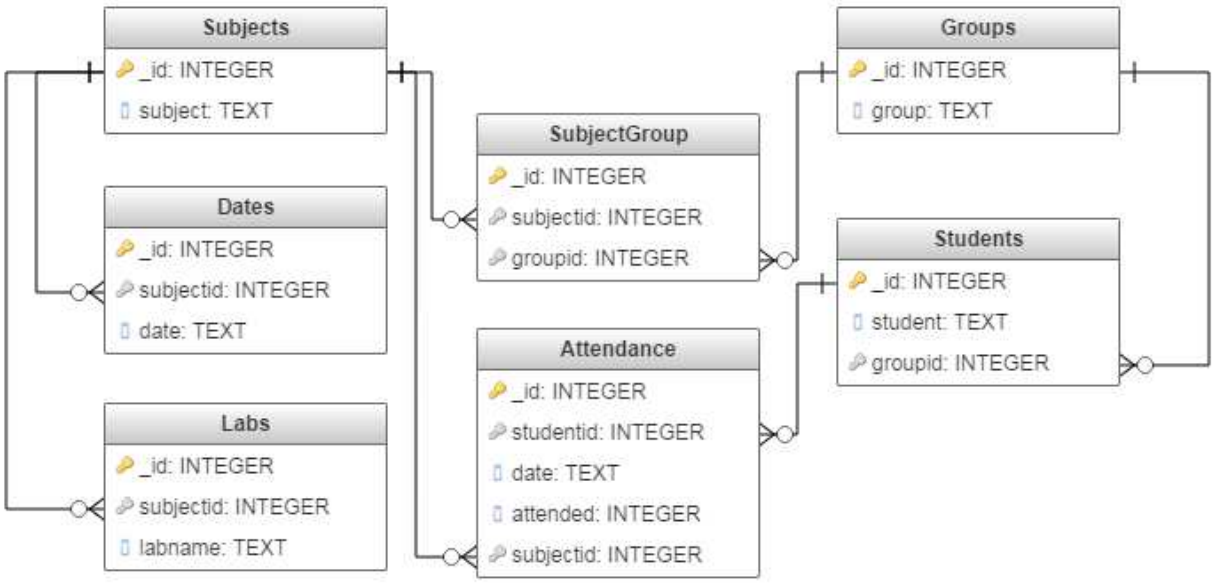

Рисунок 2.3.1 – Диаграмма базы данных

Для доступа к базе используется класс GradebookDBHelper (код класса представлен в приложении А). Этот класс наследуется от класса SQLiteOpenHelper. В методе onCreate(SQLiteDatabase) добавляются таблицы в базу данных.

2.5 Расписание

Для хранения и удобного доступа к данным о расписании преподавателя используются классы Timetable и Table (код классов представлен в приложении А). Класс Table хранит расписание для конкретного дня недели и предоставляет доступ к этим данным. В свою очередь класс Timetable хранит список экземпляров класса Table, тем самым предоставляет доступ к полному расписанию на каждый день недели. Диаграмма этих двух классов представлена на рисунке 2.5.1.

Класс Table содержит два списка с названиями предметов, первый список oddSubjectList содержит предметы, которые идут по нечетной неделе, а список evenSubjectList – предметы, которые идут по четной неделе.

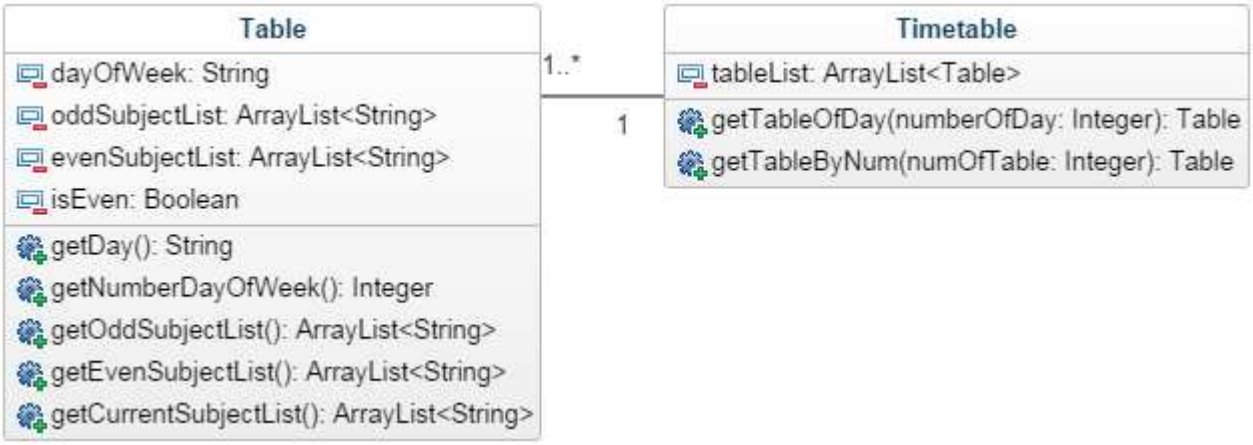

Рисунок 2.5.1 – Диаграмма классов Table и Timetable

Переменная dayOfWeek содержит название дня недели для которого представлено расписание. Переменная isEven показывает четность недели.

Вышеперечисленные переменные помечены модификатором доступа private. Для доступа к дню недели для которого представлено расписание необходимо вызвать функцию getDay(). С помощью функций getOddSubjectList() и getEvenSubjectList() можно получить список предметов для нечетной и четной недель соответственно. Функция getCurrentSubjectList() возвращает список предметов для текущей недели.

В классе Timetable, переменная tableList содержит список расписаний для конкретных дней недели. В конструктор класса Timetable передается экземпляр класса Document, который содержит html-страницу с расписанием преподавателя, вошедшего в систему. Экземпляр класса Document получается с помощью выражения Jsoup.connect(url).get(), где url – это адрес страницы расписания преподавателя. С помощью функций, предоставляемых классом Document, можно извлекать данные из страницы, такие как четность недели, предметы, группы. В конструкторе происходит парсинг страницы с расписанием преподавателя, на основе полученных данных создаются экземпляры класса Table для каждого дня недели и добавляются в список tableList.

3 Тестирование

При запуске приложения появляется окно авторизации, предлагающее ввести фамилию и инициалы преподавателя, пытающегося войти в систему. Были протестированы следующие варианты:

- нажатие кнопки войти без ввода какой-либо информации;
- ввод фамилии для которой нет данных расписания на сайте sfu-kras.ru;
- вход без доступа в интернет.

При первых двух вариантах на экран выводится сообщение о том, что такого преподавателя не существует. Без доступа в интернет, при нажатии на кнопку «Войти» выводится сообщение «Отсутствует интернет соединение» (рисунок 3.1).

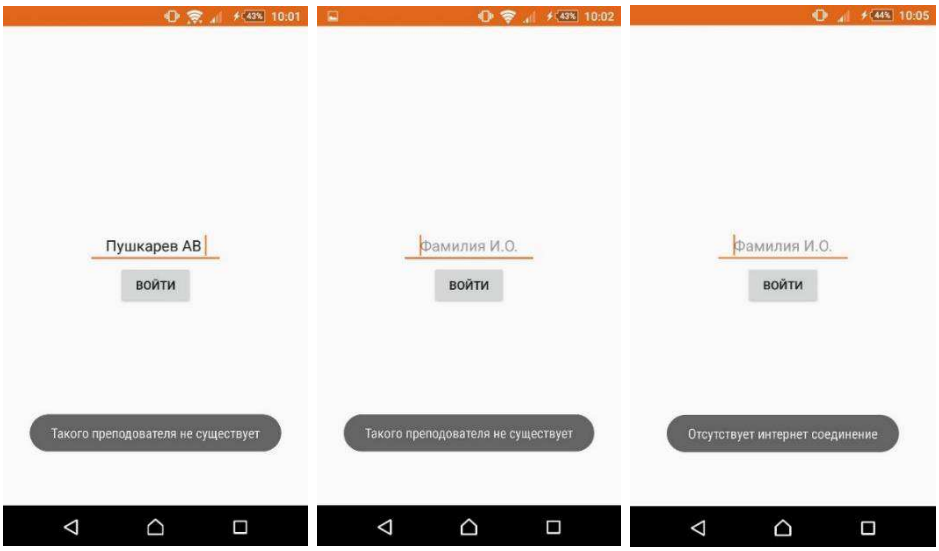

Рисунок 3.1 – тестирование авторизации

Для тестирования приложения войдем под именем Казаков ФА. После успешного входа появляется окно с расписанием. Листая влево и вправо можно переключатся между днями недели (рисунок 3.2). Недостатком является то, что если у преподавателя по одной неделе есть предмет, а по другой нету, будет выводится пустое поле в ту неделю, когда предмета нету.

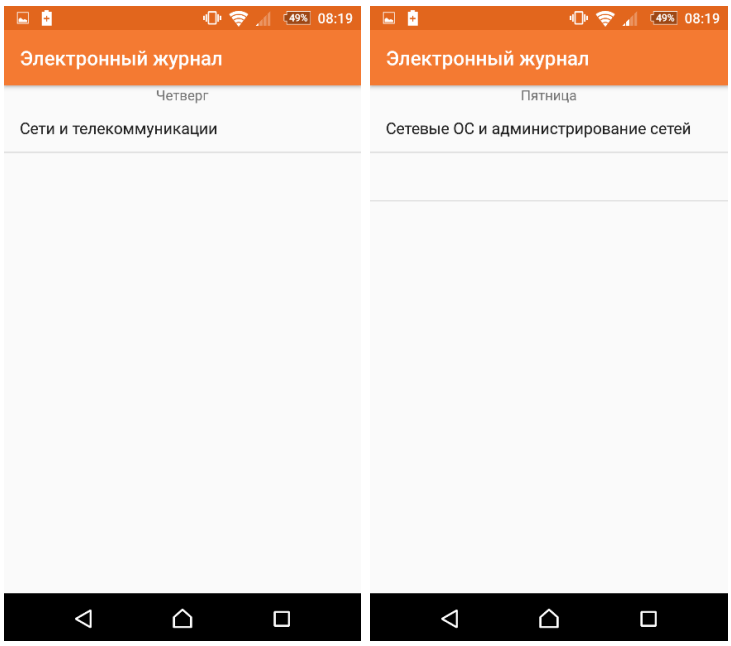

Рисунок 3.2 – Расписание преподавателя

Сделав слайд от левой стороны экрана вправо, откроется меню (рисунок 3.3) с выбором трех возможных вариантов: просмотр расписания, журналов и возможность выйти и зайти под новым именем.

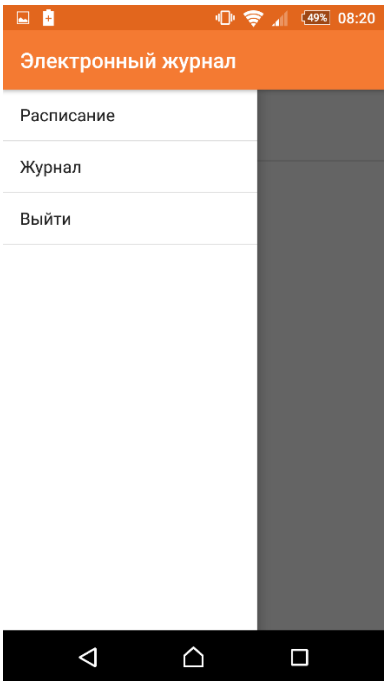

Рисунок 3.3 – Боковое меню выбора

Выбрав пункт меню «Журнал», откроется список предметов, преподаваемых преподавателем (рисунок 3.4).

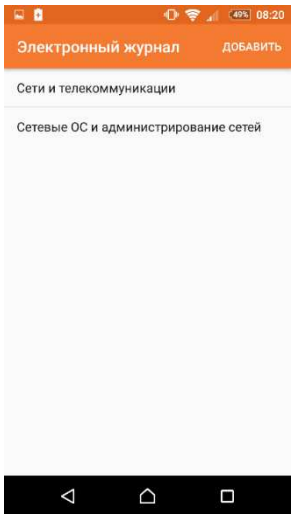

Рисунок 3.4 – Список предметов

Предметы добавляются, основываясь на расписании. Зайдя под разными пользователями и сверив предметы с расписанием, убеждаемся, что предметы добавляются корректно, не происходит дублирования и добавляются все предметы. Также в этом окне доступны функции добавления новых предметов, удаления и возможность переименовать предмет (рисунок 3.5).

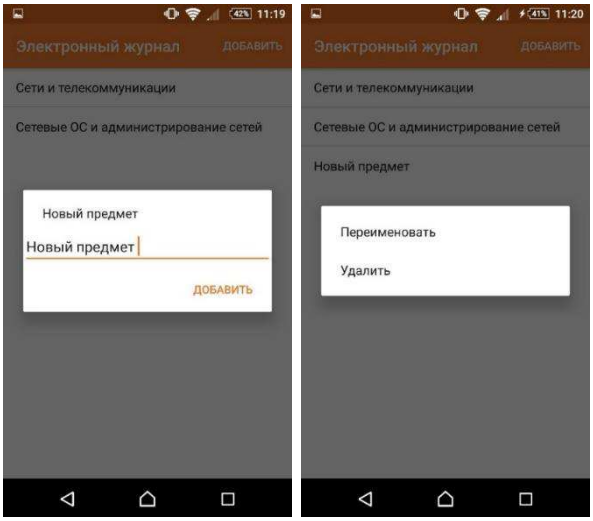

Рисунок 3.5 – Функции для редактирования предметов

При нажатии на один из предметов, появляется окно с группами (рисунок 3.6), обучающимися на данном предмете. Группы также берутся из расписания преподавателя.

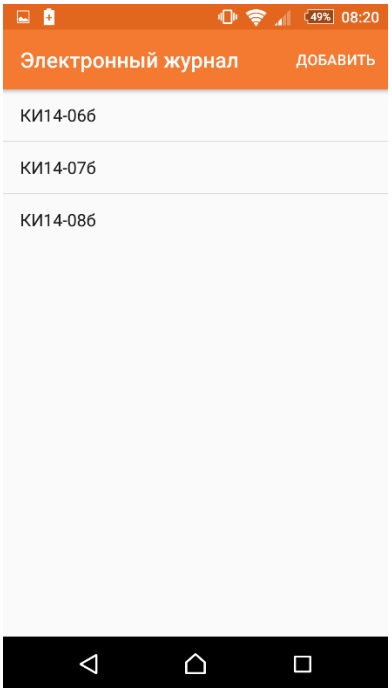

Рисунок 3.6 – Список групп, обучающихся на предмете

Сверив расписание и полученный список групп, убеждаемся в корректности добавления групп в приложение.

Также, как и в окне с предметами, в окне групп доступны функции удаления, добавления и возможность переименовать группу. Все функции работают корректно и без ошибок.

При нажатии на одну из групп открывается список студентов, обучающихся в данной группе. Из-за отсутствия информации в свободном доступе о студентах, обучающихся в той или иной группе, изначально список пуст. Добавление студентов необходимо производить вручную с помощью кнопки «Добавить» (рисунок 3.7). Все функции доступные в данном окне, такие как добавление, удаление и возможность переименовать, выполняются корректно и без ошибок.

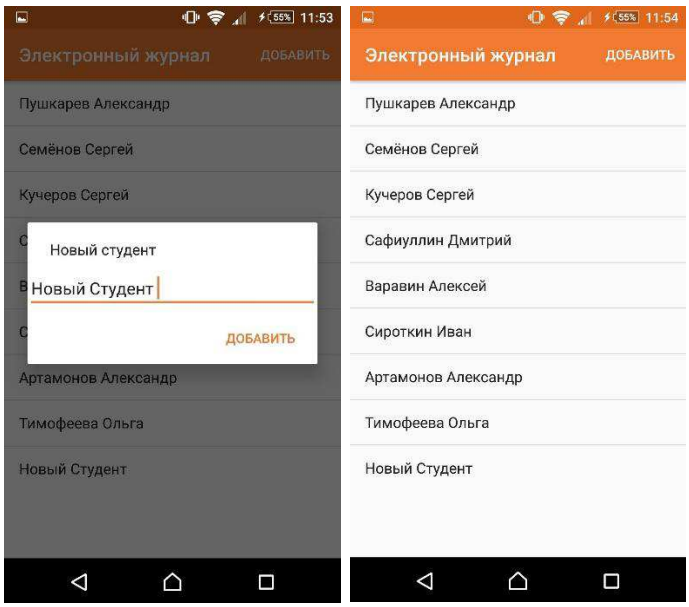

Рисунок 3.7 – Добавление студента

При нажатии на определенного студента открывается список с датами, в которые в соответствии с расписанием проводится предмет. При долгом нажатии на дату открывается контекстное меню, позволяющее отметить присутствие студента в тот или иной день. Если отметить присутствие студента в тот или иной день, этот день закрасится зеленым цветом (рисунок 3.8).

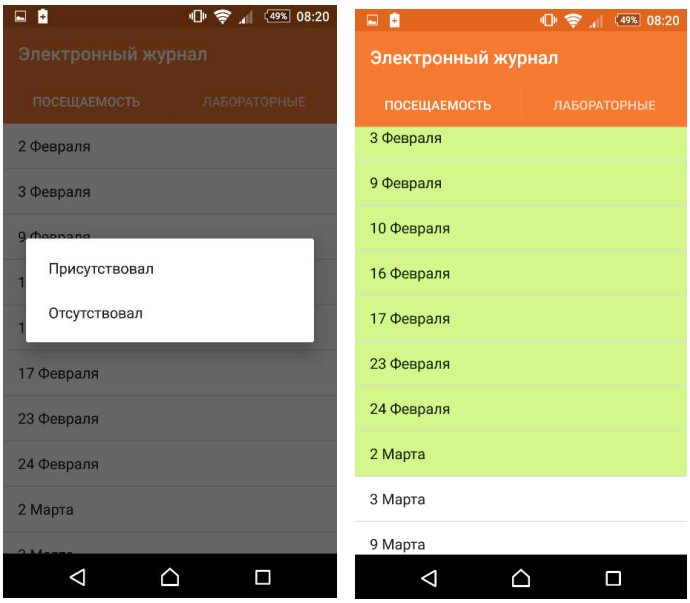

Рисунок 3.8 – Функция учета посещаемости

В данном окне находится две вкладки. В одной отмечается посещаемость, в другой - выполнение лабораторных работ (рисунок 3.9).

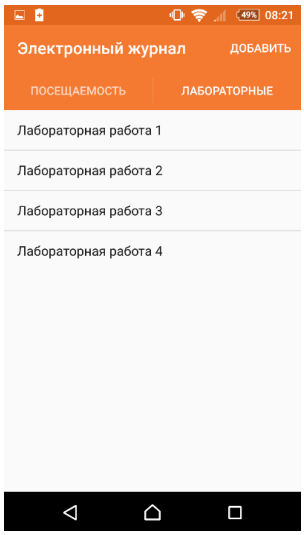

Рисунок 3.9 – список лабораторных работ

Перейти на другую вкладку можно нажав на нее, либо пролистнув по экрану. Список лабораторных работ изначально пуст, добавить лабораторную можно с помощью кнопки «Добавить». При долгом нажатии появляется контекстное меню с выбором действий: отметить выполнение или невыполнение лабораторной работы, переименовать или удалить работу (рисунок 3.10).

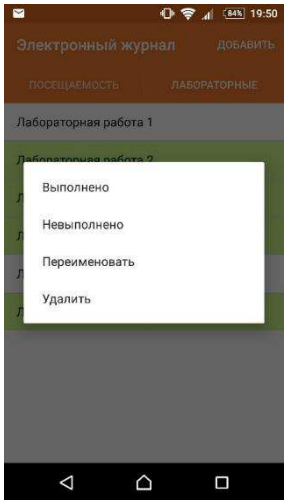

Рисунок 3.10 – Функция учета успеваемости

Когда отмечают лабораторную работу как выполненную, она загорается зеленым цветом. Все функции работают корректно и без ошибок.

В результате тестирования ошибки не были обнаружены. Все необходимые функции работают исправно.

ЗАКЛЮЧЕНИЕ

В результате данной работы было создано приложение Электронный журнал. Приложение позволяет преподавателю просматривать расписание, отмечать посещаемость и успеваемость студентов.

Все требования технического задания выполнены и протестированны.

В будущем данное приложение может быть интегрировано с электронными курсами и электронным журналом, что предоставит удобный интерфейс для учета посещаемости и успеваемости студентов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Шилдт, Герберт Java: руководство для начинающих: пер. с англ. / Г. Шилдт – Москва: ООО «И.Д. Вильямс», 2015. – 624 с.

2. Дейтел, Пол Android для разработчиков: отдельное изд. / П. Дейтел, Х. Дейтел, А. Уолд. – Санкт-Петербург: Питер, 2016. – 512 с.

3. Java HTML парсер [Электронный ресурс] – Режим доступа: https://jsoup.org/

4. Официальный сайт для Android-разработчиков [Электронный ресурс] – Режим доступа: https://developer.android.com/index.html

5. Журнал преподавателя [Электронный ресурс] – Режим доступа: https://play.google.com/store/apps/details?id=fizika03.Teacher.demo

6. Журнал учителя [Электронный ресурс] – Режим доступа: https://play.google.com/store/apps/details?id=fizika03.Teacher.demo

7. Study Journal [Электронный ресурс] – Режим доступа: https://play.google.com/store/apps/details?id=com.drprog.sjournal

8. Техническое задание [Электронный ресурс] – Режим доступа: http://www.prj-exp.ru/patterns/pattern_tech_task.php

9. UML диаграммы [Электронный ресурс] – Режим доступа: https://pro-prof.com/

10. Расписание преподавателей [Электронный ресурс] – Режим доступа: http://edu.sfu-kras.ru/timetable#teachers

ПРИЛОЖЕНИЕ А

Листинг класса GradebookDBHelper

```
public class GradebookDBHelper extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "GradebookDatabase5";
    public static final int DATABASE_VERSION = 15;

    public static final String TABLE_SUBJECTS = "Subjects";
    public static final String TABLE_GROUPS = "groups";
    public static final String TABLE_STUDENTS = "Students";
    public static final String TABLE_SUBJECTGROUP = "SubjectGroup";
    public static final String TABLE_DATES = "Dates";
    public static final String TABLE_ATTENDANCE = "Attendance";
    public static final String TABLE_LABS = "Labs";

    public static final String SUBJECTS_KEY_ID = "_id";
    public static final String SUBJECTS_KEY_NAME = "subject";
    public static final String GROUPS_KEY_ID = "_id";
    public static final String GROUPS_KEY_NAME = "name";
    public static final String STUDENTS_KEY_ID = "_id";
    public static final String STUDENTS_KEY_NAME = "student";
    public static final String STUDENTS_KEY_GROUPID = "groupid";
    public static final String SUBJECTGROUP_KEY_ID = "_id";
    public static final String SUBJECTGROUP_KEY_SUBJECTID = "subjectid";
    public static final String SUBJECTGROUP_KEY_GROUPID = "groupid";
    public static final String DATES_KEY_ID = "_id";
    public static final String DATES_KEY_SUBJECTID = "subjectid";
    public static final String DATES_KEY_DATE = "date";
```

```
    public static final String ATTENDANCE_KEY_ID = "_id";
    public static final String ATTENDANCE_KEY_STUDENTID = "studentid";
    public static final String ATTENDANCE_KEY_DATE = "date";
    public static final String ATTENDANCE_KEY_ATTENDED = "attended";
    public static final String ATTENDANCE_KEY_SUBJECTID = "subjectid";
    public static final String LABS_KEY_ID = "_id";
    public static final String LABS_KEY_SUBJECTID = "subjectid";
    public static final String LABS_KEY_NAME = "labname";
    public static final String LABS_KEY_DONE = "done";

    public GradebookDBHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        sqLiteDatabase.execSQL("create table " + TABLE_SUBJECTS + "(" +
                SUBJECTS_KEY_ID + " integer primary key, " +
                SUBJECTS_KEY_NAME + " text not null)");
        sqLiteDatabase.execSQL("create table " + TABLE_GROUPS + "(" +
                GROUPS_KEY_ID + " integer primary key, " +
                GROUPS_KEY_NAME + " text not null)");
        sqLiteDatabase.execSQL("create table " + TABLE_SUBJECTGROUP + "(" +
                SUBJECTGROUP_KEY_ID + " integer primary key, " +
                SUBJECTGROUP_KEY_SUBJECTID + " integer not null, " +
                SUBJECTGROUP_KEY_GROUPID + " integer not null, " +
                "foreign key(" + SUBJECTGROUP_KEY_SUBJECTID + ") references " +
TABLE_SUBJECTS + "(" + SUBJECTS_KEY_ID + "), " +
                "foreign key(" + SUBJECTGROUP_KEY_GROUPID + ") references " +
TABLE_GROUPS + "(" + GROUPS_KEY_ID + "))");
```

```
        sqLiteDatabase.execSQL("create table " + TABLE_STUDENTS + "(" +
                STUDENTS_KEY_ID + " integer primary key, " +
                STUDENTS_KEY_NAME + " text not null, " +
                STUDENTS_KEY_GROUPID + " integer not null, " +
                "foreign key(" + STUDENTS_KEY_GROUPID + ") references " +
TABLE_GROUPS + "(" + GROUPS_KEY_ID + "))");
        sqLiteDatabase.execSQL("create table " + TABLE_DATES + "(" +
                DATES_KEY_ID + " integer primary key, " +
                DATES_KEY_SUBJECTID + " integer not null, " +
                DATES_KEY_DATE + " text not null)");
        sqLiteDatabase.execSQL("create table " + TABLE_ATTENDANCE + "(" +
                ATTENDANCE_KEY_ID + " integer primary key, " +
                ATTENDANCE_KEY_STUDENTID + " integer not null, " +
                ATTENDANCE_KEY_DATE + " text not null, " +
                ATTENDANCE_KEY_ATTENDED + " integer not null, " +
                ATTENDANCE_KEY_SUBJECTID + " integer not null)");
        sqLiteDatabase.execSQL("create table " + TABLE_LABS + "(" +
                LABS_KEY_ID + " integer primary key, " +
                LABS_KEY_SUBJECTID + " integer not null, " +
                LABS_KEY_DONE + " integer not null, " +
                LABS_KEY_NAME + " text not null)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_SUBJECTS);
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_GROUPS);
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_STUDENTS);
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_SUBJECTGROUP);
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_DATES);
```

```
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_ATTENDANCE);
        sqLiteDatabase.execSQL("drop table if exists " + TABLE_LABS);
        onCreate(sqLiteDatabase);
    }
}
```

Листинг класса Timetable

```
public class Timetable{

    ArrayList<Table> tableList;
    boolean isEvenWeek;

    @RequiresApi(api = Build.VERSION_CODES.N)
    Timetable(Document document)
    {
        tableList = new ArrayList<Table>();
        Elements timeTableElements =
document.getElementsByTag("tbody").first().children();
        String nameWeek = document.select(".content").first().children().get(3).text();
        String name = nameWeek.split("\\s")[1];
        if(name.compareTo("чётная") == 0){
            isEvenWeek = true;
        }
        else{
            isEvenWeek = false;
        }
        String day = "";
        ArrayList<String> oddList = new ArrayList<String>();
        ArrayList<String> evenList = new ArrayList<String>();
```

```
for (int i = 0; i < timeTableElements.size(); i++) {
    Element el = timeTableElements.get(i);
    if (el.className().compareTo("heading heading-section") == 0){
        day = el.text();
        continue;
    }
    if(el.className().compareTo("table-center") == 0){
        Elements subjectsOfDay = el.getElementsByTag("b");
        if(subjectsOfDay.size() == 2){
            oddList.add(subjectsOfDay.get(0).text());
            evenList.add(subjectsOfDay.get(1).text());
        }
        else {
            Elements childTableElements = el.children();
            if(childTableElements.get(2).text().isEmpty())
            {
                oddList.add("");
                evenList.add(subjectsOfDay.text());
            }
            else{
                if (childTableElements.size() < 4)
                {
                    oddList.add(subjectsOfDay.text());
                    evenList.add(subjectsOfDay.text());
                }
                else {
                    oddList.add(subjectsOfDay.text());
                    evenList.add("");
                }
            }
```

```
            }
        }
        if((i+1)==timeTableElements.size() ||
el.nextElementSibling().className().compareTo("heading heading-section") == 0){
            tableList.add(new Table(day,new ArrayList<String>(oddList),new
ArrayList<String>(evenList),isEvenWeek));
            day = "";
            oddList.clear();
            evenList.clear();
        }
    }
}

public Table getTableOfDay(int numberOfDay){
    for (Table table:tableList) {
        if(table.getNumberDayOfWeek() == numberOfDay)
        {
            return table;
        }
    }
    return null;
}

public Table getTableByNum(int numOfTable){
    return tableList.get(numOfTable);
}

public int size(){
    return tableList.size();
}
```

```
    public boolean IsEvenWeek(){ return isEvenWeek;}
}
```

Листинг класса Table

```
public class Table{
    private String dayOfWeek;
    private ArrayList<String> oddSubjectList;
    private ArrayList<String> evenSubjectList;
    private boolean isEvenWeek;

    Table(String _dayOfWeek, ArrayList<String> _oddSubjectList, ArrayList<String>
_evenSubjectList, boolean _isEvenWeek)
    {
        dayOfWeek = _dayOfWeek;
        oddSubjectList = _oddSubjectList;
        evenSubjectList = _evenSubjectList;
        isEvenWeek = _isEvenWeek;
    }

    public String getDay(){
        return dayOfWeek;
    }
    public int getNumberDayOfWeek() {
        switch (dayOfWeek){
            case "Воскресенье" : return  1;
            case "Понедельник" : return  2;
            case "Вторник" : return  3;
            case "Среда" : return  4;
```

```
            case "Четверг" : return  5;
            case "Пятница" : return  6;
            case "Суббота" : return  7;
            default: return 1;
        }
    }
    public ArrayList<String> getOddSubjectList(){
        return oddSubjectList;
    }
    public ArrayList<String> getEvenSubjectList(){
        return evenSubjectList;
    }
    public ArrayList<String> getCurrentSubjectList() {
        if(isEvenWeek){
            return evenSubjectList;
        }
        return oddSubjectList;
    }
}
```

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Вычислительная техника
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

А. И. Легалов
подпись инициалы, фамилия
«19» 06 20 17 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника
код и наименование направления

Разработка мобильного приложения электронный журнал
тема

Руководитель	Васильев подпись, дата	ст. преп. должность, ученая степень	В. С. Васильев инициалы, фамилия
Выпускник	17.06.17 подпись, дата		А. В. Пушкарев инициалы, фамилия
Нормоконтролер	19.06.17 подпись, дата		В.И. Иванов инициалы, фамилия
Консультант	подпись, дата		Г.А. Доррер

Красноярск 2017