**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.

**КУРСОВАЯ РАБОТА**

СИСТЕМА РАСПИСАНИЯ ДЛЯ УНИВЕРСИТЕТА

Всего листов\_\_\_\_\_

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

Леквеишвили Д.М.

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

Руководитель

к.т.н., доц. Азарченков А.А.

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

Брянск 2017

**Министерство образования и науки РФ**

**Брянский государственный технический университет**

*Кафедра «Информатика и программное обеспечение»*

**Задание**

**на курсовой проект**по курсу *«Технология разработки программных продуктов»*

студенту группы 13-ИВТ1 Леквеишвили Д.М.

**ТЕМА КУРСОВОГО ПРОЕКТА:**

«СИСТЕМА РАСПИСАНИЯ ДЛЯ УНИВЕРСИТЕТА»

**Состав работы:**

1. **Введение.** Цель работы, задачи, объект и предмет исследования.
2. **Разработка и анализ требований.** Описание предметной области; обзор и анализ известных источников; анализ и формализация поставленной задачи; декомпозиция задачи на отдельные подзадачи; расчет трудоемкости.
3. **Функциональные и иные требования**.
4. **Проектная часть.** Архитектура. Выбранные средства разработки. Проектные модели.
5. **Тестирование.** Методика тестирования. Описание тестов.
6. **Заключение.** Итоги и перспективы развития выполненной работы.
7. Список **актуальной** (год издания ≥ 2012) литературы не менее 15 наименований. На все источники должны быть ссылки в тексте.

**Срок сдачи курсового проекта:** 2017г.

**Защита курсового проекта:** 2017г.

Руководитель курсового проектирования: к.т.н., доц. Азарченков А.А.

Консультант: к.т.н., доц. Панус Д.Н.

Студент группы 13-ИВТ1: Леквеишвили Д.М.

Аннотация

В данной работе согласно заданию была спроектирована система расписания для университета.

Основная часть работы включает в себя следующие разделы:

1. разработка и анализ требований – содержит описание предметной области; обзор и анализ известных источников; анализ и формализацию поставленной задачи; декомпозицию задачи на отдельные подзадачи;
2. функциональные и иные требования – содержит описание требований к разрабатываемому программному продукту;
3. проектная часть – содержит выбор и обоснование принципиальных проектных решений, методов и процедур, описание обязательных проектных процедур;
4. тестирование – содержит описание тестов, особенности процедур тестирования разработанного программного продукта на предмет соответствия функциональным требованиям, оценку результатов тестирования.

В заключении приведены краткие выводы по результатам выполненной работы и предложения по их использованию, а также указана ценность полученных решений.

Содержание

[Введение 4](#_Toc480727072)

[1. Разработка и анализ требований 5](#_Toc480727073)

[1.1. Описание и анализ исследуемой проблемы 5](#_Toc480727074)

[1.1.1. Описание текущей ситуации 5](#_Toc480727075)

[1.1.2. Описание предметной области 5](#_Toc480727076)

[1.2. Обзор аналогов 6](#_Toc480727077)

[1.2.1. Органайзеры 6](#_Toc480727078)

[1.2.2. Специализированное ПО для университетов 10](#_Toc480727079)

[1.3. Описание базовой архитектуры проектируемого программного обеспечения 16](#_Toc480727080)

[1.3.1. Модель 17](#_Toc480727081)

[1.3.2. Представление 17](#_Toc480727082)

[1.3.3. Контроллер 18](#_Toc480727083)

[1.4. Моделирование предметной области 18](#_Toc480727084)

[1.5. Оценка трудоёмкости 24](#_Toc480727085)

[2. Функциональные и иные требования 28](#_Toc480727086)

[2.1. Назначение разработки 28](#_Toc480727087)

[2.1.1. Для кого предназначена система 28](#_Toc480727088)

[2.1.2. Каким образом результаты проекта решают эти проблемы 28](#_Toc480727089)

[2.2. Требования к разрабатываемому программному продукту 28](#_Toc480727090)

[2.2.1. Авторизация 28](#_Toc480727091)

[2.2.2. Создание нового события 29](#_Toc480727092)

[2.2.3. Поиск события 30](#_Toc480727093)

[2.2.4. Просмотр события 30](#_Toc480727094)

[2.2.5. Удаление события 31](#_Toc480727095)

[2.2.6. Редактирование события 31](#_Toc480727096)

[2.3. Специальные требования 32](#_Toc480727097)

[3. Проектная часть 33](#_Toc480727098)

[3.1. Структура 33](#_Toc480727099)

[3.2. Описание средств разработки 34](#_Toc480727100)

[3.3. Макеты интерфейсов 34](#_Toc480727101)

[3.4. Проектные модели 3](#_Toc480727102)

[4. Тестирование программного продукта 6](#_Toc480727103)

[4.1. Описание тестов 6](#_Toc480727104)

[4.1.1. Модуль авторизации 6](#_Toc480727105)

[4.1.2. Модуль редактирования события 7](#_Toc480727106)

[Заключение 9](#_Toc480727107)

[Список литературы 10](#_Toc480727108)

Введение

Цель данной работы – закрепление и расширение навыков применения современных подходов к разработке программных продуктов, выбора и описания архитектуры ПП, документирования и тестирования в рамках разработки автоматизированной системы расписания для университета. Задача проектирования – анализ предметной области и проектирование программного продукта в соответствие с разработанными функциональными требованиями.

Актуальность разработки 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.

Объект исследования данной работы – работа системы расписания. Предмет исследования – основные аспекты в работе расписания.

# Разработка и анализ требований

## 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

Исходя из назначения и доступных функций можно выделить следующие категории аналогов:

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

### Органайзеры

Представленные системы можно сравнить по функционалу, но он не должен быть излишним для, поэтому будут сравниваться только схожие с поставленным проектом функции.

#### 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.

##### Сотрудничество

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.

##### Работа с расписанием

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), редактировать и удалять события. Интерфейс приложения довольно сложен и не позволяет быстро найти нужное событие по заданным критериям (Рис. 2). Фильтр по времени, участникам и аудиториям присутствует.

|  |  |
| --- | --- |
| /Users/maksimgrisutin/Desktop/2017-04-17 06.29.47.jpg | /Users/maksimgrisutin/Desktop/2017-04-17 06.29.58.jpg |
| Рис. 1. Создание события | Рис. 2. Просмотр событий |

#### Microsoft Outlook

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

Так же Microsoft Outlook является почтовым клиентом.

##### Сотрудничество

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

##### Работа с расписанием

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. 3), редактирования и удаления событий. Интерфейс в данной программе отображает больше информации (Рис. 4), чем в предыдущем аналоге.

|  |  |
| --- | --- |
| /Users/maksimgrisutin/Desktop/2017-04-17 06.55.26.jpg | /Users/maksimgrisutin/Desktop/2017-04-17 06.55.31.jpg |
| Рис. 3. Создание события в Microsoft Outlook | Рис. 4. Просмотр расписания в Microsoft Outlook |

### Специализированное ПО для университетов

Данные системы подходят для вузов значительно больше чем Органайзеры, так как разрабатывались исключительно для расписания Университетов. Аналогов очень мало, поэтому была рассмотрена одна система расписаний для университетов.

#### Rvuzov

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].

##### Сотрудничество

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$.

##### Работа с расписанием

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. 5).

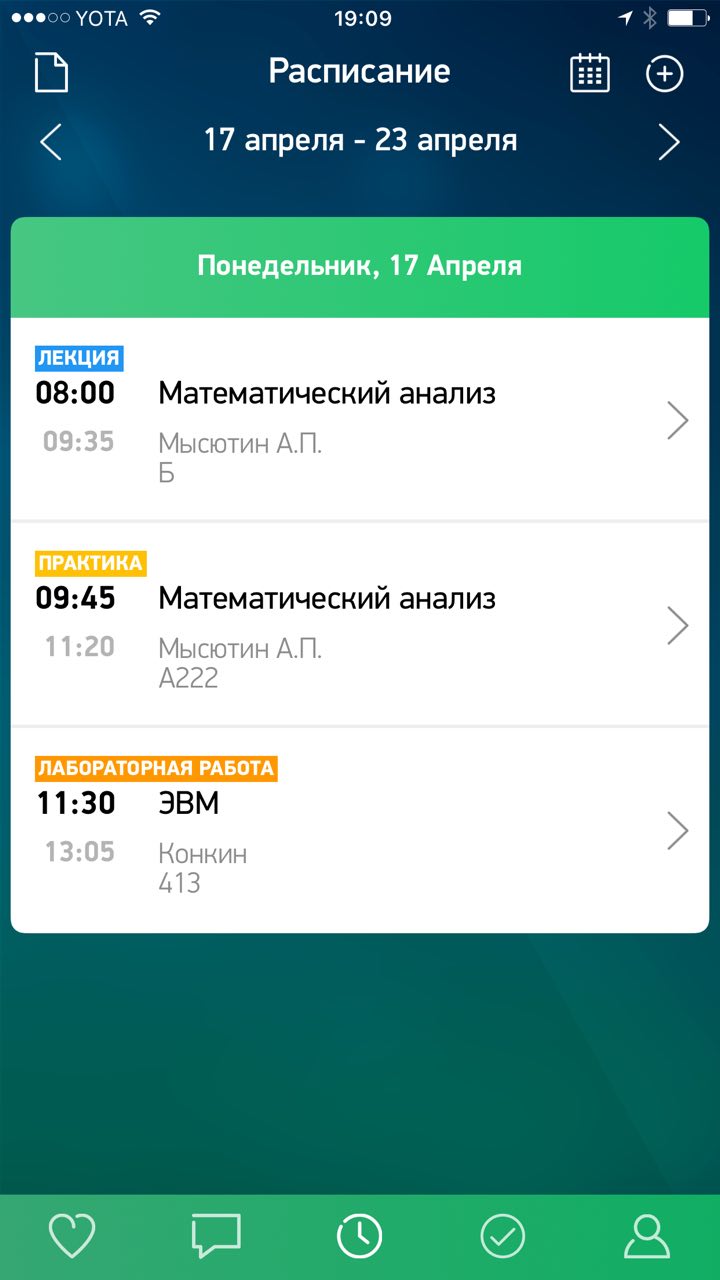


Рис. 5. Просмотр события в Rvuzov

Также присутствует расширенный просмотр события (Рис. 6) с возможностью просмотра детальной информации, такой как время события, имя преподавателя, аудитория.

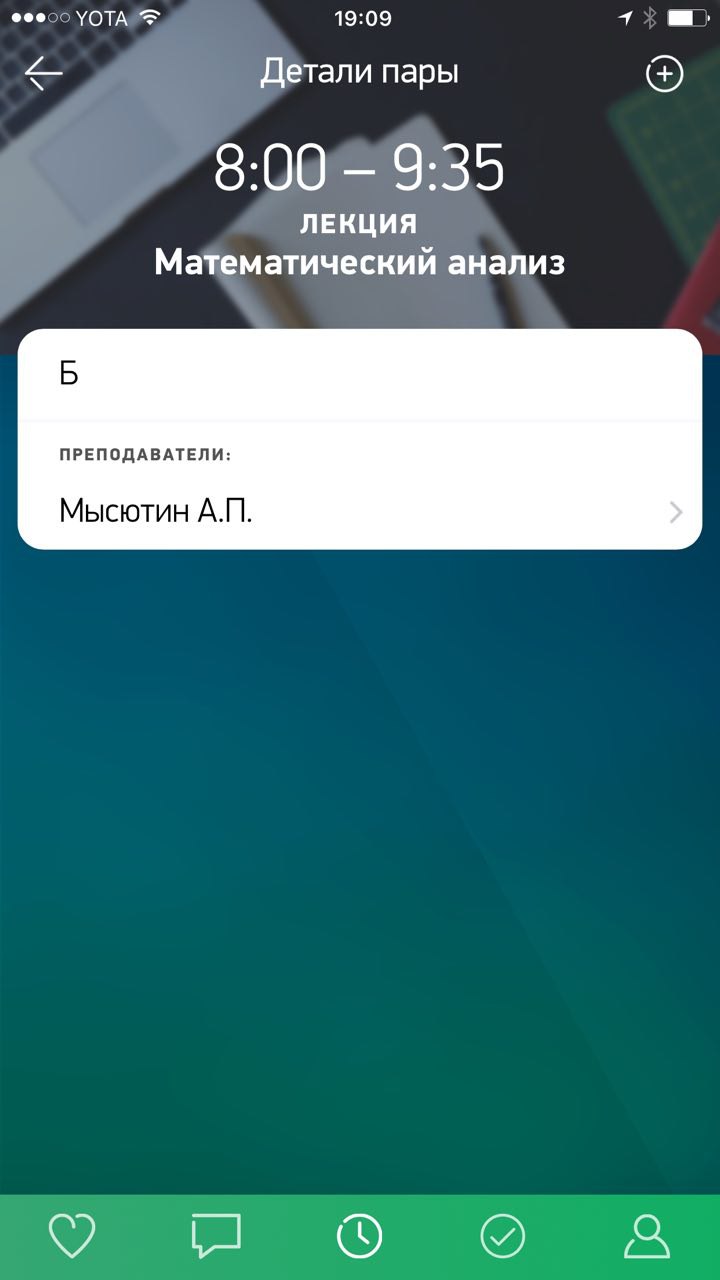


Рис. 6. Расширенный просмотр события в Rvuzov

Создание событий с возможностью создания повторяющихся по четным и нечетным неделям, позволяет добавлять в расписание пары (Рис. 7).

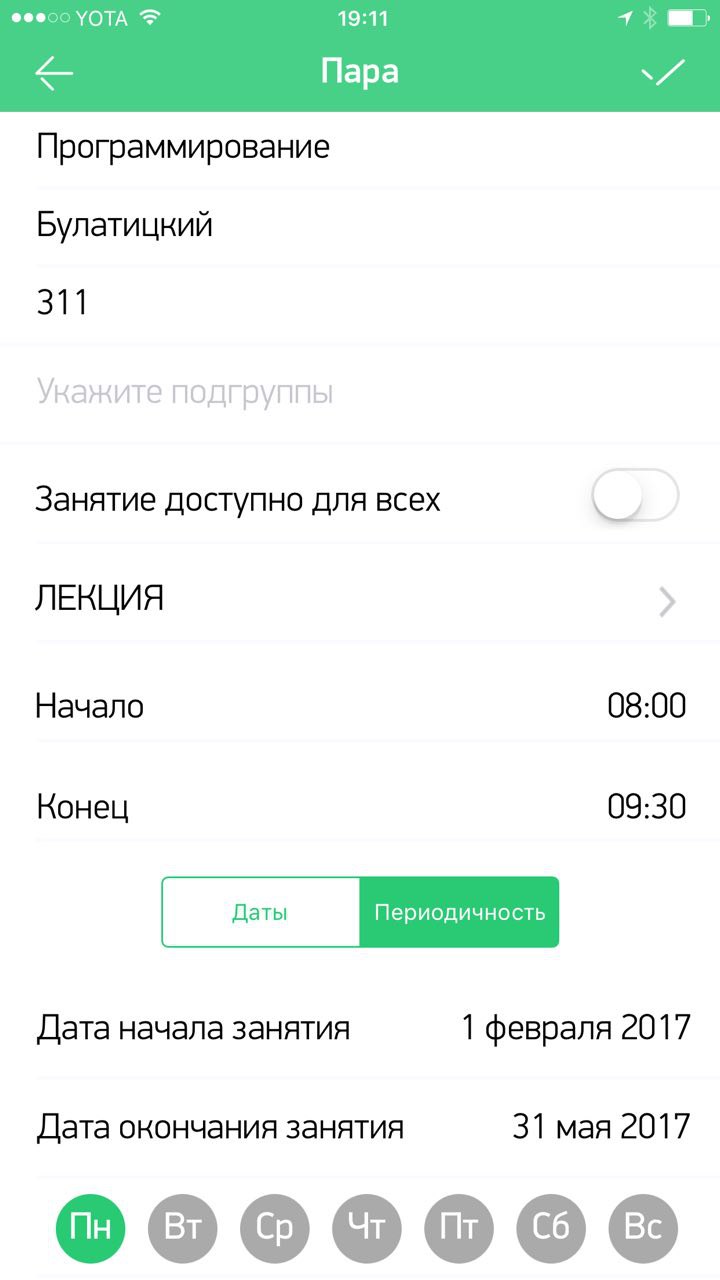


Рис. 7. Создание события в Rvuzov

#### Выводы

##### Сравнительная таблица

В табл. 1 представлена сводная таблица с результатами анализа систем расписаний.

Таблица 1

Характеристики систем расписаний

| Характеристика | Google Календарь | Microsoft Outlook | Rvuzov |
| --- | --- | --- | --- |
| Сотрудничество | не сотрудничает с учебными заведениями, предлагает платную подписку. | полностью бесплатная система без ограничений. | сотрудничает с учебными заведениями. |
| Интеграция БД расписания | не поддерживает интеграцию БД | не поддерживает интеграцию БД | поддерживает интеграцию усилиями разработчиков |
| Возможность создания, редактирования, удаления событий | присутствует, но с ограничениями по установке аудитории | присутствует | присутствует, плюс возможность создавать вне учебного события старостам. |
| Задание ролей | невозможность создать роли (кроме роли владельца) | Можно самому задать нужные роли | заранее заданные все нужные для университета роли, с возможностью добавит нужные |

##### Общие выводы

Рассмотренные системы схожи по функциям, достоинствам и недостаткам, поэтому в обзоре внимание концентрируется на отличительных особенностях аналогов и реализации целей разрабатываемой системы. Из всех рассмотренных аналогов, Rvuzov больше всех соответствует требованиям, предъявляемым к разрабатываемой системе.

Интерфейс в данной программе отображает больше информации, но в тоже время, и не перегружен, что выгодно выделяет приложение от других аналогов. Данная система имеет предустановленное разделение на нужные роли (студент, преподаватель). Так же во всех системах есть большой недостаток – невозможность использовать систему без авторизации, что говорит о том, что готовые решения не подходят для университета.

**Достоинства**

* реализованы дополнительные функции, такие как: комментарии, задачи, звонки;
* разбиение пользователе на роли достаточно гибкое;
* взаимодействие с пользователем оптимизировано, а система отлажена, т.к. платформы функционируют уже несколько лет и имеют по нескольку сотен тысяч пользователей.

**Недостатки**

* не учитываются особенность конкретного университета:
  + пользователи не привязаны к группам;
  + пользователи не привязаны к реальным идентификаторам;
* нет возможности использовать систему без авторизации;
* оплата студентами, использовавшими систему, ежемесячной подписки.

## Описание базовой архитектуры проектируемого программного обеспечения

На (рисРис. 8) изображена схема архитектуры разрабатываемой системы.



Рис. 8. Архитектура системы

У многих студентов и преподавателей есть устройства на системе Android, способные подключаться к интернету и запускать простые приложения. Поэтому, было решено писать мобильное приложение под платформу Android. Для реализации данного приложения предварительно выбрана MVC архитектура. Слово MVC – бэкроним для Model, View, Controller. Несомненным плюсом MVC является единая глобальная архитектура приложения. Даже в сложных системах, разработчики (как те, которые разрабатывали систему, так и вновь присоединившиеся) могут легко ориентироваться в программных блоках. Например, если возникла ошибка в логике обработки данных, разработчик сразу отбрасывает 2-блока программы (controller и view) и занимается исследованием 3-го (model). [2]

Архитектура MVC вводит следующие уровни:

### Модель

Предоставляет (представлению и контроллеру):

* + Данные;
  + Методы работы с данными (далее — опционально, в зависимости от реализации):
    - Запросы к базам данных;
    - Валидация данных (проверка на корректность);
    - Бизнес-логика (в случае «активной» модели).
* Нуждается в следующем:
  + В представлении (т. к. не может самостоятельно демонстрировать данные и результаты их обработки);
  + В контроллере (т. к. не имеет точек взаимодействия с пользователем).
* Может иметь множество различных представлений и контроллеров;
* Отвечает на запросы изменением своего состояния. При этом, в модель может быть встроено автоматическое оповещение «наблюдателей» (обычно через «аппарат событий»).

### Представление

* Отвечает за получение необходимых данных от модели и отправку их пользователю;
* Не обрабатывает введённые данные пользователя;
* Может влиять на состояние модели, через отправку ей сообщений.

### Контроллер

* Обеспечивает «связь» между пользователем и системой; контролирует и направляет:
  + Данные от пользователя к системе;
  + Реакцию системы — пользователю.
* Использует модель и представление, для необходимого действия;
* В случае «пассивной» модели — реализует бизнес-логику. [3]

## Моделирование предметной области

На Рис. 9. представлена use case диаграмма, на которой изображены пользователи и основные варианты использования разрабатываемой системы.

Из анализа требований к системе и работы с заказчиком получилось выделить основные варианты использования:

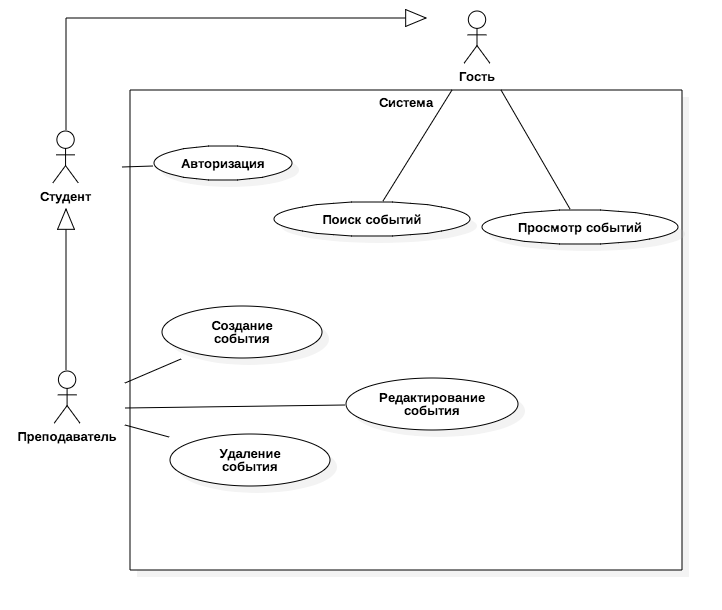


Рис. 9. Диаграмма вариантов использования

На Рис. 10 изображена диаграмма деятельности для варианта использования «Авторизация». С её помощью описаны действия пользователя, позволяющие ему авторизоваться в системе, и ответные действия системы.

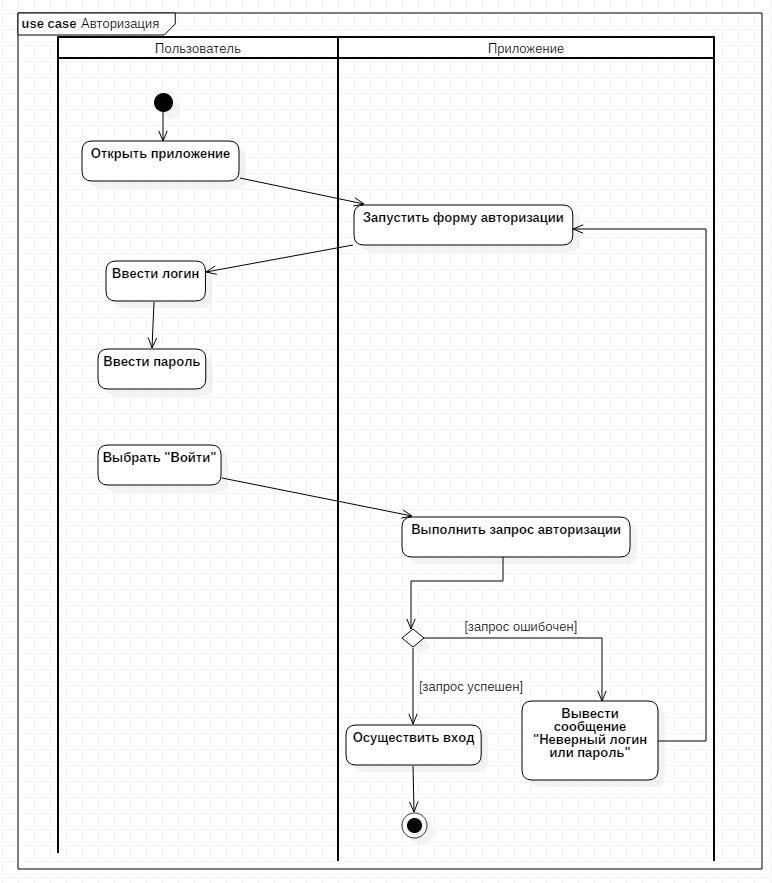


Рис. 10. Диаграмма деятельности для варианта использования   
«Авторизация»

На рис 11 изображена диаграмма последовательности для варианта использования «Авторизация».

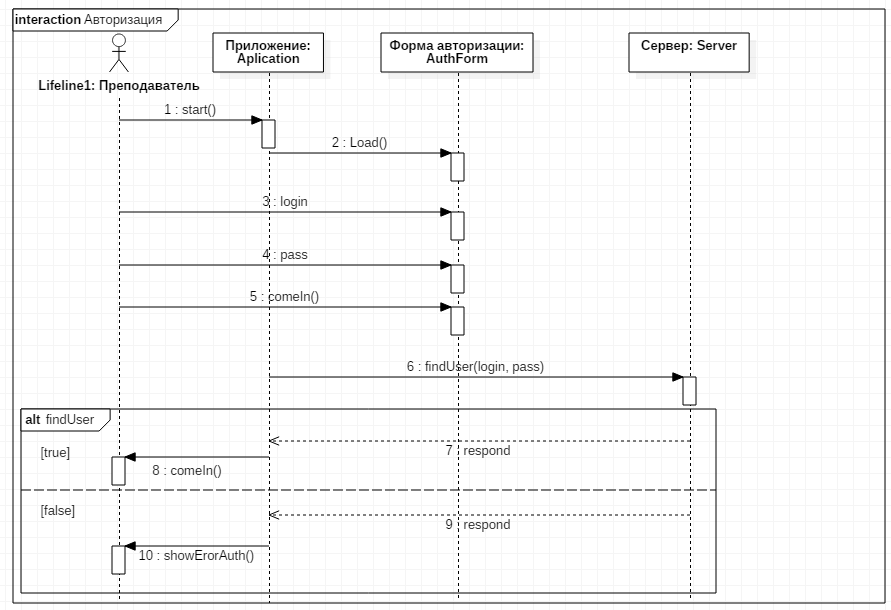


Рис. 11. . Диаграмма последовательности для варианта использования  
«Авторизация»

На рис 12 изображена диаграмма коммуникаций для варианта использования «Авторизация».

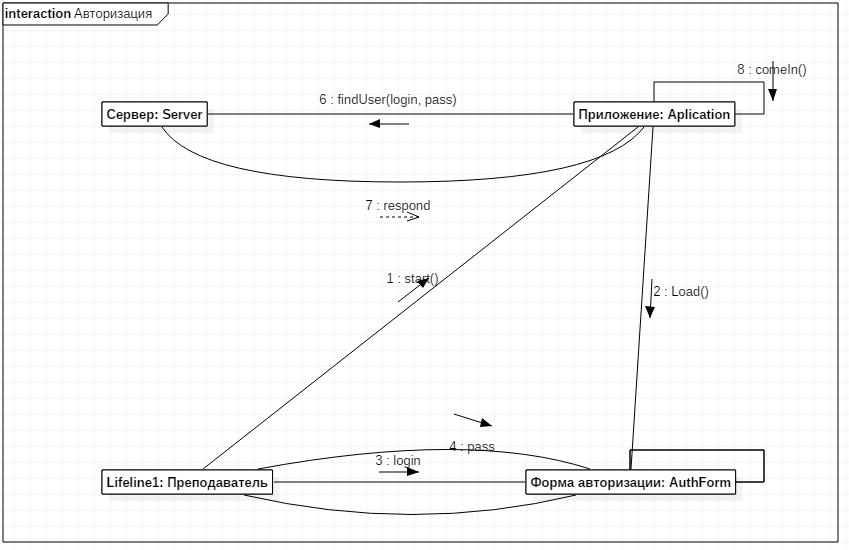


Рис. 12. Диаграмма коммуникаций для варианта использования  
«Авторизация»

На рис 13 изображена диаграмма деятельности для варианта использования «Удаление». С её помощью описаны действия пользователя, позволяющие ему удалить событие, и ответные действия системы.

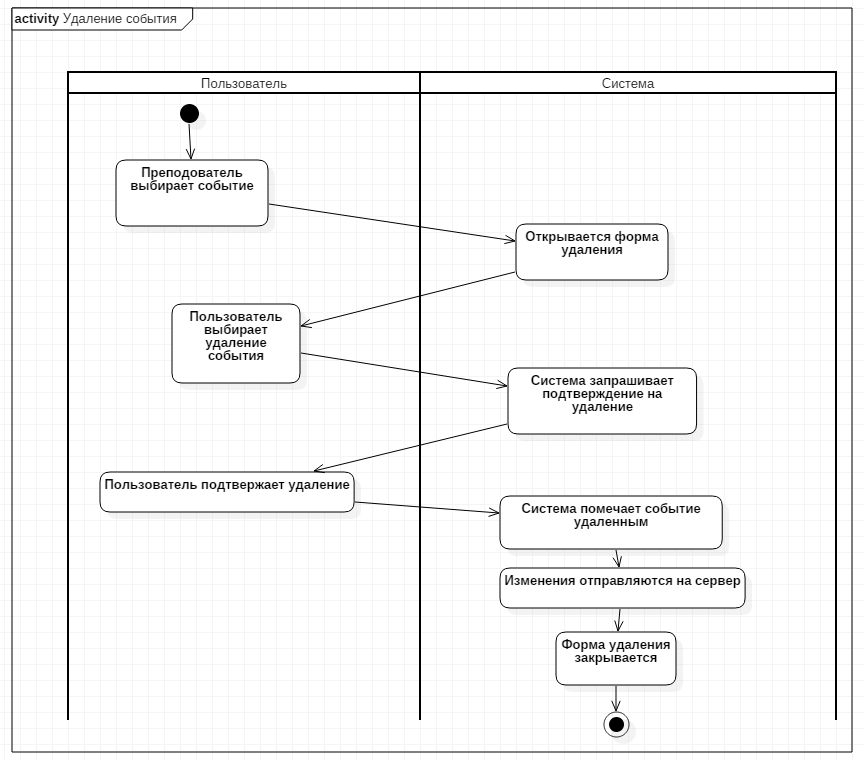


Рис. 13. Диаграмма деятельности для варианта использования   
«Удаление»

На рис 14 изображена диаграмма последовательности для варианта использования «Удаление».

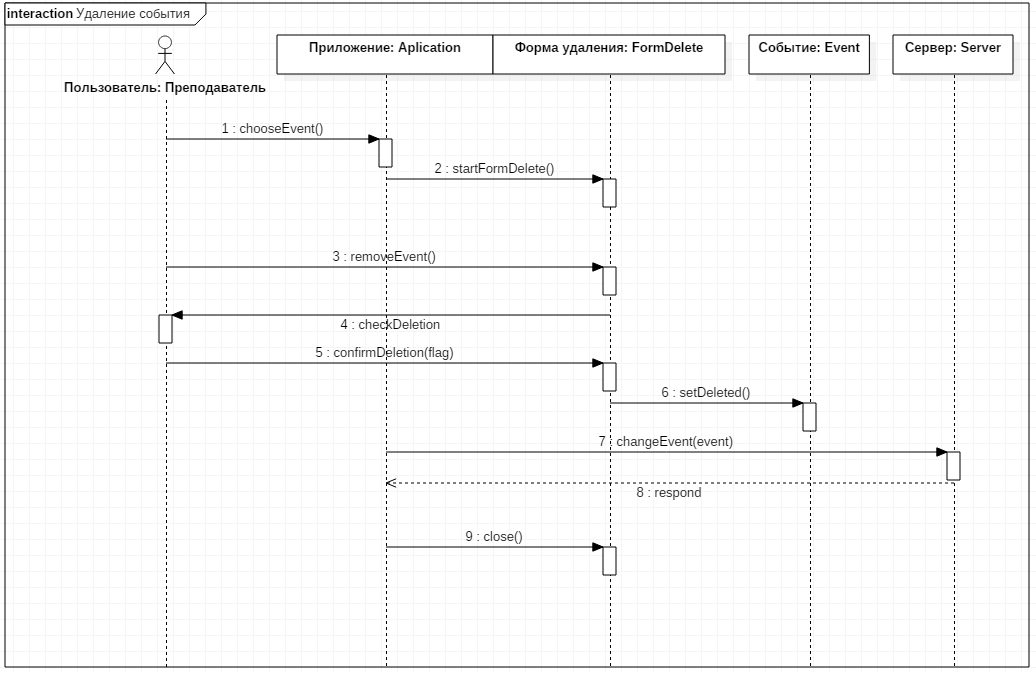


Рис. 14. . Диаграмма последовательности для варианта использования  
«Удаление»

На рис 15 изображена диаграмма коммуникаций для варианта использования «Удаление».

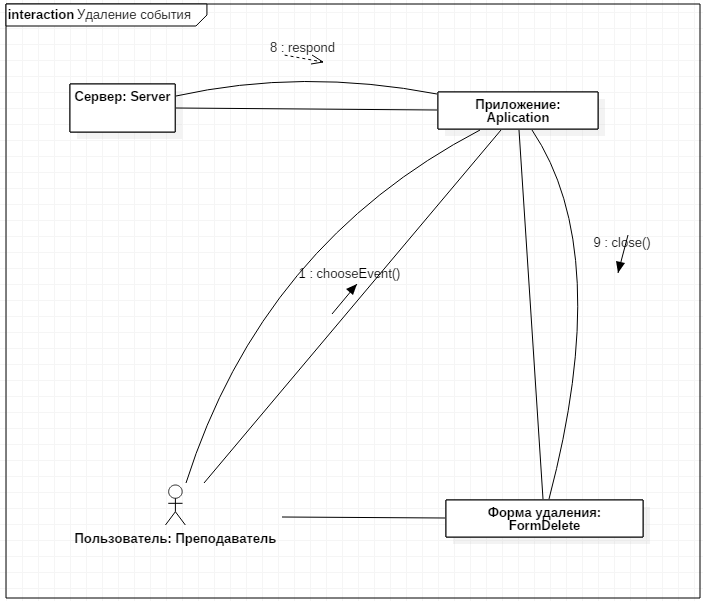


Рис. 15. Диаграмма коммуникаций для варианта использования   
«Удаление»

## Оценка трудоёмкости

Трудоёмкости разработки является одной из важных характеристик проекта, поэтому необходимо рассчитать её применительно к данному проекту. Т.к. спецификация требований определена с помощью вариантов использования, для определения оценки трудоёмкости разработки программного проекта применим подход на основе модели вариантов использования.

**Классификация вариантов использования по степени сложности**

Каждый вариант использования классифицируется по степени сложности в зависимости от числа транзакций (табл. 2). При этом учитываются как основной, так и альтернативные сценарии.

Таблица 2

Сложность и коэффициенты вариантов использования

| Тип варианта использования | Описание | Коэффициент |
| --- | --- | --- |
| Простой | Не больше 3 транзакций | 5 |
| Средний | 4-7 транзакций | 10 |
| Сложный | Более 7 транзакций | 15 |

Получим следующее распределение вариантов использования (табл. 3).

Таблица 3

Сложность вариантов использования

| № п/п | Вариант использования | Количество транзакций | Тип варианта использования |
| --- | --- | --- | --- |
|  | Авторизация | 19 | Сложный |
|  | Создание нового события | 18 |
|  | Редактирование события | 13 |
|  | Удаление события | 10 |
|  | Поиск события | 9 |
|  | Просмотр события | 5 | Средний |

Количество вариантов использования по сложности:

* сложные – 5;
* средние – 1.

**Расчет суммарного числа точек нескорректированных вариантов использования**

Для каждого из классов сложности сначала вычислим произведение числа вариантов использования определенной сложности и коэффициента для этой сложности, получив таким образом суммарное число точек нескорректированных вариантов использования (UUCP):

**Расчёт коэффициента технической сложности**

Коэффициент технической сложности (TCF) отражает сложность проекта и опыт команды и используется для корректности первоначального числа UUCP. TCP вычисляется на основе факторов, заданных в табл. 4.

Таблица 4

*Технические факторы и их вес*

|  |  |  |
| --- | --- | --- |
| Фактор | Вес | Оценка |
| Распределенная система | 2 | 4 |
| Требования быстрого ответа и высокой пропускной способности | 1 | 1 |
| Эффективность оперативной работы конечного пользователя | 1 | 1 |
| Сложная внутренняя обработка | 1 | 3 |
| Требование повторного использования кода | 1 | 5 |
| Простота установки | 0.5 | 5 |
| Простота использования | 0.5 | 5 |
| Переносимость | 2 | 5 |
| Легкость изменений | 1 | 5 |
| Параллельное выполнение | 1 | 3 |
| Включение специальных средств безопасности | 1 | 1 |
| Прямой доступ к программам независимых производителей | 1 | 1 |
| Специальные средства обучения пользователя | 1 | 1 |

Факторам в зависимости от значимости даны оценки от 0 (фактор несущественен) до 5 (основной фактор). Значение TCP вычисляется по следующей формуле:

**Расчёт коэффициента фактора окружения**

Фактор окружения (EF) вычисляется с помощью табл. 5 по следующей формуле:

Таблица 5

*Факторы окружения для команды и их вес*

|  |  |  |
| --- | --- | --- |
| Фактор | Вес | Оценка |
| Знакомство с технологиями Интернета | 1.5 | 3 |
| Опыт в прикладной области | 0.5 | 1 |
| Опыт объектно-ориентированного программирования | 1 | 2 |
| Способности ведущего аналитика | 0.5 | 2 |
| Мотивация | 1 | 2 |
| Стабильные требования | 2 | 2 |
| Персонал с частичной занятостью | -1 | 3 |
| Сложный язык программирования | -1 | 2 |

**Расчёт трудоемкости**

Рассчитаем окончательное число точек вариантов использования (UCP) по формуле:

Большая часть факторов имеет значение меньше 3, поэтому на единицу UCP следует определить от 20 до 28 человеко-часов. Таким образом, получим от 7677 до 10748 человеко-часов. При условии, что в месяце 21 рабочий день, а продолжительность рабочего дня составляет 8 часов, получим, что трудоёмкость проекта составляет от 46 до 64 человеко-месяцев.

# Функциональные и иные требования

## Назначение разработки

### Для кого предназначена система

Данная система предназначена для студентов и преподавателей БГТУ, имеющих устройства на системе Android.

### Каким образом результаты проекта решают эти проблемы

Данная система могла бы повысить качество взаимодействия студентов и преподавателей. Также она помогла бы быстро получать изменения в расписании.

Автоматизированная система предоставляет справочную информацию:

* занятость той или иной аудитории в конкретный момент времени;
* расписание преподавателя на конкретный день недели или всю неделю;
* расписание для конкретной группы на конкретный день или неделю.

## Требования к разрабатываемому программному продукту

### Авторизация

**Краткое описание**

Вариант использования «Авторизация» позволяет пользователям (преподавателям, студентам) войти в систему и получить соответствующие права.

**Предусловие**

В информационной системе ВУЗа должна содержаться идентификационная информация о студентах и преподавателях.

**Описание потока событий**

1. Пользователь открывает приложение.
2. Система запускает форму авторизации.
3. Пользователь вводит логин.
4. Система проверила логин.
5. Пользователь вводит пароль.
6. Система проверила пароль.
7. Пользователь выбрал «Войти».
8. Система производит запрос авторизации.
9. Система осуществляет вход пользователя.
10. Вариант использования заканчивается.

**Альтернативные потоки событий**

Альтернативный поток 1: ввод недопустимого логина.

1. Система выводит сообщение «Некорректный логин».
2. Пользователь вводит логин (переход к пункту 4).

Альтернативный поток 2: Запрос авторизации завершился неудачей.

1. Система выводит уведомление (переход к пункту 3).

### Создание нового события

**Краткое описание**

Вариант использования «Создание нового события» позволяет преподавателям создавать новые события.

**Предусловие**

Преподаватель авторизовался.

**Описание потока событий**

1. Преподаватель выбирает создание события.
2. Система запускает форму создания события.
3. Преподаватель выбирает тип события.
4. Система проверяет все допустимые даты.
5. Система показывает допустимые даты.
6. Преподаватель выбирает дату.
7. Система проверяет все допустимые времена.
8. Система показывает допустимые времена.
9. Преподаватель выбирает время начала события.
10. Система проверяет свободные аудитории для события.
11. Система показывает свободные аудитории для события.
12. Преподаватель выбирает аудиторию.
13. Система проверяет доступных участников для события.
14. Система показывает доступных участников для события.
15. Преподаватель выбирает участников события.
16. Преподаватель создает событие.
17. Новое событие отправляется на сервер.
18. Вариант использования заканчивается.

### Поиск события

**Краткое описание**

Вариант использования «Поиск события» позволяет пользователю найти события с заданными параметрами.

**Описание потока событий**

1. Пользователь выбирает просмотр расписания.
2. Система запускает форму поиска.
3. Пользователь выбирает тип расписания для студентов.
4. Пользователь выбирает группу.
5. Пользователь выбирает преподавателя.
6. Пользователь выбирает дату.
7. Пользователь делает нажимает кнопку «Поиск».
8. Система отображает события, удовлетворяющие поисковому запросу.
9. Вариант использования заканчивается.

### Просмотр события

**Краткое описание**

Вариант использования «Просмотр события» позволяет пользователю просмотреть события.

**Предусловие**

Пользователь авторизовался или был выполнен поиск.

**Описание потока событий**

1. Пользователь выбирает просмотр расписания.
2. Система запрашивает День у сервера.
3. Система создает День с данными из сервера.
4. Система показывает День.
5. Вариант использования заканчивается.

### Удаление события

**Краткое описание**

Вариант использования «Удаление события» позволяет преподавателю удалять ранее созданным им события или события, которыми он владеет.

**Предусловие**

Преподаватель авторизовался.

**Описание потока событий**

1. Пользователь выбирает событие.
2. Система открывает форму удаления.
3. Пользователь выбирает удаление события.
4. Система удаляет событие из списка событий.
5. Вариант использования заканчивается.

### Редактирование события

**Краткое описание**

Вариант использования «Редактирование события» позволяет преподавателю редактировать ранее созданным им события или события, которыми он владеет.

**Предусловие**

Преподаватель авторизовался.

**Описание потока событий**

1. Пользователь выбирает событие.
2. Система загружает данные события с сервера.
3. Система запускает форму редактирования событий.
4. Пользователь изменяет дату.
5. Пользователь нажимает кнопку «Сохранить изменения».
6. Система проверяет измененные значения.
7. Система сохраняет изменения и применяет их к событию.
8. Вариант использования заканчивается.

**Альтернативные потоки событий**

Альтернативный поток событий 1: Преподаватель изменяет категорию.

1. Пользователь изменяет аудиторию. (переход к пункту 4).

Альтернативный поток событий 2: Преподаватель изменяет время.

1. Пользователь изменяет время. (переход к пункту 4).

Альтернативный поток событий 3: Преподаватель изменяет участников события.

1. Пользователь изменяет участников события. (переход к пункту 4).

Альтернативный поток событий 4: Система обнаружила ошибку в измененных данных.

1. Пользователь изменяет аудиторию. (переход к пункту 4).

## Специальные требования

К системе также предъявляется ряд общих требований, не связанных с действиями конкретной категории пользователей.

1. Система должна проверять корректность всех данных, вводимых пользователем (формат email-адресов, наличие недопустимых символов и т.д.).
2. Система должна использовать JWT – «json web tokens» для авторизации.[4]
3. Должно осуществляться автоматическое хеширование последнего просмотренного расписания.

# Проектная часть

## Структура

На рис. 16 изображено распределение элементов программного обеспечения по аппаратным узлам компьютерной системы. Элементы, относящиеся к серверу, распределены между соответствующими программными серверами [5].

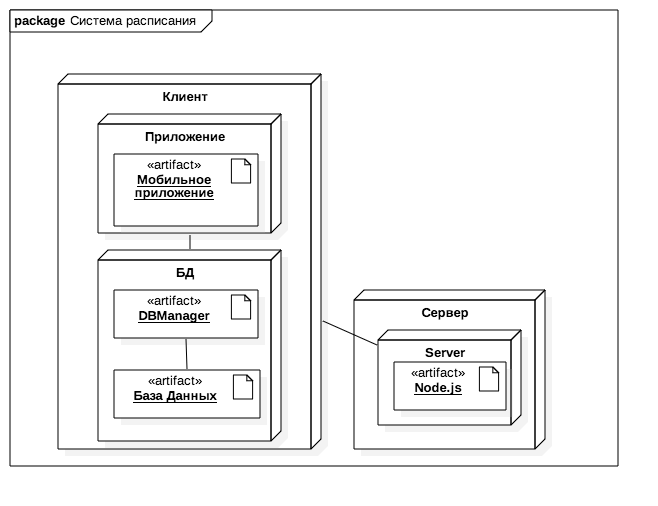


Рис. 16. Диаграмма размещения

На рис. 17 представлена диаграмма компонентов фрагмента системы. На ней изображены модули, представляющие интерфейсы доступа для просмотра событий, а так же кеширования их в базе данных.

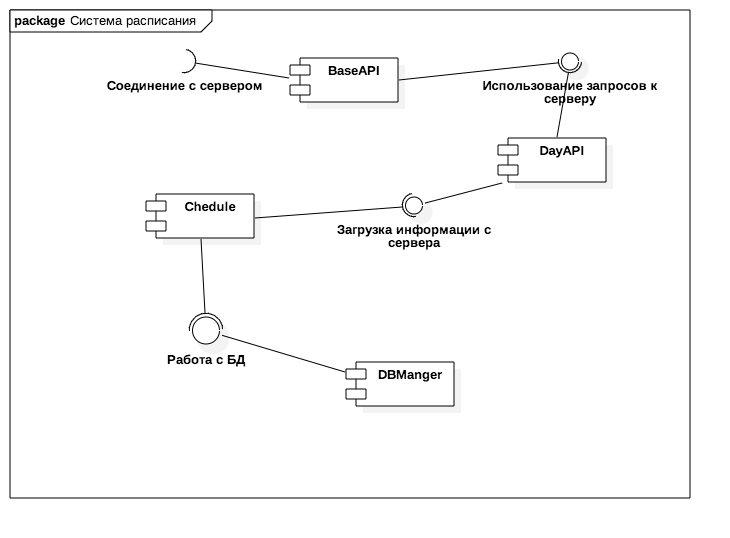


Рис. 17. Диаграмма компонентов системы

## Описание средств разработки

Для разработки системы планируется использовать язык Java и среду разработки Android Studio [6]. Android Studio доступна для бесплатного пользования. Java предоставляет необходимые возможности для программирования сервисов и содержит в себе различные библиотеки структур данных и программных компонентов [7].

## Макеты интерфейсов

Разрабатываемая система подразумевает работу с пользователями, поэтому должна предоставлять графический интерфейс для выполнения требуемых функций. Следовательно, одной из задач проектирования системы является разработка эффективного пользовательского интерфейса. [8]

Существует четыре основных критерия качества любого интерфейса:

1. скорость работы пользователей;
2. количество человеческих ошибок;
3. скорость обучения;
4. субъективное удовлетворение пользователей.

Подразумевается, что соответствие интерфейса задачам пользователя является неотъемлемым свойством интерфейса.

Рассмотрим требования к интерфейсу, выполнение которые позволяет повысить его ценность по основным критериям, и конкретные решения, часть которых продемонстрирована на макете [9].

**Скорость работы пользователей**

* длительность интеллектуальной работы – необходимо уменьшить влияние факторов, снижающих скорость принятия решений, с помощью методов:
  + непосредственное манипулирование объектами системы;
  + задержка фокуса внимания;
  + ограничение количества элементов для выбора (пункты меню, элементы списков);
* длительность физических действий – основная часть работы с системой осуществляется при помощи мыши, поэтому желательно сократить перемещения курсора и ускорить его наведение:
  + элементы управления должны иметь размеры, достаточные для того, чтобы по ним можно было быстро попасть;
  + элементы, используемые для выполнения одной последовательности действий необходимо располагать недалеко друг от друга;
* длительность реакции системы – система должна уведомлять пользователя о ходе выполнения операций, которые длятся дольше 5 с;
* удобство навигации – пункты меню и элементы списков должны образовывать логичную и очевидную для пользователя иерархию [10].

**Человеческие ошибки**

* безопасность действий – система должна блокировать потенциально опасные до получения подтверждения;
* проверка параметров – необходимо проверять корректность вводимых параметров по мере их указания;
* ограничение действий – система должна предоставлять пользователю только действия, возможные в данный момент времени при данных условиях, и никакие другие.

**Обучение работе с системой**

* понятность системы – в системе должны использоваться термины, пиктограммы и последовательности операций, характерные для предметной области и веб-приложений;
* обучающие материалы – система должна предоставлять документацию с описанием собственных возможностей и инструкциями по выполнению базовых операций. Также справочной информацией должны быть снабжены все сервисы системы.

**Субъективное удовлетворение**

* эстетика – интерфейс системы должен быть выполнен в едином стиле с использованием базовых констант и пропорций, единой цветовой гаммы;
* минимизация количества сообщений об ошибках – необходимо блокировать последовательности действий или комбинации параметров, которые могут привести к выводу сообщения об ошибке.

Исходя из вышеперечисленных требований в программе Sketch был создан действующий макет системы, позволяющий оценить эффективность работы с системой[11]. Ниже представлены макеты некоторых ключевых экранов мобильного-приложения.

На рис. 18 – 22 изображены некоторые макеты экранов системы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| /Users/maksimgrisutin/Desktop/Снимок экрана 2017-04-18 в 13.30.59.png | | /Users/maksimgrisutin/Desktop/321.png | | /Users/maksimgrisutin/Desktop/Simulator Screen Shot 18 апр. 2017 г., 13.28.27.png |
| Рис. 18. Экран авторизации пользователя | | Рис. 19. Экран создания события | | Рис. 20. Экран просмотра событий |
| /Users/maksimgrisutin/Desktop/Simulator Screen Shot 18 апр. 2017 г., 13.28.32.png  Рис. 21. Экран поиска событий | | /Users/maksimgrisutin/Desktop/Simulator Screen Shot 18 апр. 2017 г., 13.28.42.png  Рис. 22. Экран настроек пользователя | | |

## Проектные модели

На рис. 23 представлена диаграмма классов. Изображены классы, используемые при отображении расписания. Данная диаграмма позволяет генерировать заготовки классов системы, поэтому является одной из наиболее ценных диаграмм модели проекта [12].

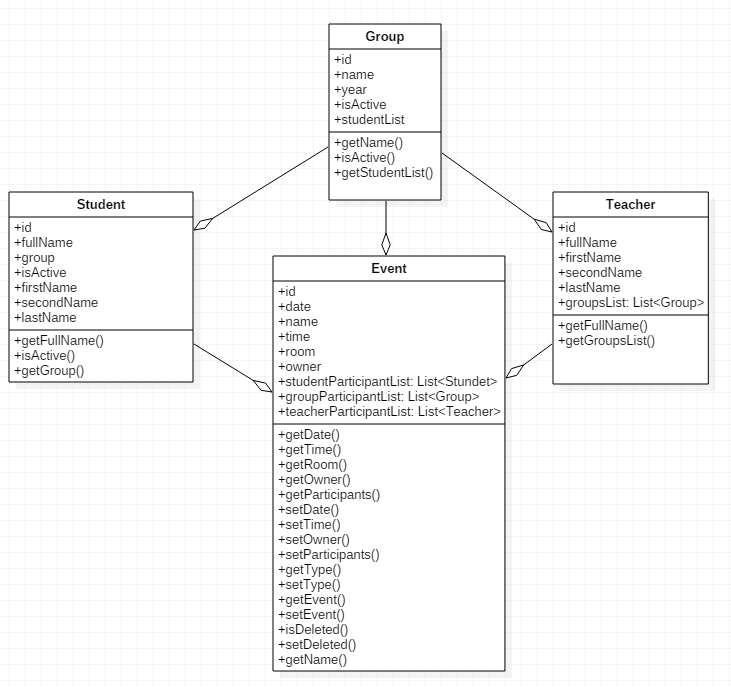


Рис.23. Диаграмма классов

На рис. 24 изображена диаграмма объектов генератора шаблона. На ней представлена возможная иерархия объектов, полученная в процессе конструирования шаблона задания. В отличие от диаграммы классов на ней изображены реальные объекты и связи между ними, а не классы [13].

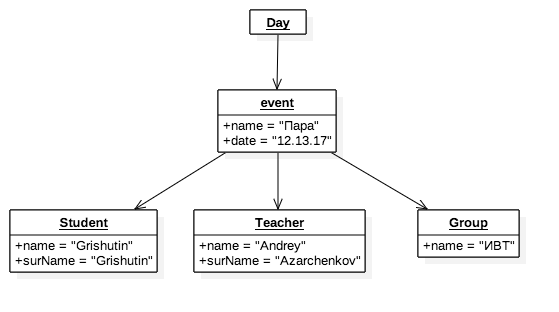
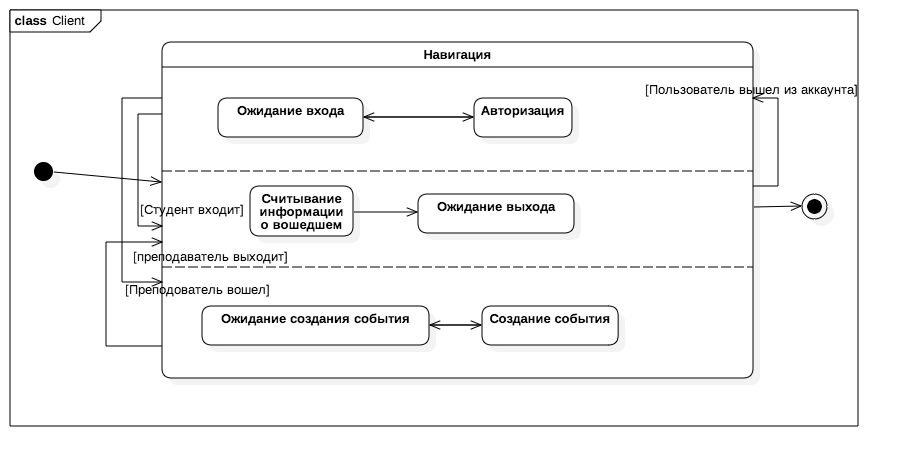


Рис.24. Диаграмма объектов

На рис. 25 изображена диаграмма состояний мобильного приложения (личного кабинета пользователя). Состояние приложения зависит от того вошел пользователь или нет, а также под какой ролью вошел пользователь.

Рис. 25. Диаграмма состояний

На рис 26 изображена диаграмма состояний объекта «Событие». Смена состояний происходит при конфигурации параметров, их проверке и истечении срока хранения.

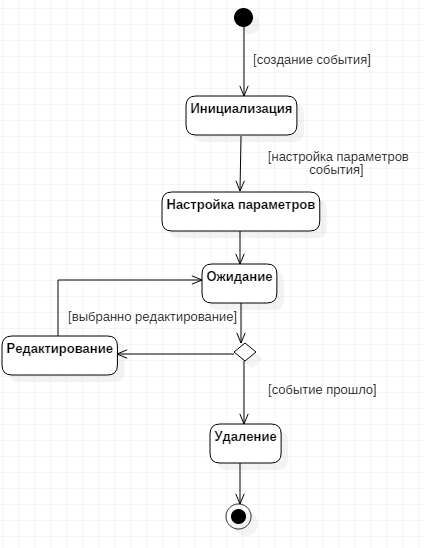


Рис. 26. Диаграмма состояний объекта «Событие»

# Тестирование программного продукта

## Описание тестов

Тестирование – это неотъемлемая составляющая процесса разработки ПО. Его цель – найти как можно больше проблем, и чем серьёзнее найденные проблемы, тем лучше. [14] Для данной системы решено проводить функциональное тестирование методом «чёрного ящика», т.к. оно позволяет установить соответствие разработанного программного обеспечения исходным функциональным требованиям [15].

Для тестирования выбраны модули авторизации и редактирования события.

### Модуль авторизации

Данный модуль осуществляет функцию авторизации существующего пользователя. Входными данными являются логин и пароль пользователя.

Требования к паролю:

* не менее 8 символов;
* наличие спецсимволов.

Логин должен содержать только символы английского алфавита и цифры.

В табл. 6 приведены классы эквивалентности с кратким описанием.

Таблица 6

Классы эквивалености

|  |  |  |
| --- | --- | --- |
| №п/п | Описание | Допустимость |
|  | Длина пароля меньше 8 символов | недопустимый |
|  | Длина пароля больше 8 символов | допустимый |
|  | Пароль содержит спецсимволы | допустимый |
|  | Пароль не содержит спецсимволы | недопустимый |
|  | Логин содержит только латиницу и цифры | допустимый |
|  | Логин содержит символы, кроме латиницы и цифр | недопустимый |

В табл. 7 приведены граничные условия с кратким описанием.

Таблица 7

Граничные условия

|  |  |  |
| --- | --- | --- |
| №п/п | Описание | Допустимость |
| 1. | Длина пароля равна 8 символам | допустимый |

Составим тестовые наборы в соответствии с выделенными классами эквивалентности (табл. 8). Полужирным шрифтом выделены недопустимые классы эквивалентности.

Таблица 8

Тестовые наборы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  п/п | Исходные данные | | Номера классов эквивалентности | Ожидаемый результат |
| логин | пароль |
|  | sdfxcsdf | d@$k | **1**, 3, 5 | ошибка авторизации |
|  | dsf21sdf | ald723hs5 | 2, **4**, 5 | ошибка авторизации |
|  | sfфвл3 | dfg82f3$4 | 2, 3, **6** | ошибка авторизации |
|  | jsdf83sd | a6@f#dвs | 3, 5, 7 | успешная авторизация |
|  | df92fsj3 | sdвf82!3#4 | 2, 3, 5 | успешная авторизация |

### Модуль редактирования события

Данный модуль осуществляет функцию редактирования существующего события. Входными данными являются дата, время, участники и аудитория.

Требования к дате:

* не раньше текущего числа;
* нельзя указывать праздники.

Требования к времени:

* возможность ставить только в рабочие часы университета (8:00 – 20:00).

Участники и аудитория изначально выбираются из действующего списка, поэтому её не нужно проверять.

Таблица 9

Классы эквивалености

| №п/п | Описание | Допустимость |
| --- | --- | --- |
|  | Дата выпадает на праздники | недопустимый |
|  | Дата не выпадает на праздники | допустимый |
|  | Дата указана не задним числом | допустимый |
|  | Дата указана задним числом | недопустимый |
|  | Время указано на рабочие часы университета | допустимый |
|  | Время указано на не рабочие часы университета | недопустимый |

Составим тестовые наборы в соответствии с выделенными классами эквивалентности (табл. 10). Полужирным шрифтом выделены недопустимые классы эквивалентности.

Таблица 10

Тестовые наборы

| №  п/п | Исходные данные | | | Номера классов эквивалентности |
| --- | --- | --- | --- | --- |
| Текущая Дата | Дата | Время |
|  | 15.04.17 | 3.05.17 | 12:00 | **1**, 3, 5 |
|  | 15.04.17 | 15.04.17 | 7:00 | 2, 3, **6** |
|  | 15.04.17 | 14.04.17 | 12:00 | 2, **4**, 5 |
|  | 15.04.17 | 3.05.17 | 21:00 | **1**, 3, **6** |
|  | 15.04.17 | 13.04.17 | 24:00 | 2, **4**, **6** |
|  | 15.04.17 | 16.04.17 | 12:00 | 2, 4, 6 |

Тестовые наборы 1 – 5 должны приводить к обнаружению ошибки входных данных, шестой набор должен успешно пройти проверку.

Заключение

В данной работе была обоснована актуальность выбранной темы, изучены основные соответствующие теоретические положения, научная и справочная литература, произведён анализ собранных данных. На основе анализа предметной области разработаны функциональные требования к разрабатываемой системе. В соответствии с предъявляемыми требованиями спроектирована система расписания для университета:

* определена структура системы;
* выбраны средства разработки;
* созданы макеты интерфейсов системы;
* описаны проектные модели некоторых объектов системы;
* разработаны процедуры тестирования отдельных модулей системы.

Данный проект можно использовать как основу для разработки соответствующей системы или отдельных её модулей.

Список литературы

1. Хабрахабр, статья «Как студенту иметь актуальное расписание, как старостам и преподавателям отправлять сообщения сразу всей группе и другое» 29.12.2014 – Режим доступа: https://habrahabr.ru/company/raspisanie/blog/246867 (дата обращения: 14.04.2017).
2. Хабрахабр, статья «Model View Controller (MVC) опыт использования, выводы» – Режим доступа: https://habrahabr.ru/post/249263/ (дата обращения: 14.04.2017).
3. Свободная энциклопедия Википедия, статья «Model-View-Controller», 2016 – Режим доступа: https://ru.wikipedia.org/wiki/Model-View-Controller (дата обращения: 13.04.2017).
4. Свободная энциклопедия Википедия, статья «JSON Web Token», 2017 – Режим доступа: https://en.wikipedia.org/wiki/JSON\_Web\_Token (дата обращения: 13.04.2017).
5. Хабрахабр, статья «Архитектура мобильного клиент-серверного приложения» 26.12.2014 – Режим доступа: https://habrahabr.ru/post/246877 (дата обращения: 15.04.2017).
6. Харди Б. Android программирование для профессионалов / Харди Б, 2016 – 636с.
7. Сьерра К. Изучаем Java / В. А. Усов, 2012. – Москва: Эксмо, 2010. – 717с.
8. Клифтон Я. Проектирование Пользовательского Интерфейса Android / Клифтон Я, 2017. – ДМК: 2017 – 452с.
9. Головач, В.В. Дизайн пользовательского интерфейса / В.В. Головач, 2015. – 147с.
10. Тодд Варфел, Прототипирование. Практическое руководство / Тодд Варфел – Манн, Иванов и Фербер, 2013. – 240с.
11. Хабрахабр, статья «10 советов по прототипированию в Sketch» 28.03.2015 – Режим доступа: https://habrahabr.ru/post/254215/ (дата обращения: 14.04.2017).
12. Джессе Реззел, Роналд Коч, Диаграмма классов / Джессе Реззел, Роналд Коч – Книга по требованию, 2013. – 162с.
13. Хасан Гома, UML-проектирование систем реального времени параллельных и распределенных приложений / Хасан Гома – ДМК Пресс, 2011. – 704с.
14. Р. Блэк, Ключевые процессы тестирования. / Р. Блэк – Лори, 2014. – 538с.
15. Бейзер, Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем / Б.Бейзер. – СПб.: Питер, 2004. – 318с.