Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новосибирский государственный технический университет»

Кафедра систем сбора и обработки данных

**Отчет ПО ПРАКТИКЕ**

Производственная практика: преддипломная практика

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование практики в соответствии с учебным планом)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

09.03.02 Информационные системы и технологии

Направление подготовки, профиль: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Воронов

Виталий Владимирович

|  |  |
| --- | --- |
| Выполнил:  Буханцев Максим Олегович  Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(Ф. И. О.)*  АТ-04  Группа \_\_\_\_\_\_\_\_\_\_\_\_\_  АВТФ  Факультет …………….  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *подпись*  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. | Проверил:  Руководитель от НГТУ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(Ф. И. О.)*  Балл: \_\_\_\_\_\_\_\_\_\_, ECTS\_\_\_\_\_\_\_\_\_\_\_\_\_,  Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «отлично», «хорошо», «удовлетворительно», «неуд.», «зачтено», «не зачтено»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *подпись*  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г. |

Новосибирск 2024

**СОДЕРЖАНИЕ**

[1 ВВЕДЕНИЕ 6](#_Toc166436483)

[2 ГЛАВА 1. ИЗУЧЕНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 7](#_Toc166436484)

[2.1 Что такое информационные системы: виды и цели 7](#_Toc166436485)

[2.2 Анализ предметной области 8](#_Toc166436486)

[2.3 Существующие решения 10](#_Toc166436487)

[2.3.1 Bitrix 24 10](#_Toc166436488)

[2.3.2 amoCRM 12](#_Toc166436489)

[2.3.3 1C Фитнес-клуб 15](#_Toc166436490)

[2.3.4 Сравнительный анализ 17](#_Toc166436491)

[3 ГЛАВА 2. ПРОЕКТИРОВНАИЕ СИСТЕМЫ 19](#_Toc166436492)

[3.1 Методология проектирования 19](#_Toc166436493)

[3.2 Используемых технологий 20](#_Toc166436494)

[3.2.1 Язык программирования 20](#_Toc166436495)

[3.2.2 Выбор фреймворка 21](#_Toc166436496)

[3.2.3 Система сборки 22](#_Toc166436497)

[3.2.4 База-данных 23](#_Toc166436498)

[3.3 Анализ бизнес-процессов 24](#_Toc166436499)

[3.4 Варианты использования информационной системы 26](#_Toc166436500)

[3.5 Описание основных функций 40](#_Toc166436501)

[3.6 Модели ключевых классов информационной системы 43](#_Toc166436502)

[4 ГЛАВА 3 РАЗРАБОТКА СИСТЕМЫ. 46](#_Toc166436503)

[4.1 Создание интерфейса системы 46](#_Toc166436504)

[4.1.1 Формы регистрации 46](#_Toc166436505)

[4.1.2 Форма данных спортсмена 49](#_Toc166436506)

[5 ЗАКЛЮЧЕНИЕ 49](#_Toc166436507)

[6 СПИСОК ЛИТЕРАТУРЫ 50](#_Toc166436508)

[7 ПРИЛОЖЕНИЕ 52](#_Toc166436509)

ВВЕДЕНИЕ

В наше время спортивные учреждения сталкиваются с различными проблемами главной из которых является эффективная работа с информацией при ограниченных временных возможностях. Чаще всего тренер спортивного учреждения совмещает в себе роли самого тренера, руководителя спортивного учреждения и менеджера по работе с клиентами. Задача совмещать в себе все эти роли очень трудна и возможность снять часть нагрузки благодаря автоматизации работы менеджера является просто необходимой для спортивного учреждения.

Конкретные проблемы, с которыми сталкиваются спортивные учреждения это: работа с информацией о поступивших звонках, заказчиках и их документах, работа с информацией о спортсменах, учёт оплат, учёт посещений тренировок, план мероприятий и тренировок, и все эти проблемы должен решать один человек, который при всём этом отвечает и за тренировочный процесс, а так же обязан качественно оказывать свои профильные услуги.

Каждая из этих проблем имеет значение для бизнеса, например учёт посещений тренировок. Если не будет учёта, то при ситуации с их непосещением, у нас не будет возможности отреагировать и связаться с клиентом узнать причину отсутствия спортсмена, тем самым показать свою заинтересованность и возможно удержать клиента, который хотел отказаться от услуг спортивного учреждения. Другой пример — это проблема доведения звонков на пробные занятие до посещения учреждения, при отсутствии базы данных поступивших звонков мы лишаем бизнес потенциальных клиентов, при должной обработке этой базы, что на прямую влияет на его успех. Учёт оплаты абонементов с посещениями в ручном режиме будет тратить много времени, которое могло бы эффективно расходоваться на другие задачи. В информационной системе контроль этих процессов можно автоматизировать и свести время, затрачиваемое на эти процедуры, к минимуму.

Целью работы является реализация информационной системы автоматизирующей функции сказанные выше. Основные требования к системе: минимизация времени, затрачиваемого на административные процессы, удобный и простой интерфейс.

Для реализации указанной цели требуется решить следующие задачи:

* Проанализировать бизнес-процессы предметной области
* Проанализировать существующие решения
* Спроектировать программное решение в соответствии с требованиями к системе:
  + Создать модель бизнес-процессов
  + Формализовать требования к системе
  + Создать диаграммы вариантов использования
  + Создать диаграммы деятельности
  + Создать диаграммы взаимодействий
* Выбрать стек технологий для решения задачи
  + Язык
  + Фреймворки
  + Базу-данных
* Реализовать используя выбранные технологии и спроектированные модели прототип приложения.

ГЛАВА 1. ИЗУЧЕНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

**Что такое информационные системы: виды и цели**

Понятие информационная система довольно широкое понятие и необходимо внести ясность почему к разрабатываемой мной программе относиться данное определение.

Как говориться в [ISO](https://ru.wikipedia.org/wiki/ISO)/IEC 2382:2015 [1] информационная система – это система обработки информации совместно с соответствующими организационными ресурсами (человеческими, техническими, финансовыми и т. д.), которая обеспечивает и распространяет информацию [2].

И цель создания информационных систем — это обеспечение определённых людей необходимой информацией, для решения потребностей определённой предметной области, а результатом функционирования информационных систем является информационная продукция – которая представляет собой документы, информационные массивы, базы данных и информационные ресурсы [3].

В разработанной системе происходит обработка информации и её помещение в базу данных, для использования данных с целью улучшения качества оказываемых услуг в спортивном учреждении и выявления, и реализации конкурентных преимуществ для малого и среднего бизнеса.

Было рассмотрено отношения правового аппарата к информационным системам. Система относиться к информационной системе для обеспечения типовой деятельности, к которой относятся такие системы как система бухгалтерского учета, справочные и правовые системы, официальные сайты организаций, иное. Система является информационной системой персональных данных согласно [4].

В системе содержаться Ф.И.О. и контактные данные сотрудников и пользователей. А так же система является корпоративная информационная системой и информационная система общего пользования согласно [5] одновременно, ведь доступ к определённой и информации такой как оплата абонементов будет доступен только пользователям с определённым уровнем доступа, а информация о расписании тренировка будет обще доступная.

Классификацию информационной системы спортивных учреждений для дальнейшего проектирования.

По архитектуре система является клиент-серверной так как пользователи отправляют запросы с требуемой информацию на сервер, на котором работает СУБД, какая именно будет сказано далее в пункте её выбора. В свою очередь клиент-серверные информационные системы можно разделить на двухзвенные и многозвенные в нашем случае это будет многозвенная структура, так как прежде, чем клиент обратиться к СУБД он будет обращаться к веб-серверу.

По степени автоматизации система, это автоматизированная информационная система, ведь персоналу такой как тренер необходимо вносить правки в расписание и вести учёт спортивных успехов учеников, к сожалению, вручную. Как пример бедующего усовершенствования, добавление на грамоты QR кодов, которые можно просто отсканировать и занести результаты соревнований в системы.

Так же система информационно справочная, она выдаёт информацию в удобном виде, который бы позволял извлекать из неё пользу. В системе нет обработки данных сложными алгоритмами, так одна из преследуемых целей информационной системы для спортивных учреждений — это простота в использовании и уменьшения затрат персонала на решение повседневных, обыденных задач, которые в основном приставляют из себя рутину ответов по расписанию и ведения журнала присутствующих с обратной связью для родителей.

Предметная область спортивные учреждения и организации и решение их проблем.

**Анализ предметной области**

Спортивная организация - юридическое лицо независимо от его организационно-правовой формы, осуществляющее деятельность в области физической культуры и спорта в качестве основного вида деятельности, в том числе имеющее право на оказание физкультурно-оздоровительных услуг. Положения настоящего Федерального закона, регулирующие деятельность физкультурно-спортивных организаций, применяются соответственно к индивидуальным предпринимателям, осуществляющим деятельность в области физической культуры и спорта в качестве основного вида деятельности, в том числе оказывающим физкультурно-оздоровительные услуги [6].

Деятельность спортивных же организаций представлять из себя построение спортивной подготовки и его выполнения с целью достижения индивидуальных спортивных результатов спортсменов, организация и проведение спортивных мероприятий и соревнований, в рамках одной спортивной дисциплины [7].

Рассматривая эти определение было выделены и реализованы такие средства как, организации спортивной подготовки, средства планирование, добавления спортивных мероприятия и соревнования, которые является неотъемлемой частью спортивной подготовки, учёт спортивных достижений спортсмена на мероприятиях и соревнованиях, реализованы средства отслеживания подготовки спортсмена, такие как учёт посещения тренировок и возможность отображения результатов на тренировке.

Классификации спортивных организаций разбивается на несколько пунктов:

1. Размер спортивных организаций;
2. Форма собственности;
3. Принципы распределения прибыли;
4. Цели создания и функционирования

1. Спортивные организации делятся на малые 50 человек, средние 100 человек и крупные более 300 человек. Спроектированная система создана для малых и средних спортивных организаций, в ней реализованы средства и инструменты, которые позволяют, реализации конкурентных преимуществ в сфере оказания индивидуальных специализированных услуг в рамках спорта. Данные функции позволяют оказывать индивидуальный подход в отличии от крупных организаций, информационные системы которых направленны на эффективную продажу услуг как можно большему количеству людей, без оказания персонализированного подхода.

2. По форме собственности спортивные организации делятся на государственные, муниципальные и частные. В созданной информационной системе учтена специфика работы частных и муниципальных организаций, которые как раз и относятся к малым и средним спортивным организациям.

3. По отношению к прибыли, коммерческие и некоммерческие. Так как упор в разработанной системе был сделан на малый и средний спортивные организации, самым распространённым отношением к прибыли в таких организациях является коммерческое отношения, в связи с этим в системе реализованы такие функции как отслеживание оплаты за месяц или иной период, а также других взносов таких как членских взнос или взнос за выступления на турнире.

4. Целью создания и функционирования является основным для понятия различий в спортивных организациях.

* Организации, осуществляющие учебно-массовую и физкультурно-спортивную работу;
* Организации, организующие возможность занятий спортом спорт площадки, парки;
* Организации, обеспечивающие медицинское обслуживание;
* Организации по подготовки физкультурных кадров;
* Организации по обеспечению сферы физической культуры и спорта необходимой информацией и проводящие пропаганду.

Все вышеперечисленные виды организаций предоставляют большой спектр спортивных услуг, направленных на развитие физического здоровья как детей, так и взрослых, вовлечения в спорт и здоровый образ жизни. Так как для спортивных организаций разработано не так много специализированных решений, а специфика работы в довольно большая. Я разработал информационную систему, которая по моему мнению могла бы помочь своим функционалом решить проблемы, возникающие перед спортивными организациями.

Для решения проблемы с набором детей и текучкой. Была создана база клиентов и звонящих.

Проблема контроля ежемесячной оплаты и оплаты мероприятий и соревнования. Реализована функция создания эвента оплаты и её отслеживания.

Проблема с информированием клиентов о различных мероприятиях и требуемых для участия в них условиях. Реализована функция информирования клиентов мероприятиях и отслеживания выполненных условия для участия в них. А также возможность отслеживания заявок для тренеров и выполнения условий.

Проблема с отслеживанием успехов спортсменов на соревнованиях. Реализована база и интерфейс для просмотра результатов участия на мероприятиях и спортивных соревнованиях.

Проблема с отслеживанием посещения тренировок и успехов на них. Реализован интерфейс, в котором можно увидеть количество посещённых тренировок и когда, это может видеть как клиент-родитель, так и тренер.

Проблема с планированием тренировок, мероприятиями и соревнованиями. Реализован интерфейс в виде календаря в котором можно расписать тренировки и мероприятия, и он будет виден как тренеру, так и клиентам, при изменении или добавлении в который будет информироваться клиенты.

Решение всех этих проблем и рутины занимает большое количество времени, а также часть данных не эффективно используется или вовсе не собирается. Теперь же благодаря реализованной информационной системе, работа малых и средних спортивных организаций станет более простой и эффективной [8].

**Существующие решения**

**Bitrix 24**

Одним из вариантов решения задач разработки приложения является использование готовых решений, рассмотрим первое из них Bitrix24 [9].

Все продукты компании Bitrix представленные на рисунке 1 выполнены на основе Bitrix Framwork который представляет из себя некоторую платформу, созданную на языке PHP. Из которой уже и строятся готовые продукты компании.

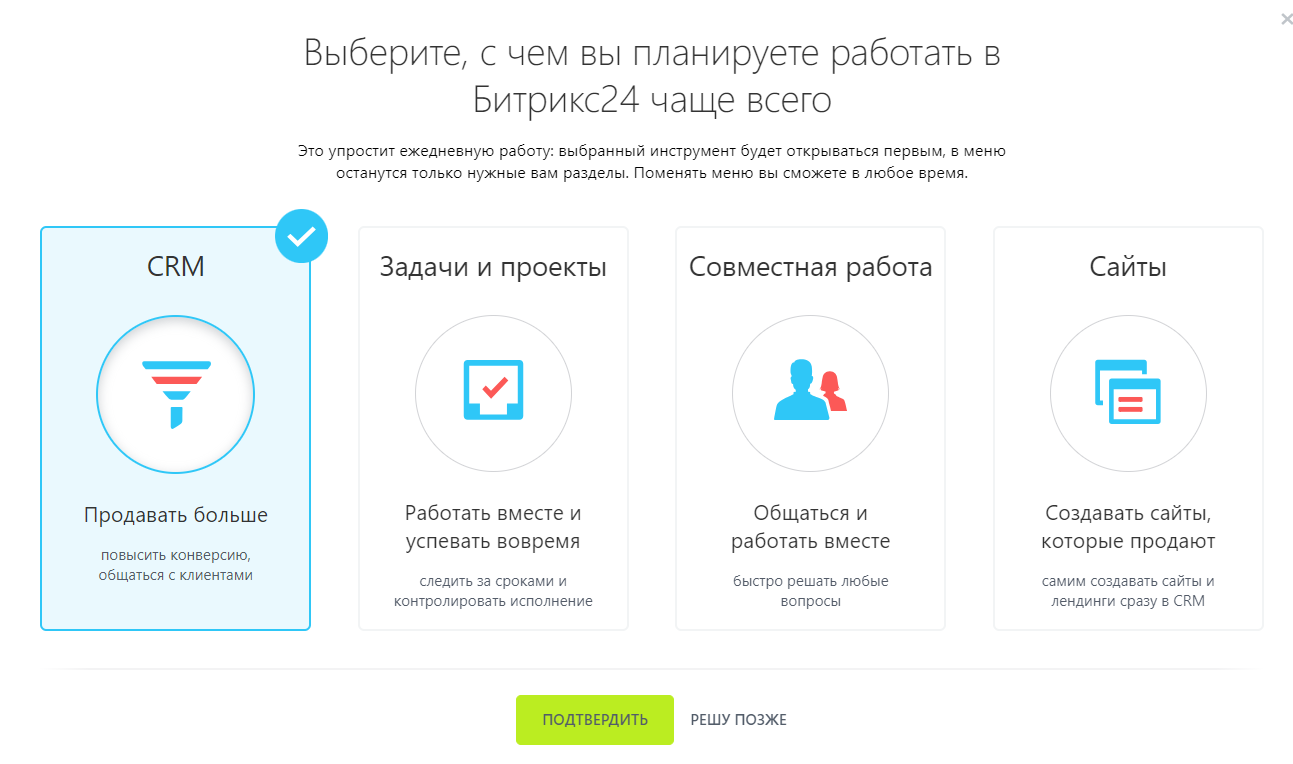


Рисунок 1 – Продукты компании Bitrix

Рассмотрим по подробнее CRM систему предлагаемую Bitrix24

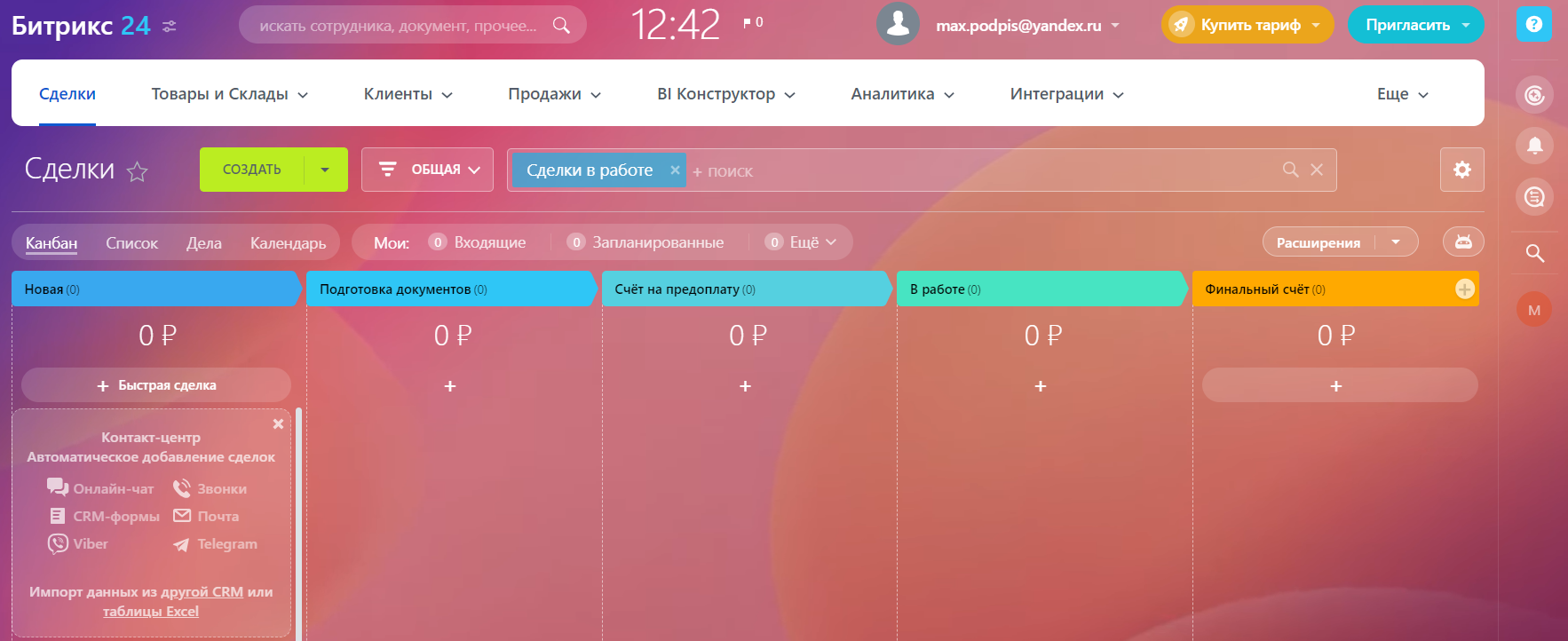


Рисунок 2 – Рабочий стол Bitrix

Уже в бесплатном варианте мы можем ознакомиться с основным функционалом системы представленным на рисунке 2 на главном рабочем столе, однако отдельные его части открываются при покупке подписки.

Программные продукты Bitix обладают большим функционалом, и возможностью интеграций с множеством русскоязычных программных решений приставленных на рисунке 3.



Рисунок 3 – Примеры платных расширений для Bitrix

Однако в платных решения просматривается тенденция, когда тебе необходима одна конкретная функция, но она идёт с в комплекте с другим, не нужным тебе, за которые ты ещё и заплатишь с ценообразованием можно ознакомиться на рисунке 4.

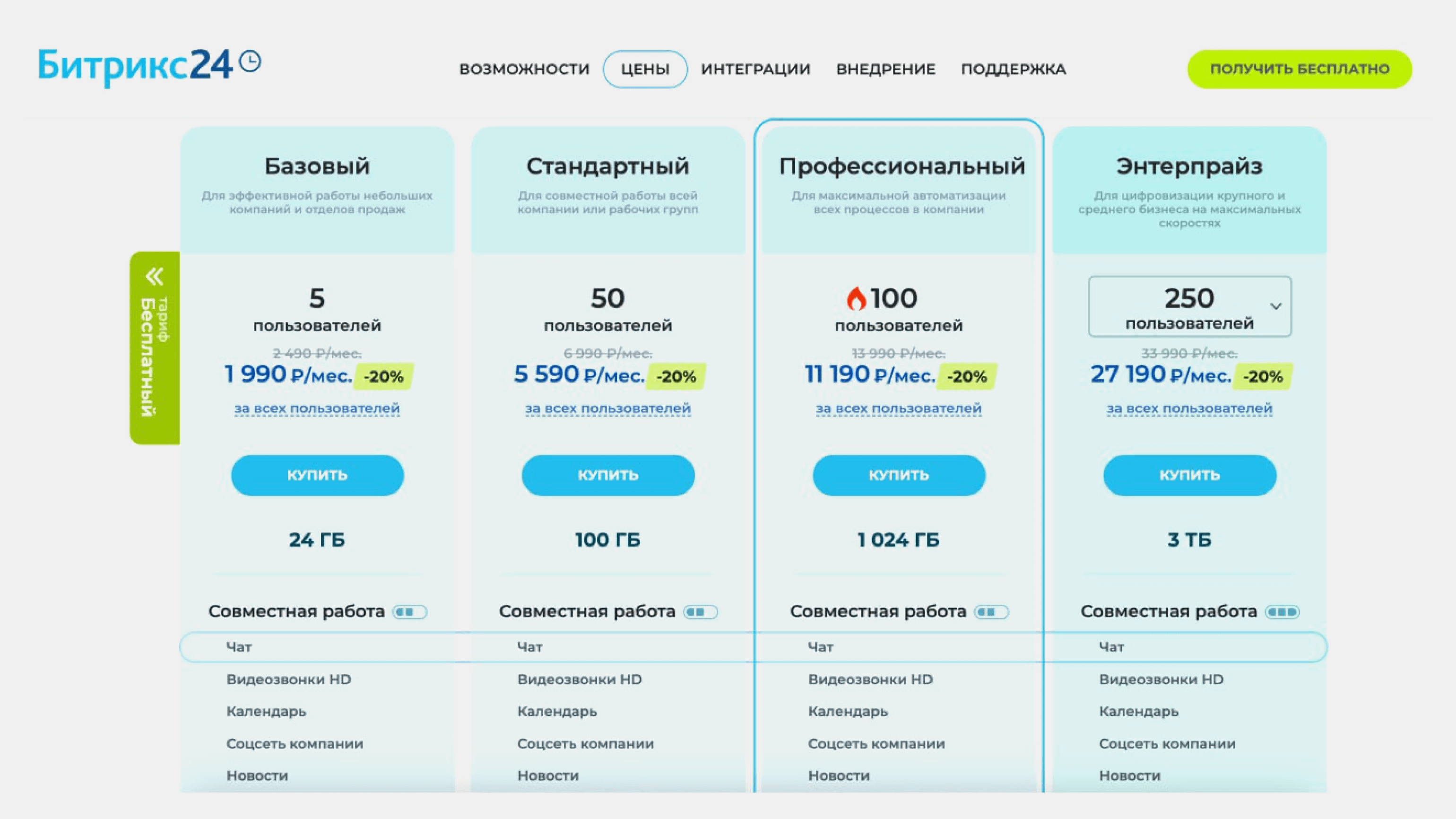


Рисунок 4 – Тарифные планы компании Bitrix

Из плюсов можно выделить продукция является русскоязычной и документация также вся на русском языке, и огромное количество интеграций с документацией так же на русском языке. Минусы сложность в настройке и управлении программным продуктом, что ведёт к необходимости оплаты профессионала по настройке, доплаты за интеграции излишний функционал со старта, когда он может быть тебе не нужен [10].

**amoCRM**

Следующая информационная система называется AmoCRM [11]. При первом знакомстве необходимо зарегистрироваться на сайте по форме представленной на рисунке 5 и сразу же провести первоначальную настройку, вы получаете 14 дней тестового полного доступа к системе. Данный ход является хитрим, так как в полной версии мы знакомимся с некоторым функционалом, который можем считать базовым, но после перехода на базовую версию функционал, который был доступен в полном доступе тестового режима нет.

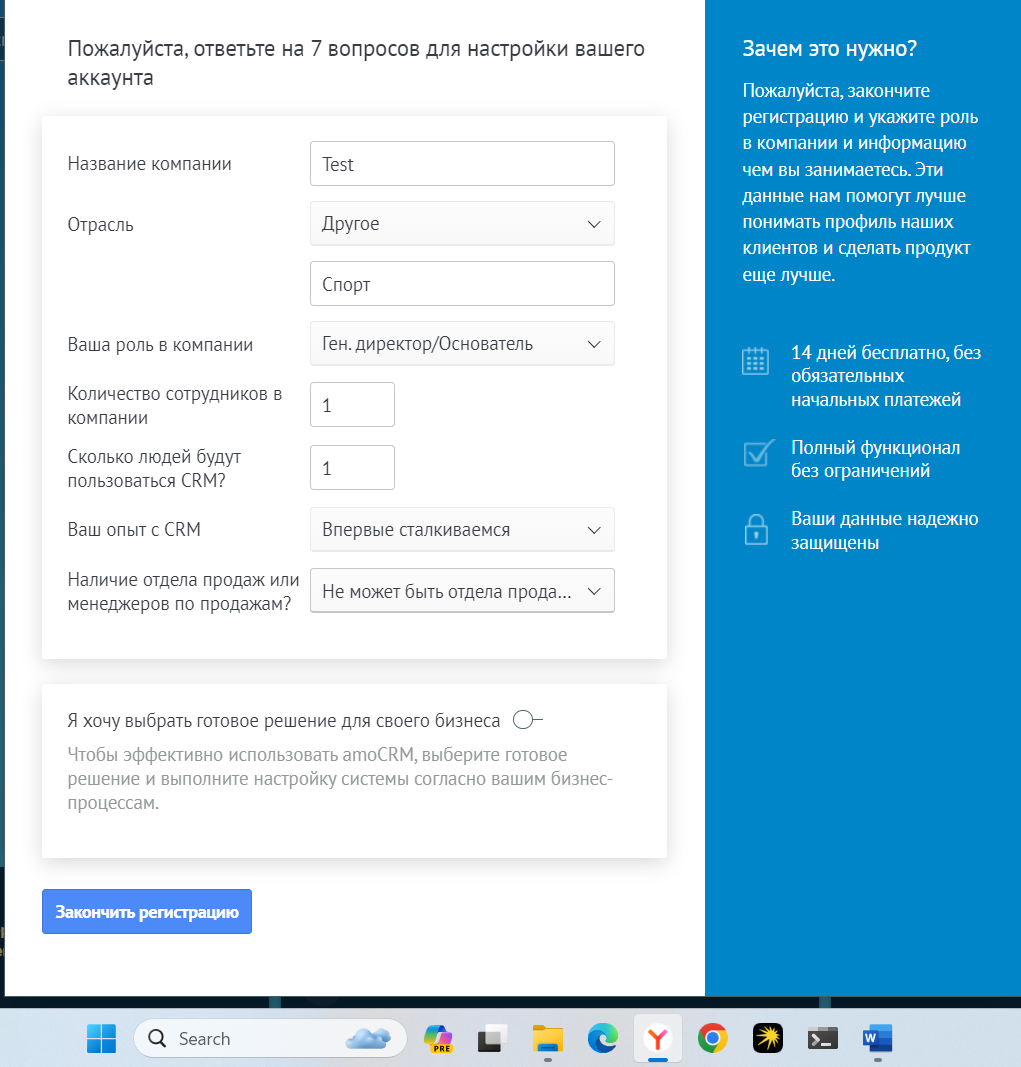


Рисунок 5 – Стартовый экран компании AmoCRM

Система схожа по функционалу с Bitrix но, если Bitrix имеет под собой одноимённый фреймворк на основе которого как из блоков можно строить информационные системы для решения различных задач связанных с продажами и работой с некоторой информацией. В AmoCRM является более узко направленной информационной системой она концентрируется на продажах, аналитике, интеграцией телефонией с всем представленным функционалом можно ознакомиться на рисунке 6.

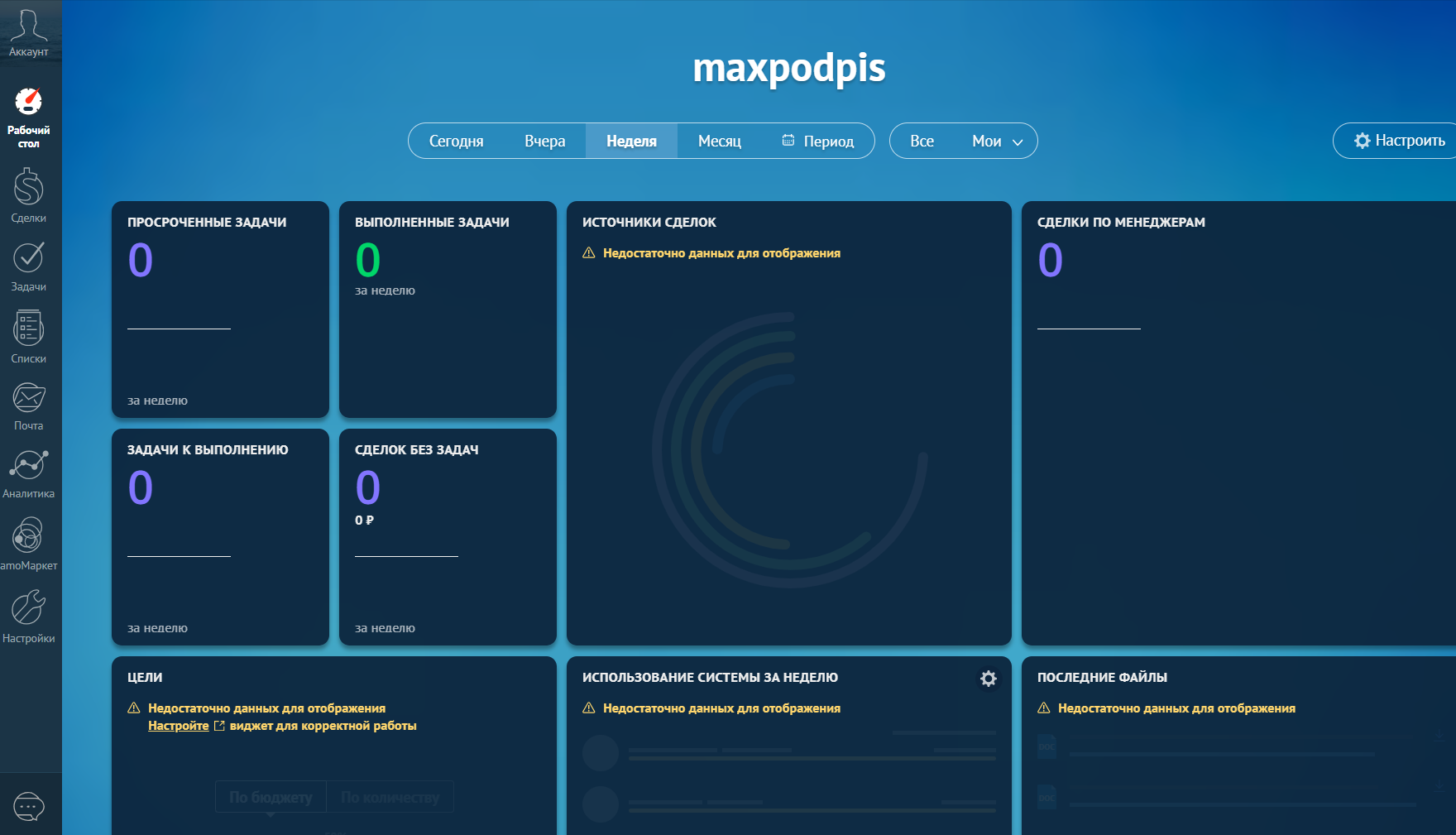


Рисунок 6 – Рабочий стол компании AmoCRM

Такая система проста и подойдёт малому и среднему бизнесу и так же как и в Bitrix есть интеграции с другими программными решениями которые можно увидеть на рисунке 7 в меню настройки.

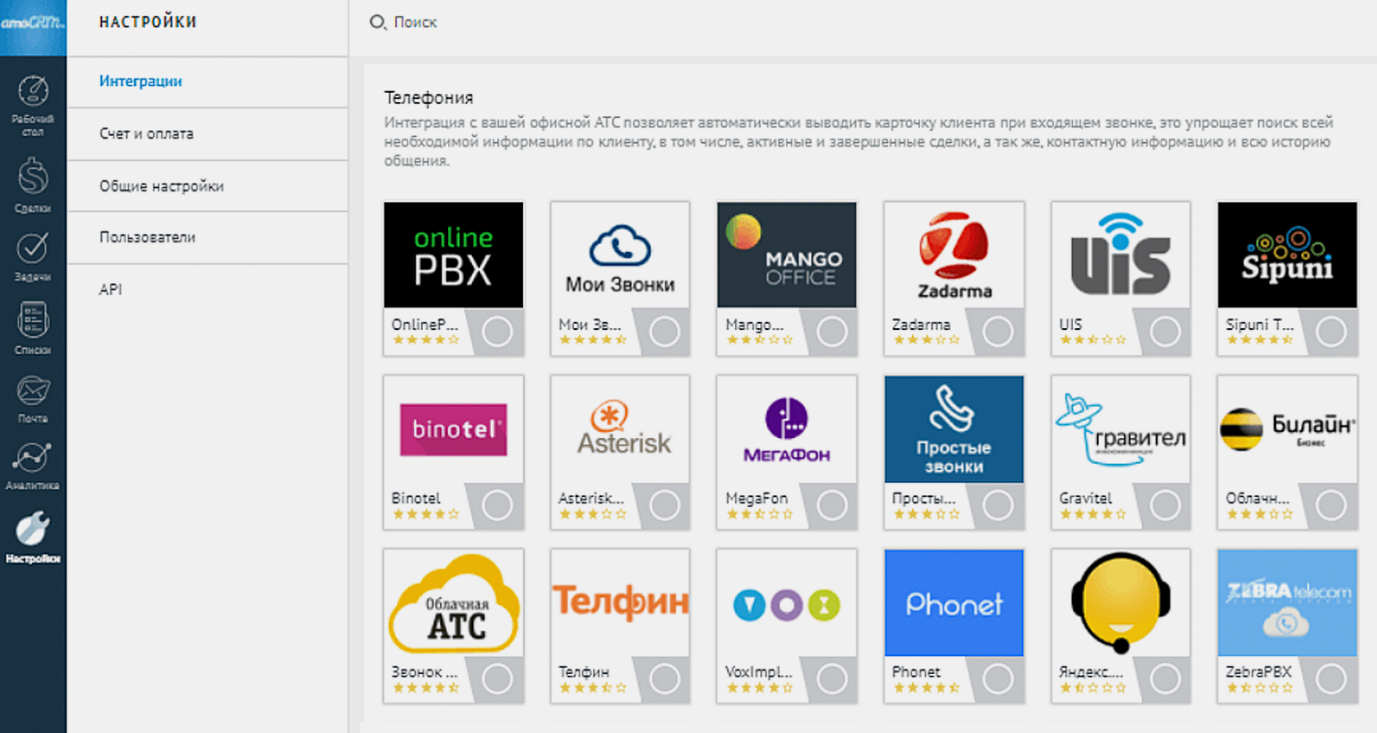


Рисунок 7 – Расширения компании AmoCRM

Однако и эта система не лишена минусов, в ней нет функциональности управления подпроцессами и параллельными бизнес-процессами, по этому бизнесу с сложной воронкой продаж данная система не подойдёт, а также нету функциональности множественного выполнения и фиксации результата, например спортивной организации это система абонементов, каждый месяц пришлось бы создавать отдельные продажи под каждого пользователя, а также необходимо отслеживание оплаты соревнований и членских взносов. Так как реализованная эта структура в AmoCRM не подходит и пришлось бы совершать излишнюю работу и усложнять работу с программой. Стоит отметить и то, что в базовой версии нет некоторых интеграций с привычными сервисами такими как “Яндекс Метрика” и “Google Таблицы” за них необходимо доплачивать. А так же некоторая работа в базовом тарифе не автоматизирована, как пример работа с карточками в разделе “Сделка”, в базовом тарифе перемещение карточек между колонками нужно производить в ручную и создание клона существующей карточки так же не возможно данный подход к системе организации функционала системы можно ознакомиться на рисунке 8 [12].

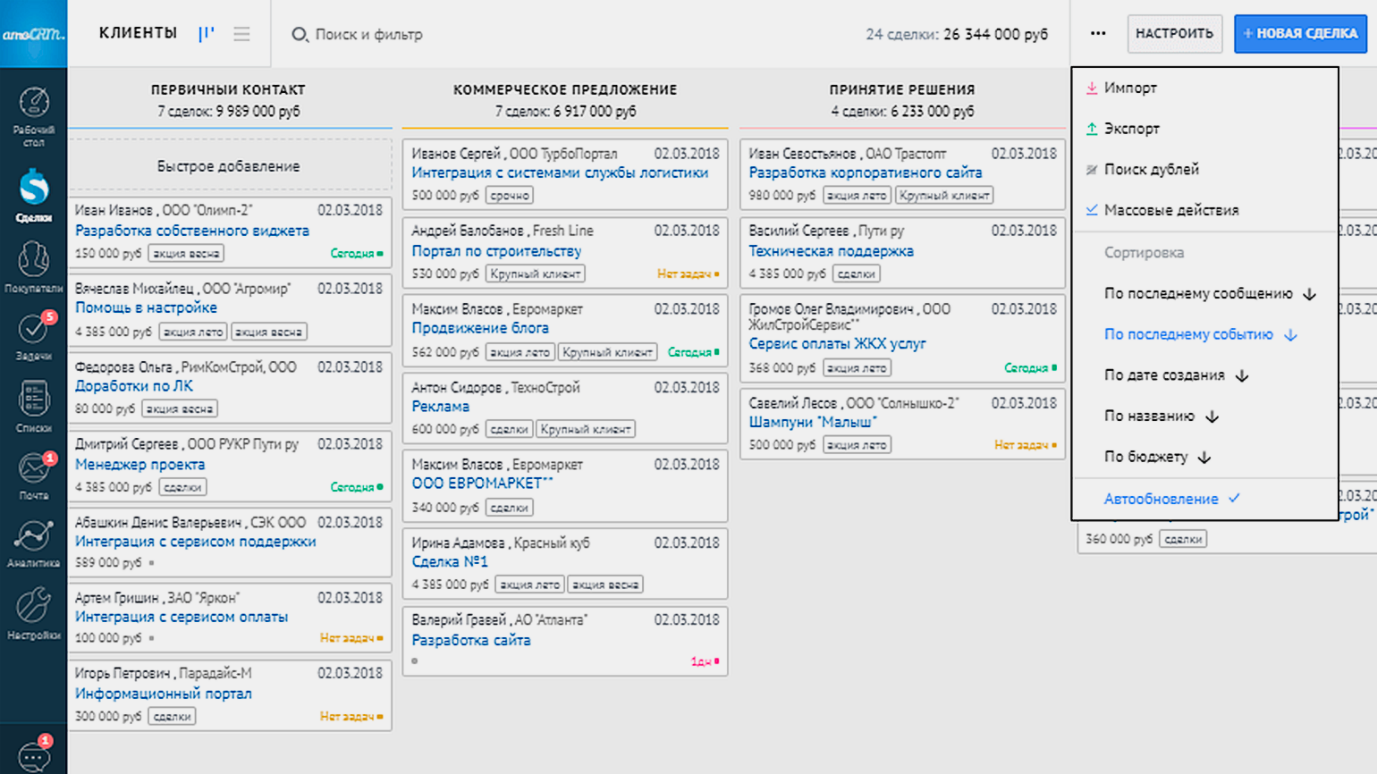


Рисунок 8 – Пример карточной системы в AmoCRM

**1C Фитнес-клуб**

Из всех систем 1С Фитнес-клуб [13] может решить больше всех задач, стоящих перед спортивными организациями со всеми этими решениями, можно ознакомиться на рисунке 9. В данной системе нет такого сильного перекоса в продажи как у систем, представленных ранее, но данный модуль тоже имеется, и он может предложить большую, я бы даже сказал в каких-то случаях, например для малого и среднего бизнеса излишнюю функциональность часть из которых можно увидеть на рисунке 10. Данное решения обладает всеми теме же интеграциями что и другие системы, а также уникальные, собственные такие как 1С Бухгалтерия, что для крупных организаций является огромным плюсом. Техподдержка, а также курсы по обучению системе предлагаемые компанией 1С позволят освоит данную программу, а при проблемах быстро их решить [14]. Конечно, за всё это приходиться платить, скорость обработки ваших вопросов зависит от вашего тарифа, а обучение для компании не является бесплатным.



Рисунок 9 – Преимущества решения 1С Фитнес-клуб

Так же эта система уже может предложить возможности отслеживания действий персонала и его работы. Что является плюсом

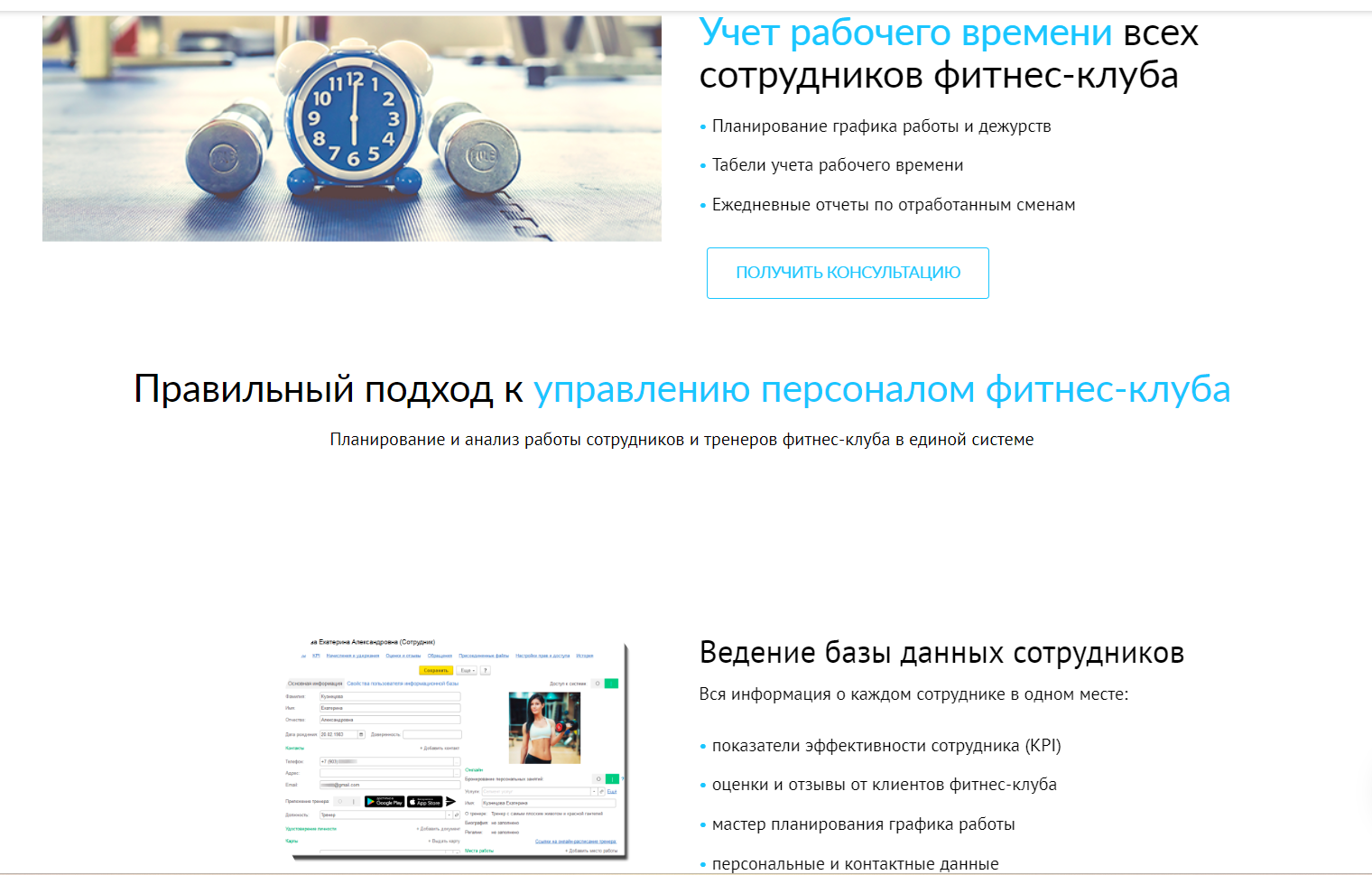


Рисунок 10 – Особенности системы 1С Фитнес-клуб

А также есть отдельные пакетные решения для различных спортивных организаций как пример показано пакетное решения для школы единоборств на рисунке 11.



Рисунок 11 – Пакет решений 1С Фитнес-клуб

Однако всё же при всей своей избыточности и сложности в освоении, подержании и обучении данные решения ориентированы на большой бизнес, и некоторые особенности, которые существуют в малом и среднем не реализованы, так как для массового потребителя, например в фитнес зале они являются избыточными и ненужными. В малом же и среднем бизнесе эти бы решения такие как наличие карточки спортсмена и отметки посещений, позволят оказывать более индивидуальный подход к заказчику что является конкурентным преимуществом и большим таких небольших организаций [15].

**Сравнительный анализ**

С учетом всего выше сказанного про информационные системы различных компаний, проведём сравнение между собой и выделим то, что необходимо реализовать в нашей системе. Чтобы она являлась уникальным решением, которое помогало автоматизировать задачи малым и средним спортивным организациям и позволяло реализовывать их конкурентные преимущества.

Сравнение произведено на примере задач, которые поставлены в ходе работы.

Таблица 1 – Сравнение информационных решений

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1С Фитнес-клуб | Bitrix 24 | amoCRM |
| Информация о поступивших звонках | + | + | + |
| Информация о заказчиках | + | - | - |
| Информация о спортсменах | - | - | - |
| Учёт посещений | - | - | - |
| План тренировок | + | - | - |
| План мероприятий | - | - | - |
| Учёт финансов | + | + | + |
| Акции | + | - | - |

Рассмотрены 2 наиболее популярные на российском рынке CRM системы Bitrix 24 и amoCRM, а также конфигурация 1С Фитнес-зал.

CRM системы хорошо выполняют свои профильные задачи, которые связаны с обработкой заказов через заявки на сайте или другими способами. Так же у них есть возможность отслеживания оплаты и хранения информации о заказчиках. Однако данный функционал не полностью соответствует требованиям учреждений выполняются они в ограниченном формате, информация о заказчике представляет собой только контактные данные, без хранения документов необходимых для занятия спортом, оплата фиксируется как за оказанную услугу, без абонементной системы и учёта посещений [16].

Лучшая ситуация с решением от компании 1С, их конфигурация фитнес-зал почти полностью удовлетворяет задачам, однако и у неё есть недостатки. Отсутствует, разделение заказчика и спортсмена. В фитнес-залах как правило заказчик и спортсмен один и тот же человек, в спортивном же учреждении обучаются в основном дети, заказчиками которых являются родители. Следовательно информация заказчика и спортсмена, может принадлежать, разным людям. Ещё одним минусом является отсутствие возможности планирования мероприятий таких как спортивные соревнования, это не тренировка, где необходима функция отметки посещений, а мероприятие, перед которым необходимо собрать пакет документов, при этом каждый документ является проверяемым пунктом чек листа и имеет сроки. Сложность в обращении с данной системой является не маловажным фактором почему данный вариант нельзя выбрать как решение нашей проблемы, необходимость настройки и поддержки системы в рабочем состоянии будет занимать время пользователя, что не является нашей задачей. Согласен что в некоторых моментах данное решение является более профессиональным, так как модуль 1С Бухгалтерия с которым интегрируется 1С Фитнес-зал обладает большим функционалом. Но для нашей задачи такая глубина и сложность излишняя, напомню что у нас спортивное учреждения в котором роль управленца, менеджера по работе с клиентами сочетает в себе тренер и простата используемой программы играет не последнюю роль, ведь траты времени на освоение 1С и различных её модулей с постоянной работой с ней и поддержании её в рабочем состоянии требует большого вложении временного ресурса [17].

В ходе исследования решений других компаний, я пришёл к выводу что ни одно из решений полностью не решает поставленной цели и задач. Одни из них концентрируются на узко специализированных областях, другие же наоборот являются излишними и сложными в настройке и обслуживании более направленнее на решение задач больших компаний. В созданной информационной системе взяты лучшие части от каждой из систем, в какой-то степени упрощены. Сделано это для эффективного выполнения поставленных задач, и решения проблем спортивных организаций, а также помощи в реализации конкурентных преимуществ малого и среднего бизнеса. В реализованной системе был сделан упор на простоту и эффективность, чтобы обучение использовании информационной системой было простым, а также лёгким в обслуживании и содержании, что не как бы не отразилось на её эффективности. В системе было усовершенствована работа с информацией о спортсменах и заказчиках, с позиции клиентоцентричности. А также сочетание функций менеджера и тренера одним человеком.

ГЛАВА 2. ПРОЕКТИРОВНАИЕ СИСТЕМЫ

**Методология проектирования**

Методология, применяемая при разработке данной информационной системы инкрементно-итерационная. Данная модель является для создания данной информационной системы благодаря сочетанию плюсов данных подходов [18].

Плюсы:

* Простота контроля разработки. Созданный минимальный жизнеспособный продукт позволяет отразить все этапы в дипломной работе на всех этапах его написания.
* Получение первичного отклика от пользователей системы, для формирования гипотез для дальнейшего развития продукта.
* Тесная обратная связь, которая позволяет вносить необходимые функции и удалять избыточные не используемые функции, перегружающие интерфейс.
* Отсутствие необходимости больших затрат на тестирование. Дипломная работа — это проект одного человека и этот плюс весомый, так как временные затраты уделяемы на тестирование не такие большие.
* Тестирование минимальной версии системы происходит сразу.

Минусы:

* Отсутствие фиксированного бюджета и сроков. Сложно определить этап, на котором система будет полностью готовой, всегда есть идеи и чем дополнить. В связи с этим сложно сочетать разработку и написание диплома.
* Полностью выполненная работа, далека так как говорилось ранее всегда есть что добавить или изменить.
* В процессе разработки написание технической документации, которая является основой диплома уделяется мало времени.

Несмотря на наличие минусов данная методология по моему мнению лучше всего подходит для проектирования информационной системы для спортивных организаций, так как сфера достаточна нова с точки зрения внедрения информационных систем и требования для системы формируются по ходу её создания и эксплуатации, проверки бизнес-гипотез [19]. Инкрементно-итерационная модель показана на рисунке 12.



Рисунок 12 – Структура “Инкрементно-итерационной методологии”

* Планирование и требования – это определение и постановка задач, которые должна решать данная система, анализ схожих систем.
* Проектирование – это определение какие технологии будут использоваться в системе, анализ бизнес-процессов, описание вариантов использования, описание основных функций и написание на основе всех анализов моделей ключевых классов и диаграмм взаимодействий.
* Разработка – это воплощение спроектированного приложения с помощью выбранных инструментов.
* Тестирование – это проверка работы созданной информационной системы, на вывод корректных данных и удобство взаимодействия с ними.
* Оценка – использование системы спортивной организацией с выявлением проблем и новых требований [20].

**Используемых технологий**

Технологий, с помощью которых выполнена система, с точки зрения выбора и соответствия поставленным задачам и для решения проблем. Были рассмотрены такие языки программирования как Python, C++, PHP, Java. А также их фреймворки и библиотеки специализированные для решения поставленной задачи.

**Язык программирования**

Python популярный язык, с преимуществами в виде простаты в изучении и наличии большого количества библиотек для аналитики данных. Однако он по-прежнему является не таким производительным языком как C++ и Java. Стоит учитывать и перспективу дальнейшей разработки, да создание простейшего прототипа на python намного легче чем на C++, однако в дальнейшем может возникнуть потребность в переписывании ранее созданного кода и добавление функциональности, или добавление мобильной платформы. При работе с такими требованиями лучше выбрать другой язык программирования. Популярность данного языка у начинающих может послужить дурную службу, так как найти опытного разработчика сложно, и при текущей ситуации на рынке с трендом на завышение опыта может не повезти на человека, преувеличившего свой опыт и некомпетентного в критичных ситуациях при разработке.

С C++ другая ситуация, преимущество языка в виде скорости скомпилированных программ, нивелируется сложность работы с ним и вероятностью ошибок, и их критичностью. Некоторая свобода, которую даёт данный язык программирования требует большого профессионализма, и строго наказывает за оплошности в коде, а хорошие профессионалы стоят дорого. Создание же прототипа на данном языке, выполняется дольше и сложнее чем на других. Данный язык хорош, когда у вас уже есть команда профессионалов готовых к экспериментам и интересным задачам. В других случаях он является не выигрышным вариантом.

PHP прекрасный язык на котором уже реализовано большое количество схожих информационных систем, тот же самый Bitrix Framwork написан на данном языке. Однако я хотел бы иметь возможность простой разработки под мобильные платформы так как сейчас смартфон стал неотделимой часть для большинства людей. Данный же язык программирования не силён в подобной стезе. Несомненно, при разработке я буду рассматривать примеры информационных систем с открытым исходным кодом, созданных на PHP и заимствовать удачные решения.

Java именно на этом языке я собираюсь остановиться так как он не настолько сложен в освоении как C++, проблемы быстродействия не такие большие как у python, создание прототипа с большим количеством помарок не столь вероятно так как язык строго типизирован, а проблем с мобильной разработкой как у PHP нет, ведь java используется для разработки приложений под Android.

Ведь проблема универсальности программных кодов решена в Java в рамках концепции

виртуальной Java-машины (JVM, от Java Virtual Machine). С одной стороны, это позволяет создавать достаточно универсальные программы (в том смысле, что они могут использоваться с разными операционными системами). Но, с другой стороны, платой за универсальность является снижение скорости выполнения программ, однако оно не столь велико как в python. А также я обладаю знаниями достаточными для реализации данного проекта.

**Выбор фреймворка**

Выбор языка программирования — это ещё не всё ведь реализация информационной системы на стандартном языке заняла бы огромное время, для таких задач в языках реализованы фреймворки и библиотеки. В них уже реализованная необходимая функциональность, которая необходима нам для создания нашей системы. В Java в большинстве широко применяемых фреймворков используется инверсия управления и внедрение зависимостей основные преимущества такого подхода это:

* Сокращение объёма связующего кода;
* Упрощение конфигурирования приложения;
* Возможность управлять общими зависимостями в одном хранилище;
* Улучшенная тестируемость;
* Стимулирование качественных проектных решений;

Внедрение зависимостей в свою очередь реализуется разными фреймворками

JBoss Seam Framework

Основан Гэвином Кингом (Gavin King), создателем библиотеки Hibernate ORM, и представляет собой еще один полнофункциональный каркас, ориентированный на внедрение зависимостей. Он поддерживает разработку пользовательских интерфейсов веб-приложений (JSF), уровень бизнес-логики (EJB 3) и прикладной интерфейс JPA для сохраняемости данных. Основное отличие каркаса Seam Framework от Spring состоит в том, что он построен полностью на стандартах JEE. Сервер приложений JBoss также способствовал донесению идей Seam Framework до организации JCP и появлению спецификации JSR-299 (Contexts and Dependency Injection for the Java ЕЕ Platfonn -Контексты и внедрение зависимостей для платформы Java ЕЕ).  
Обработка большого объема трафика и одновременное использование ресурсов различными пользователями. Широкий спектр корпоративных функций, таких как встроенная поддержка кластеризации, балансировки нагрузки, кэширования и отработки отказа, которые необходимы для крупномасштабных приложений, тоже может стать выбором именно Jboss [21].

Google Guice

Следующим распространенным каркасом для внедрения зависимостей является

Google. Поддерживаемый поисковым гигантом Google, Guice представляет собой облегченный каркас, предназначенный для внедрения зависимостей при управлении конфигурацией приложений. Он также стал эталонной реализацией спецификации JSR-330 (Dependency lnjection for Java - Внедрение зависимостей для Java). Основным отличием является легковесность и конфигурация без XML [22]. Однако количество материалов для обучения данному фреймворку и задач которых на нём решали не так богато как на Spring.

Spring [24]

Это первоначальный создатель внедрения зависимостей, с самым большим количеством полезных надстроек над собой которые позволяют упростить создание приложений во много раз если сравнивать spring с вышеуказанными фреймворками можно выделить такие плюсы: [23]

– Легкий вес и гибкость: Spring предоставляет легкий и гибкий фреймворк, который позволяет разработчикам выбирать только необходимые модули для своего приложения, что приводит к повышению производительности.   
– Инверсия управления (IoC): контейнер IoC от Spring упрощает создание объектов и управление зависимостями, упрощая интеграцию компонентов и написание слабосвязанного кода.  
– Аспектно-ориентированное программирование (AOP): Поддержка AOP в Spring обеспечивает модульную разработку, отделяя сквозные задачи, такие как ведение журнала, управление транзакциями и безопасность, от бизнес-логики.  
– Комплексная экосистема: Spring предлагает обширную экосистему библиотек и инструментов, таких как Spring Boot, Spring Data и Spring Security, которые предоставляют дополнительную функциональность и упрощают процесс разработки.  
– Активное сообщество: Spring располагает большим и активным сообществом разработчиков, предоставляющим богатый багаж знаний, ресурсов и поддержки. То есть вероятность столкнуться с такой проблемой которая не встречалась и не решалась раньше очень небольшая.

К минусам можно причислить:

– Сложность конфигурации: Spring требует конфигурации на основе XML или Java, что может быть сложным и отнимать много времени для сложных приложений. В случае нашей информационной системы, это не столь значимый минус.  
– Производительность во время выполнения: хотя Spring со временем улучшился, его производительность во время выполнения все еще может вызывать беспокойство по сравнению с легковесными фреймворками.  
– Управление зависимостями: управление зависимостями между различными модулями Spring может быть сложным и приводить к проблемам совместимости.  
– Тяжелая документация: Из-за обширных функций и опций документация Spring может быть перегруженной и сложной для навигации для некоторых разработчиков.

**Система сборки**

Система сборки нужна для конфигурирования сложных многомодульных проектов, а так как мы используем Spring, проект и будет таковым. Система сборки служит для решения проблем, когда проект располагается более чем в одном каталоге, так же довольно часто различные модули написаны с помощью разных технологий и языков, система сборки автоматизирует процесс, сбирая его в нужном порядке и применяет к каждому модулю свой набор инструментов, так же упрощает проблему масштабирования [25].

Рассмотрим две самые популярные и системы сборки Maven [26] и Gradel [27]. Начнём с последней из них, потому что она являться наиболее новой.

Gradle — это система автоматизации сборки, которая является полностью открытым исходным кодом и использует концепции, которые вы видите в Apache Maven и Apache Ant. Она использует специфический язык, основанный на языке программирования Groovy, что отличает ее от Apache Maven, который использует XML для конфигурации проекта. Она также определяет порядок выполнения задач с помощью направленного ациклического графа. Gradle и впервые выпустили его в 2007 году, а в 2013 году он был принят компанией Google в качестве систе

мы сборки для проектов Android. Она была разработана для поддержки много проектных больших сборок. Она также позволяет дополнять вашу сборку, поскольку знает, какие части вашего проекта обновляются. Задачи, которые зависят от обновленных частей повторно не выполняются.

Maven используется для автоматизации сборки проектов с использованием Java. Он помогает составить схему сборки конкретного программного обеспечения, а также его различных зависимостей. Он использует XML-файл для описания проекта, который вы собираете, зависимостей программного обеспечения от сторонних модулей и частей, порядка сборки, а также необходимых плагинов. Существуют заранее определенные цели для таких задач, как упаковка и компиляция. Maven загружает библиотеки и плагины из различных репозиториев, а затем помещает их все в кэш на вашей локальной машине.

Gradel является более сложным и функциональным решением которое больше подойдёт крупным компаниям для решения больших задач с ним сложнее работать и обучаться его применению. Maven же в свою очередь более простой и лёгкий, с помощью Maven вы можете легко определить метаданные и зависимости вашего проекта. Maven обеспечивает простое, но эффективное управление зависимостями, а поскольку он имеет структуру каталогов для ваших проектов, у вас есть своего рода стандартная схема для всех ваших проектов. Он использует декларативный XML-файл для своего POM-файла и имеет множество плагинов, которые вы можете использовать.

Для создания информационной системы для спортивных организаций, нам не нужны столь большие возможности gradel и более простой maven будет достаточен. При желании и росте проекта возможен переход от Maven к Gradel [28].

**База-данных**

Так как в информационной системе будет происходить работа с данными, необходимо определиться с тем как это будет происходить и как данные будут храниться.

Рассмотрим реляционные и нереляционные базы данных.

Реляционные базы данных представляют из себя совокупность таблиц взаимосвязанных между собой определённым образом, каждый столбец представляет из себя некоторое свойство, которое относиться ко всем строкам, которые являются уникальными записями. Для взаимодействия был создан язык SQL и на данный момент является стандартом для реляционных баз данных.

Нереляционные базы данных отличаются от реляционных, применением модели, которая оптимизирована для хранения определённого типа содержимого. Например, в виде JSON документов, графов или ключ-значений. Запросы осуществляются так же с помощью других языков не SQL.

Типы NoSQL баз данных бывают:

* Хранилище документов. Поля представлены различными форматами XML, JSON.
* Хранение данных в столбцах. Упорядочение данных происходит с помощью ключей.
* Хранение “ключ-значение”. Каждый набор данных обладает уникальным ключом.
* Хранение содержимого графов. Данные хранятся в узлах, а также определяется связь между узлами.
* Хранение временных рядов. Наборы значений привязаны к определённым моментам времени.
* Хранилище внешних индексов. Выступает в роли посредника служит для ускорения доступа к другим базам.

Нереляционны базы специфичны и адаптированы под работу с определёнными задачами и целями, данные хранящиеся в них также подчиняются этим правилам. В нашем же случае для решения поставленных задач, мы не нуждаемся в особенных преимуществах данных баз данных и будет использоваться реляционная база данных. Реляционные базы данных тоже имеют различные СУБД [29]. Выбор будет происходить между PostgreSQL [31] и MySQL [30]. Так как каждая из этих СУБД реализует базовые функции сконцентрируемся на их различие.

* Типы данных: PostgreSQL поддерживает более широкий спектр современных типов данных, включая массивы, hstore (хранилище типа “ключ-значение”) и JSONB (бинарный JSON), которые обеспечивают более гибкие и эффективные возможности хранения данных. С другой стороны, MySQL имеет ограниченный набор типов данных.
* Индексирование: В MySQL по умолчанию используется тип индекса B-tree, который хорошо подходит для большинства юзкейсов. PostgreSQL имеет более совершенную систему индексирования, чем MySQL, включая поддержку индексов B-tree, GiST (Generalized Search Tree. Обобщенное поисковое дерево [32]) и GIN (Generalized Inverted Index. Обобщенный обратный индекс [33]). Они предоставляют больше возможностей для оптимизации производительности запросов и поиска данных.
* Репликация: PostgreSQL и MySQL поддерживают репликацию баз данных, но методы и возможности репликации различны. PostgreSQL поддерживает мульти-мастер (с несколькими мастерами) репликацию, в то время как MySQL в основном поддерживает репликацию master-slave (ведущий-ведомый).

Я сделал выбор в пользу PostgreSQL как более универсальной системы, которая не сильно отстаёт по производительности от MySQL. Значимыми плюсами для создания информационной системы я считаю развитую систему индексирования по сравнению с MySQL, это может повысить производительность при выполнении сложных запросов. Расширенная поддержка типов данных, таких как массивы и JSONB может привести к более эффективному хранению и получению данных [34].

**Анализ бизнес-процессов**

Для решения выделенных проблем проанализированы бизнес-процессы и составлена BPMN диаграмма [35], представленная на рисунке 12.

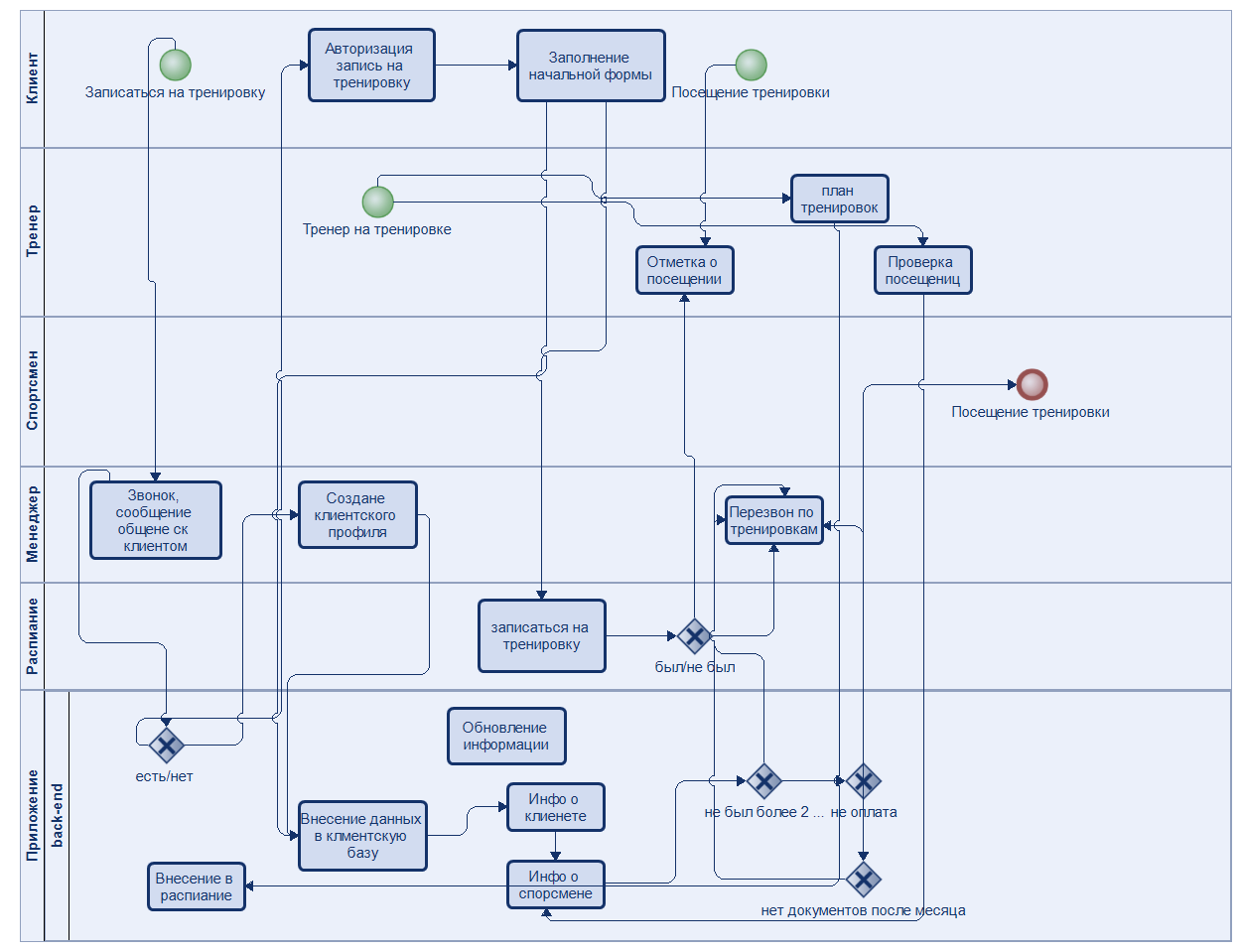


Рисунок 12 – BPMN диаграмма информационной системы для спортивной организации

В ролях были выделены отдельно менеджер и тренер, для разграничения функциональных требований от семантики действий, однако как говорилось ранее данные функции может исполнять один сотрудник.

Формализованы требования к системе

Назначение системы

* + Автоматизация деятельности работников спортивного учреждения.
  + Автоматизация учета посещения тренировок.
  + Автоматизация сбора информации о заказчиках.

Цель создания системы

* + Упрощение координации между заказчиком, тренером
  + Упрощение контроля сбора информации, её хранения и эффективного использования
  + Упрощение сбора информации о заказчиках
  + Освобождение сотрудников от ручного заполнения бланков и хранения информации индивидуально в бумажном формате
  + Доступ к расписанию тренировок и информации о спортсмене в цифровом виде.

Краткие сведения об объекте автоматизации.

Объектом автоматизации являются спортивные учреждения и организации и лица взаимодействующие с организацией с целью получения спортивных услуг.

Требования, предъявляемые к системе к её структуре и функциональности.

Система должна обеспечить:

* ввод, сбор, хранение, вывод, редактирование и обработку информации;
* функционал всех пользователей формируется согласно их ролям. Роли определяются исходя из специфики, должности и обязанности работников-пользователей и заказчиков-пользователей;
* безопасное использование приложения;
* работу с документированной информацией;
* язык системы – русский

В число ролей пользователей должны входить:

* + Заказчик
  + Менеджер
  + Тренер

У роли менеджера будет часть общих возможностей с другими ролями, это необходимо сделать с целью возможности совмещения функций, которые содержат другие роли для небольших учреждений, в которых тренера совмещают в себе данные задачи. Для пользователей заказчиков требований будет меньше, поскольку от них необходимо только умение пользоваться телефон или персональным компьютером.

Выделим функциональные блоки системы:

* + Функциональный блок заказчика;
  + Функциональный блок менеджера;
  + Функциональный блок тренера;

Рассмотрим каждый функциональный блок в отдельности

Блок заказчика должен выполнять следующие задачи:

* + Запись на пробную тренировку
  + Покупка абонементов на различное количество тренировок
  + Просмотр расписания тренировок и их посещения
  + Заполнение редактирования контактной информации и документов
  + Заполнение редактирования информации спортсмена

Блок менеджера:

* Возможность записывать заказчиков на пробные тренировки
* Проверка и редактирования контактной информации и документов заказчиков
* Проверка и редактирования контактной информации и документов спортсменов
* Просмотр расписания и записанных на тренировку и посетивших её
* Просмотр абонементов и оставшихся тренировок заказчика
* Отслеживание и планирование акций

Блок тренера:

* Проверка и редактирования информации и документов заказчика
* Проверка и редактирования информации и документов спортсмена
* Составление и редактирование тренировок и мероприятий
* Проверка и отметка посещений тренировок

Выполнение работ по созданию системы выполняется в три этапа:

* разработка методологической базы проекта;
* разработка Системы для целей опытной эксплуатации. Ввод системы в опытную эксплуатацию;
* доработка Системы для целей промышленной эксплуатации Ввод системы в промышленную эксплуатацию.

**Варианты использования информационной системы**

На основе изученных бизнес-процессов и сформированных требований к системе была разработана диаграмма вариантов использования [36], отражающая основные функции системы на рисунке 13.

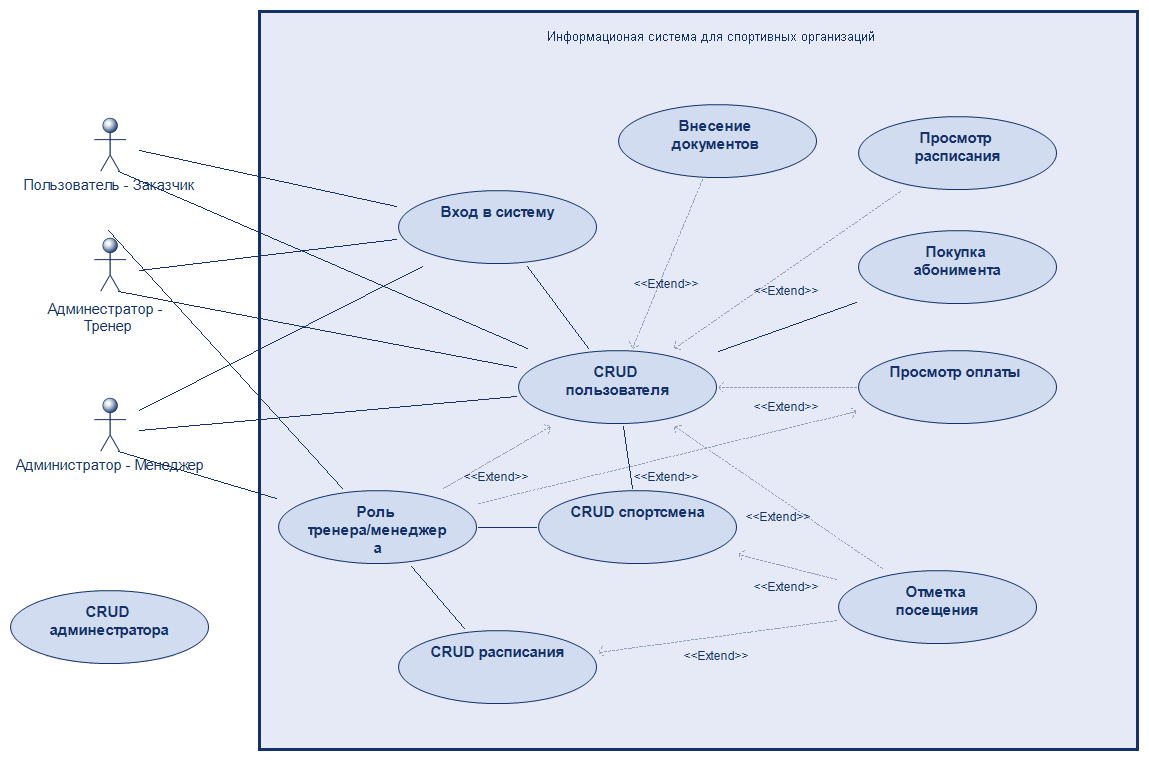


Рисунок 13 – Диаграмма вариантов использования для спортивных учреждений

Данная диаграмма отражает общее представление о вариантах использования системы, каждый вариант использования также отдельно подробно описан по действиям действующих лиц [37]. Описания представлены в таблице 2.

Таблица 2 – Варианты использования системы спортивных учреждений

|  |  |
| --- | --- |
| Вариант использования | Вход в систему |
| Краткое описание | Данный вариант использования позволяет установить какими возможностями будет обладать пользователь. |
| Основной поток событий | 1. Система запрашивает авторизацию 2. Ввод имени и пароля 3. Регистрация успешно можно переходить к доступным данным |
| Подчиненные потоки | 2А. Не успешный ввод пароля повтор   1. Система запрашивает пароль и имя вновь 2. Повторить 3 раза предложить сбросить пароль   2Б. Предложение зарегистрироваться   1. Перевод на CRUD пользователя |
| Альтернативный поток | 2А.А. Не успешный ввод пароля повтор   1. Система запрашивает пароль и имя вновь 2. Повторить 3 раза предложить сбросить пароль   2Б-2В.А. Предложение зарегистрироваться   1. Перевод на CRUD пользователя |
| Предусловия | - |
| Постусловия | Если вариант использования завершится успешно, пользователь переходит на личную страничку с своим функционалом и данными |
| Ограничения | - |
| **Вариант использования** | **CRUD пользователя** |
| Краткое описание | Данный вариант использования позволяет зарегистрироваться новому пользователю, изменить или удалить уже имеющиеся данные. Сведения включают в себя его ФИО, дату рождения, email, номер телефона и дополнительную информацию.  Система создает личный кабинет для каждого нового пользователя. Далее там будет отображаться данные о тренировках, оплате и расписание. Также он может видеть контактные данные тренера или менеджера. |
| Основной поток событий | 1. Пользователь или администратор запрашивает требуемое действие (создать нового пациента, редактировать данные уже существующие,). 2. Сообщаем свой выбор системе. 3. Согласно выбору выполняется один из подчиненных потоков (создать нового пользователя, редактировать данные, удалить пользователя). |
| Подчиненные потоки | 2А. Создать нового пользователя   1. Система запрашивает данные пользователя. 2. Вносим данные. 3. Сообщаем о необходимости сохранить данные о пользователе. 4. Система сохраняет данные и создает личный кабинет пользователя. 5. Управление передается на шаг 1 основного потока событий.   2Б. Редактировать данные   1. Запрашиваем информацию пользователе. 2. Система выводит всю имеющуюся информацию о пользователе. 3. Администратор сообщает о необходимости внести изменения. 4. Система предоставляет доступ для изменения данных. 5. Вносим изменения в данные. 6. Сообщаем системе о необходимости сохранить данные о пользователе. 7. Система сохраняет данные. 8. Управление передается на шаг 1 основного потока событий.   2В. Удалить пользователя   1. Запрашиваем информацию о пользователе. 2. Сообщаем системе о необходимости удалить все данные пользователя. 3. Подтверждаем удаление данных. 4. Система удаляет данные.   Управление передается на шаг 1 основного потока событий. |
| Альтернативный поток | 2А.А. Данный о пользователе уже существует   1. Система отображает всю информацию о пользователе. 2. Управление передается на шаг 1 основного потока событий.   2Б-2В.А. Данный пользователь не найден в системе   1. Система информирует о том, что данного пользователя нет в системе. 2. Предлагает создать нового пользователя (если заходит клиент)   Управление передается на шаг 1 основного потока событий |
| Предусловия | - |
| Постусловия | Если вариант использования завершится успешно, то данные о новом пользователе будут сохранены в системе, изменения в данных пользователе будут сохранены и требуемая информация будет удалена. В противном случае система гарантирует, что изменения в системе не будут произведены. |
| Ограничения | - |

|  |  |
| --- | --- |
| **Вариант использования** | **Роль тренера/менеджера** |
| Краткое описание | Данный вариант использования позволяет создать нового тренера/менеджера с их привилегиями, возможностям и доступом к информации и их личному кабинету возможность изменить или удалить уже имеющиеся данные. Сведения включают в себя его ФИО, дату рождения, email, номер телефона и дополнительную информацию, уровень админ доступа к различным функциям (например назначать других админов)  Система создает личный кабинет. Далее там будет отображаться данные о группах посещении, оплате и доступ к редактированию расписания и доступом к информации пользователей. Также он может видеть контактные данные тренера менеджера и пользователя. |
| Основной поток событий | 1. Администратор запрашивает требуемое действие (создать нового тренера менеджера, редактировать данные уже существующие,). 2. Сообщаем свой выбор системе. 3. Согласно выбору выполняется один из подчиненных потоков (создать нового тренера/менеджера, редактировать данные, удалить). |
| Подчиненные потоки | 2А. Создать нового тренера/менеджера   1. Система запрашивает данные тренера/менеджера. 2. Вносим данные. 3. Сообщаем о необходимости сохранить данные о тренера/менеджера. 4. Система сохраняет данные и создает личный кабинет. 5. Управление передается на шаг 1 основного потока событий.   2Б. Редактировать данные   1. Запрашиваем информацию тренера/менеджера. 2. Система выводит всю имеющуюся информацию о тренере/менеджере. 3. Администратор сообщает о необходимости внести изменения. 4. Система предоставляет доступ для изменения данных. 5. Вносим изменения в данные. 6. Сообщаем системе о необходимости сохранить данные о тренере/менеджере. 7. Система сохраняет данные. 8. Управление передается на шаг 1 основного потока событий.   2В. Удалить тренера/менеджера   1. Запрашиваем информацию о тренере/ менеджере 2. Сообщаем системе о необходимости удалить все данные тренера/менеджера. 3. Подтверждаем удаление данных. 4. Система удаляет данные.   Управление передается на шаг 1 основного потока событий. |
| Альтернативный поток | 2А.А. Данный тренера/менеджера уже существует   1. Система отображает всю информацию о тренере/менеджере. 2. Управление передается на шаг 1 основного потока событий.   2Б-2В.А. Данный тренера/менеджера не найден в системе   1. Система информирует о том, что данного тренера/менеджера нет в системе. 2. Предлагает создать нового тренера/менеджера   Управление передается на шаг 1 основного потока событий |
| Предусловия | Вход по администратором |
| Постусловия | Если вариант использования завершится успешно, то данные о новом тренера/менеджера будут сохранены в системе, изменения в данных тренера/менеджера будут сохранены и требуемая информация будет удалена. В противном случае система гарантирует, что изменения в системе не будут произведены. |
| Ограничения | Возможность создавать данные доступы есть не у всех а у определённых тренеров/менеджеров и администратора |

|  |  |
| --- | --- |
| Вариант использования | CRUD Спортсмена |
| Краткое описание | Данный вариант использования позволяет зарегистрировать новому клиенту или тренеру/менеджеру, изменить или удалить уже имеющиеся данные. Сведения включают в себя его ФИО, дату рождения, email, номер телефона при наличии, достижения, посещённые тренировки и дополнительную информацию,.  Система создает страничку для каждого нового спортсмена. Далее там будет отображаться данные о тренировках, их посещении и расписание, достижение. |
| Основной поток событий | 1. Пользователь или администратор запрашивает требуемое действие (создать нового спортсмена, редактировать данные уже существующие,). 2. Сообщаем свой выбор системе. 3. Согласно выбору выполняется один из подчиненных потоков (создать нового спортсмена, редактировать данные, удалить спортсмена). |
| Подчиненные потоки | 2А. Создать нового пользователя   1. Система запрашивает данные пользователя. 2. Вносим данные. 3. Сообщаем о необходимости сохранить данные о спортсмене. 4. Система сохраняет данные и создает страницу спорсмена. 5. Управление передается на шаг 1 основного потока событий.   2Б. Редактировать данные   1. Запрашиваем информацию о спортсмене. 2. Система выводит всю имеющуюся информацию о спортсмене. 3. Администратор сообщает о необходимости внести изменения. 4. Система предоставляет доступ для изменения данных. 5. Вносим изменения в данные. 6. Сообщаем системе о необходимости сохранить данные о спортсмене. 7. Система сохраняет данные. 8. Управление передается на шаг 1 основного потока событий.   2В. Удалить спортсмена   1. Запрашиваем информацию о пользователе. 2. Сообщаем системе о необходимости удалить все данные спортсмена. 3. Подтверждаем удаление данных. 4. Система удаляет данные.   Управление передается на шаг 1 основного потока событий. |
| Альтернативный поток | 2А.А. Данный о спортсмене уже существует   1. Система отображает всю информацию о спортсмене. 2. Управление передается на шаг 1 основного потока событий.   2Б-2В.А. Данный спортсмен не найден в системе   1. Система информирует о том, что данного пользователя нет в системе. 2. Предлагает создать нового спортсмена.   Управление передается на шаг 1 основного потока событий |
| Предусловия | Авторизованный пользователь или тренер или менеджер (администратор) |
| Постусловия | Если вариант использования завершится успешно, то данные о новом спортсмена будут сохранены в системе, изменения в данных пользователе будут сохранены и требуемая информация будет удалена. В противном случае система гарантирует, что изменения в системе не будут произведены. |
| Ограничения | - |
| Вариант использования | CRUD Расписания |
| Краткое описание | Данный вариант использования позволяет создать тренировку в расписании. |
| Основной поток событий | 1. Тренер запрашивает требуемое действие (создать новую тренировку, редактировать данные уже существующие,) 2. Сообщаем свой выбор системе. 3. Согласно выбору выполняется один из подчиненных потоков (создать нового спортсмена, редактировать данные, удалить тренировку). |
| Подчиненные потоки | 2А. Создать новую тренировку   1. Система запрашивает данные. 2. Вносим данные. 3. Сообщаем о необходимости сохранить данные о тренировке. 4. Система сохраняет данные и создает тренировку в расписании. 5. Управление передается на шаг 1 основного потока событий.   2Б. Редактировать данные   1. Запрашиваем информацию о тренировке. 2. Система выводит всю имеющуюся информацию о тренировке. 3. Сообщаем о необходимости внести изменения. 4. Система предоставляет доступ для изменения данных. 5. Вносим изменения в данные. 6. Сообщаем системе о необходимости сохранить данные о тренировке. 7. Система сохраняет данные. 8. Управление передается на шаг 1 основного потока событий.   2В. Удалить тренировку   1. Запрашиваем информацию о пользователе. 2. Сообщаем системе о необходимости удалить все данные спортсмена. 3. Подтверждаем удаление данных. 4. Система удаляет данные.   2Г. Отметка о тренировке   1. Вносятся отметки о посещении тренировками спортсменами   Управление передается на шаг 1 основного потока событий. |
| Альтернативный поток | 2А.А. Данный о тренировке уже существует на это время   1. Система отображает всю информацию о спортсмене. 2. Управление передается на шаг 1 основного потока событий.   2Б-2В.А. Данная тренировке не найдена в системе   1. Система информирует о том, что данного пользователя нет в системе. 2. Предлагает создать нового спортсмена. 3. Управление передается на шаг 1 основного потока событий |
| Предусловия | Авторизованный пользователь или тренер или менеджер (администратора) |
| Постусловия | Если вариант использования завершится успешно, то данные о тренировке будут сохранены в системе, изменения в данных тренировки будут сохранены и требуемая информация будет удалена. В противном случае система гарантирует, что изменения в системе не будут произведены. |
| Ограничения | - |

|  |  |
| --- | --- |
| Вариант использования | Внесение документов |
| Краткое описание | Данный вариант использования позволяет добавить документы в личный кабинет пользователя |
| Основной поток событий | 1. Добавить документы в личном кабинете 2. Сообщаем свой выбор системе. 3. Согласно выбору выполняется один из подчиненных потоков (Добавление документов ). 4. Или выбор удалить документы |
| Подчиненные потоки | 2В. Удалить документы   1. Запрашиваем информацию о пользователе. 2. Сообщаем системе о необходимости удалить все данные спортсмена. 3. Подтверждаем удаление данных. 4. Система удаляет данные.   Управление передается на шаг 1 основного потока событий. |
| Альтернативный поток | - |
| Предусловия | Авторизованный пользователь или тренер или менеджер (администратора) |
| Постусловия | Если вариант использования завершится успешно, то документ о будет сохранены в системе, или требуемая информация будет удалена. В противном случае система гарантирует, что изменения в системе не будут произведены. |
| Ограничения | - |
| Вариант использования | Оплата тренировки |
| Краткое описание | Данный вариант использования позволяет пользователю оплатить тренировку. |
| Основной поток событий | 1. Пользователь запрашивает разрешение оплатить 2. Система представляет доступ к оплате 3. Пользователь вводит сумму. 4. Система сохраняет историю пополнений. |
| Подчиненные потоки | - |
| Альтернативный поток | 2А. Система отклонила запрос   1. Система отклонила запрос на пополнение. (недостаточная сумма) 2. Вариант использования завершается. |
| Предусловия | Пользователь должен войти в систему. |
| Постусловия | Если вариант использования завершится успешно, то у пользователя отобразится баланс. В противном случае, счёт не будет пополнен. |
| Ограничения | Пользователи могут пополнить только свой счёт, тренер/менеджером не могут менять баланс |

**Описание основных функций**

Среди вариантов использования выделены основные функции, требующие более подробного описания, они визуализированы на диаграммах действий [38], представленных [39] на рисунках с 14 по 17.

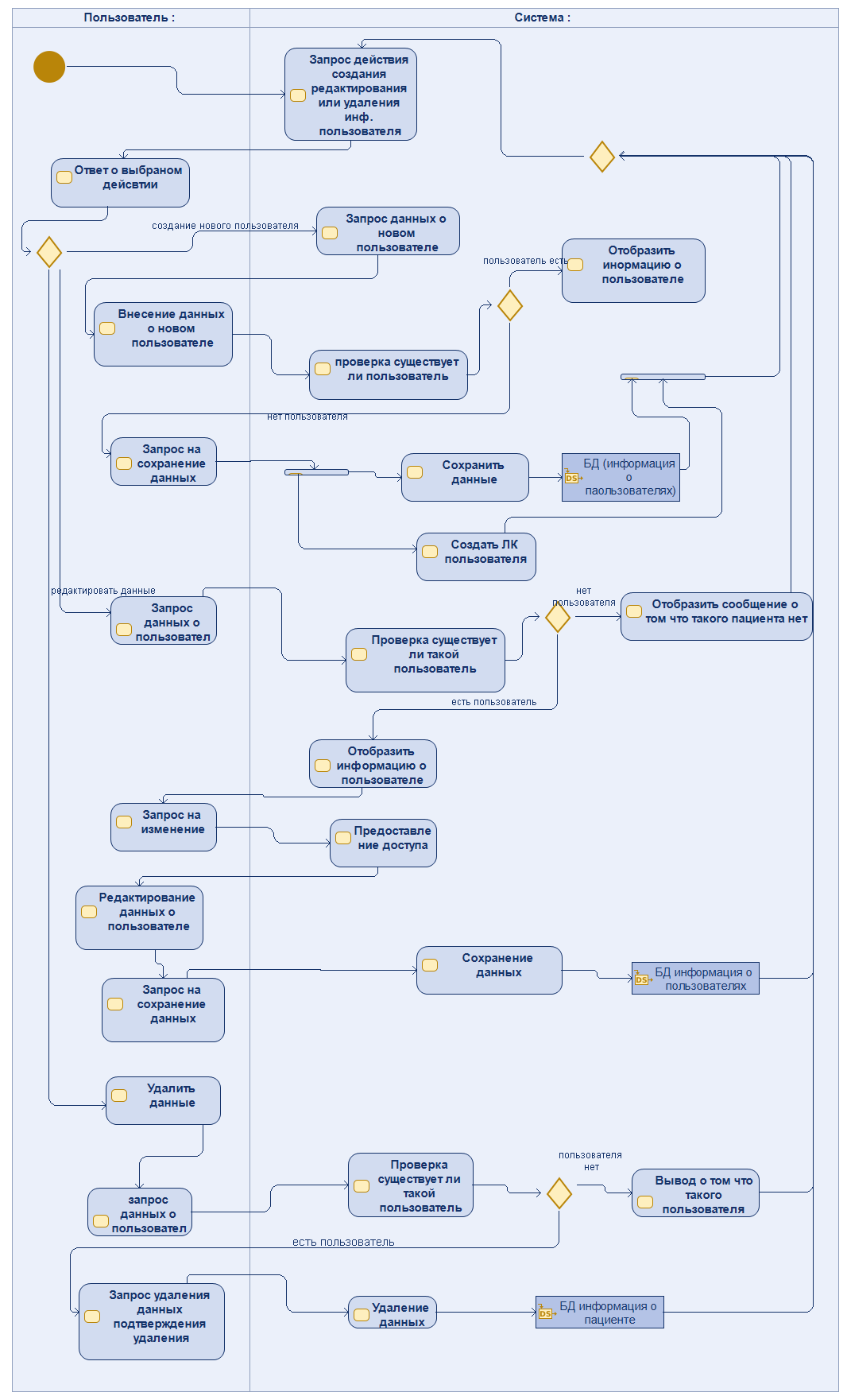


Рисунок 14 – Диаграмма деятельности «CRUD данных о пользователях»

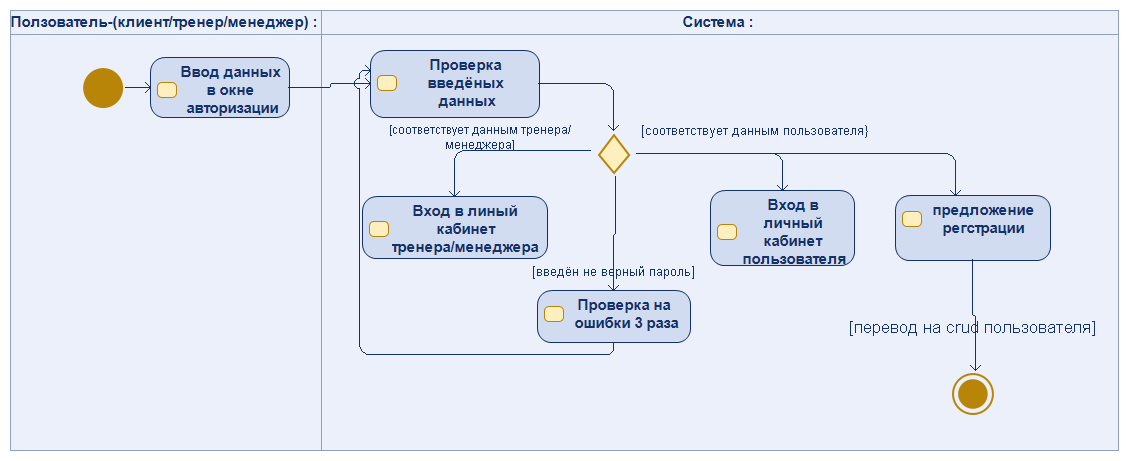


Рисунок 15 – Диаграмма деятельности «Вход в систему»

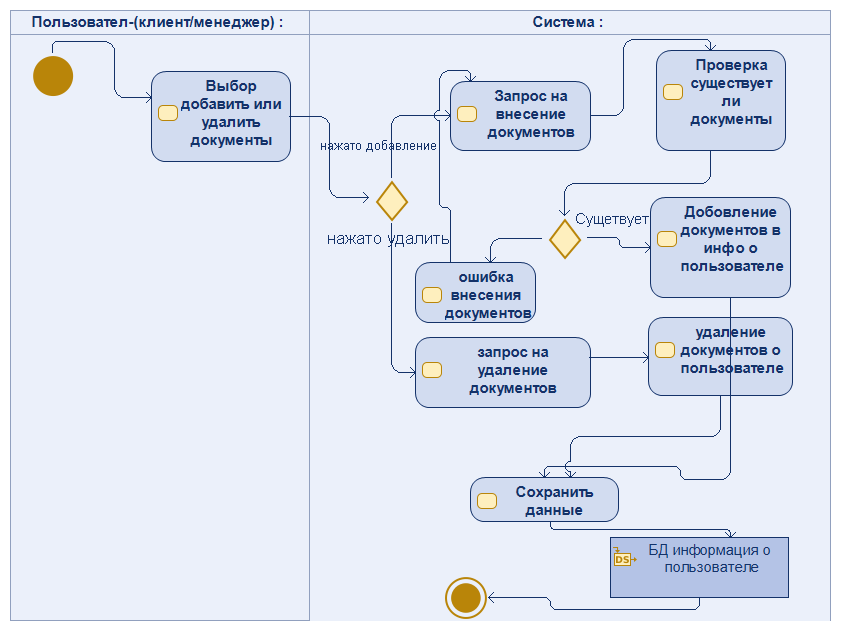


Рисунок 16 – Диаграмма деятельности «Внесение документов»

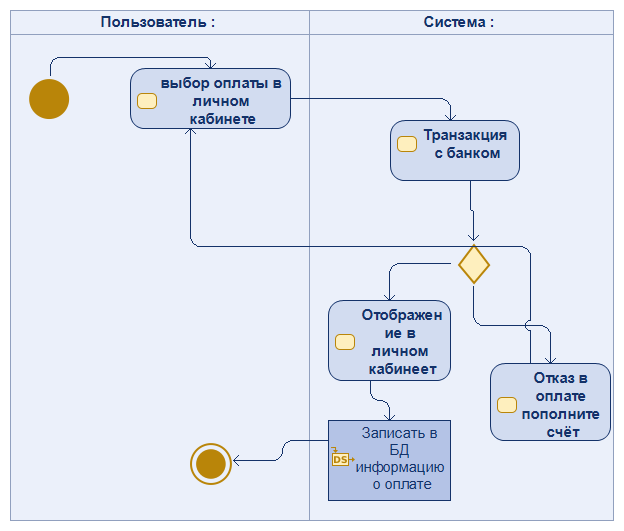


Рисунок 17 – Диаграмма деятельности «Оплаты»

**Модели ключевых классов информационной системы**

На основе вышеописанных диаграмм и данных проведён анализ и спроектирована архитектура программного решения функционала информационной системы.

Создана модель ключевых классов информационной системы [40] [41], представленная на рисунках 18 и 19, 21. На рисунках 20 и 22 показаны диаграммы взаимодействий [42] [43].

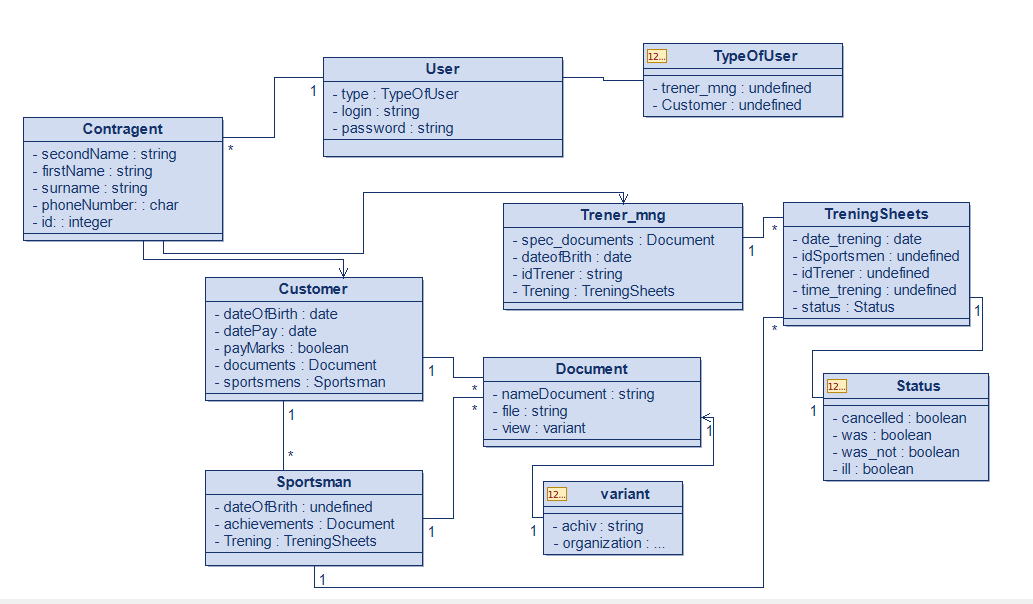


Рисунок 18 – Диаграмма Key Abstractions

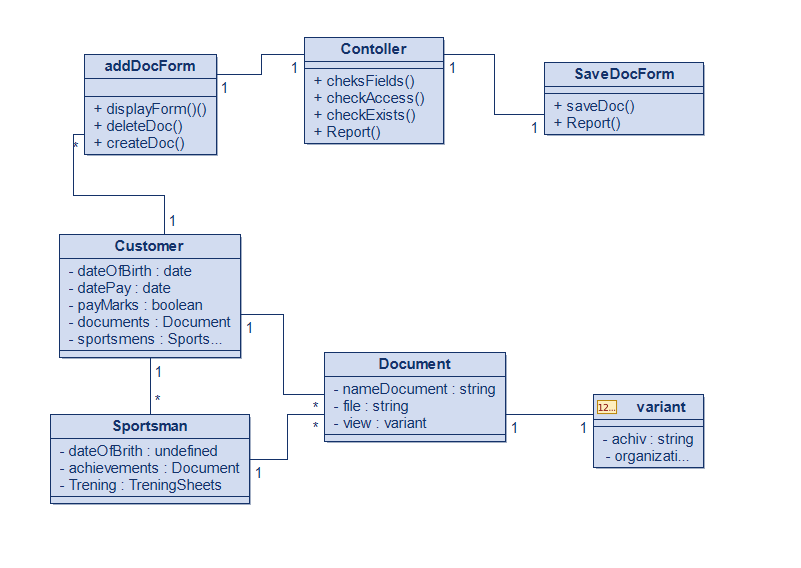


Рисунок 19 – Диаграмма классов VOPC для варианта «Добавления документа»

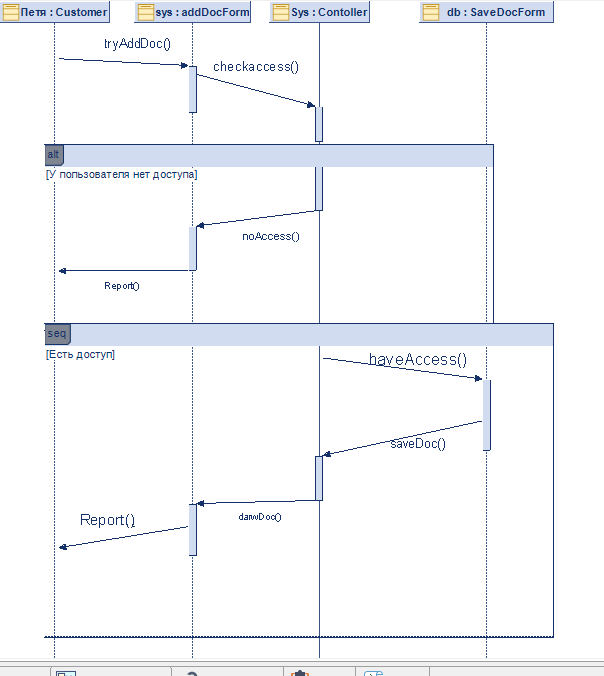


Рисунок 20 – Диаграмма взаимодействия для варианта «Добавление документа»



Рисунок 21 – Диаграмма классов VOPC для варианта «Транзакции с банком»

Так же спроектируем диаграммы взаимодействий

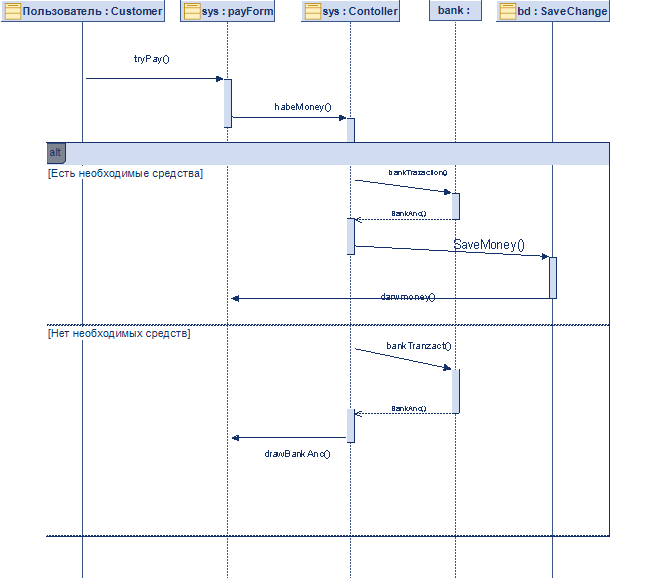


Рисунок 22 – Диаграмма взаимодействия для варианта «Транзакции с банком»

ГЛАВА 3 РАЗРАБОТКА СИСТЕМЫ.

**Создание интерфейса системы**

**Формы регистрации**

Начальная форма регистрации бывает 2 видов. Один полностью онлайн показанный на рисунке 23



Рисунок 23 – Форма онлайн регистрации

При этом варианте происходит переход к расписанию, где можно отметить на какое число и тренировку записываются, в соответствии с возрастом ребёнка и свободным местом на тренировках, так же есть выбор между тренерами.

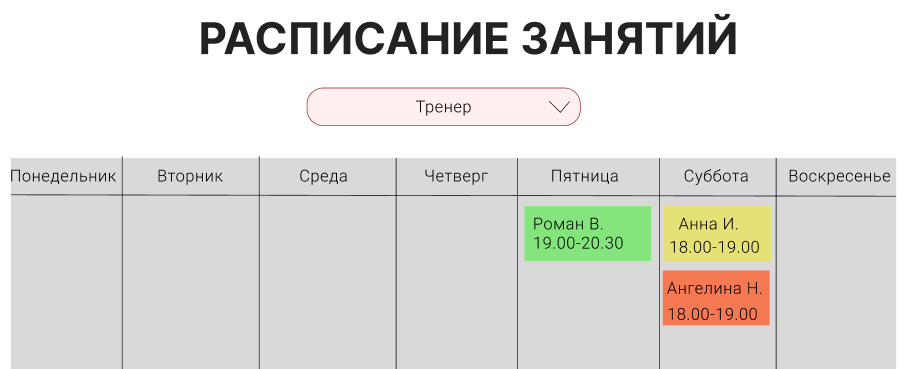


Рисунок 24 – Расписание групп

При втором варианте, показанном на рисунке 25 заполнение происходит сотрудником спортивной организации который совершает звонок по номеру клиента, он уточняет ФИО ребёнка и дату рождения, а так же отмечает запись на тренировку в расписании с рисунка 24



Рисунок 24 – Форма регистрации через звонок

Отличной частью от других информационных систем это наличие карточки спортсмена с успехами как показано на рисунке 25.

**Форма данных спортсмена**

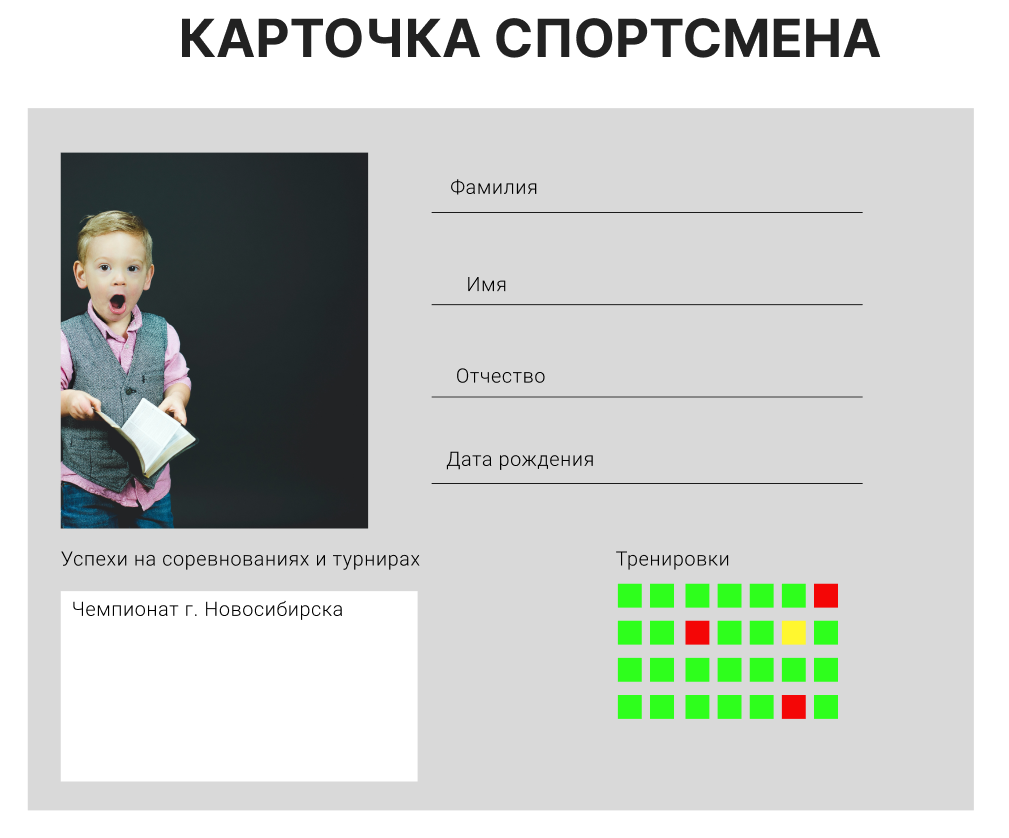


Рисунок 25 – Карточка спортсмена

В ней можно увидеть посещённые тренировки, а также успехи на прошлых турнирах эти данные, служат для хранения и не утери этой информации, а так же решения тренера, о выстраивании дальнейшего тренировочного плана.

ЗАКЛЮЧЕНИЕ

В ходе данной работы была создана информационная система путём выполнения следующих задачи.

В результат исследования предметной области выявлены особенности спортивных организаций, были проанализированы бизнес-процессы и на их основе создана BPMN диаграмма.

При анализе решений от других компаний применяя их к спортивным организациям, были найдены минусы и преимущества. Которые утонены при проектировании и разработке моей информационной системы для спортивных организаций.

Далее в ходе проектирования системы создавались различные диаграммы, такие как диаграмма использования для понимания что необходимо реализовать в системе, выделения определённых действий отражённых на диаграмме действий, что позволило понять какие функции необходимо реализовать, также выделены классы с помощью диаграмме классов и их связи и взаимодействия показанных на диаграммах взаимодействий.

Рассматривались также различные средства для разработки системы, в ходе их сравнения выделены наиболее удовлетворяющие требованиям.

Реализация системы показал, что есть места, которые не были рассмотрены в ходе проектирования, а создание внешнего интерфейса, через который будет осуществлять взаимодействие с системой показало, что это также требует большого количества времени, навыков и понимания что к чему ведёт.

Был реализован интерфейс формы онлайн записи с помощью, которой с спортивной организацией взаимодействуют новые пользователи. Как идея по улучшению качества оказания индивидуальных услуг был создан карточка спортсмена, в который отображены основные данные, требующиеся для тренера.

Идеи по совершенствованию и модернизации системы.

Как и идеи для модернизации системы я бы хотел добавить ботов в telegram и whatsapp, через которых так же можно было бы записаться, а так де отслеживать посещение тренировки детьми, для родителей. Например, через отмети тренером, тех кто есть в начале на ней, отсылались бы сообщения для родителей. Так же эту систему возможно автоматизировать созданием приложения с карточками спортсмена с QR кодами, которые можно сканировать при входе в зал, что также отправляет сообщение родителю. Добавить автоматические сообщения бота, который интересуется почему вы так долго не ходите, что может показать заинтересованность спортивной организации в клиенте. Хранение и анализ успехов учеников в различных мероприятиях и соревнованиях также можно анализировать, так как многие родители снимают видео выступлений, получиться накапливать видеотеку о росте спортсмена. Добавление различных тестов на психологию также можно внедрить в систему, что позволит отслеживать влияние помологической подготовки или индивидуально подхода к юным спортсменам. Часть данных идей реализованы в других видах информационных систем или социальных сетей, но это не мешает собрать их в едино и помочь бизнесу связанному со спортом конкурировать за время детей.

СПИСОК ЛИТЕРАТУРЫ

* + - 1. Information technology — Vocabulary [Электронный ресурс]. – URL: [iso.org/obp/ui/en/#iso:std:iso-iec:2382:ed-1:v2:en](https://www.iso.org/obp/ui/en/#iso:std:iso-iec:2382:ed-1:v2:en)
      2. Виды информационных система [Электронный ресурс]. – URL: <https://ru.wikipedia.org/wiki/Информационная_система>
      3. Определение информационной системы по ГОСТ [Электронный ресурс]. – URL: [ГОСТ 7.0-99 СИБИД. Информационно-библиотечная деятельность, библиография. Термины и определения - docs.cntd.ru](https://docs.cntd.ru/document/1200004287) - № 3.1.30.
      4. Федеральный закон "О персональных данных" [Электронный ресурс]. – URL: <https://www.consultant.ru/document/cons_doc_LAW_61801/> - п. 10 ст. 3 №152-ФЗ
      5. Федеральный закон "Об электронной подписи" [Электронный ресурс]. – URL: <https://www.consultant.ru/document/cons_doc_LAW_112701/> - п. 12, п. 13 ст. 2 №63-ФЗ
      6. Федеральный закон "О физической культуре и спорте в Российской Федерации" [Электронный ресурс]. – URL: <https://www.consultant.ru/document/cons_doc_LAW_73038/b819c620a8c698de35861ad4c9d9696ee0c3ee7a/> - п. 30 ст. 2 №87-ФЗ
      7. Федеральный закон "О физической культуре и спорте в Российской Федерации" [Электронный ресурс]. – URL: <https://www.consultant.ru/document/cons_doc_LAW_73038/b819c620a8c698de35861ad4c9d9696ee0c3ee7a/> - п. 15, п. 18, п. 19 ст. 2 №87-ФЗ
      8. Ковалевский Г. И. Статья «Правовой статус спортивных организаций» [Электронный ресурс]. – URL: <https://eee-science.ru/item-work/2022-3291/> - cтр. 3 – стр. 5.
      9. Официальный сайт Bitrix24 [Электронный ресурс]. – URL: <https://www.bitrix24.ru/>
      10. Обзор Bitrix24 [Электронный ресурс]. – URL: <https://habr.com/ru/companies/trinion/articles/257795/>
      11. Официальный сайт AmoCRM [Электронный ресурс]. – URL: <https://www.amocrm.ru/>
      12. Обзор на AmoCRM [Электронный ресурс]. – URL: <https://skillbox.ru/media/management/obzor-amocrm-onlaynservisa-dlya-upravleniya-prodazhami/>
      13. Официальный сайт 1C: Фитнес-клуб [Электронный ресурс]. – URL: [https://www.fitness1c.ru/](https://www.fitness1c.ru/?utm_source=yandex&utm_medium=cpc&utm_campaign=86618372&utm_content=13956699466&utm_term=1%D1%81%20%D1%84%D0%B8%D1%82%D0%BD%D0%B5%D1%81%20%D0%BA%D0%BB%D1%83%D0%B1&calltouch_tm=yd_c:86618372_gb:5178819422_ad:13956699466_ph:44352503800_st:search_pt:premium_p:1_s:none_dt:desktop_reg:65_ret:44352503800_apt:none&yclid=3042637538226339839)
      14. Обучающие материалы от компании 1С: Фитнес-клуб [Электронный ресурс]. – URL: <https://www.fitness1c.ru/kurs-zapusk-fitnes-studii-i-kluba#part4>
      15. Обзор на 1C: Фитнес-клуб [Электронный ресурс]. – URL: <https://vc.ru/u/553644-1s-fitnes-klub/717342-kak-1s-fitnes-klub-pomogaet-prinimat-upravlencheskie-resheniya>
      16. Сравнение CRM систем AmoCRM и Bitrix[Электронный ресурс]. – URL: <https://vc.ru/services/161490-amocrm-ili-bitriks24-vybiraem-i-sravnivaem-dve-populyarnye-crm>
      17. Отзывы о решении 1С: Фитнес-клуб [Электронный ресурс]. – URL: <https://crmindex.ru/products/1c_fitness>
      18. Методологии проектирования [Электронный ресурс]. – URL: <https://otus.ru/journal/informacionnye-sistemy-opredelenie-i-metodologii-sozdaniya/#Информационное_обеспечение>
      19. Итерационный и инкременационый методы [Электронный ресурс]. – URL: <https://testirovshik.com/chem-iterativnaja-model-otlichaetsja-ot-inkrementnoj/#:~:text=%D0%98%D1%82%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F%20%D0%BF%D0%BE%D0%B4%D1%80%D0%B0%D0%B7%D1%83%D0%BC%D0%B5%D0%B2%D0%B0%D0%B5%D1%82%20%D0%BF%D0%BE%D1%81%D1%82%D0%B5%D0%BF%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5%20%D0%BF%D1%80%D0%B8%D0%B1%D0%BB%D0%B8%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5%20%D1%86%D0%B8%D0%BA%D0%BB%D0%B0%D0%BC%D0%B8,%D0%B8%20%D0%B7%D0%B0%D1%82%D0%B5%D0%BC%20%D0%BE%D0%B1%D1%8A%D0%B5%D0%B4%D0%B8%D0%BD%D0%B8%D1%82%D1%8C%20%D0%B2%20%D1%86%D0%B5%D0%BB%D0%BE%D0%B5>
      20. Итерационная модель [Электронный ресурс]. – URL: <https://studfile.net/preview/16493243/page:4/>
      21. Документация JBoss Seam Framework [Электронный ресурс]. - URL: <https://docs.jboss.org/seam/2.3.1.Final/reference/html_single/#Book-Preface> п. Introduction to JBoss Seam.
      22. Документация Google Guice [Электронный ресурс]. - URL: <https://github.com/google/guice/wiki/> п. Overview и п. Motivation
      23. Понятие внедрения зависимостей [Электронный ресурс]. - URL: <https://en.wikipedia.org/wiki/Dependency_injection>
      24. Документация Spring [Электронный ресурс]. - URL: <https://docs.spring.io/spring-framework/reference/index.html> п. Overview и п. Core
      25. О системах сборки [Электронный ресурс]. - URL: <https://bazel.build/basics/build-systems>
      26. Документация Maven [Электронный ресурс]. - URL: <https://maven.apache.org/guides/getting-started/index.html>
      27. Документация Gradel [Электронный ресурс]. - URL: <https://docs.gradle.org/current/userguide/userguide.html>
      28. Сравнение Gradel и Maven [Электронный ресурс]. - URL: <https://habr.com/ru/companies/otus/articles/593903/>
      29. Сравнение реляционных и нереляционных баз данных [Электронный ресурс]. - URL: <https://aloa.co/blog/relational-vs-non-relational-database-pros-cons#:~:text=A%20relational%20database%20is%20structured,of%20a%20laundry%20list%20order>.
      30. Документация MySQL [Электронный ресурс]. - URL: <https://dev.mysql.com/doc/refman/8.3/en/what-is-mysql.html>
      31. Документация PostgreSQL [Электронный ресурс]. - URL: <https://www.postgresql.org/docs/current/intro-whatis.html>
      32. Об обобщенном поисковом дереве [Электронный ресурс]. - URL: <https://www.postgresql.org/docs/current/gist-intro.html>
      33. Об обобщенном обратном индексе [Электронный ресурс]. - URL: <https://www.postgresql.org/docs/current/gin-intro.html>
      34. Сравнение MySQL и PostgreSQL [Электронный ресурс]. - URL: <https://habr.com/ru/companies/otus/articles/722304/>
      35. Что такое BPMN диаграммы и их создание [Электронный ресурс]. - URL: <https://www.ibm.com/blog/bpmn/>
      36. Что такое диаграммы вариантов использования и их создание [Электронный ресурс]. - URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>
      37. Создание таблиц по диаграмме вариантов использования [Электронный ресурс]. - URL: <https://practicum.yandex.ru/blog/chto-takoe-use-case-kak-ih-napisat/>
      38. Что такое диаграмма действий и их создание [Электронный ресурс]. - URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>
      39. Примеры создания диаграмм действий [Электронный ресурс]. - URL: <https://creately.com/blog/ru/uncategorized-ru/учебник-по-диаграмме-активности/>
      40. Что такое диаграмма классов и их создание [Электронный ресурс]. - URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>
      41. Подробнее про отношение между классами в диаграмме [Электронный ресурс]. - URL: <https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/>
      42. Что такое диаграмма взаимодействий их создание [Электронный ресурс]. - URL: <https://www.geeksforgeeks.org/interaction-overview-diagrams-unified-modeling-language-uml/>
      43. Примеры создания диаграмм взаимодействий [Электронный ресурс]. - URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-interaction-overview-diagram/>

ПРИЛОЖЕНИЕ