**Содержание**

[Перечень условных обозначений, символов и терминов 5](#_Toc102761556)

[Введение 6](#_Toc102761556)

[1 Анализ и моделирование работы центра по организации мероприятий 7](#_Toc102761557)

[1.1 Описание деятельности центра по организации мероприятий 7](#_Toc102761558)

[1.2 Разработка функциональной модели центра по организации мероприятий 11](#_Toc102761559)

[1.3 Анализ требований к разрабатываемому программному средству. Спецификация функциональных требований. 14](#_Toc102761559)

[1.4 Разработка информационной модели деятельности центра по организации мероприятий 15](#_Toc102761559)

[1.5 Модели представления программного средства и их описание 16](#_Toc102761559)

[2 Проектирование и конструирование программного средства 21](#_Toc102761562)

[2.1 Постановка задачи 21](#_Toc102761563)

[2.2 Обоснование выбора компонентов и технологий для реализации программного средства 22](#_Toc102761564)

[2.3 Архитектурные решения 23](#_Toc102761564)

[2.4 Описание алгоритмов, реализующих ключевую бизнес-логику разрабатываемого программного средства 24](#_Toc102761564)

[2.5 Проектирование пользовательского интерфейса 28](#_Toc102761564)

[2.6 Методы и средства, используемые для обеспечения безопасности данных 29](#_Toc102761564)

[3 Тестирование и проверка работоспособности программного средства 30](#_Toc102761569)

[4 Руководство по развертыванию и использованию программного средства 31](#_Toc102761557)

[4.1 Руководство по установке(развёртыванию) программного средства 31](#_Toc102761558)

[4.2 Руководство пользователя 31](#_Toc102761559)

[Заключение 37](#_Toc102761574)

[Список использованных источников 38](#_Toc102761575)

# Перечень условных обозначений, символов и терминов

|  |  |
| --- | --- |
| БД (база данных) | – представленная в объективной форме совокупность самостоятельных материалов, систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины |
| Информационная система | – система, предназначенная для хранения, поиска и обработки информации, и соответствующие организационные ресурсы, которые обеспечивают и распространяют информацию |
| *API* (*application programming interface*) | – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой |
| *CRUD* | – [акроним](https://ru.wikipedia.org/wiki/%D0%90%D0%BA%D1%80%D0%BE%D0%BD%D0%B8%D0%BC), обозначающий четыре базовые функции, используемые при работе с [базами данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85): создание ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *create*), чтение (*read*), модификация (*update*), удаление (*delete*) |
| *IDEF0* | – методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов |
| *Java* | – строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией |
| *REST* | – это механизм вызова удаленных процедур (RPC), подобно SOAP, где вызовы осуществляются посредством простых обращений по протоколу HTTP и без огромных пространств имен SOAP в XML; |
| *SQL* | – язык структурированных запросов, декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных |
| *UML* | – язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур |

**Введение**

В настоящее время на рынке приложений представлено множество продуктов, помогающих пользователям решать огромное количество повседневных задач. Особое внимание можно уделить программным средствам, предназначенным для автоматизации работы планирования и организации мероприятий. Организация мероприятий является отличным методом к продвижению брендов, в особенности среди больших компаний. Но не каждое мероприятие будет эффективно. Более того, неправильный выбор мероприятия и плохая организация могут негативно отразиться на имидже бренда и неправильно его коммуницировать. Организация мероприятий требует максимального внимания как к клиентам, так и к процессу подготовки мероприятий. Автоматизация работы агентств позволит увеличить их производительность, минимизировать риски и влияние человеческого фактора, экономить время на обработке даже сложных заказов.

Целью данного проекта является создание информационной системы для повышения эффективности проведения мероприятий за счёт управления процессом планирования.

Объектом исследования данного проекта являются процессы работы организации по проведению мероприятий. Предметом исследования является автоматизация процессов предоставления услуг.

Задачи проекта:

* исследование основных процессов проведения мероприятий, определение требований к системе, организация работы с базой данных, разработка моделей представления системы;
* разработка архитектурных решений для приложения, разработка пользовательского интерфейса, описание основных алгоритмов программы;
* тестирование программного средства;
* описание работы приложения и инструкции по развертыванию.

Актуальность данной работы состоит в том, что подобные программные продукты позволяют пользователям и сотрудникам экономить время и качественно и точно организовать делегирование конкретных задач конкретным людям.

# 1 Анализ и моделирование работы центра по организации мероприятий

## Описание деятельности центра по организации мероприятий

В современном мире важнейшим ресурсом является время, поэтому многие пытаются оптимизировать различные процессы. Ведущую роль играют беспроводная и проводная сеть Интернет, которая является не только средством экономии времени и источником информации, но и универсальной средой для общения, развлечения и обучения. Количество пользователей Интернетом растет с каждым днем. Наибольшую популярность приобрели в наши дни web-сайты. Веб-сайтом называют некую сущность, совокупность веб-документов, объединенных в единую структуру, которая расположена по конкретному доменному адресу. Веб-сайт со стороны пользователя воспринимается как единый целый организм, он обязательно имеет собственный индивидуальный адрес.

Организация мероприятий сегодня - корпоративный праздник, запуск нового проекта, прием VIP-гостей, презентация компании - рассматривается как один из главных элементов коммуникации компаний и организаций, важный инструмент формирования и управления репутацией. Работа агентств по организации мероприятий представляет собой постоянный поток заказов и взаимодействие с десятками посредников. Подготовка одного мероприятия может занимать несколько месяцев, поэтому отслеживать действия сотрудников становится сложно [1]. Среди особенностей такого бизнеса отмечаются:

* изменчивость требований. Коррективы в план организации мероприятия вносятся в большинстве случаев. Поэтому менеджерам необходимо оперативно учитывать их и уточнять;
* высокая конкуренция. Чтобы удерживать позиции на рынке, бизнес должен стать надёжным и узнаваемым в своей нише;
* зависимость результата от работы персонала;
* высокий средний чек.

Технологии стремятся сделать планирование мероприятий более простым и эффективным. Автоматизация является ключевым фактором для успешной работы агентств и организаторов мероприятий. Она позволяет эффективно планировать, организовывать и выполнять все задачи, связанные с проведением мероприятия. Автоматизация упрощает координацию с участниками, ведение документации и аналитику результатов. Это повышает эффективность и уменьшает риск ошибок, что в свою очередь позволяет уделять больше внимания креативным аспектам мероприятия, позволяет организаторам мероприятий автоматизировать весь процесс управления мероприятиями, начиная с [**регистрации**](http://blog.hubilo.com/create-event-registration-form/)**,**[**продвижения мероприятия**](https://www.hubilo.com/event-promotion)изаканчивая[**обратной связью**](http://blog.hubilo.com/how-to-ask-perfect-event-feedback/). Первоначально программное обеспечение для управления мероприятиями использовалось для логистики и бюджетирования. В конце концов, он включал такие функции, как регистрация и [**маркетинг по электронной почте.**](https://www.hubilo.com/event-marketing)

Однако проблема с традиционным программным обеспечением для управления мероприятиями заключалась в том, что каждый процесс, даже оцифрованный, должен был выполняться вручную. Простым решением проблемы стало появление программного обеспечения для автоматизации событий. Организаторы мероприятий теперь могут автоматизировать все те задачи, которые изначально выполнялись вручную с помощью программного обеспечения для регистрации событий, будь то создание веб-сайта мероприятия, настройка [**регистрации событий**](https://www.hubilo.com/event-registration) или рассылка почтовых сообщений вашим посетителям.  Самым большим преимуществом автоматизации мероприятий является упрощение различных процессов, связанных с планированием мероприятий. Как упоминалось ранее, несмотря на то, что различные процессы, связанные с планированием мероприятий, были оцифрованы, организаторам мероприятий стало трудно справиться с ситуацией. Вы можете управлять регистрацией, продажей билетов, маркетинговыми кампаниями по электронной почте, кампаниями по продвижению событий и т. д. из одного места. Кроме того, поскольку весь процесс автоматизирован, на самом деле не нужно быть настолько технически подкованным, чтобы использовать программное обеспечение для автоматизации событий. Достаточно одного прогона, чтобы ознакомиться со всеми функциями программного обеспечения.

Оптимизировать деятельность компании позволит внедрение современной CRM-системы. Программы обладают расширенным функционалом для согласованной работы персонала над заказами, создания и ведения клиентской базы и выстраивания взаимоотношений с клиентами в мессенджерах, соцсетях, почте [2].

Данные системы активно используются в сфере организации выставок, конференций, презентаций и праздников. Они способны выполнять целый ряд функций в [решении задач бизнеса](https://sbercrm.com/blog/sales/tpost/06r8f8xhy1-vibor-sistemi-upravleniya-zadachami). Так специализированные программные продукты обеспечивают:

* прозрачность заказов. Ни одна деталь не останется без внимания, если своевременно вносить поправки и дополнения в план мероприятия. Доступ к заказу имеют все задействованные в его подготовке сотрудники. Сведения хранятся в единой базе, поэтому даже новый менеджер легко включится в работу над проектом;
* повышение клиентоориентированности;
* лёгкость управления агентством. Руководитель ставит задачи и контролирует сроки их выполнения прямо в программе. Общение между сотрудниками производится в чате, а отчёты формируются автоматически и максимально быстро;
* увеличение заказов.

Одной из наиболее успешных и популярных программ у владельцев отдельных и сетевых event-агентств является «MeetingHand» (рисунок 1.1).

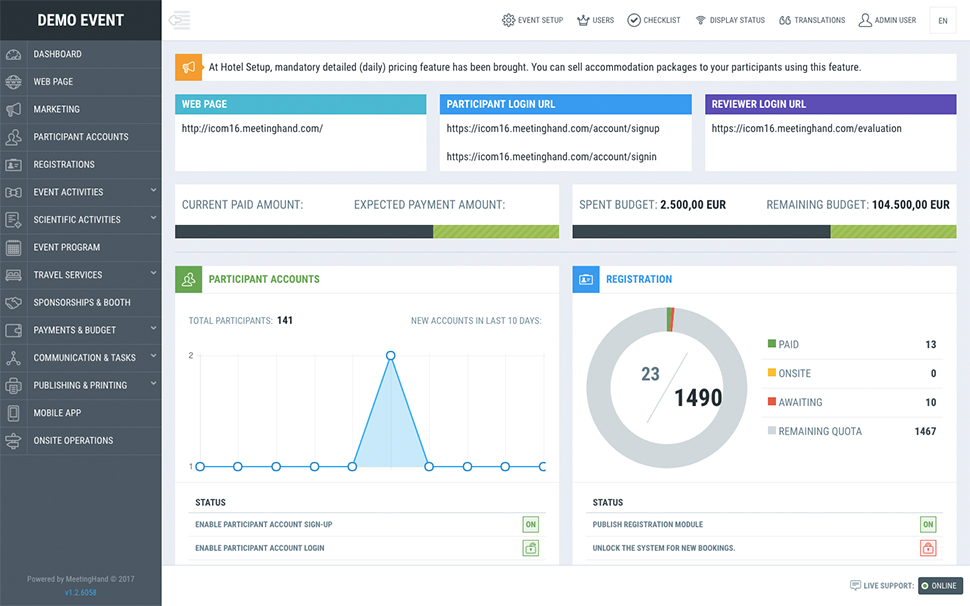


Рисунок 1.1 – Интерфейс программы «MeetingHand»

MeetingHand - это онлайн-платформа для управления мероприятиями, которая помогает менеджерам планировать, организовывать и продвигать свои проекты. Некоторые из ее функций включают управление регистрацией мероприятий, маркетинг, создание веб-сайта управление посетителями и продажа билетов. Пользователи MeetingHand также могут создавать дополнительные события и планировать конкретные мероприятия для участников, такие как семинары, экскурсии и ужины на конференциях. Организаторы могут повысить вовлеченность аудитории с помощью push-уведомлений и функций онлайн-чата в мобильных приложениях продукта [3].

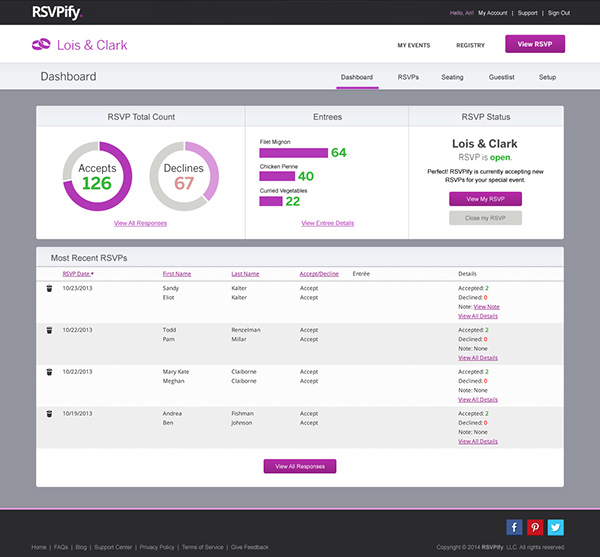


Рисунок 1.2 – Интерфейс программы «RSVPify»

RSVPify - это облачная платформа сквозного управления мероприятиями для проведения личных и виртуальных событий (рисунок 1.2). Ее функции включают управление списками гостей, продажу билетов и встраиваемые формы RSVP для регистрации участников. Кроме того, программа предлагает составление схемы рассадки с помощью перетаскивания для планирования оптимальной рассадки, опции меню, позволяющие гостям делиться выбором блюд и диетическими потребностями, а также проверку возраста для мероприятий, имеющих возрастные ограничения. Программное обеспечение предлагает функцию вторичных мероприятий. RSVPify предлагает библиотеку поддержки, видеоуроки и круглосуточную поддержку пользователей на основе тикетов.

## Разработка функциональной модели деятельности центра по организации мероприятий

Наиболее удобным языком моделирования бизнес-процессов является IDEF0, где система представляется как совокупность взаимодействующих работ или функций. В этом случае функции системы анализируются независимо от объектов, которыми они оперируют, что позволяет более четко смоделировать логику и взаимодействие процессов организации [4].

На рисунке 1.3 представлена контекстная диаграмма проекта верхнего

уровня. Она определяет главный процесс. Входная информация: запрос на проведение мероприятия, разрешение на проведение мероприятия. Информация обрабатывается и конвертируется в конечную цель: проведённое мероприятие, чек-оплата, акт выполненных работ. Управляющий механизм: сотрудники, клиент, оборудование. Механизм ограничения: нормативные документы законы РБ, общие условия проведения, прайс-лист.

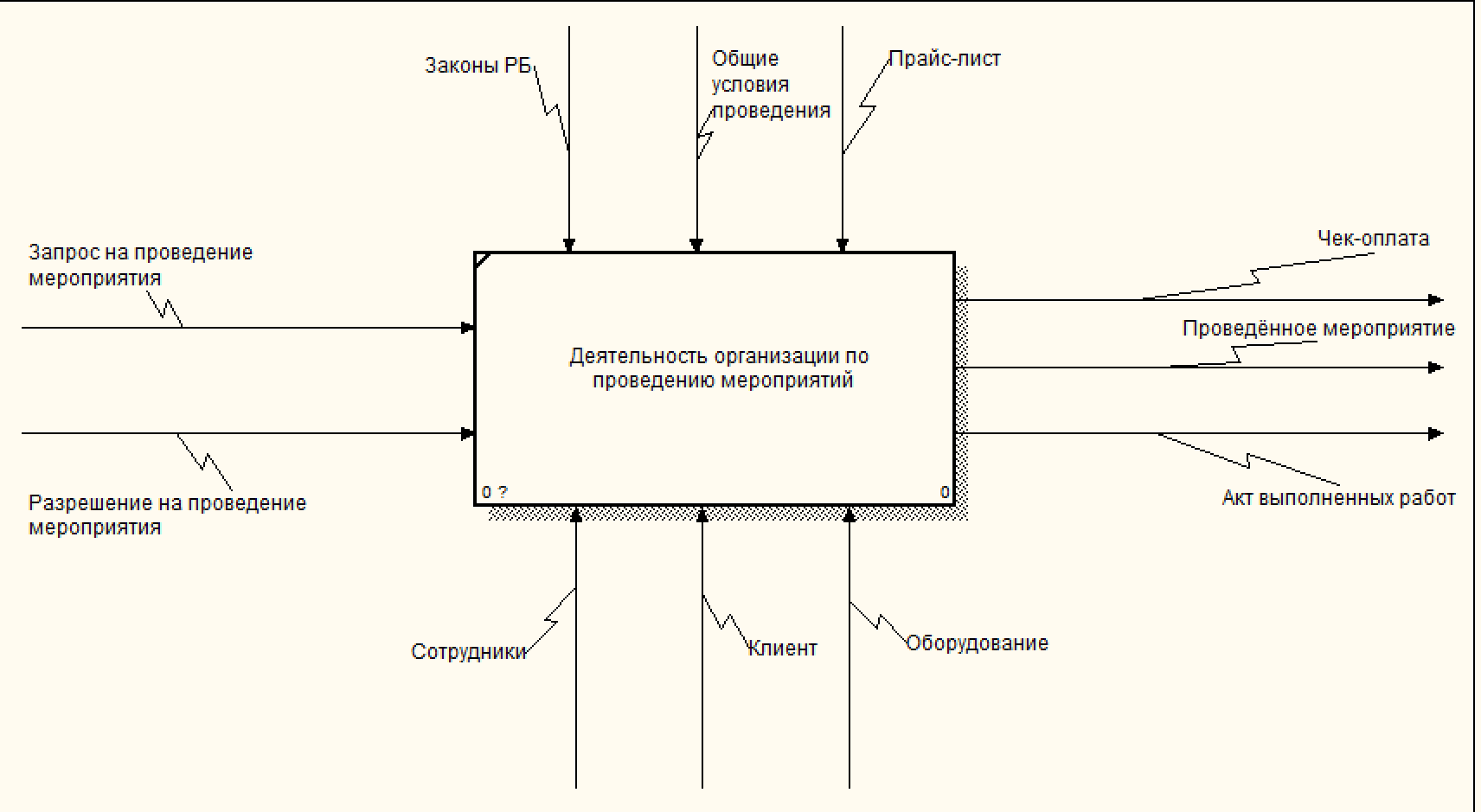


Рисунок 1.3 – Контекстная диаграмма верхнего уровня

Более подробно содержание раскрывается на дочерней диаграмме. Далее будем пошагово раскрывать наш процесс. Для начала декомпозируем процесс на компоненты-задачи (рисунок 1.4):

* спланировать мероприятие;
* провести мероприятие;
* произвести расчёт и составить акт выполненных работ.

Конечными целями системы по результатам выполнения процесса является проведённое мероприятие, получение чек-оплаты и акта выполненных работ.

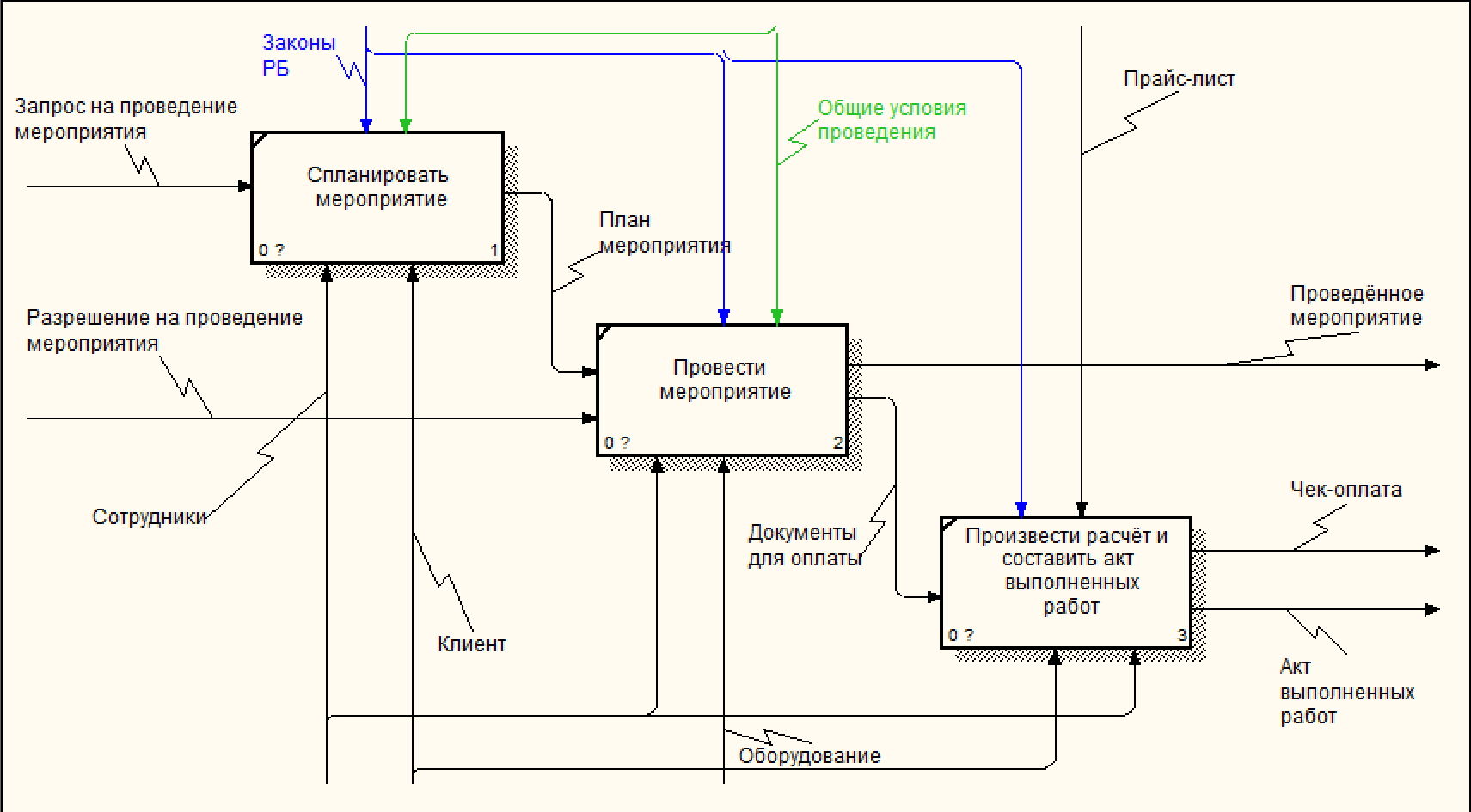


Рисунок 1.4 – Декомпозиция блока контекстной диаграммы

На следующем этапе разработки модели выполняется декомпозиция подпроцесса «Провести мероприятие». На данном уровне существуют следующие подпроцессы (рисунок 1.5):

* скомплектовать мероприятие;
* провести мероприятие;
* подвести итоги мероприятия.

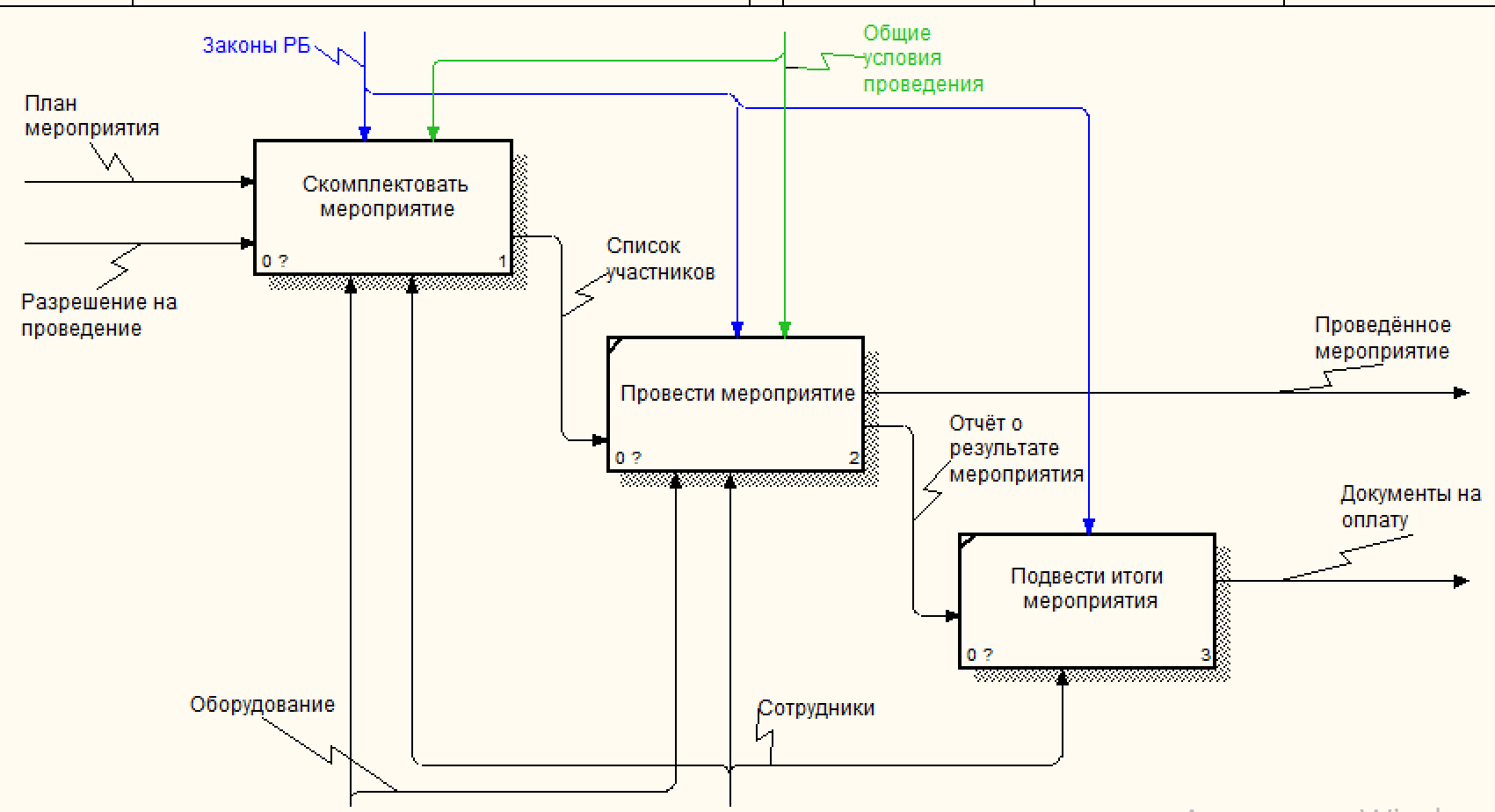


Рисунок 1.5 – Декомпозиция блока «Провести мероприятие»

На следующем этапе разработки модели выполняется декомпозиция подпроцесса «Скомплектовать мероприятие». На данном уровне существуют следующие подпроцессы (рисунок 1.6):

* подготовить документы к комплектованию мероприятия;
* зарегистрировать участников в БД;
* разослать приглашение участникам;
* завершить комплектование мероприятия.

После последовательного выполнения всех подпроцессов будут достигнуты конечные цели процесса: составлены и оформлены документы на оплату и проведено мероприятие.

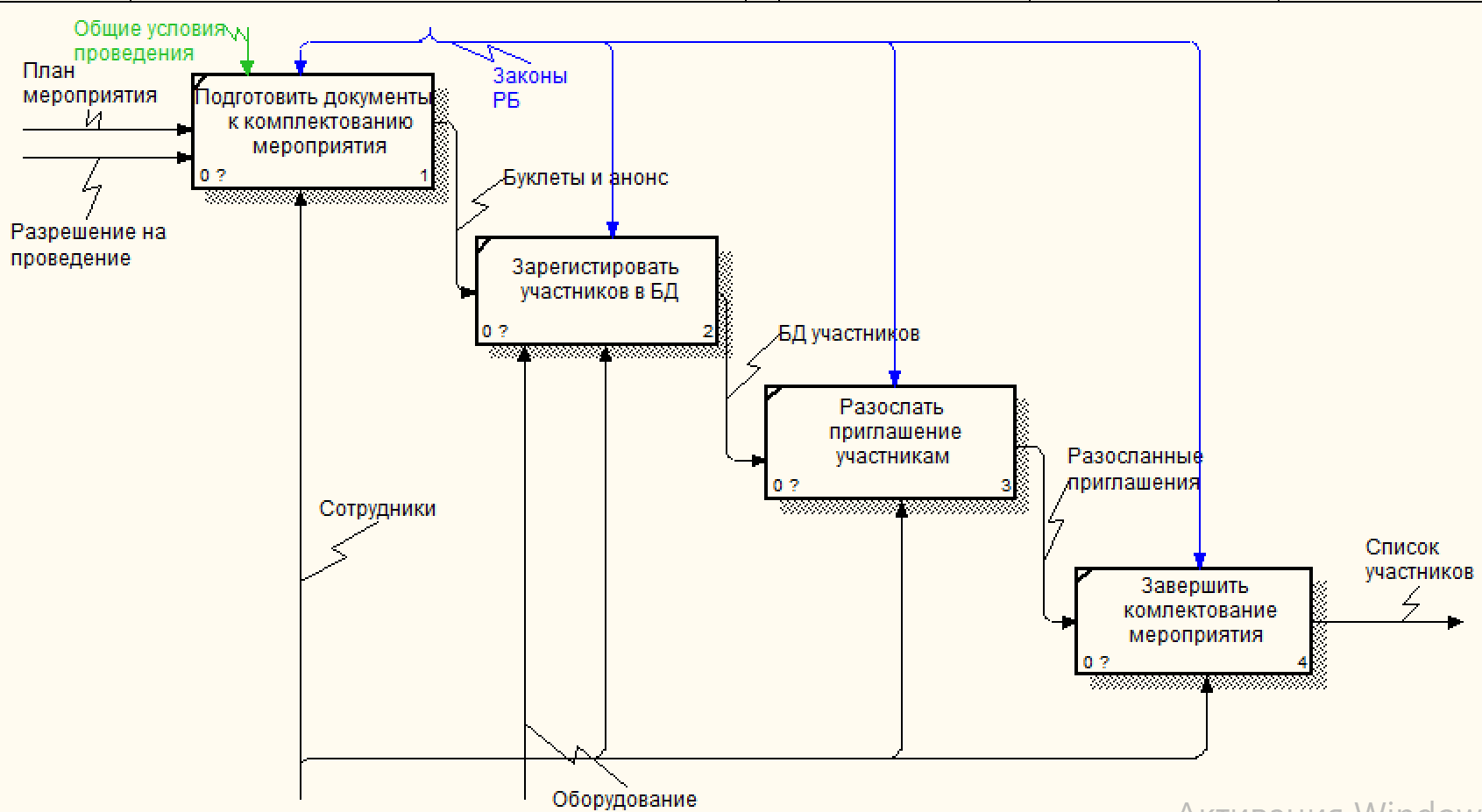


Рисунок 1.6 – Декомпозиция блока «Скомплектовать мероприятие»

Целью построения функциональной модели является описание всех необходимых процессов с точностью, достаточной для однозначного моделирования деятельности системы. Уровень декомпозиции рассмотренных диаграмм достаточен для отражения цели моделирования, и на диаграммах нижнего уровня в качестве наименований работ используются элементарные функции (с точки зрения пользователя системы). Из представленных декомпозиций можно сделать вывод о том, как устроена и на сколько сложна деятельность организации по проведению мероприятий. Таким образом, автоматизация некоторых процессов облегчит работу и повысит эффективность.

## Анализ требований к разрабатываемому программному средству. Спецификация функциональных требований

Функционал разрабатываемой системы можно описать при помощи диаграммы вариантов использования. Диаграмма вариантов использования – диаграмма, отражающая отношения между актёрами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне. Основное назначение диаграммы — описание функциональности и поведения, позволяющее заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему. Диаграмма вариантов использования является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними.

Диаграмма претендентов для данного проекта представлена на рисунке 1.7:

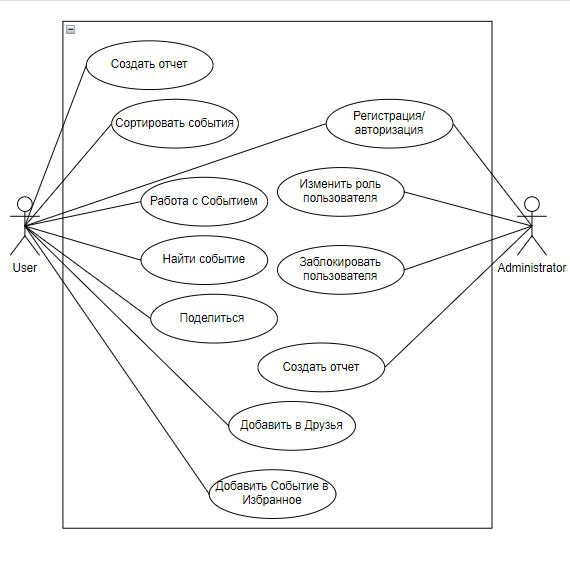


Рисунок 1.7 – Диаграмма вариантов использования системы

На диаграмме вариантов использования представлены следующие актеры: пользователь и администратор. Администратору доступны следующие действия: изменение поли пользователя, блокировка пользователя, создание отчёта по мероприятиям. Клиенту доступны следующие действия: работа с событиями, поиск необходимого события, сортировка и возможность поделиться этим событием, создание отчёта и добавление друзей.

|  |  |
| --- | --- |
| **Роль** | **Описание** |
| Пользователь | Пользователь – зарегистрированный пользователь. У него есть права на:   1. Зарегистрироваться (Если пользователь впервые заходит в приложение) и авторизироваться (Все последующие сессии) 2. Добавление событий 3. Возможность добавить пользователей в событие 4. Создать отчет по событиям 5. Добавить другого пользователя в друзья 6. Найти событие 7. Добавить событие в избранное 8. Изменить данные события (Время проведения, имя, тд) |
| Администратор | Администратор – зарегистрированный пользователь, которому были выданы определенные права другим администратором или программно через консоль разработчика. Имеет права (сверх пользовательских) на :   1. Блокировку пользователей 2. Смена роли пользователя 3. Создание администраторского отчета (по всем событиям всех пользователей) 4. Просмотр всех аккаунтов пользователей 5. Просмотр всех событий пользователей |

## Разработка информационной модели фитнес-центра

Под моделью информационной системы в общем случае понимается формализованное описание системы на определенном уровне абстракции. Каждая модель определяет конкретный аспект системы, использует набор диаграмм и документов заданного формата [5]. Сущности описывают объекты, являющиеся предметом деятельности предметной области, а связи – это потоки данных, информации, документов, распоряжений и прочего. Каждая сущность может быть декомпозирована – и быть представлена, в свою очередь, как совокупность более мелких и быстрых процессов и обмена информацией, требуемого для их реализации.

При создании информационной модели были выделены следующие сущности:

* friend\_request;
* comment;
* user;
* user\_friends;
* user-friends\_requests;
* event.

Графическое представление базы данных реализовано на платформе MySQL Workbench (рисунок 1.8).

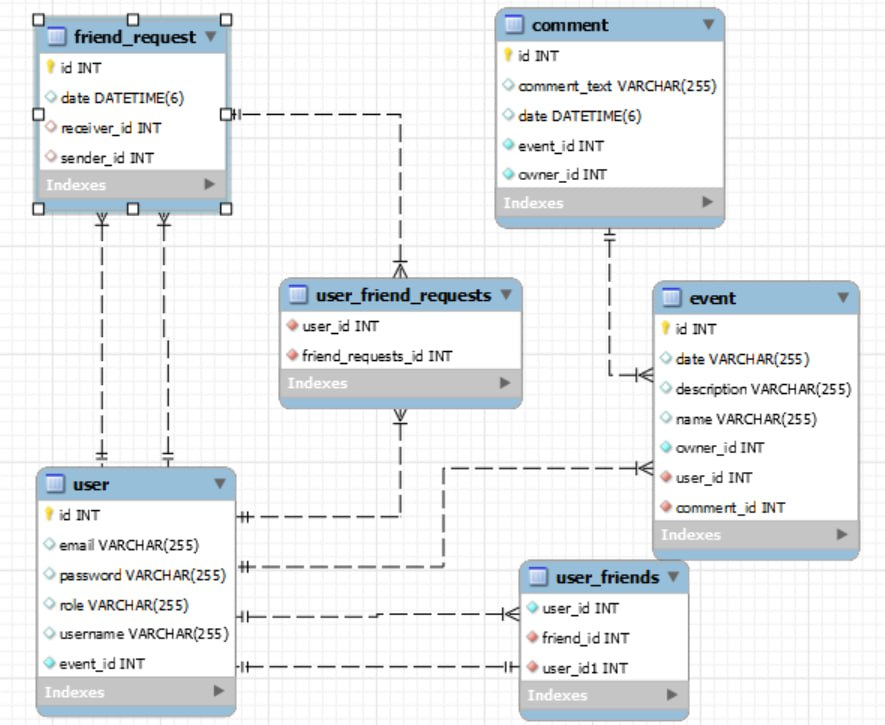


Рисунок 1.8 – Информационная модель системы

работы event-агентства

Как итог, была спроектирована база данных для дальнейшей работы программы. Все отношения соответствуют третьей нормальной форме, что означает, что таблица находится во второй нормальной форме и каждый не ключевой атрибут не транзитивно зависит от первичного ключа, что является главным правилом третьей нормальной формы. Это можно доказать тем, что одному значению не ключевого поля может соответствовать другое значение, имеющее аналогичный тип данных – и это не будет являться логической ошибкой.

## Модели представления программного средства и их описание

UML – [язык](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) [графического](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) описания для [объектного моделирования](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) в области [разработки программного обеспечения](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F), [моделирования бизнес-процессов](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%B8%D0%B7%D0%BD%D0%B5%D1%81-%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1), [системного проектирования](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и отображения [организационных структур](https://ru.wikipedia.org/wiki/%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0). Он является языком широкого профиля, использующий графические обозначения для создания [абстрактной модели](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D1%81%D1%82%D1%80%D0%B0%D0%BA%D1%82%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C) [системы](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), называемой UML-моделью. Существует два основных типа диаграмм UML: структурные диаграммы и поведенческие диаграммы. Эти варианты существуют для представления многочисленных типов сценариев и диаграмм, которые используют разные типы людей [6]. [Структурные диаграммы](https://support.office.com/en-us/article/create-a-uml-static-structure-diagram-bb11b3df-8830-4e31-a437-332585da7ce8)  представляют статическую структуру программного обеспечения или системы, они также показывают различные уровни абстракции и реализации. Они используются, чтобы помочь визуализировать различные структуры, составляющие систему, например, базу данных или приложение (Диаграмма классов, диаграмма развёртывания).Поведенческие диаграммы показывают функциональные возможности системы и демонстрируют, что должно происходить в моделируемой системе. Основное внимание здесь уделяется динамическим аспектам системы программного обеспечения или процесса (диаграмма деятельности, вариантов использования, последовательности).

**1.5.1** Диаграмма последовательности

Диаграмма последовательности, моделирует взаимодействие между объектами в едином сценарии использования. Она иллюстрирует, как различные части системы взаимодействуют друг с другом для выполнения функции, а также порядок, в котором происходит взаимодействие при выполнении конкретного случая использования.

Диаграмма последовательности описывает поведение только одного варианта использования. Рассмотрим диаграмму последовательности отображающую процесс построения графика (рисунок 1.9).

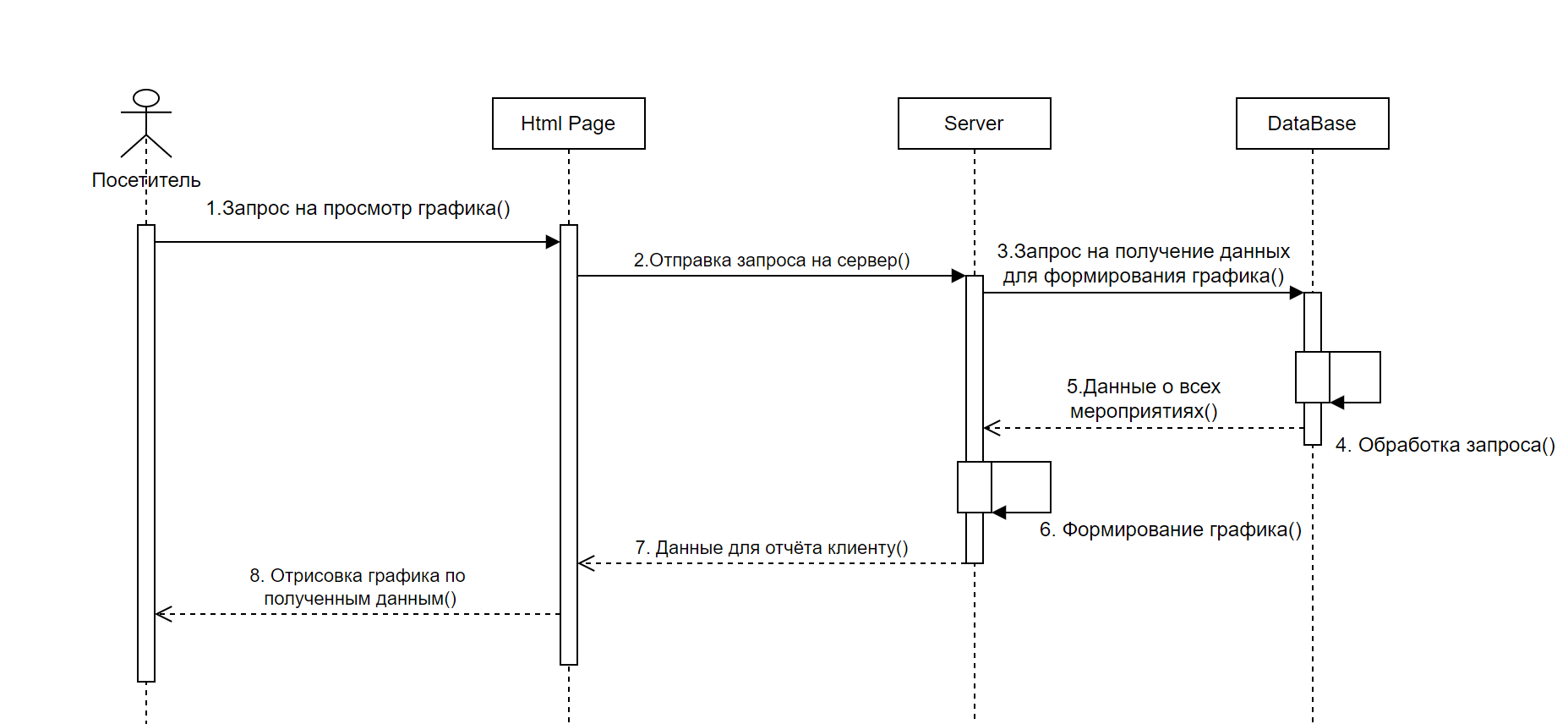


Рисунок 1.9 – Диаграмма последовательности для просмотра статистики

Взаимодействуя с компонентами интерфейса, спроектированного при помощи формата HTML, пользователь выбирает необходимое действие и инициирует отправку на сервер. На сервере, после прохождения проверки, формируется запрос, обрабатываемый СУБД. На выходе сервер формирует объект в случае корректного выполнения. Объект отправляется на клиент, который формирует график на основе полученного набора данных (в случае получения строки выводит сообщение об ошибке). Таким образом, диаграмма иллюстрирует переходы контроля управления между объектами используя инструменты и компоненты моделирования при помощи UML.

**1.5.2** Диаграмма состояний

Рассмотрим схему состояния программы в процессе создания мероприятия (рисунок 1.10).

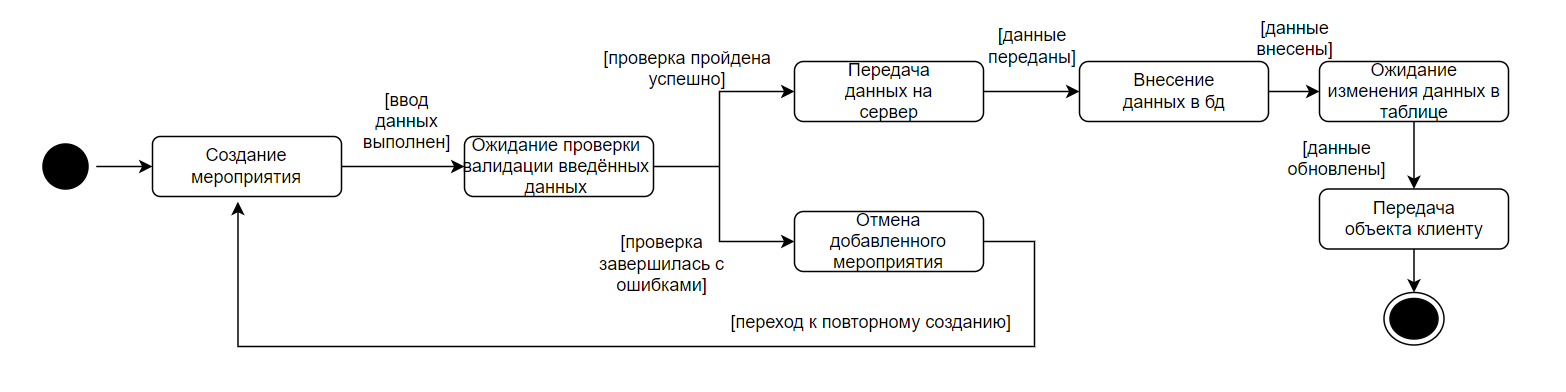


Рисунок 1.10 – Схема состояния при создании мероприятия

  Процесс начинается с начального блока создания мероприятия, в случае данной программной реализации это нажатие кнопки добавления мероприятия в меню менеджера. Далее программа переходит в состояние ожидание проверки введенных данных, если же данные прошли проверку на валидацию, данные вносятся в базу данных и серверная часть ожидает изменение данных в модели. Дальнейшие действия заключаются в передачи объекта на клиентскую часть, которая в свою очередь весь цикл ожидает поступления данных и их обновления.

**1.5.3** Диаграмма развёртывания

Диаграммы развертывания используются для визуализации аппаратных процессоров/узлов/устройств системы, каналов связи между ними и размещения программных файлов на этом аппаратном обеспечении.

В качестве узлов выделяются серверное приложение, клиентское приложение и средство управления базами данных. Между узлами установлены коммуникационные ассоциации, реализуемые через протокол HTTP (рис 1.11):

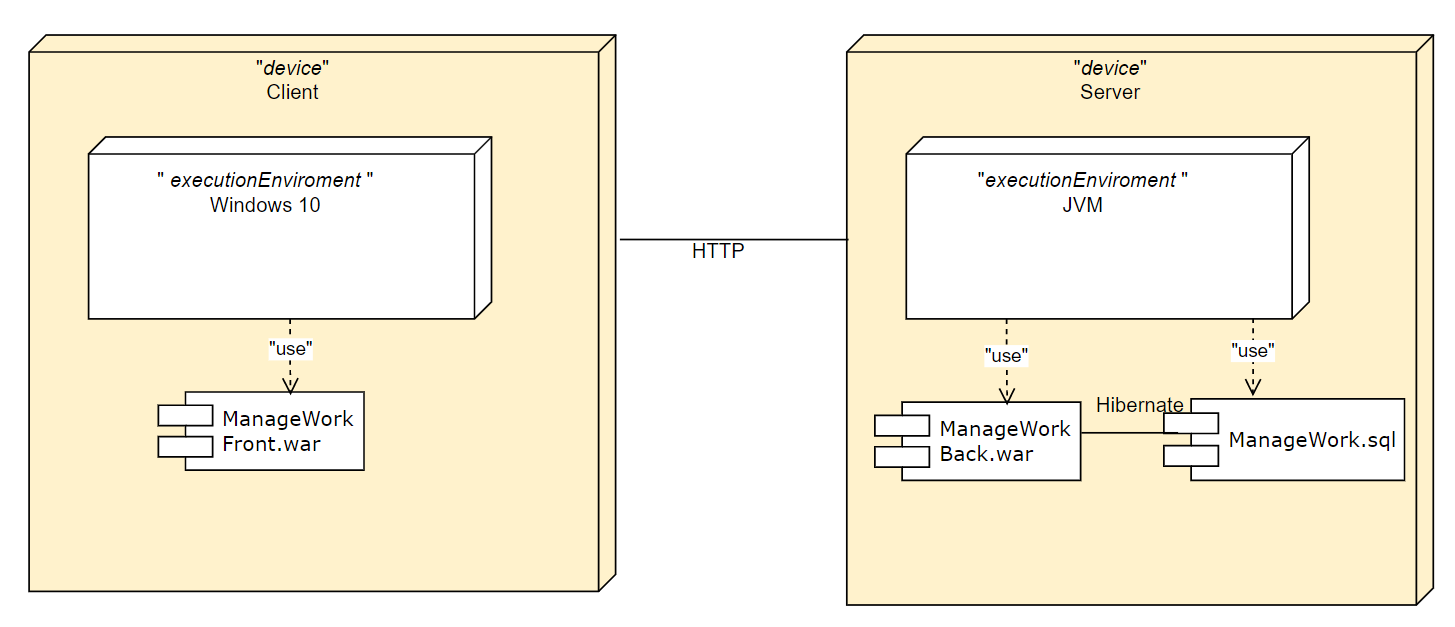


Рисунок 1.11 – Диаграмма развёртывания системы

**1.5.4** Диаграмма компонентов

Диаграмма компонентов описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. На рисунке 1.12 представлена диаграмма компонентов для разрабатываемого программного средства.

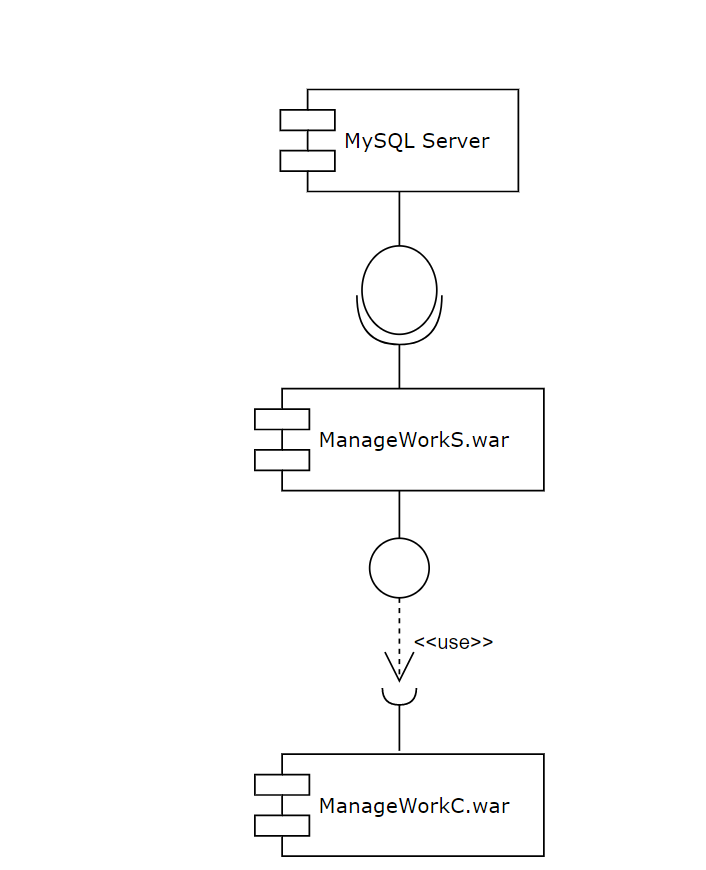


Рисунок 1.12 – Диаграмма компонентов системы

В диаграмме компонентов имеется пакет ManageWorkC, в котором хранятся все необходимые файлы для работы клиентской части программы.  В пакете ManageWorkS хранятся все необходимые файлы для правильной работы серверной части приложения. MySQL Server содержит базу данных, с которой будет работать данное программное средство.

# 2 Проектирование и конструирование программного средства

## 2.1 Постановка задачи

Основной задачей, поставленной в начале работы над данным проектом, является разработка программного средства, автоматизирующее и оптимизирующее процессы работы организации по проведению мероприятий. Подзадачей будет считаться разработка интуитивно понятного интерфейса, который упрощал бы взаимодействие пользователей с сервисом, не всегда способных сформулировать свои запросы. Также интерфейс можно считать за средство взаимодействия между менеджером и клиентом.

Таким образом, для реализации данного проекта необходимо выполнить следующие основные задачи:

− изучить основные процессы;

– создать интуитивно понятный пользовательский интерфейс;

– разработать базу данных и приложение;

– протестировать полученное приложение.

Разработанное в ходе выполнения проекта приложение будет доступно внутри event-агенства. С целью более эффективной работы с информацией в приложении требуется реализовать разделение пользователей на роли Администратора и Пользователя. Система позволит корректно и быстро осуществлять взаимодействие всех ролей.

При проектировании системы необходимо выполнить моделирование с использованием следующих стандартов:

− IDEF0 – функциональное моделирование процессов предметной области решаемой задачи (не менее чем 4 уровня);

− IDEF1.X – информационное моделирование;

− UML 2.0 – модели представления системы на основе UML

Бизнес-логика системы должна быть реализована на серверной части приложения. Бизнес-логика приложения реализована на основе фреймворка Spring. База данных должна быть приведена к 3-ей нормальной форме. Приложение должно задействовать функционал стороннего сервиса. Также должен быть реализован процесс защиты хранимой и передаваемой информации.

## 2.2 Обоснование выбора компонентов и технологий для реализации программного средства

Для разработки была выбрана наиболее популярная и удобная среда разработки: IntelliJ IDEA. Она имеет ряд преимуществ: по ходу написания кода программа анализирует его и предлагает, как может код быть дополнен, решены существующие ошибки; поддержка широкого круга фреймворков и платформ, а именно Spring Framework, Node.js, AngularJS, Java EE; интеграция с системами контроля версий, например, Git, Subversion, Mercurial, CSV; инструменты для работы с базами данных; инструменты запуска тестов и анализа покрытия кода, включая JUnit, ScalaTest, Cucumber, TestNG.

Что касается базы данных, то в моём приложение использовалась популярная СУБД с моделью «клиент-сервер» – MySQL. MySQL - это популярный сервер баз данных, используемый в разных приложениях. SQL означает язык структурированных запросов - (S)tructured (Q)uery (L)anguage, который MySQL использует для коммуникации с другими программами. Сверх того, MySQL имеет свои собственные расширенные функции SQL для того чтобы обеспечить пользователям дополнительный функционал [7]. Основные преимущества MySQL: простота в использовании; обширный функционал; безопасность; масштабируемость; скорость.

Spring – это популярный фреймворк для разработки на Java, который используют для создания корпоративных приложений, например, CRM. Spring позволяет создавать приложения из обычных объектов Java (plain old Java objects, POJO) и применять к ним корпоративные сервисы. Эта возможность относится к модели программирования Java SE и полностью или частично –  к Java EE. Приложения Java обычно состоят из объектов, которые работают совместно. Таким образом объекты приложения находятся в зависимости друг от друга.

Spring Framework обеспечивает комплексную модель разработки и конфигурации для современных бизнес-приложений на Java - на любых платформах. Ключевой элемент Spring - поддержка инфраструктуры на уровне приложения: основное внимание уделяется "водопроводу" бизнес-приложений, поэтому разработчики могут сосредоточиться на бизнес-логике без лишних настроек в зависимости от среды исполнения.

В отличие от других платформ, Spring фокусируется на нескольких областях функционирования приложений и предоставляет для них широкий спектр дополнительных функций.

Одной из основных особенностей Spring Framework является использование паттерна Dependency Injection (DI, внедрение зависимости). DI помогает намного проще реализовывать необходимую приложениям функциональность, а также позволяет разрабатывать слабо связанные классы, делая их более универсальными.

Hibernate — это решение объектно-реляционного сопоставления для сред Java. Термин [объектно-реляционное отображение](https://en.wikipedia.org/wiki/Object-relational_mapping) относится к методу отображения данных из представления объектной модели в представление реляционной модели данных и наоборот [8].

Hibernate не только заботится об отображении классов Java в таблицы базы данных и типов данных Java в типы данных SQL, но также предоставляет средства запроса и извлечения данных. Это может значительно сократить время разработки, затрачиваемое на ручную обработку данных в SQL и JDBC. Цель разработки Hibernate состоит в том, чтобы избавить разработчика от 95% общих задач программирования.

Hibernate может быть не лучшим решением для ориентированных на данных приложений, которые используют только хранимые процедуры для реализации бизнес-логики в базе данных, он наиболее полезен с объектно-ориентированными моделями предметной области и бизнес-логикой на среднем уровне на основе Java. Тем не менее, Hibernate, безусловно, может помочь удалить или инкапсулировать SQL-код, зависящий от поставщика, и поможет с распространенной задачей перевода набора результатов из табличного представления в граф объектов.

Java Persistence API – это набор классов и методов для постоянного хранения огромных объемов данных в базе данных, предоставляемой корпорацией Oracle. Чтобы уменьшить нагрузку на написание кодов для управления реляционными объектами, программист следует структуре «JPA Provider», которая позволяет легко взаимодействовать с экземпляром базы данных. Здесь необходимые рамки приняты JPA. Java Persistence API является источником для хранения бизнес-сущностей как реляционных сущностей.

## 2.3 Архитектурные решения

При проектировании программного продукта были использованы паттерны проектирования. Паттерн проектирования – повторяемая архитектурная конструкция, представляющая собой решение проблемы проектирования в рамках некоторого часто возникающего контекста.

Singleton – это шаблон (паттерн) проектирования, который дает гарантию, что у класса будет всего один экземпляр класса и предоставляет глобальную точку доступа к экземпляру данного класса.

Отсюда две особенности, характерные для практически каждой реализации паттерна Singleton:

* приватный конструктор. Ограничивает возможность создания объектов класса за пределами самого класса.
* публичный статический метод, который возвращает экземпляр класса. Данный метод называют getInstance. Это глобальная точка доступа к экземпляру класса.

Builder – это порождающий паттерн проектирования, который позволяет создавать сложные объекты пошагово. Строитель дает возможность использовать один и тот же код для получения разных представлений объектов.

В разработке данного программного средства были использованы модели Репозиторий и MVC. Репозиторий представляет собой все объекты определенного типа в виде концептуального множества. Репозиторий – это то, что на самом деле является моделью – набор методов, реализующих бизнес-логику приложения. Метод в репозитории может включать в себя десяток разных запросов к БД для получения набора данных, необходимого в приложении, плюс их обработку.  Его поведение похоже на поведение коллекции, за исключением более развитых возможностей для построения запросов. В свою очередь, следуя модели MVC нужно разделять систему на три составные части. Их, в свою очередь, можно называть модулями или компонентами. У каждой составной компоненты будет свое предназначение.

Модель. Это основная логика приложения. Отвечает за данные, методы работы с ними и структуру программы. Модель реагирует на команды из контроллера и выдает информацию и/или изменяет свое состояние. Она передает данные в представление.

Представление. Задача компонента – визуализация информации, которую он получает от модели. View отображает данные на уровне пользовательского интерфейса. Например, в виде таблицы или списка. Представление определяет внешний вид приложения и способы взаимодействия с ним.

Контроллер. Он обеспечивает взаимодействие с системой: обрабатывает действия пользователя, проверяет полученную информацию и передает ее модели. Контроллер определяет, как приложение будет реагировать на действия пользователя. Также контроллер может отвечать за фильтрацию данных и авторизацию [9].

 На рисунке 2.1 представлена логическая схема MVC-приложения:

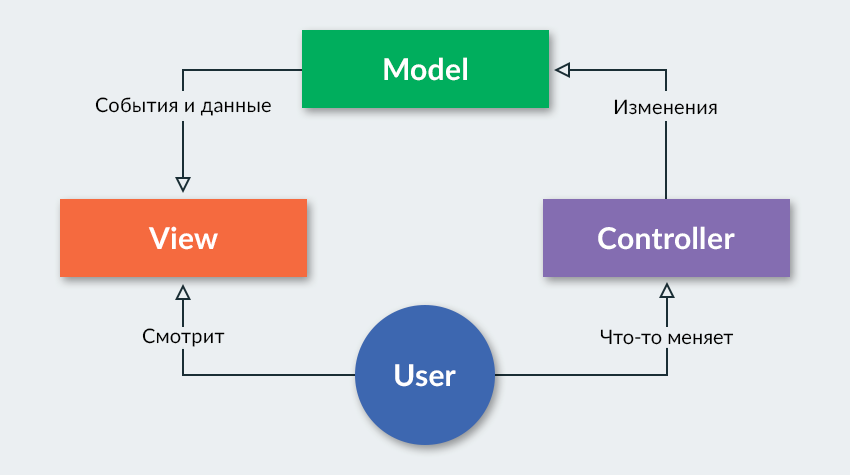


Рисунок 2.1 – Логическая схема MVC-приложения

Внедрение модели помогает упростить код большого приложения, сделать его понятным и структурированным, облегчить поддержку, тестирование, повторное использование элементов. Также модель позволяет организовать независимую работу различных отделов, которые занимаются разработкой своей части программного продукта.

## Описание алгоритмов, реализующих ключевую бизнес-логику разрабатываемого программного средства

 Построение блок-схем алгоритмов, выполняющихся в системе, является также важным этапом в разработке ее модели. Одним из основных алгоритмов, имеющих большое значение для работы пользователя, является алгоритм регистрации и авторизации. При прохождении регистрации и авторизации происходит работа с объектами учетных записей пользователей (рис 2.2).

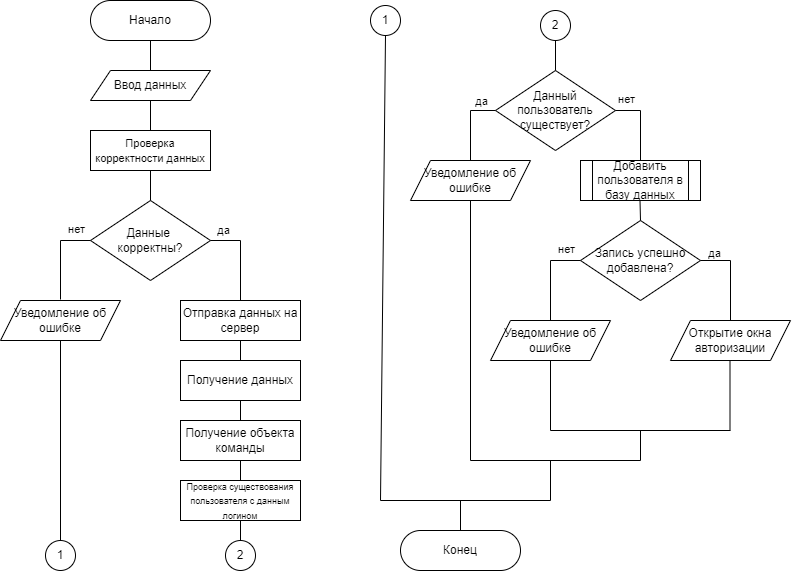


Рисунок 2.2 – Диаграмма активностей

Выполнение алгоритма начинается с ввода всех необходимых для регистрации данных (логин, пароль, адрес электронной почты). Далее идет проверка на корректность ввода, которая включает в себя совпадение паролей при начальном и повторном вводах. Если данные корректны, то вся информация о будущем пользователе отправляется на сервер, где выполняются все необходимые методы по их обработке. Если данные введены неверно, то пользователь получит уведомление об ошибке. После этого введенный логин сравнивается со всеми записями в базе данных, для проверки на существование такого имени пользователя в таблице. Если данный логин является уникальным, то пользователь успешно создал учетную запись. В противном случае он получит уведомление об ошибке.

## Проектирование пользовательского интерфейса

Пользовательский интерфейс – это набор программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером. Основу такого взаимодействия составляют диалоги. Для разработки интерфейса были использованы HTML, CSS.

HTML (или язык гипертекстовой разметки) - это стандартный язык разметки для создания веб-страниц и веб-приложений. Каскадные таблицы стилей (CSS) и JavaScript образуют триаду краеугольных технологий для всемирной паутины [10]. Веб-браузеры получают HTML-документы с веб-сервера или из локального хранилища и преобразуют их в мультимедийные веб-страницы. Он предоставляет средства для создания структурированных документов, обозначая структурную семантику для текста, такого как заголовки, абзацы, списки, ссылки, цитаты и другие элементы. HTML-элементы очерчены тегами, написанными с использованием угловых скобок. Тег сообщает веб-браузеру, где начинается и заканчивается элемент, тогда как атрибут описывает характеристики элемента.

CSS – язык таблиц стилей, который позволяет прикреплять стиль (например, шрифты и цвет) к структурированным документам (например, документам HTML и приложениям XML). Обычно CSS-стили используются для создания и изменения стиля элементов веб-страниц и пользовательских интерфейсов, написанных на языках HTML и XHTML, но также могут быть применены к любому виду XML-документа, в том числе XML, SVG. В таблице стилей мы прописываем стили (правила) оформления. Каждое правило состоит из селектора и блока объявлений. Селектор определяет, на какие части документа распространяется правило. Блок объявлений помещается в фигурные скобки и состоит из одного или более объявлений, разделенных точкой с запятой. Внешние таблицы стилей хранятся в виде файлов с расширением .css. Внутренние таблицы стилей - это инструкции CSS, помещаемые прямо в заголовок конкретной страницы .html. Встроенные стили - это фрагменты CSS, записанные в самом коде HTML.

## Методы и средства, используемые для обеспечения безопасности данных

Лучшие решения по обеспечению безопасности данных включают такие методы защиты данных, как шифрование данных, управление ключами, разделение данных на подмножества и маскирование данных, а также контроль доступа привилегированных пользователей.

Большинство веб-приложений требуют, чтобы их пользователи аутентифицировали себя, запрашивая у них имя пользователя и пароль. Они сравнивают предоставленные пользователем учетные данные с данными, хранящимися в их базе данных, и, если учетные данные совпадают, пользователю предоставляется доступ. По статистике, 55% пользователей сети используют один и тот же пароль для большинства веб-сайтов [11]. Это означает, что, если веб-сайт, хранящий пароль в виде простого текста, будет скомпрометирован, хакер сможет получить доступ не только к учетной записи на этом веб-сайте, но и ко всем учетным записям в социальных сетях, электронной почте. Пароль является важным механизмом аутентификации пользователя, поэтому пароли должны храниться в базе данных в виде хэшей, а не в виде обычного текста. Хэш-функция превращает любой объем данных в «fingerprint» фиксированной длины, который нельзя обратить вспять. «Fingerprint» фиксированной длины называется хеш-значением или хеш-кодом. Уникальная особенность хеш-функций заключается в том, что любое изменение входных данных приводит к совершенно другому хэш-коду. Криптографические хэш-функции используются для реализации хеширования паролей.

Spring Security это Java/JavaEE framework, предоставляющий механизмы построения систем аутентификации и авторизации, а также другие возможности обеспечения безопасности для корпоративных приложений, созданных с помощью Spring Framework. Фреймворк применяется для защиты приложений на Spring. В нем представлены базовые инструменты безопасности, которые без труда расширяются для решения разных задач [12]. Фреймворк Spring Security представляет собой всеобъемлющее решение по обеспечению безопасности, реализующее возможность аутентификации и авторизации как на уровне веб-запросов, так и на уровне вызовов методов, дополнительная интеграция с Spring Web MVC.

# Тестирование и проверка работоспособности программного средства

Тестирование программных продуктов - это процесс проверки соответствия программного продукта его требованиям и ожиданиям пользователей. Это важный этап в разработке программного обеспечения, который позволяет выявлять ошибки и дефекты в продукте перед его выпуском на рынок.

Тестирование может проводиться на разных этапах жизненного цикла программного продукта, начиная от тестирования отдельных модулей до тестирования конечного продукта. В зависимости от цели и типа программного продукта, тестирование может включать в себя функциональное тестирование, тестирование производительности, тестирование безопасности, тестирование совместимости и другие виды тестирования.

Тестирование приложений выполняется с использованием кодов с целью обнаружения ошибок в программном обеспечении. Оно помогает повысить производительность разрабатываемых продуктов, снизить затраты и сократить время производства. Тестирование приложений может затрагивать графический интерфейс, функциональность, базу данных и многое другое. Для проверки корректности работы приложения было проведено тестирование, отражающее возможные исключительные ситуации. Была рассмотрена возможность некорректной авторизации или регистрации с попыткой входа и регистрации без заполнения всех предложенных полей или, когда одно из полей было заполнено некорректно, такая же проверка действует на поля при добавлении друзей и мероприятий (рисунок 3.1).



Рисунок 3.1 – Вход без заполнения всех существующих полей

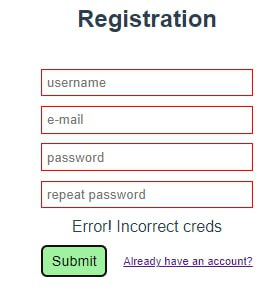


Рисунок 3.2 – Регистрация без заполнения всех существующих полей

# 4 Руководство по развертыванию и использованию программного средства

## 4.1 Руководство по установке (развёртыванию) программного средства

Обязательным требованием, чтобы запустить приложение, является поддержка языка Java с помощью программы Ultimate IDEA, поддержка MySQL Workbench, Node.js. Для установки программного средства пользователю необходимо установить менеджер пакетов npm, входящий в состав Node.js., npm – это инструмент командной строки, который помогает взаимодействовать с онлайн-платформами, такими как браузеры и серверы. Эта утилита помогает в установке и удалении пакетов, управлении версиями и зависимостями, необходимыми для запуска проекта. Чтобы использовать npm, нужно сначала установить node.js, так как они связаны. После установки Node.js пользователь через терминал программного средства устанавливает менеджер пакетов npm, прописав команду npm install.

Далее необходимо развернуть БД на сервере СУБД. Для этого можно использовать MySQL Workbench. Открыв программу, необходимо скопировать скрипт генерации базы данных из приложения В и запустить его. При правильном выполнении скрипта должны появиться зеленые галочки в консоли. После создания базы данных переходим к настройке программы. Скачиваем архив с проектом или клонируем его. После этого мы можем развернуть наш проект, но до этого следует скачать JDK 19 для запуска и для среды открытия проекта IntelliJ IDEA Ultimate. Далее открываем проект в IntelliJ IDEA, выбираем JDK 17 в разделе ProjectStructure->Project->SDK. Проверяем подключение к базе данных. Для этого нажимаем вкладку Database справа в верхнем углу. При успешном подключении база данных отобразится в проекте. После настройки проекта можно приступать к работе. Для этого следует поочередно запустить модули сервера и клиента.

## 4.2 Руководство пользователя

После проведения всех этапов из предыдущего подраздела, пользователь попадает на страницу с входом в приложение, где его просят ввести адрес электронной почты и пароль. Если у пользователя всё ещё нет аккаунта он может зарегистрировать или же войти в аккаунт с помощью Google (рисунок 4.1-4.2).

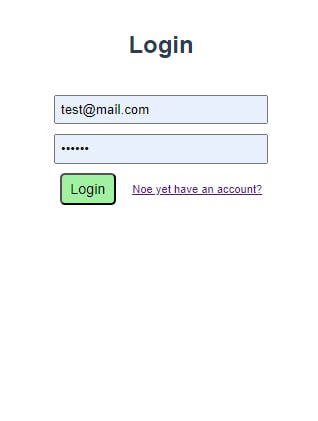


Рисунок 4.1 – Страница входа в приложение

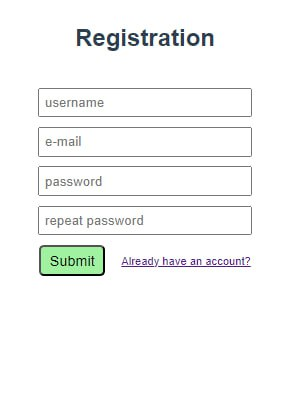


Рисунок 4.2 – Регистрация нового пользователя

После входа в приложение пользователь может при необходимости изменить свои личные данные перейдя на вкладку «Профиль» (рисунок 4.3).

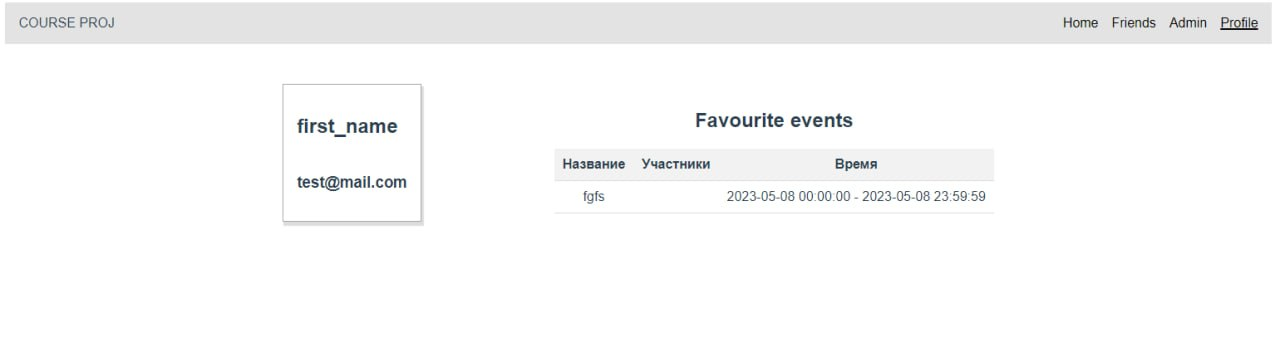


Рисунок 4.3 – Редактирование личных данных

Функционал пользователя включает в себя добавление мероприятия вручную или экспорт данных о мероприятии из Excel. На вкладке «Планер» пользователь может ознакомиться с календарём запланированных мероприятий и при необходимости редактировать данные по мероприятию (рисунок 4.4).

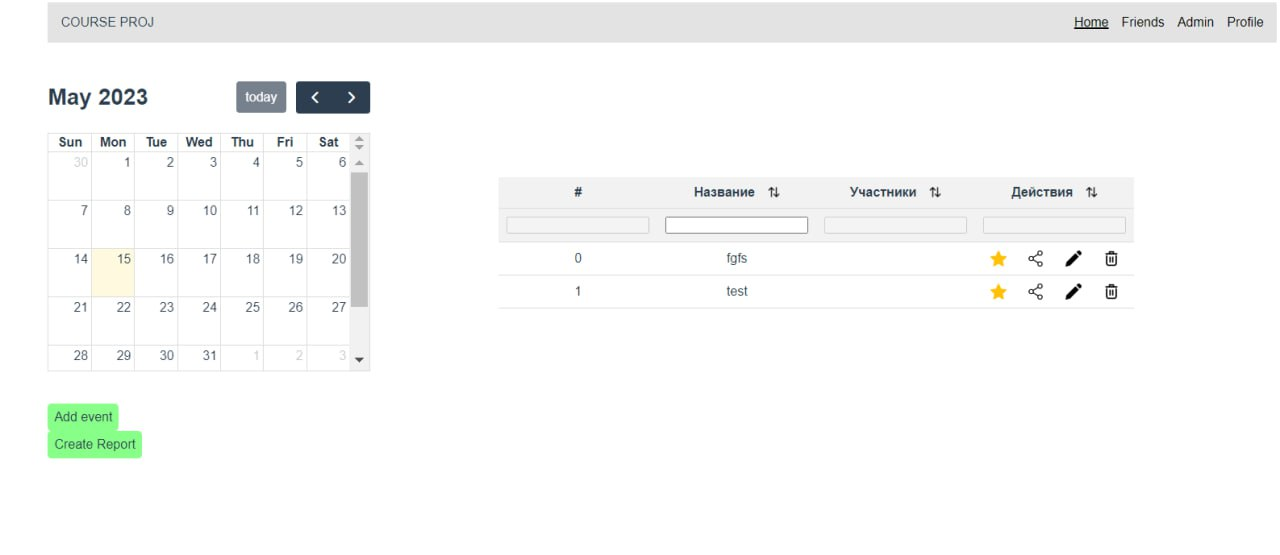


Рисунок 4.4 – Страница с мероприятиями

После добавления мероприятий, предусмотрена возможность добавления необходимого нам мероприятия в категорию «Избранное» и поиск по данной категории (рисунок 4.5).

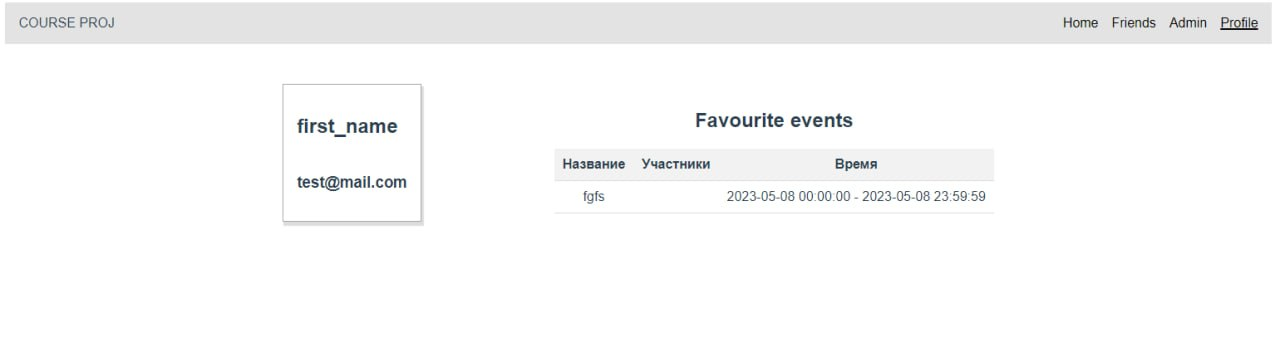


Рисунок 4.5 – Добавление мероприятия в избранные

Предусмотрена возможность добавления друзей. Для этого необходимо заполнить его имя, адрес электронной почты и дату рождения. А также принять заявку в друзья и просмотреть текущих друзей (рисунок 4.6-4.8).

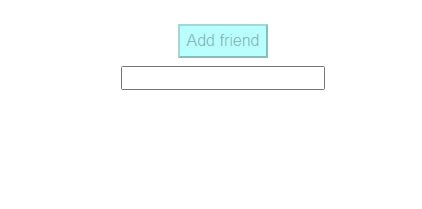


Рисунок 4.6 – Добавление друга

Чтобы добавить новое мероприятие пользователь должен заполнить поля о дате проведения, времени, названии и добавить описание (рисунок 4.9).

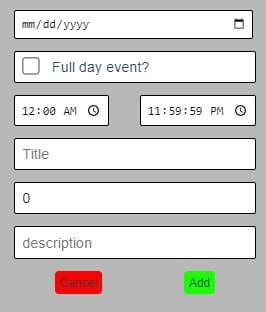


Рисунок 4.9 – Добавление мероприятия

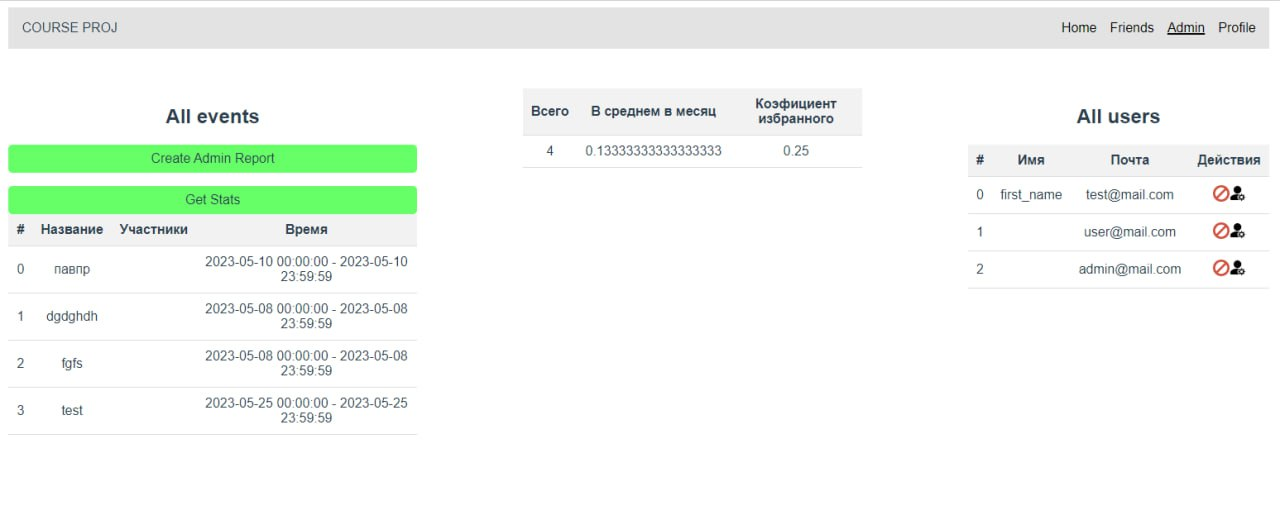


Рисунок 4.9 – Панель Администратора

Администратор имеет возможность блокировать пользователей или делать их администраторами, просматривать статистику по событиям и всем мероприятиям пользователей, а также сделать специализированный отчет рисунок 4.9).

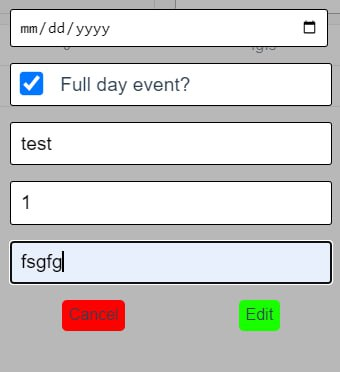


Рисунок 4.10 – Окно редактирования мероприятия

# Заключение

В ходе работы над данным проектом была достигнута поставленная цель, в рамках которой было разработано программное средство для автоматизации работы агентства по организации мероприятий. Данная программа была выполнена в полной мере, что позволит организатору автоматизировать свою работу, в том числе такие операции, как поиск свободных дат, оформление договора, учет полученных платежей, получение отчетов по результатам работы.  Внедрение данной программы позволит компании значительно снизить затраты на оплату труда персонала, увеличить скорость обслуживания клиентов, а также точность и достоверность получаемых данных. Также были подробно изучены особенности внедрения информационных технологий в сферу планирования организации мероприятий, в результате чего выделены основные методы решения поставленных задач.

Полученная информация позволила построить функциональную модель IDEF0 и описать диаграммы UML, с помощью которых было выполнено проектирование системы. С учетом результатов решения предыдущих подзадач осуществлено описание информационной модели системы с доказательством приведения ее к третьей нормальной форме.

Разработанный программный продукт позволяет удобно и надёжно хранить данные и работать с ними. Предусмотрено два уровня доступа к данным. Предусмотрены механизмы поиска, сортировки и просмотра информации в табличном виде.

При разработке данного программного продукта была учтена логика пользователя и проанализированы вероятные сценарии работы программы. Важной особенностью программы является легкая модифицируемость, в дальнейшем это поможет вносить как незначительные, так и большие изменения, и модификации. Это безусловно огромный плюс, ведь без развития любой программный продукт устаревает и приходит в ненадобность. Поэтому созданное программное приложение нацелено на продолжительное использование и открыто для модификаций и успешного развития.

# Список использованных источников

1. Франкель, Н. Event-маркетинг. Всё об организации и продвижении событий. – 1-е изд. – СПб. : Питер, 2017 – 371 с.
2. CRM для event-агенства [Электронный ресурс]. – Режим доступа: https://sbercrm.com/blog/business/tpost/dckm11vuo1-crm-dlya-event-agentstva
3. Программы для event-менеджмента [Электронный ресурс]. – Режим доступа: https://picktech.ru/blog/tips/7-besplatnykh-programm-dlya-event-menedzhmenta
4. Функциональное моделирование на базе стандарта IDEF0. Учебный курс – Минск: 2002 – 35 с.– Режим доступа: <https://studfile.net/preview>
5. База данных как информационная модель предметной области [Электронный ресурс]. – Режим доступа – <https://helpiks.org/8-34437.html>
6. Гради, Б. Язык UML. Руководство пользователя. – 2-е изд. – Москва, 2006. – 483 с.
7. MySQL [Электронный ресурс]. - Электронные данные. – Режим доступа: <https://www.hostinger.ru/rukovodstva/shto-takoje-mysql/>
8. Hibernate ORM. Руководство пользователя. [Электронный ресурс]. – Режим доступа – https://docs.jboss.org/hibernate/Hibernate\_User\_Guide.html
9. Блох, Д. Java. Эффективное программирование – М.: «Лори», 2002 – 224с.
10. Обзор HTML [Электронный ресурс]. – Режим доступа – https://chenweixiang.github.io/2017/04/26/html.html
11. Как сохранить пароль в базе данных [Электронный ресурс]. – Режим доступа –https://www.geeksforgeeks.org/store-password-database/
12. Уоллс, K. Spring в действии. – 3-е изд. – М. : ДМК Пресс, 2013 –752 с.