**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФГБОУ ВО

**«БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Кафедра** «Информатика и программное обеспечение»

**«У Т В Е Р Ж Д А Ю»**

Зав. кафедрой «И и ПО», к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подвесовский А.Г.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2017г.

**ОТЧЕТ ПО ПРЕДДИПЛОМНОЙ ПРАКТИКЕ**

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ РАСПИСАНИЯ КАФЕДРЫ ДЛЯ ПЛАТФОРМЫ ANDROID

Всего листов\_\_\_\_\_

Выполнил студент гр.13-ИВТ1 зач. кн.№\_\_

Леквеишвили Д.М.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017

Дипломный руководитель

асс. Панус Д.Н.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017 г.

Брянск 2017

**БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Направление 230100 – Информатика и вычислительная техника**

**профиль «Программное обеспечение вычислительной техники**

**и автоматизированных систем»**

**ЗАДАНИЕ**

**на дипломную работу**

студенту группы 13ИВТ-1

***Леквеишвили Давиду Мерабовичу***

1. Тема работы  *«Мобильное приложение расписания кафедры для платформы Android»*

Утверждена приказом по БГТУ № \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ от \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

1. Срок предоставления законченной работы  *июнь 2017 г.*
2. Исходные данные

Цель работы

Инструментальные средства  *Android Studio, StarUML*

Основные требования

Дата выдачи задания \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2017г.

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подвесовский А.Г.

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Панус Д.Н.

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2017г.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Леквеишвили Д.М.

АННОТАЦИЯ

В данном дипломном проекте представлены следующие главы и их описания.

**Анализ требований.** В данном разделе описан обзор предметной области, приведены основные термины и понятия, рассмотрена актуальность темы и выявлены основные проблемы. Кроме того, раздел включает в себя обзор программ-аналогов с описанием достоинств и недостатков, относящихся к теме дипломного проекта. Так же в данном разделе описан процесс моделирования разрабатываемой системы с его описанием. Описано техническое задание с перечислением требований к бедующей системе.

**Экономический анализ.** Включает в себя описание затрат на разработку и себестоимость программного продукта.

**Разработка программного продукта.** Включает в себя описание архитектуры системы, модели базы данных, схемы низкоуровневого проектирования на примере UML, описание этапов проектирования интерфейса, а также руководство для конечного пользователя.

**Экспериментальная часть.** В данном разделе, проводятся тестирование и проверка на соответствие системы заявленным в техническом задании требованиям, а также приводится описание исключительных ситуаций, которые могут возникнуть в ходу работы программы

**Организационная часть.** ДОБАВИТЬ ТЕКСТ

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc483160436)

[1. АНАЛИЗ ТРЕБОВАНИЙ 5](#_Toc483160437)

[1.1. Описание и анализ исследуемой проблемы 5](#_Toc483160438)

[1.1.1. Описание текущей ситуации 5](#_Toc483160439)

[1.1.2. Описание предметной области 5](#_Toc483160440)

[1.2. Обзор аналогов 6](#_Toc483160441)

[1.2.1. Google Календарь 6](#_Toc483160442)

[1.2.2. Microsoft Outlook 8](#_Toc483160443)

[1.2.3. Ruzov 10](#_Toc483160444)

[1.3. Функциональная модель разрабатываемой системы 13](#_Toc483160445)

[1.4. Функциональные требования 14](#_Toc483160446)

[1.4.1. Авторизация и роли 14](#_Toc483160447)

[1.4.2. Возможности роли «Гость» 15](#_Toc483160448)

[1.4.3. Возможности роли «Студент» 15](#_Toc483160449)

[1.4.4. Возможности роли «Преподаватель» 15](#_Toc483160450)

[1.5. Выводы 16](#_Toc483160451)

[2. ФУНКЦИОНАЛЬНЫЕ И ИНЫЕ ТРЕБОВАНИЯ 17](#_Toc483160452)

[2.1. Назначение разработки 17](#_Toc483160453)

[2.2. Требования к программе или программному продукту 17](#_Toc483160454)

[2.2.1. Функциональные требования 17](#_Toc483160455)

[2.2.2. Требование к технической документации 18](#_Toc483160456)

[2.3. Детальные требования 18](#_Toc483160457)

[2.4. Этапы разработки 23](#_Toc483160458)

[2.5. Порядок контроля и приемки 23](#_Toc483160459)

[3. ПРОЕКТНАЯ ЧАСТЬ 24](#_Toc483160460)

[3.1. Архитектура подсистемы оповещения 24](#_Toc483160461)

[3.1.1. Подсистема оповещения в системе учета посещаемости и успеваемости 24](#_Toc483160462)

[3.2. Описание средств разработки 25](#_Toc483160463)

[3.2.1. Описание СУБД 25](#_Toc483160464)

[3.2.2. Язык программирования 25](#_Toc483160465)

[3.2.3. Microsoft Visual Studio 25](#_Toc483160466)

[3.2.4. Технология ASP.NET 27](#_Toc483160467)

[3.2.5. Библиотека Bootstrap 28](#_Toc483160468)

[3.3. Модель данных программы 28](#_Toc483160469)

[4. ТЕСТИРОВАНИЕ 33](#_Toc483160470)

[4.1. Выбор методики тестирования 33](#_Toc483160471)

[4.2. Эквивалентное разбиение 33](#_Toc483160472)

[4.3. Анализ граничных значений 34](#_Toc483160473)

[4.4. Проектирование тестов 35](#_Toc483160474)

[4.4.1. Тестирование модуля просмотра наблюдения 35](#_Toc483160475)

[4.4.2. Тестирование модуля создания наблюдения 35](#_Toc483160476)

[4.4.3. Тестирование модуля редактирования наблюдения 36](#_Toc483160477)

[4.4.4. Тестирование модуля удаления наблюдения 36](#_Toc483160478)

[4.4.5. Тестирование отправки уведомлений 36](#_Toc483160479)

[4.4.6. Тестирование отправки уведомлений по указанным каналам связи 37](#_Toc483160480)

[ЗАКЛЮЧЕНИЕ 38](#_Toc483160481)

[СПИСОК ЛИТЕРАТУТЫ 39](#_Toc483160482)

# ВВЕДЕНИЕ

Актуальность разработки электронной системы расписания для университета обусловлена темпами технологического прогресса и высоким техническим оснащением как студентов, так и преподавателей, что говорит о том, что есть необходимость в улучшении процесса получения информации о расписании с использованием сетевых технологий.

**Целью** данной дипломной работы является автоматизация предоставления информации о расписании, путем разработки мобильного приложения для платформы Android.

Для достижения этой цели были поставлены следующие **задачи**:

1. Рассмотреть предметную область.
2. Рассмотреть программы аналоги, выявить их сильные и слабые

стороны.

1. Спроектировать будущее приложение согласно полученным

данным.

1. Разработать мобильное приложение.
2. Протестировать полученное мобильное приложение и исправить найденные ошибки.

**Объектом** исследования является работа системы расписания.

**Предметом** исследования в работе является основные аспекты в работе расписания.

# АНАЛИЗ ТРЕБОВАНИЙ

Данная глава содержит описание требований и обзор имеющихся аналогов. Здесь приведены основные понятия, описан процесс отправки оповещений на текущий момент. Также рассматриваются основные проблемы при отправке оповещений и способ их решения.

## Описание и анализ исследуемой проблемы

### Описание текущей ситуации

Все студенты и преподаватели пользуются расписанием университета,

но данное расписание не имеет возможности быстрого переноса, создания, изменения событий с последующим уведомлением всех участников. Сейчас расписание печатают и вывешивают в университете. Эта работа довольно трудоемкая. Также нет возможности быстро уведомить всех участников о переносе, изменении, создании новых событий. Некоторые студенты и преподаватели фотографируют или вбивают расписание вручную в свои смартфоны.

### Описание предметной области

Разрабатываемая система охватывает задачу работы с расписанием. Разработке расписания посвящено довольно много публикаций и статей. Правда большая часть рассматривает простое расписание, не заточенное под нужды университета.

Нужной работой является публикация «Как студенту иметь актуальное расписание, как старостам и преподавателям отправлять сообщения сразу всей группе» [1].

Данная публикация позволила прийти к результатам:

* произведен тщательный анализ проблемы и возможных вариантов её решения;
* обоснована актуальность системы;
* выбран метод вывода графической информации (схемы, диаграммы, иллюстрации);

Тем не менее, работа обладает принципиальным недостатком – она мало охватывает проектирование архитектуры. Это существенно снижает её ценность

в рамках разрабатываемой системы.

## Обзор аналогов

Для того чтобы выделить важные функции проектируемой подсистемы, необходимо провести сравнение уже существующих систем. Для данной подсистемы можно выделить два класса систем, которые мы будем считать аналогами:

* большие программы органайзеры;
* специализированное ПО для расписания университетов.

К первой группе относятся примеры, рассмотренные в пунктах 1.2.1 – 1.2.2, а второй группы в пункте 1.2.3.

### Google Календарь

Google Календарь – сервис для планирования встреч, событий и дел. Он позволяет задавать время встречи, создавать повторяющиеся мероприятия, устанавливать напоминания, а также приглашать других участников (им высылается сообщение по электронной почте) [1].

Напоминания о событиях можно получать по электронной почте и с помощью Push уведомлений. Работа с Календарём осуществляется в окне браузера через веб-интерфейс, а все данные хранятся на сервере Google, поэтому получить доступ к расписанию можно с любого компьютера, подключенного к интернету.

Можно создавать несколько календарей, а также автоматически помечать официальные праздники.

Одно из ключевых преимуществ этого сервиса – возможность совместного использования календаря. Его можно показывать избранным пользователям, что позволяет планировать общие встречи. На январь 2017 года на платформе зарегистрировано 300 тысяч пользователей.

Проект предлагает бесплатное использование, но с ограничениями: нет возможности создавать общее расписание с функциями создания, редактирования, удаления событий. За использование данных функций, предлагают заплатить за каждого пользователя 5$.

Для университета нужно будет использовать ежемесячную подписку 5$, что бы данная система соответствовала требованиям.

В данной системе имеется возможность фильтровать события по нужным нам группам, студентам, преподавателям. Невозможность присоединить свою базу данных к данной системе делает данную систему сложной для пользования, так как придется вручную вбивать все события. Также присутствует невозможность создания ролей для пользователей и распределения обязанностей между ролями (кроме владельца).

Есть возможность создавать «рис 1.1.1», редактировать и удалять события.

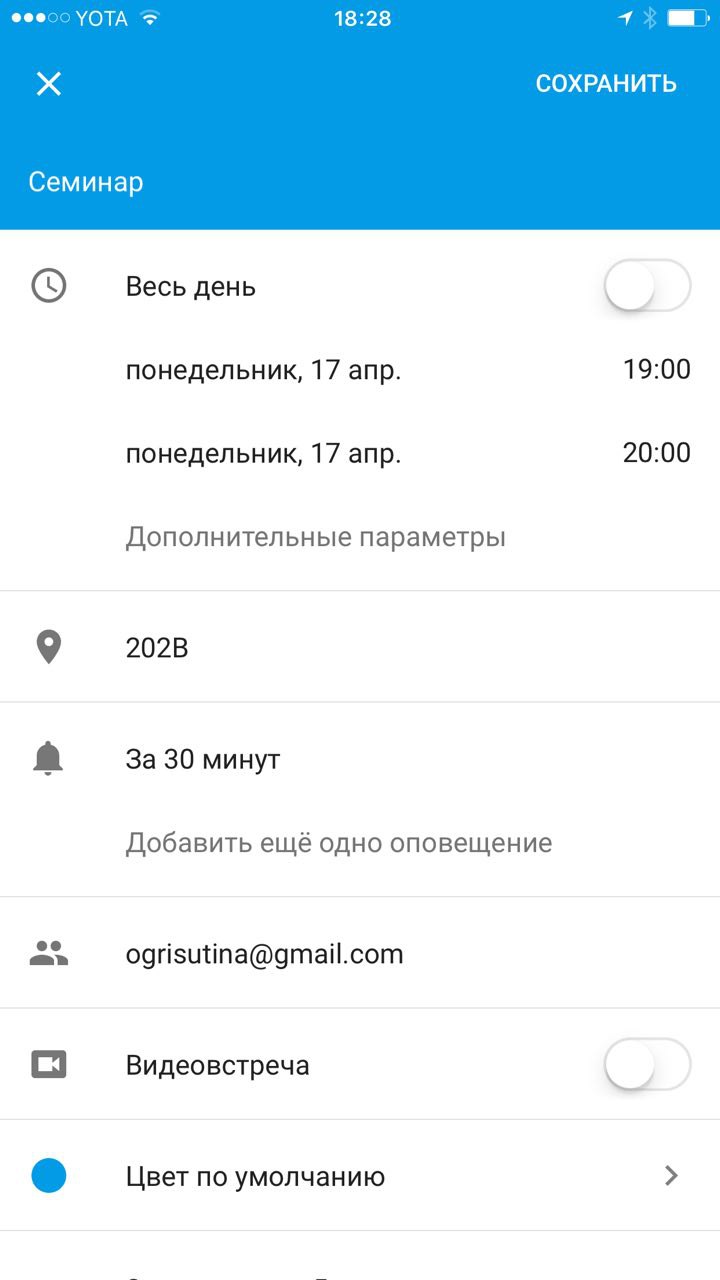


Рис. 1.1.1. Создание события в Google Календарь

Интерфейс приложения «рис 1.1.2» довольно сложен и не позволяет быстро найти нужное событие по заданным критериям. Фильтр по времени, участникам и аудиториям присутствует.



Рис. 1.1.2. Просмотр события в Google Календарь

### Microsoft Outlook

Microsoft Outlook является полноценным органайзером, предоставляющим функции календаря, планировщика задач, записной книжки и менеджера контактов. Кроме того, программа позволяет отслеживать работу с документами пакета Microsoft Office для автоматического составления дневника работы.

Microsoft Outlook не имеет подписки и предоставляется бесплатно. Данная форма сотрудничества подходит университету и выгодно выделяет данную программу от предыдущего аналога.

В данной системе имеется возможность фильтровать события по нужным нам группам, студентам, преподавателям, как и в предыдущем аналоге. Невозможность присоединить свою базу данных к данной системе делает невозможным загрузку данных из базы данных университета. Как и в предыдущем приложении здесь имеется возможность создания «рис 1.2.1», редактирования и удаления событий. Интерфейс в данной программе отображает больше информации «рис 1.2.2», чем в предыдущем аналоге.

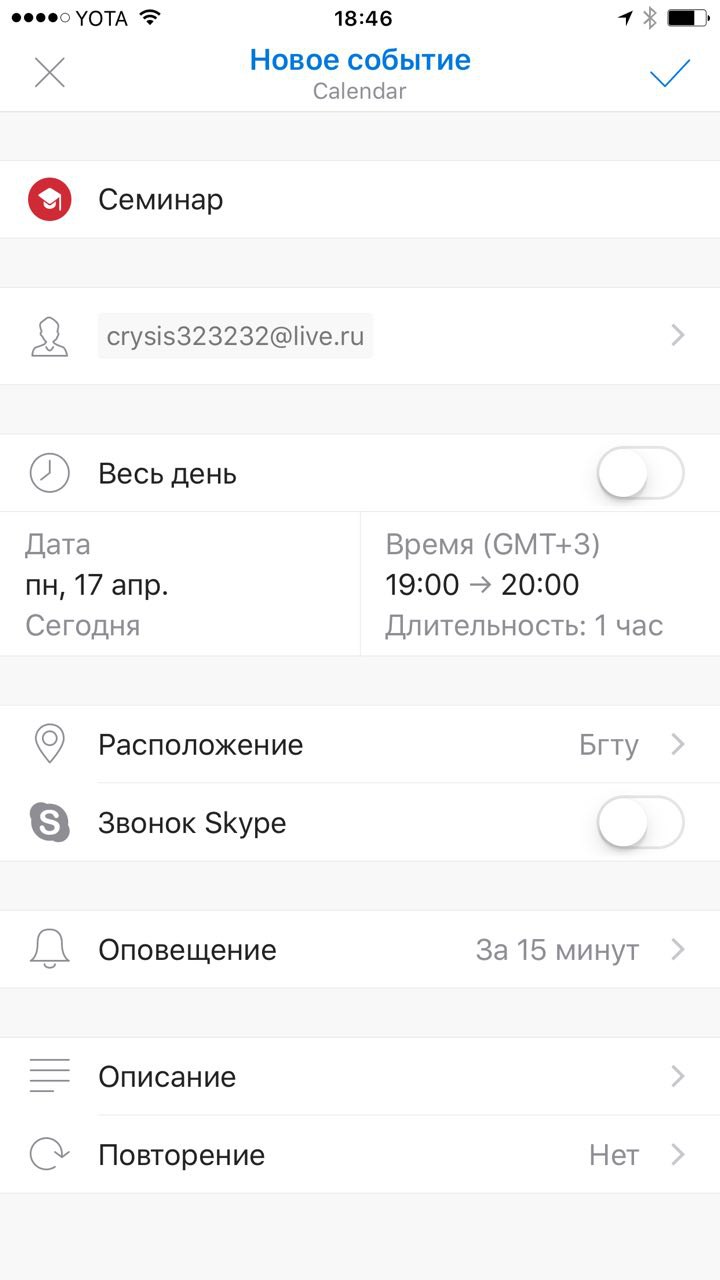


Рис. 1.2.1. Создание события в Microsoft Outlook

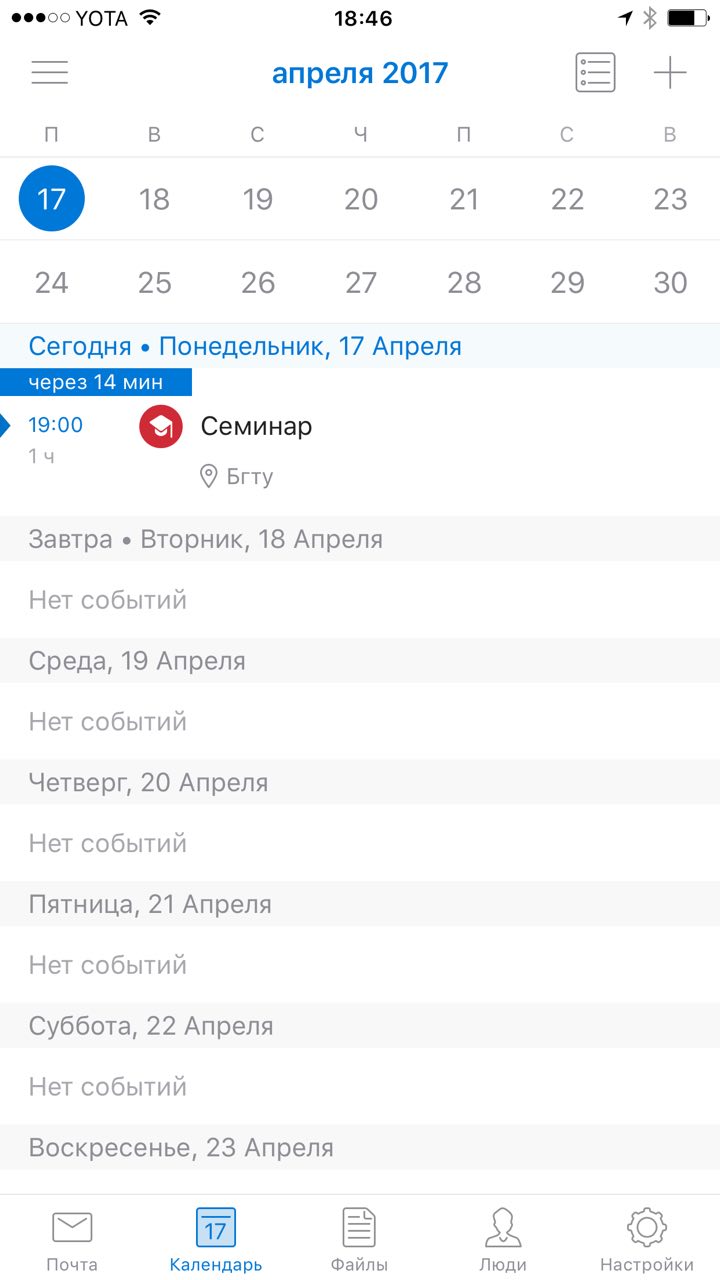


Рис. 1.2.2. Просмотр расписания в Microsoft Outlook

### Ruzov

Rvuzov – расписание занятий для студентов. Приложение, позволяющее студентам и преподавателям пользоваться мобильным расписание где угодно. Специализированное специально для университетов ПО, позволят быстро интегрировать систему в учебный план [1].

Данная система сотрудничает с университетами совершенно бесплатно, но что бы добавлять, заменять или удалять события нужно приобрести подписку. На выбор есть ежемесячная подписка 1.99$ и годовая 6.99$.

Данная система позволяет производить поиск событий по группам, студентам, преподавателям и, что выгодно отличает ее от аналогов, кабинетам. Интеграция БД здесь значительно упрощена, создатели системы могу помочь с переносом БД в их систему. На экране приложения присутствует вся нужная информация, не перегружая интерфейс и позволяя быстро начать работать с программой «рис 1.3.1».

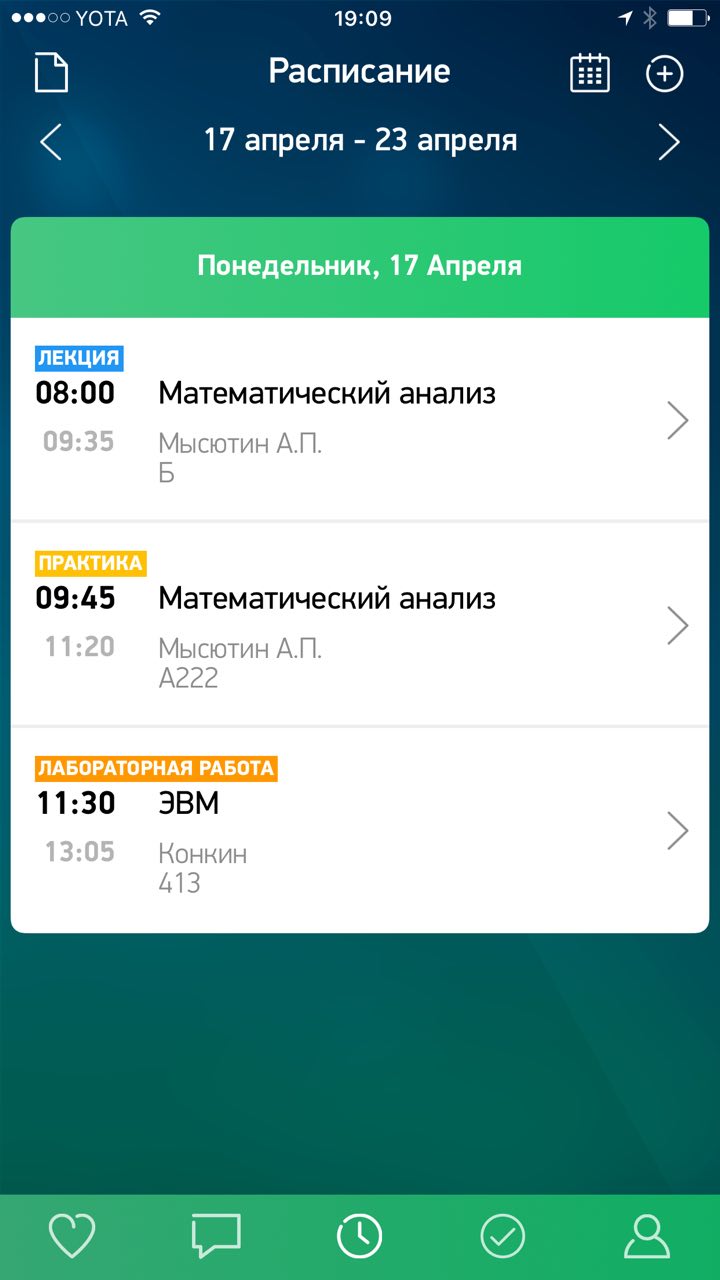


Рис. 1.3.1. Просмотр события в Rvuzov

Также присутствует расширенный просмотр события «рис 1.3.2» с возможностью просмотра детальной информации, такой как время события, имя преподавателя, аудитория.

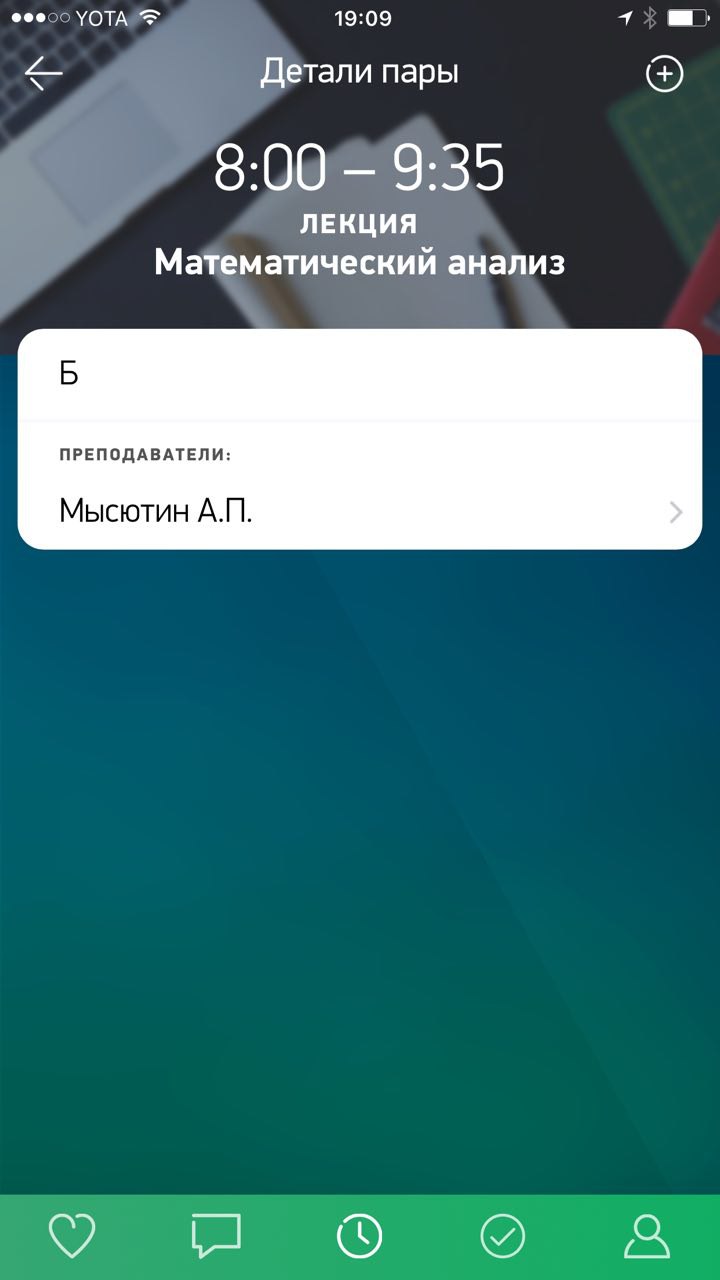


Рис. 1.3.2. Расширенный просмотр события в Rvuzov

Создание событий с возможностью создания повторяющихся по четным и нечетным неделям, позволяет добавлять в расписание пары «рис 1.3.3».

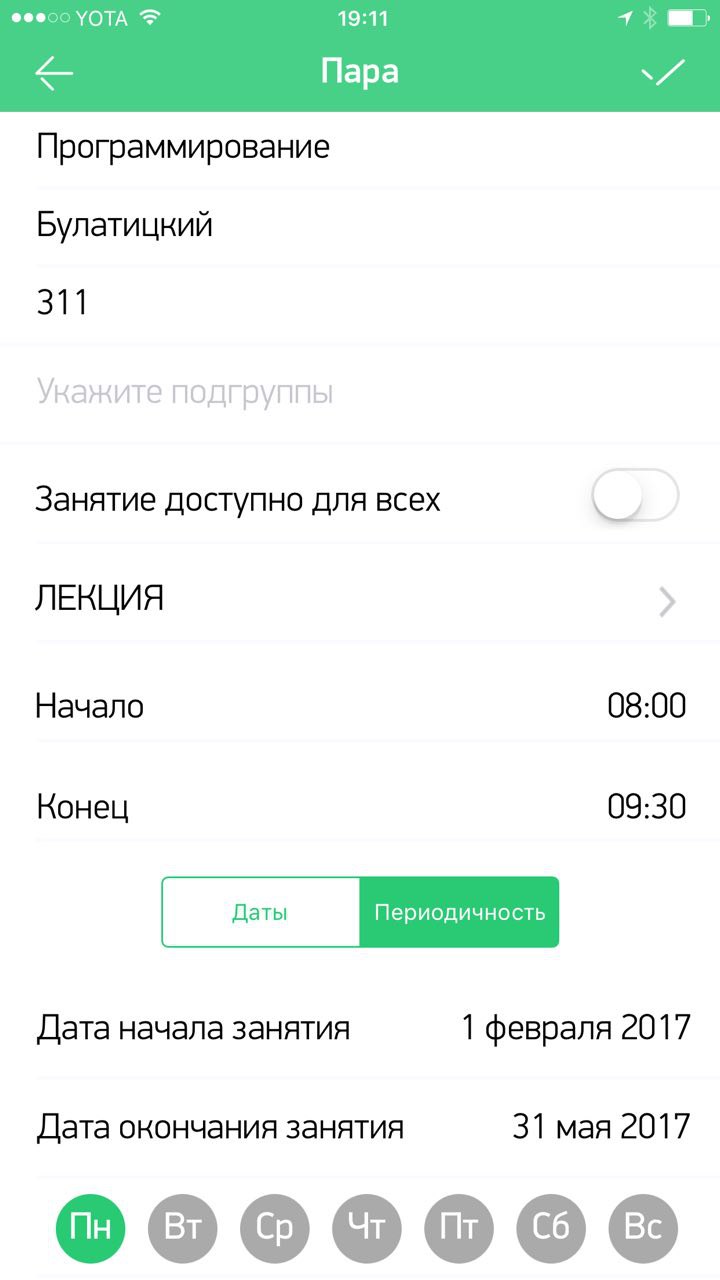


Рис. 1.3.3. Создание события в Rvuzov

## Функциональная модель разрабатываемой системы

В данном разделе описаны функциональные требования разрабатываемой системы на основе технического задания.

Для моделирования разрабатываемой системы управления расписанием была выбрана диаграмма вариантов использования [9].

Диаграмма вариантов использования определяет поведение системы с точки зрения пользователя. Проектируемая подсистема представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью вариантов использования.

Диаграмма вариантов использования мобильного приложения расписания кафедры для платформы Android представлена на рис. 1.3.



Рис. 1.3. Диаграмма вариантов использования

Неавторизованный пользователь может просматривать события в системе расписания, а также формировать поисковые запросы по интересующим критериям.

После авторизации, в зависимости от прав доступа в системе, пользователь получает определенный набор доступных ему операций.

Студент наследует весь функционал неавторизованного пользователя, а также получает возможность быстрого просмотра своего расписания. Данная опция доступна на отдельном экране в мобильном приложении.

Быстрый просмотр своего расписания подразумевает просмотр ближайших событий для авторизовавшегося пользователя. Например, если студент группы 13-ИВТ1 авторизуется в данном приложении, то в этом разделе будут отображаться ближайшие пары группы 13-ИВТ1.

Преподаватель также наследует весь вышеуказанный набор опций, а также получает доступ к редактированию, удалению и созданию новых событий в системе расписания.

В процессе редактирования или создания события введенная информация проверяется на актуальность, а также на возможность применения внесенных изменений. При попытке создания события в занятое время в конкретной аудитории приложение оповестит пользователя о невозможности создания такого события. Так же в системе предусмотрены и другие проверки на вводимые данные.

При выборе конкретного события преподаватель может удалить событие из системы расписания. Из мобильного приложения событие будет удалено.

## Функциональные требования

### Авторизация и роли

Пользоваться приложением могут как авторизованные, так и неавторизованные пользователи. Приложение должно поддерживать следующие роли пользователей:

* гость;
* студент;
* преподаватель.

### Возможности роли «Гость»

Неавторизованный пользователь по умолчанию имеет права доступа «Гость».

Гость имеет возможность:

* просматривать события;
* производить поиск по критериям;
* включать и выключать систему локальных уведомлений.

### Возможности роли «Студент»

После авторизации приложение автоматически определяет по учетным данным в системе права доступа. В случаи если пользователь «Студент», то он имеет возможность:

* просматривать события;
* производить поиск по критериям;
* включать и выключать систему локальных уведомлений;
* быстрого просмотра персонального расписания.

### Возможности роли «Преподаватель»

После авторизации приложение автоматически определяет по учетным данным в системе права доступа. В случаи если пользователь «Преподаватель», то он имеет возможность:

* просматривать события;
* производить поиск по критериям;
* включать и выключать систему локальных уведомлений;
* быстрого просмотра персонального расписания;
* создания нового события;
* редактирования имеющихся событий;
* удаления событий.

## Выводы

На основании приведенного исследования, были сделаны выводы о необходимости написания мобильного приложения, т.к. программы аналоги не адаптированы для университета или же не поддерживают необходимый функционал.

**Целью** данной дипломной работы является автоматизация предоставления информации о расписании, путем разработки мобильного приложения для платформы Android.

Для достижения этой цели были поставлены следующие **задачи**:

1. Рассмотреть предметную область.
2. Рассмотреть программы аналоги, выявить их сильные и слабые

стороны.

1. Спроектировать будущее приложение согласно полученным

данным.

1. Разработать мобильное приложение.
2. Протестировать полученное мобильное приложение и исправить найденные ошибки.

**Объектом** исследования является работа системы расписания.

**Предметом** исследования в работе является основные аспекты в работе расписания.

# ЭКОНОМИЧЕСКИЙ АНАЛИЗ

Перед началом разработки любой программной системы необходимо выяснить, насколько целесообразна ее разработка, как с точки зрения полезности, так и с точки зрения экономической эффективности. В технико-экономической части работы рассматриваются вопросы организации работ по созданию и внедрению программной системы, а также приводится расчёт ее себестоимости.

## Организационная структура проекта

Организационная структура проекта (OBS) приведена на «рис. 2.1»



***Рис. 2.1. Организационная структура проекта***

## Календарный план проекта

Для оценки расходов на реализацию проекта в числе прочих необходимо определить временные затраты на его реализацию. Для определения временных затрат проекта необходимо разработать календарный план проекта. Был сформирован состав работ проекта, определена их длительность, а также распределение ресурсов по ним. При разработке календарного плана были учтены ограничения и допущения, накладываемые на проект заказчиком.

Для реализации проекта необходимо выполнить следующие работы:

1. Сбор требований Заказчика к разрабатываемому ПО.
2. Разработка и согласование технического задания.
3. Разработка дизайна.
4. Разработка ПО.
5. Тестирование ПО.

Распределение человеческих ресурсов по работам проекта и степень их загрузки приведены в табл. 2.1.

**Таблица 2.1**

***Структура общего времени на создание программного продукта***

| **№**  **этапа** | **Этап работ** | **Ответственные исполнители (занятость на этапе)** | **Длительность, дней** |
| --- | --- | --- | --- |
| 1 | Сбор требований Заказчика к разрабатываемому ПО | * Консультант (постановщик задач) [100%] * Руководитель проекта [40%] | 2 |
| 2 | Разработка и согласование технического задания | * Консультант (постановщик задач) [100%] * Руководитель проекта [40%] * Разработчик [40%] | 4 |
| 3 | Разработка дизайна | * Разработчик [20%] * Дизайнер [100%] * Консультант (постановщик задач) [20%] | 3 |
| 4 | Разработка ПО | * Руководитель проекта [40%] * Разработчик [100%] | 14 |
| 5 | Тестирование ПО | * Руководитель проекта [20%] * Тестировщик [100%] | 3 |

При реализации данного проекта работы выполняются последовательно. Диаграмма Ганта приведена на «рис. 2.2.1» и «рис. 2.2.2»



***Рис. 2.2.1. Табличное представление Диаграммы Ганта***



***Рис. 2.2.2. Графическое представление Диаграммы Ганта***

Исходя из длительности работ и коэффициента загрузки членов проектной команды, определим их трудозатраты при реализации проекта табл. 2.2.

**Таблица 2.2**

***Трудозатраты членов проектной команды***

|  |  |  |
| --- | --- | --- |
| **№** | **Исполнитель** | **Трудозатраты, человеко-часов** |
| 1 | Руководитель проекта | 69 |
| 2 | Консультант (постановщик задач) | 53 |
| 3 | Разработчик | 130 |
| 4 | Тестировщик | 24 |
| 5 | Дизайнер | 24 |

* 1. **Расчёт затрат на разработку продукта**

Расчет затрат на создание и внедрение ПО включает следующие составляющие с последующим их графическим представлением в виде круговой диаграммы:

* заработная плата исполнителей работ по проекту – *ЗПосн*;
* отчисления на социальные нужды (страховые взносы) – *Нзп*;
* арендные платежи за производственные (офисные) помещения – *Апм*;
* амортизация используемых основных средств и нематериальных активов – *А*;
* расходы на модернизацию и приобретение основных средств – *Рмод*;
* расходы на приобретение необходимого ПО – *РПО*;
* расходы на интернет, связь – *Ртел*;
* расходы на канцелярские товары и расходные материалы – *Рр.м.*;
* прочие расходы – *Пр.р.*.

### Расчёт заработной платы исполнителей работ по созданию программного продукта

Основная ЗП определяется по формуле:

|  |  |
| --- | --- |
|  | (2.1) |

где *M* – месячная зарплата (*руб.*), *T* – общие трудозатраты (*чел.-ч*), *Чр* – число рабочих дней в месяц, *tр.д. –* продолжительность рабочего дня в часах, *П* – процент премии. В данной работе *Чр =* 21 *день*, *tр.д.=* 8*ч*, *П* = 0.

Значение месячной заработной платы (М), суммарные трудозатраты членов, а также рассчитанная по формуле 2.1 основная заработная плата проектной команды приведены в табл. 2.3.

**Таблица 2.3**

***Основная заработная плата членов проектной команды***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Исполнитель** | **Месячная заработная плата (М), руб.** | **Трудозатраты, человеко-часов** | **, руб.** |
| 1 | Руководитель проекта | 50 000 | 69 | 20 535 |
| 2 | Консультант (постановщик задач) | 35 000 | 53 | 11 041 |
| 3 | Разработчик | 35 000 | 130 | 27 083 |
| 4 | Тестировщик | 20 000 | 24 | 2 857 |
| 5 | Дизайнер | 25 000 | 24 | 3 571 |

Суммарное значение основной заработной платы проектной команды на период реализации проекта составит 65 087 (руб.).

Дополнительная заработная плата берется в размере 15% от основной.

ЗПдоп=9 763 (руб.).

Общая заработная плата составит 74 850 (руб.).

Проект реализуется в небольшой ИТ-компании, где доля вспомогательного и административного персонала по отношению к основному персоналу не велика. Большая часть административного персонала задействована в проектной деятельности в качестве руководителей проекта. Кадровый учет, бухгалтерский и налоговый учет в Компании отдан на аутсорсинг. Затраты на аутсорсинг войдут в прочие расходы. В связи с этим примем заработную плату обслуживающего персонала равной 0 руб.

* + 1. ***Расчёт отчислений на социальные нужды (страховые взносы)***

Теперь можно рассчитать величину отчислений на социальные нужды (страховые взносы), которые начисляются на заработную плату и в 2017 г. для организаций, осуществляющих деятельность в области информационных технологий, составляют 14% по выплатам в пределах 75 тыс. руб. Структура отчислений на социальные нужды (страховые взносы) приведена в табл. 2.4.

**Таблица 2.4**

Структура отчислений на социальные нужды (страховые взносы)

|  |  |
| --- | --- |
| Пенсионный фонд Российской Федерации | 8,0% |
| *для лиц 1966 года рождения и старше* |  |
| страховые взносы на страховую часть трудовой пенсии | 8,0% |
| *для лиц 1967 года рождения и моложе* |  |
| страховые взносы на страховую часть трудовой пенсии | 2,0% |
| страховые взносы на накопительную часть трудовой пенсии | 6,0% |
| Фонд социального страхования Российской Федерации | 2,0% |
| Федеральный фонд обязательного медицинского страхования | 4,0% |

Таким образом, *Нзп=* 10 479(*руб.*).

* + 1. ***Арендные платежи за производственные (офисные) помещения***

Компания, реализующая проект, арендует офисные помещения в г. Брянск.

Стоимость аренды составляет 12 000 руб в месяц за 32 м2.

Арендная плата включает в себя оплату как площади занимаемых Компанией помещений, так и электроэнергии, отопления, водоснабжения, кондиционирования и уборки помещений, вывоза и утилизации технико-бытовых отходов, парковочных мест на автостоянке.

На каждого члена проектной команды приходится 4,5 м2 арендуемого офисного помещения. На период данного проекта члены проектной команды в других проектах не задействованы.

Затраты на арендные платежи рассчитываются по формуле: длительность проекта / количество рабочих дней в месяце \* стоимость аренды в месяц.

Исходя из изложенного выше, затраты на аренду помещений, отнесенные на проект составят *Апм* = 14 857 (руб.).

* + 1. ***Амортизация используемых основных средств и нематериальных активов***

При реализации проекта задействованы следующие основные средства:

* 3 персональных компьютеров в сборе первоначальной стоимостью 30 000 (руб.) каждый.
* 2 ноутбука первоначальной стоимостью 25 000 (руб.).

Срок полезного использования для задействованных в проекте основных средств определен в 3 года. Метод начисления амортизации – линейный.

Амортизационные отчисления для персонального компьютера на 1 месяц составят

30 000 / 36 = 833,33 (руб.).

Амортизационные отчисления для ноутбука на 1 месяц составят

25000 / 36 = 694,44 (руб.).

Амортизационные отчисления по ОС, относящиеся на проект составят:

*(руб.).*

В качестве ОС используется Windows 10, которая была предустановлена на три персональных компьютера и два ноутбука. Амортизационные отчисления по РБП, относящиеся на проект составят: 0 (руб.). Суммарные амортизационные отчисления составят:

А=4 783 руб.

* + 1. ***Расходы на модернизацию и приобретение основных средств***

При реализации проекта по разработке мобильного приложения расписания кафедры для платформы Android не планируется приобретение новых и модернизация существующих основных средств.

* + 1. ***Расходы на приобретение необходимого ПО***

При реализации проекта не планируется приобретение ПО.

* + 1. ***Расходы на интернет и связь***

Так как в Компании, реализующей проект не производится биллинг и тарификация телекоммуникационных услуг в разрезе сотрудников, затраты на интернет и связь войдут в прочие затраты, рассчитываемые как процент от прямых затрат.

* + 1. ***Расходы на канцелярские товары и расходные материалы***

Затраты на расходные материалы берутся по факту и составляют = 1 500 (*руб.*). К данным затратам относятся затраты на канцтовары, тонер и бумагу для принтера и т.д.

* + 1. ***Прочие расходы***

Прочие расходы составляют 30% от суммы следующих элементов структуры затрат: *ЗПосн*, *ЗПдоп*, *Нзп*, *Апм*, *А*, *Рмод*, *РПО*, *Ртел* и *Рр.м.*.

|  |  |
| --- | --- |
|  |  |

Таким образом, *Пр.р.=* 31 940 (*руб.*).

### Расчёт себестоимости программного продукта

В себестоимость программного продукта входят следующие элементы: *ЗПосн*, *ЗПдоп*, *Нзп*, *Апм*, *А*, *Рмод*, *РПО*, *Ртел*, *Рр.м.* и *Пр.р.*.

Сложив все элементы, можно определить себестоимость программного продукта и услуг по его внедрению: *Сп.п.*= 108 409 (*руб.*).

Структура себестоимости программного продукта отражена в табл. 2.5 и представлена на «рис. 2.3.10»

**Таблица 2.5**

***Структура себестоимости программного продукта***

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Элементы себестоимости** | **Сумма (руб.)** | **% в общ. сумме себестоимости** |
| 1 | Основная заработная плата исполнителя | 65087 | 60,03 |
| 2 | Дополнительная заработная плата исполнителя | 9763 | 9 |
| 3 | Отчисления на социальные нужды (страховые взносы) | 10479 | 9,66 |
| 4 | Арендные платежи за производственные (офисные) помещения | 14857 | 13,70 |
| 5 | Амортизация используемых основных средств и нематериальных активов | 4783 | 4,41 |
| 6 | Расходы на модернизацию и приобретение основных средств | – | – |
| 7 | Расходы на приобретение необходимого ПО | – | – |
| 8 | Расходы на интернет, связь | – | – |
| 9 | Расходы на канцелярские товары и расходные материалы | 1500 | 1,38 |
| 10 | Прочие расходы | 31940 | 29,46 |
| **Итого:** | | **108409** | **100** |

***Рис. 2.3.10. Структура себестоимости программного продукта***

# РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА

Данная глава содержит описание архитектуры подсистемы, описание используемых средств разработки и описания модели данных.

## Архитектура подсистемы оповещения

### Подсистема оповещения в системе учета посещаемости и успеваемости

Проектируемый программный продукт является подсистемой в Системе Учета Посещаемости. Кроме подсистемы работы с оповещениями система учета посещаемости имеет следующие подсистемы [4]:

* подсистема учета посещаемости;
* подсистема работы с аттестациями;
* подсистема работы с учебными планами;
* подсистема работы с рабочими учебными планами;
* подсистема оповещения (находится в разработке).

Подсистема для работы оповещениями связана с другими подсистемами. Для того чтобы отправить оповещение пользователю, подсистеме требуется информация из подсистем посещаемости и успеваемости.

Чтобы наглядно показать взаимодействие подсистем используем диаграмму компонентов, представленную на рис. 3.1. Диаграмма компонентов описывает особенности физического представления системы. Она позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный и исполняемый код.

Так как система учета посещаемости студентов имеет архитектуру клиент – сервер приложения – сервер базы данных, то и модуль оповещений будет иметь архитектуру клиент – сервер приложения – сервер базы данных. Клиент будет пользоваться системой через браузер.

Проектируемый программный продукт будет использовать шаблон проектирование MVC.

## Описание средств разработки

### Описание СУБД

Microsoft SQL Server – система управления реляционными базами данных (СУБД), разработанная корпорацией Microsoft [16]. Основной используемый язык запросов – Transact-SQL, создан совместно Microsoft и Sybase.

SQL Server поддерживает зеркалирование и кластеризацию баз данных. Кластер сервера SQL – это совокупность одинаково конфигурированных серверов; такая схема помогает распределить рабочую нагрузку между несколькими серверами. Все сервера имеют одно виртуальное имя, и данные распределяются по IP-адресам машин кластера в течение рабочего цикла. Также в случае отказа или сбоя на одном из серверов кластера доступен автоматический перенос нагрузки на другой сервер.

### Язык программирования

В качестве языка программирования выбран C# [15]. Он имеет следующие преимущества:

* простота;
* объектная ориентированность;
* типовая защищенность;
* большая популярность среди разработчиков;
* кроссплатформенность .Net технологии;
* разработкой и поддержкой занимается Microsoft.

### Microsoft Visual Studio

В качестве среды разработки веб-приложения была выбрана Microsoft Visual Studio [13].

|  |  |
| --- | --- |
| E:\!Учеба\трпп\ComponentDiagram2.png | Рис. 3.1. Диаграмма компонентов подсистем СУП |

Microsoft Visual Studio – среда разработки, которая позволяет создавать приложения, работающие на платформе .net. Она объединяет в себе огромное количество функций, позволяющих осуществлять разработки для Windows всех версий, Интернета, мобильных устройств и облачных технологий.

Также Visual Studio поддерживает различные системы контроля версий, например, Team Foundation Server (TFS), которая и будет использоваться в дальнейшем. Это нужно для того, чтобы была возможность откатить изменения по проекту, если эти изменения привели к поломке веб-приложения.

### Технология ASP.NET

ASP.NET – технология создания веб-приложений и веб-сервисов от компании Microsoft. Она является составной частью платформы Microsoft .NET и развитием более старой технологии Microsoft ASP [10].

Ключевые преимущества ASP.NET:

* архитектура MVC;
* расширяемость;
* тестируемость;
* современный API-интерфейс.

Под Архитектурой MVC понимается модель MVC (model-vie-controller). Она позволяет осуществить взаимодействие пользователя с приложением в соответствии с естественным циклом: пользователь предпринимает действие, в ответ на которое приложение изменяет свою модель данных и доставляет обновленное представление пользователю.

Также модель MVC незаменима в тех случаях, когда в веб-приложении требуется объединить несколько технологий, например, исполняемый код, базу данных и HTML страницы.

Расширяемость позволяет легко заменять любой компонент аналогичным благодаря тому, что внутренние компоненты являются независимыми частями.

В качестве языка программирования было решено выбрать язык С#, т.к. этот язык был специально спроектирован и разработан для применения с .NET Framework.

### Библиотека Bootstrap

Bootstrap – это набор инструментов для верстки, который содержит в себе такие инструменты как сетки, шаблоны, определения классов для шрифтов, таблицы, формы и многое другое [11].

Преимуществами данной библиотеки являются: бесплатность, скорость работы, масштабируемость, легкая настраиваемость, большое количество готовых шаблонов, широкая сфера применения.

Библиотека проста в использовании и требует лишь базовых знаний HTML и CSS для того, чтобы начать работу с ней. Благодаря множеству готовых элементов верстка с Bootstrap занимает значительно меньше времени. Редактирование стилей производится путем создания новых css-правил, которые исполняются вместо стандартных. Добавление новых элементов не нарушает общую структуру. Можно выбрать фиксированный или резиновый шаблон документа. Также Bootstrap совместим со многими браузерами.

Шаблоны в Bootstrap позволяют настраивать стандартные элементы под свои потребности. Подключить шаблон легко – нужно только добавить вызов CSS необходимого шаблона.

Еще одним преимуществом является возможность работы с изображениями и видео.

## Модель данных программы

Проектные модели предметной области строятся с целью выявления взаимосвязей между основными элементами разрабатываемой системы. Они построены на основе объектно-ориентированного подхода к проектированию. Программное обеспечение имеет модульную структуру, и этот подход позволяет оптимальным образом описать ее. Поэтому именно объектно-ориентированный подход был выбран в качестве методологии проектного моделирования предметной области.

Диаграмма классов позволяет создавать логическое представление системы, на основе которого создается исходный код описанных классов.

На рис. 3.2. рассматриваем диаграмму классов нашей системы, в ней представлены следующие основные классы интерфейса:

* Student – студент. Эта сущность содержит в себе фамилию, имя, отчество студента, а также группу, в которой он обучается;
* TypeEvent – тип события. Эта сущность является вспомогательной и содержит типа события;
* PeriodCity – периодичность оповещений. Эта сущность включает в себя название периода и период времени, показывающий как часто пользователь хочет получать оповещения;
* MethodNtification – тип оповещения. Эта сущность является вспомогательной и содержит имя типа оповещения;
* Observation – наблюдение. Эта сущность содержит в себе тип события, периодичность оповещений, тип оповещений и студента на которого подписался пользователь.

Сущность Group уже имеется в базе данных и обозначают группу.

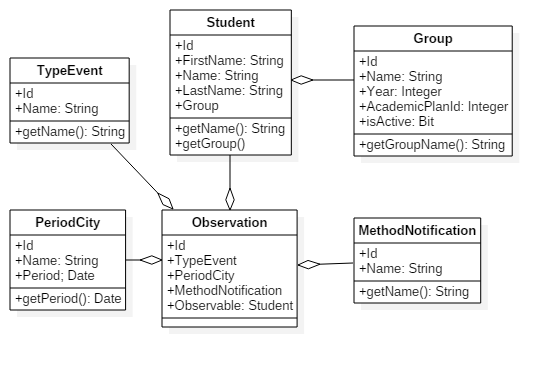


Рис. 3.2. Диаграмма классов для модуля оповещений

Диаграмма объектов подсистемы оповещения в системе учета посещаемости представлена на рис. 3.2. Ключевым объектом является наблюдение, оно связано с пользователем, который осуществляет наблюдение. В свою очередь наблюдение связано с обозреваемым объектом, типами подписки, периодичностью оповещений, а также способом оповещения. У одного пользователя может быть несколько наблюдений, что изображено на диаграмме.

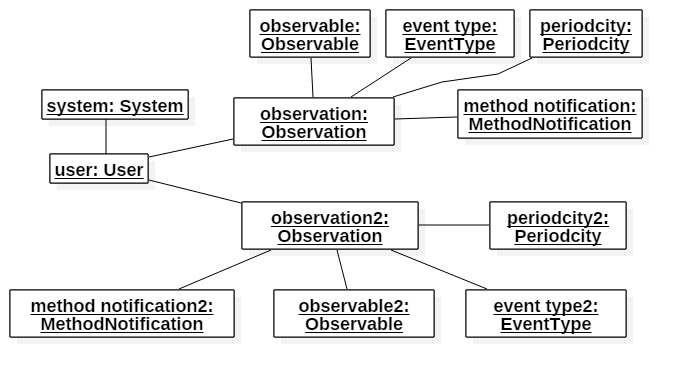


Рис. 3.2. Диаграмма объектов подсистемы оповещения

Диаграмма состояний подсистемы оповещения в системе учета посещаемости представлена на рис. 3.3.

При использовании системы основным состоянием является отображение списка наблюдений. При нажатии кнопки "Создать наблюдение" появляется диалоговое окно, в котором нужно заполнить параметры нового наблюдения. При выходе или сохранении нового наблюдения, пользователь возвращается к состоянию отображение списка наблюдений. Для перехода к состоянию редактирование наблюдение, пользователю необходимо выбрать редактирования. После завершения редактирования, пользователь возвращается к основному состоянию. Для удаления события, пользователь выбирает "Удаление наблюдения", и подтверждает свое действие в новом диалоговом окне.

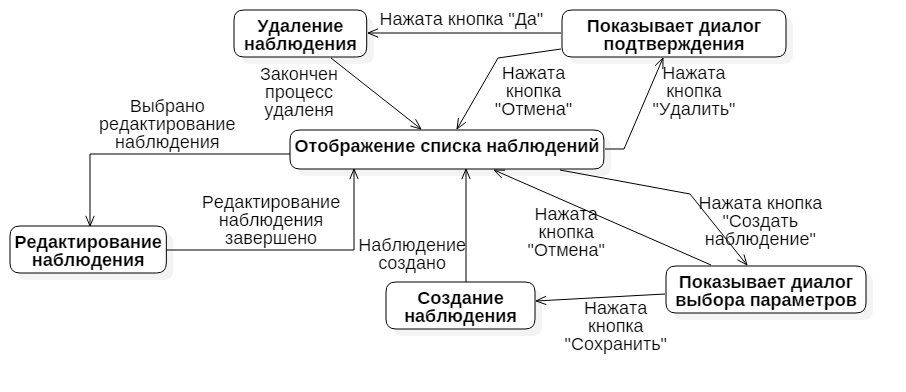


Рис. 3.3. Диаграмма состояний подсистемы оповещения

Диаграмма состояний объекта Observation представлена на рис. 3.4. Основным состоянием объекта является состояние "Активно". От состояния "Редактирование" происходит возврат к состоянию "Активно".

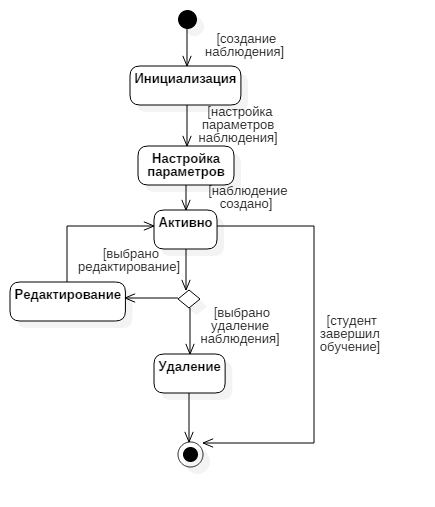


Рис. 3.4. Диаграмма состояний объекта Observation

Диаграмма состояний объекта Student представлена на рис. 3.5. Основным состоянием объекта является состояние "Обучается". После перехода к состоянию "Отчислен" или "Завершил обучение" объект Student переходит в конечное состояние.



Рис. 3.5. Диаграмма состояний объекта Student

# ЭКСЕПРЕМЕНТАЛЬНАЯ ЧАСТЬ

Целью тестирования системы является выявление ошибок в работе программы. Ошибки могут быть связаны с добавлением некорректных данных, неправильным редактированием или общим функционированием программного продукта. В данной главе описывается выбор методики тестирования и проектируются тесты.

## Выбор методики тестирования

При тестировании модуля оповещений будет применяться функциональное тестирование [12].

Функциональное тестирование — это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает ПО, какие задачи оно решает.

Функциональные требования включают в себя [14]:

* функциональная пригодность;
* точность;
* способность к взаимодействию;
* соответствие стандартам и правилам;
* защищённость.

## Эквивалентное разбиение

В данном случае определены два типа классов: допустимые классы эквивалентности, представляющие допустимые входные данные программы, и недопустимые классы эквивалентности, представляющие все остальные возможные со­стояния условий (т.е. недопустимые входные значения). Таким образом, здесь соблюден один из принципов тестирования, согласно которому необходимо фокусировать внимание на недопустимых или непредвиденных условиях (см. таблица 4.1.).

**Таблица 4.1.**

Описание классов эквивалентности

|  |  |  |
| --- | --- | --- |
| Номер класса эквивалентности | Описание | Тип (допустимый/ недопустимый) |
| 1 | Просмотр списка наблюдений, когда пользователь не авторизован. | Недопустимый |
| 2 | Добавление нового наблюдения, когда пользователь авторизован. | Допустимый |
| 3 | Редактирование наблюдения, когда пользователь авторизован. | Допустимый |
| 4 | Удаление наблюдения, когда пользователь авторизован. | Допустимый |
| 5 | Создание нового наблюдения без выбора наблюдаемого. | Недопустимый |
| 6 | Создание нового наблюдения без выбора типа оповещений. | Допустимый |
| 7 | Создание нового наблюдения без выбора способа доставки. | Допустимый |
| 8 | Создание нового наблюдения без выбора типа событий. | Допустимый |

## Анализ граничных значений

Тесты, исследующие граничные условия, приносят большую пользу, чем тесты, которые не обеспечивают этого.

Граничные условия – это ситуации, возникающие в области граничных значений входных и выходных классов эквивалентности.

В ходе анализа граничных значений было решено взять параметры наблюдения. Внимание фокусировалось не только на входных условиях, но и на пространстве получаемых результатов.

В таблице 4.2 представлены граничные значения для нашей задачи.

**Таблица 4.2.**

Описание вариантов граничных вариантов

|  |  |
| --- | --- |
| Номер варианта | Описание |
| 1 | Все параметры не заполнены. Создание и редактирование не происходит. |
| 2 | Заполнен только наблюдаемый. Создание, редактирование и удаление происходит. |
| 3 | Заполнен наблюдаемый и еще какой-либо параметр. Создание, редактирование и удаление происходит. |
| 4 | Все параметры заполнены. Создание, редактирование и удаление происходит. |

## Проектирование тестов

Для работы с подсистемой оповещения необходимо быть авторизованным, в одной из следующих ролей: администратор, преподаватель, студент или наблюдатель.

### Тестирование модуля просмотра наблюдения

Пользователь входит в личный кабинет. При входе в личный кабинет, система выводит список наблюдений пользователя, в случае первого входа он будет пустым.

Ожидаемый результат: вывод списка наблюдений текущего пользователя.

### Тестирование модуля создания наблюдения

При тестировании создания наблюдения необходимо зайти в личный кабинет. Система должна будет отобразить список текущих наблюдений. Далее пользователь нажимает на кнопку "Создать наблюдение". Система должна отобразить новое диалоговое окно. Пользователь заполняет данные для нового наблюдения и сохраняет его, при это система проверяет заполнены ли все необходимые данные и заносит данные в базу данных. Пользователь возвращается к списку наблюдений, система выводит обновленный список наблюдений.

Ожидаемый результат: наблюдение будет создано корректно, с заданными параметрами и появится в списке наблюдений.

### Тестирование модуля редактирования наблюдения

Для того, чтобы протестировать данный модуль необходимо создать наблюдение. После создания наблюдения, пользователь выбирает "Редактировать наблюдение". Система выводит диалоговое окно, как при создании наблюдения, с параметрами редактируемого наблюдения. Пользователь изменяет необходимые ему параметры, и нажимает кнопку "Сохранить". Система обновляет наблюдение в базе данных. Пользователь возвращается к списку наблюдений.

Ожидаемый результат: наблюдение будет сохранено с измененными параметрами.

### Тестирование модуля удаления наблюдения

Для того, чтобы протестировать модуль удаления необходимо чтобы было создано наблюдение. Пользователь выбирает "Удалить наблюдение". Система выводит диалоговое окно, о подтверждении удаления наблюдения. Пользователь выбирает и нажимает кнопку "Удалить". Система удаляет наблюдение из базы данных. Пользователь возвращается к обновленному списку наблюдений, в котором нет удаленного ранее наблюдения.

Ожидаемый результат: наблюдение будет удалено корректно и не будет отображаться в списке наблюдений.

### Тестирование отправки уведомлений

Для отправки уведомлений необходимо созданное наблюдение и событие, которое является индикаторным, т.е. после такого события будет произведена рассылка наблюдателям. Для тестирования отправки уведомлений, администратор должен послужить источником генерации индикаторного события, например, подписать ведомость посещаемости. Затем система произведет рассылку пользователям, которые подписаны на данное событие и определенного студента. Пользователю придет уведомление о наблюдаемом студенте, по указанному каналу связи.

Ожидаемый результат: уведомления будут отправляться при возникновении интересующих событий, а уведомления от которых пользователь отписался отправляться не будут.

### Тестирование отправки уведомлений по указанным каналам связи

Для тестирования данного модуля, необходимо созданное наблюдение. После возникновения индикаторного события в системе, пользователю будет отправлено уведомление по заданному ранее каналу связи.

Ожидаемый результат: уведомления будут отправляться через заданный канал связи.

# ОРГАНИЗАЦИОННАЯ ЧАТЬ

# ЗАКЛЮЧЕНИЕ

В рамках курсового проекта был проведен анализ предметной области, рассмотрен процесс работы подсистемы оповещения в "Системе Учета Посещаемости".

В результате выполнения курсового проекта спроектирована подсистема оповещения в системе учета успеваемости студентов и посещаемости занятий "СУП", позволяющая оптимизировать работу администратора системы.

Для проведения тестирования спроектированы необходимые тесты, которые помогут при приеме и контроле качества конечного продукта.

При необходимости подсистеме можно добавить дополнительную функциональность.

# СПИСОК ЛИТЕРАТУТЫ

1. Булатицкий Д.И. Принципы построения подсистемы оповещения в системе мониторинга успеваемости студентов и посещаемости занятий "СУП" / Д.И. Булатицкий, Е.В. Николаенко // МНПК «Перспективы развития науки и образования» – 2017. № XV (15), март. с. 97-100. Режим доступа: http://co-nf.ru/wp-content/uploads/2017/04/Sbornik\_31.03.2017.pdf
2. Булатицкий Д.И. Разработка системы мониторинга посещаемости занятий и успеваемости студентов / З.В. Борисов, Д.И. Булатицкий // Научные перспективы XXI века. Достижения и перспективы нового столетия. – 2015. № X (17), ноябрь. с. 8-11. Режим доступа: http://edu-science.ru/wp-content/uploads/2016/03/edu-17\_p1\_6-158.pdf
3. Борисов, З.В. Анализ и визуализация данных в системе учёта посещаемости и успеваемости студентов / З.В. Борисов // Материалы 70-ой студенческой научно-практической конференции. – Брянск: БГТУ, 2016. с. 314-315. Режим доступа – http://mn.tu-bryansk.ru/files/Sbornik\_70-oy\_konferentsii\_Gotova.pdf
4. Булатицкий Д.И. Применение технологий OLAP для анализа и визуализации данных в системе учёта посещаемости и успеваемости студентов / З.В. Борисов, Д.И. Булатицкий // III Международная студенческая конференция «Инновационные направления разработки и использования информационных систем и технологий» – Брянск: БГАУ, 2016.
5. Система учета успеваемости студентов UJOP – Режим доступа: http://www.podebrady.ru/2013/11/27/studijni-vysledky.
6. Информационный портал ИАСУ «Электронный университет» – Режим доступа: http://eun.bmstu.ru/products/portal.
7. Сайт «Почты России» – Режим доступа: https://www.pochta.ru.
8. Сайт интернет-магазина «Читай город» – Режим доступа: http://www.chitai-gorod.ru.
9. Орлов С.А. Технологии разработки программного обеспечения / С.А. Орлов, Б.Я. Цилькер // М: ПИТЕР, 2016г – 608с. [9]
10. Фримен А. ASP.NET MVC 4 с примерами на C# 5.0 для профессионалов /А. Фримен, С. Сандерсон. М.: ВИЛЬЯМС, 2013. – 688с.
11. Сайт по обсуждению возможностей библиотеки Bootstrap – Режим доступа: http://mybootstrap.ru.
12. Блэк Р. Ключевые процессы тестирования. Планирование, подготовка, проведение, совершенствование /Р. Блэк. М.: ЛОРИ, 2011. – 565с.
13. MSDN документация по Microsoft Visual Studio 2015 – Режим доступа: https://msdn.microsoft.com.
14. Кайсперин Л. Гибкое тестирование: практическое руководство для тестировщиков ПО и гибких команд / Л. Кайсперин, Д. Грегори // М.: «Вильямс», 2010. — 464 с.
15. Шилдт Г. C# 4.0. Полное руководство /Г. Шилдт, М.Торгерсен, C.Вилтамут, П.Голд // М.: ВИЛЬЯМС, 2015. – 1056с.
16. Петкович Д. Microsoft SQL Server 2012. Руководство для начинающих /Д. Петкович. СП.: БВХ-Петербург, 2013. – 816с.