**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

|  |  |
| --- | --- |
| КП.09.02.03.22.191.09 ПЗ |  |

ИНФОРМАЦИОННАЯ СИСТЕМА «ЙОГА КЛУБ»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Д.А. Гусаков) |

Иркутск 2022

**Содержание**

[Введение 3](file:///G:\КП__Ильин%20Артур.docx#_Toc89852049)

[1.Описание предметной области ИС 5](file:///G:\КП__Ильин%20Артур.docx#_Toc89852050)

[2.Анализ инструментальных средств разработки 7](file:///G:\КП__Ильин%20Артур.docx#_Toc89852051)

[3.Техническое задание 12](file:///G:\КП__Ильин%20Артур.docx#_Toc89852052)

[4.Проектирование ИС 13](file:///G:\КП__Ильин%20Артур.docx#_Toc89852053)

[4.1.Структурная схема ИС 13](file:///G:\КП__Ильин%20Артур.docx#_Toc89852054)

[4.2.Функциональная схема ИС 15](file:///G:\КП__Ильин%20Артур.docx#_Toc89852055)

[4.3.Проектирование базы данных 18](file:///G:\КП__Ильин%20Артур.docx#_Toc89852056)

[4.4. Проектирование интерфейса 21](file:///G:\КП__Ильин%20Артур.docx#_Toc89852057)

[5.Разработка ИС 23](file:///G:\КП__Ильин%20Артур.docx#_Toc89852058)

[5.1.Разработка интерфейса ИС 27](file:///G:\КП__Ильин%20Артур.docx#_Toc89852059)

[5.2.Разработка базы данных ИС 27](file:///G:\КП__Ильин%20Артур.docx#_Toc89852060)

[5.3.Разработка ИС 29](file:///G:\КП__Ильин%20Артур.docx#_Toc89852061)

[6.Документирование программного продукта 31](file:///G:\КП__Ильин%20Артур.docx#_Toc89852062)

[6.1.Руководство пользователя ИС 31](file:///G:\КП__Ильин%20Артур.docx#_Toc89852063)

[Заключение 39](file:///G:\КП__Ильин%20Артур.docx#_Toc89852064)

[Список используемых источников 40](file:///G:\КП__Ильин%20Артур.docx#_Toc89852065)

[Приложение А – Техническое задание 41](file:///G:\КП__Ильин%20Артур.docx#_Toc89852066)

[Приложение Б – Листинг views.py 46](file:///G:\КП__Ильин%20Артур.docx#_Toc89852067)

**Введение**

Стресс, депрессия, кризис – прочно вошли в нашу жизнь. Из-за этих факторов человек находится всегда в напряжении, не умеет дать себе отдых, он спешит и суетится, мешает своей нервозностью жить и работать другим, и, в конце концов, становится обузой для себя и своего окружения. Рано или поздно такой образ жизни ведет к нервным расстройствам. Этот замкнутый круг невозможно разорвать приемом все новых и новых медикаментов. Занятие йогой откроют человеку двери в качественно новую жизнь, в особенности она, способствует поддержанию гармоничного душевного состояния, здоровья и энергичной активной жизни.

Для того, чтобы занятия йогой были структурированы и не несли вреда для людей, решивших заняться этим безусловно полезным занятием, существуют йога клубы, в них человек может заниматься йогой не опасаясь последствий, и, кроме того, иметь возможность находится в кругу единомышленников, заводить новые приятные знакомства, совершенствуя своё тело и разум.

Для того, чтобы в йога клубе царила атмосфера гармонии и порядка, необходимо обеспечить удобство клиентов, в том числе и в сфере информационных технологий, поэтому необходимо создать информационную систему, которая полностью покрывала бы потребности клиента, и исключала бы дискомфорт и неудобство с посещением клуба. Этим обусловлена актуальность данного курсового проекта.

Целью курсового проекта является создание информационной системы «Йога клуб».

Для достижения конечного результата, необходимо решить следующие задачи:

* исследовать выбранную предметную область ИС;
* выполнить анализ инструментальных средств разработки;
* подобрать литературу и интернет-источники по выбранной теме;
* разработать техническое задание;
* выполнить проектирование ИС;
* разработать интерфейс и структуру ИС;
* составить руководство пользователя.

1. **Описание предметной области ИС**

Предметной областью курсового проекта является йога клуб.

Йога клуб – это специально оборудованное помещение, в котором создается благоприятная атмосфера для практики йоги. Йога клуб имеет штат сотрудников: Администратор и тренеры, а также множество специализированных залов.

В йога клубе существует некоторое множество направлений йоги, которое может дополняться и корректироваться. Каждому направлению соответствует свой тренер, который является специалистом в данной области, имеет соответствующее образование и опыт, а также свой специализированный для этого направления зал. Тренер проводит занятия в соответствии с расписанием. Клиенты клуба разделяются по группам, группы составляются в соответствии с выбранным направлением, уровнем физической подготовленности и возрастом клиента.

Посещение клуба регулируется системой абонементов, существует три вида абонементов:

* Разовое посещение.
* Абонемент на месяц.
* Абонемент на год.

Стоимость абонементов не зависит от выбранного направления, когда клиент покупает абонемент, он имеет право посещать любое из представленных направлений, а также менять их по своему усмотрению на период действия абонемента.

Общие представление работы йога клуба выглядит следующим образом:

1. Клиент записывается на сайте выбирая нужный абонемент.
2. Администратор принимает и обрабатывает заявки с сайта, уточняет информацию с клиентом по телефонному разговору и в соответствии с полученной информацией определяет клиента в группу и регулирует расписание.
3. Тренеры проводят занятия.
4. **Анализ инструментальных средств разработки**

Для создания информационной системы «Йога клуб» были определены следующие инструментальные средства разработки:

1. HTML.
2. CSS.
3. JavaScript.
4. Bootstrap.
5. Python.
6. Django.
7. SQLite.
8. PyCharm.

HTML — это язык разметки, который представляет простые правила оформления и компактный набор структурных и семантических элементов разметки. HTML позволяет описывать способ представления логических частей документа (заголовки, абзацы, списки и т.д.) и создавать веб-страницы разной сложности.

CSS — технология описания внешнего вида документа, оформленного языком разметки. Преимущественно используется как средство оформления веб-страниц в формате HTML и XHTML, но может применяться с любыми видами документов в формате XML, включая SVG и XUL. Каскадные таблицы стилей используются создателями веб-страниц для задания цветов, шрифтов, расположения и других аспектов представления веб-документа. Основной целью разработки CSS являлось разделение содержимого (написанного на HTML или другом языке разметки) и оформления документа (написанного на CSS). Это разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

JavaScript — это язык программирования, как правило, выполняется на стороне клиента. Он используется для взаимодействия с пользователем. Он также используется в разработке игр, настольных и мобильных приложений, в создании документов pdf и виджетов рабочего стола. Веб-браузеры имеют встроенную поддержку для этого языка.

Особенности: Основная уникальность этого языка в том, что он имеет поддержку почти у всех браузеров и имеет полную интеграцию с ними. Все действия с помощью JavaScript являются очень простыми. Нет никакой другой технологии, которая бы вмещала в себя все подобные преимущества. Например, есть не кросс-браузеры, они имеют поддержку не у всех обозревателей (это XUL, ActiveX, VBScript). Также есть и такие, которые не в полной степени интегрированы с браузером. Это Silverlight, Flash, Java. На данный момент идет большое развитие подобной технологии и в разработке находятся элементы JavaScript 2.

Bootstrap — это бесплатный фреймворк с открытым исходным кодом для создания веб-сайтов и веб-приложений. Это самый популярный фреймворк HTML, CSS и JS для разработки адаптивных и мобильных проектов в Интернете. Одним из основных преимуществ фреймворков Bootstrap, является то, что они ускоряют время разработки, сохраняя при этом качество и согласованность всего сайта. Больше не нужно переделывать каждый элемент и тратить часы на то, чтобы всё выглядело и работало правильно на разных платформах, устройствах и в браузерах.

Python — это высокоуровневый язык программирования, который используется в различных сферах IT, таких как машинное обучение, разработка приложений, web, парсинг и другие.

Преимущества:

* Кроссплатформенность.
* Большое количество библиотек.
* динамическая типизация.
* Недостатки:
* Медлительность.
* Высокое потребление памяти.
* Строгая привязанность к системным библиотекам.

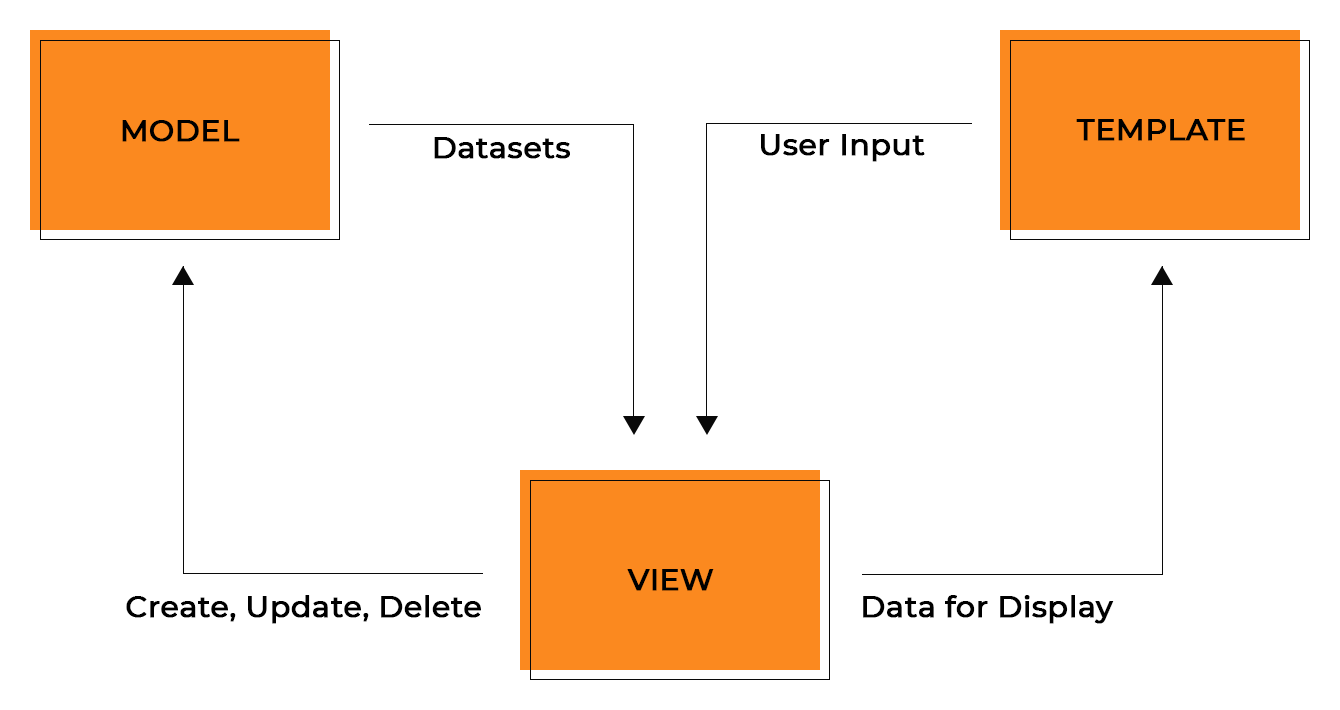
Django — это высокоуровневый Python веб-фреймворк, который позволяет быстро создавать безопасные и поддерживаемые веб-сайты.  
Создатели реализовали в Django паттерн MVT (модель-представление-шаблон), и он применяется в текущей версии фреймворка. Документация Django определяет модель (model) как «источник информации о данных, в которых содержатся ключевые поля и поведение данных». Обычно одна модель указывает на одну таблицу в базе данных. Django поддерживает базы данных PostgreSQL, MySQL, SQLite и Oracle. Представление решает три задачи: принимает HTTP-запросы, реализует бизнес-логику, определённую методами и свойствами, отправляет HTTP-ответ в ответ на запросы. То есть представление получает данные от модели и предоставляет шаблонам доступ к этим данным или предварительно обрабатывает данные и затем предоставляет к ним доступ шаблонам. Шаблоны представляют собой файлы с HTML-кодом, с помощью которого отображаются данные. Содержимое файлов может быть статическим или динамическим. Шаблоны не содержат бизнес-логики. Поэтому они только отображают данные. На рисунке 1 представлена схема, отображающая общее устройство паттерна MVT.

Рисунок 1 – Схема MVT

SQLite — компактная встраиваемая реляционная система баз данных. Является чисто реляционной системой баз данных. SQLite представляет библиотеку, которая написанна на языке C (ANSI-C) и которая реализует движок реляционных баз данных.На сегодняшний день SQLite, возможно, самая используемая система баз данных. Так, ее бд можно найти в каждом устройстве на Android, iOS, Mac, Windows 10/11, ее используются большинство распространенных браузеров - Firefox, Chrome, Safari и т.д. В отличие от других систем баз данных, как MS SQL Server, MySQL, Postgres и т.д., для SQLite не требуется сервер базы данных. SQLite представляет встраиваемый движок базы данных, который обращается напрямую к файлу базы данных на диске. Сооветственно для работы с базами данных не нужно явным образом устанавливать или как-то конфигурировать SQLite. SQLite имеет полноценную поддержку большинства возможностей, которыми обладают другие реляционные СУБД - таблицы, индексы, триггеры, представления.

PyCharm — это интегрированная среда разработки для Python, которая имеет полный комплект средств, необходимых для эффективного программирования на Python. Сейчас PyCharm распространяется в двух вариантах: платном (PyCharm Professional Edition) и бесплатном (PyCharm Community Edition). Бесплатная версия имеет открытый исходный код и распространяется под лицензией Apache 2. Это облегченная среда, которая подходит для разработки только на Python. Платный вариант представляет собой более расширенную и функциональную версию с возможностью разработки в том числе многоязычных веб-приложений, и дает возможность удаленной разработки, а также работы с базами данных.

Преимущества:

* PyCharm имеет удобный редактор кода со всеми полезными функциями: подсветкой синтаксиса, автоматическим форматированием, дополнением и отступами. PyCharm позволяет проверять версии интерпретатора языка на совместимость, а также использовать шаблоны кода.
* PyCharm позволяет быстро производить рефакторинг кода, а также использовать удобный графический отладчик.
* Утилита поддерживает все свежие версии Django, а также IronPython, Jython, Cython, PyPy wxPython, PyQt, PyGTK и многие другие инструменты.
* В PyCharm можно проводить интегрированное Unit тестирование, использовать интерактивные консоли для Python, Django, SSH, отладчика и баз данных.
* PyCharm имеет большую коллекцию плагинов, и его можно использовать в связке с разными трекерами вроде JIRA, Youtrack, Lighthouse, Redmine, Trac и так далее.
* PyCharm кросс-платформенная среда разработки: можно использовать на Linux, Windows и Mac OS.

1. **Техническое задание**

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Согласно ГОСТ 19 техническое задание должно включать следующие разделы:

1. общие сведения;
2. назначение и цели создания системы;
3. требования к системе в целом;
   1. требования к структуре и функционированию системы;
   2. требования к надежности;
   3. требования к безопасности;
   4. требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
4. требования к документированию;
5. состав и содержание работ по созданию системы.

Техническое задание на разработку приложения представлено в приложении А.

1. **Проектирование ИС**

Проектирование информационной системы позволяет разработчику получить структурированный чётки план действий, сокращает время разработки и повышает её качество, а также позволяет предусмотреть любые другие нюансы разработки.

Большая часть проектирования производится с помощью специальных схем и диаграмм, которые будут представлены далее.

* 1. **Структурная схема ИС**

Диаграмма прецедентов, представленная на рисунке 2, описывает какой функционал разрабатываемой информационной системы доступен каждой группе пользователей.

В данной информационной системе выделяется две группы пользователей: Клиенты и Персонал. Каждая из групп пользователей может пользоваться информационной системой по-своему.

****

Рисунок 2 – Диаграмма прецедентов

Диаграмма деятельности, изображённая на рисунке 3, представляет собой графическую схему рабочих процессов. Диаграмма деятельности используется для изучения бизнес-процессов с целью определения их потока и требований.

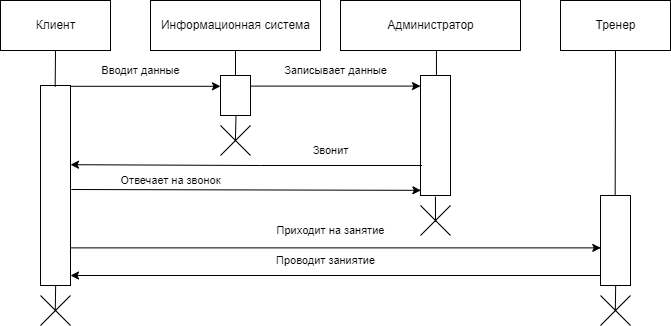


Рисунок 3 – Диаграмма деятельности

На рисунке 4 представлена диаграмма компонентов, которая визуализирует как компоненты информационной системой соединяются между собой для формирования более крупных компонентов или программных подсистем.

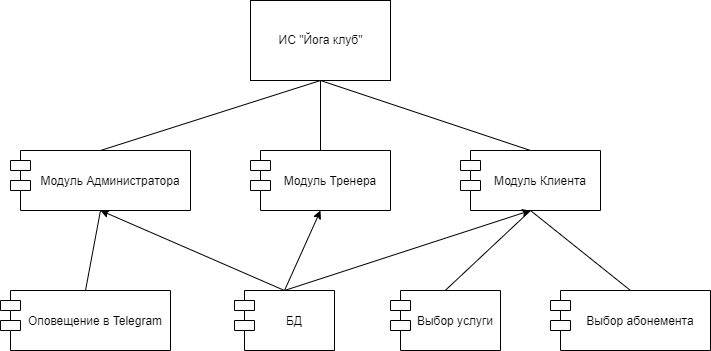


Рисунок 4 – Диаграмма компонентов

На рисунке 5 изображена диаграмма развёртывания, которая содержит графическое изображение устройств, процессов и связей между ними.

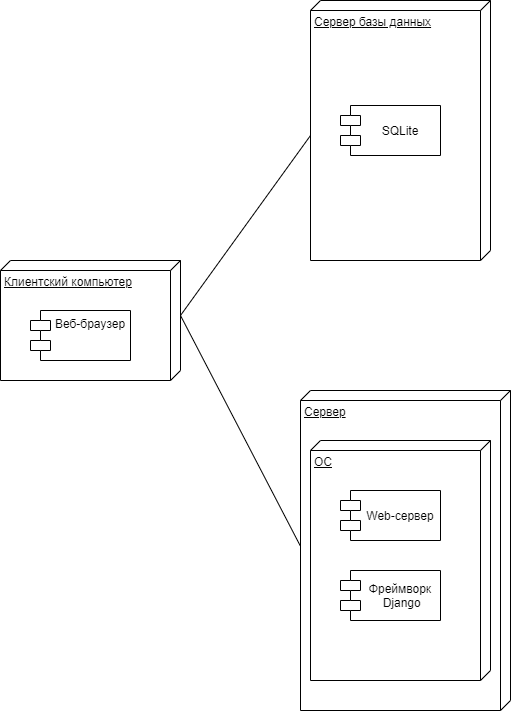


Рисунок 5 – Диаграмма развёртывания

* 1. **Функциональная схема ИС**

Контекстная диаграмма IDEF0, изображённая на рисунке 6, представляет систему как набор иерархических действий, в которой каждое действие преобразует некоторый объект или набор объектов.

Каждая IDEF0-диаграмма содержит блоки и дуги. Блоки изображают функции моделируемой системы. Дуги связывают блоки вместе и отобра­жают взаимодействия и взаимосвязи между ними.

На рисунке 7 представлена диаграмма декомпозиции, которая разбивает контекстную диаграмму на более детализированные компоненты более низкого уровня.

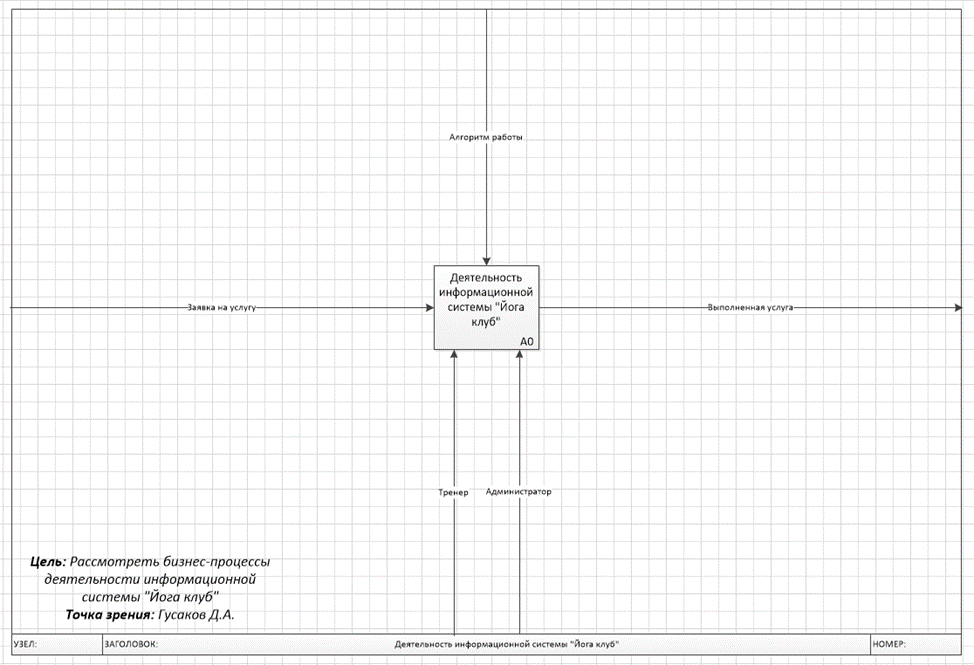
****

Рисунок 6 – Контекстная диаграмма IDEF0

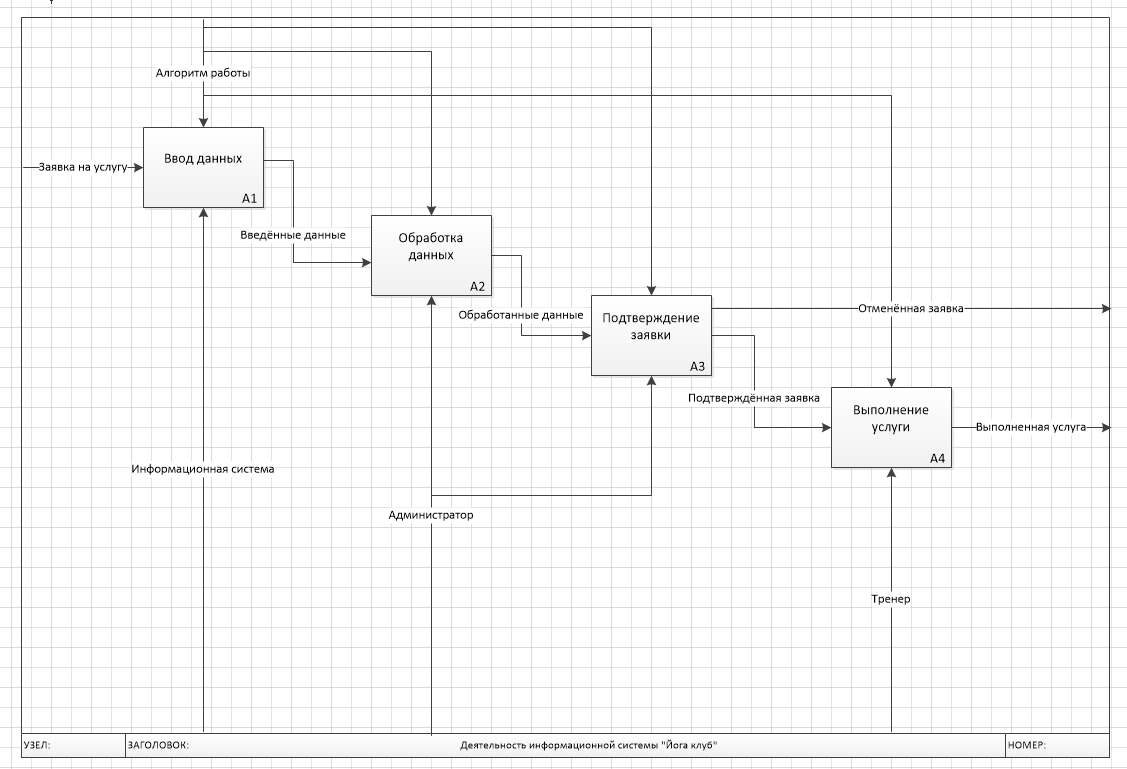


Рисунок 7 – Диаграмма декомпозиции

Диаграмма классов, изображённая на рисунке 8, предназначена для представления внутренней структуры программы в виде классов и связей между ними.

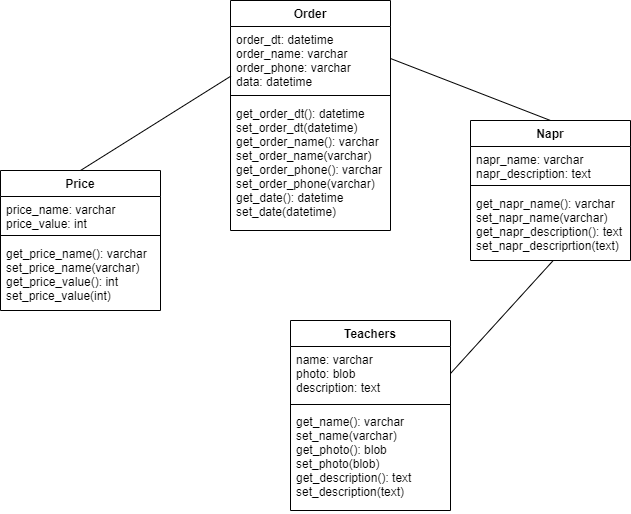


Рисунок 8 – Диаграмма классов

На рисунке 9 представлена диаграма потоков данных, предназначенная для моделированния информоционных систем с точки зрения действий над данными.

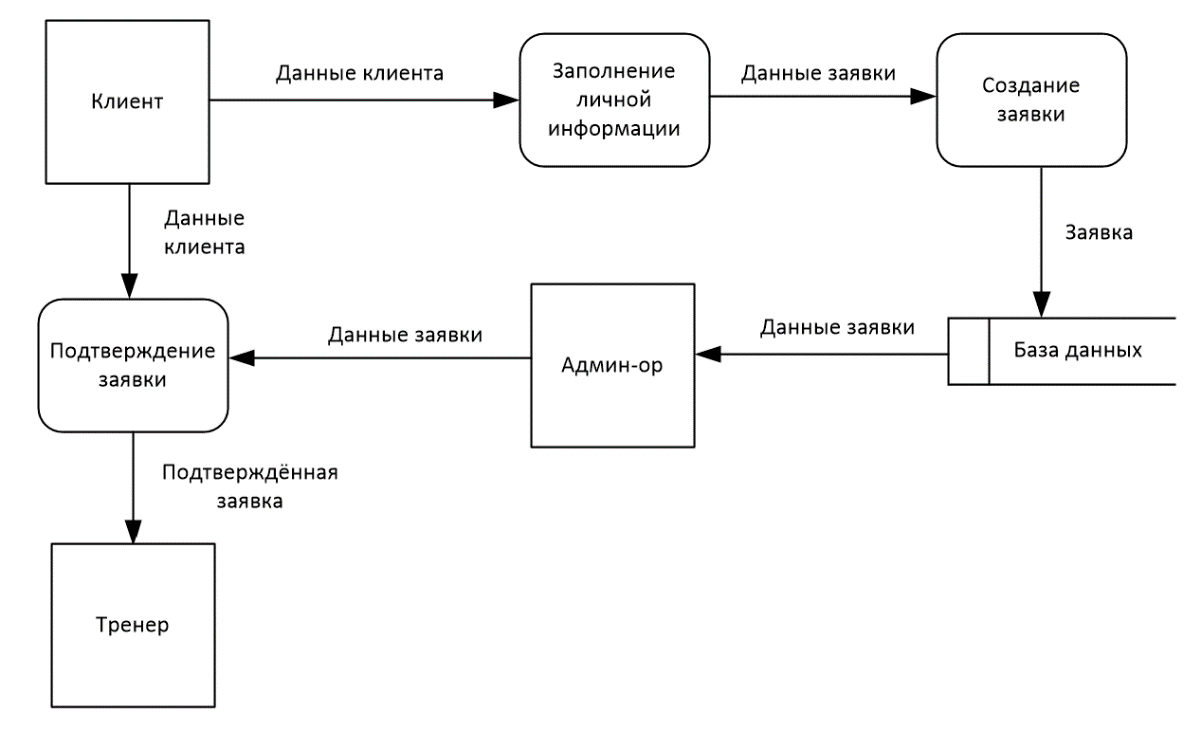


Рисунок 9 - Диаграмма потоков данных

* 1. **Проектирование базы данных**

Инфологическая модель, изображённая на рисунке 10, представляет собой отображение предметной области в виде совокупности информационных объектов и структурных связей

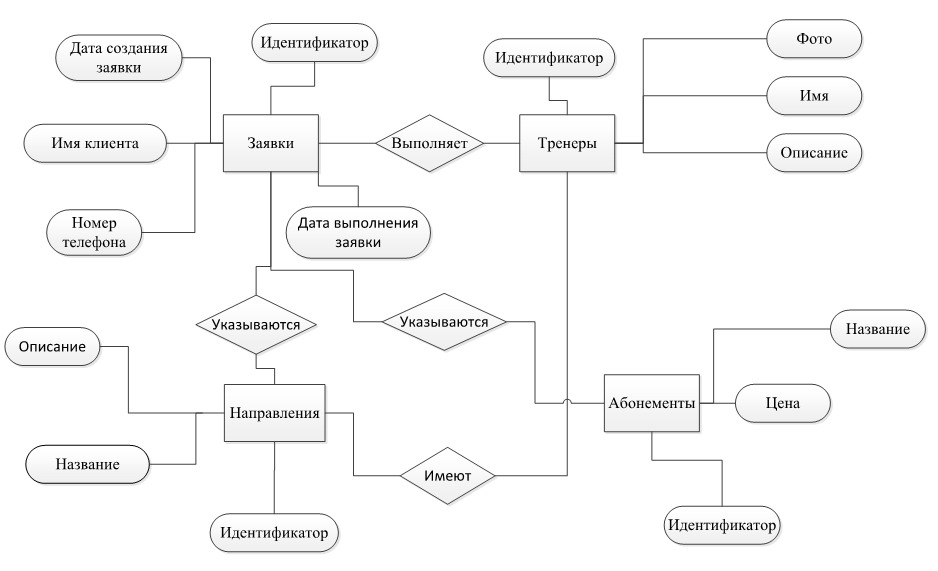


Рисунок 10 – Инфологическая модель

Даталогическая модель, изображённая на рисунке 11, представляет собой модель логического уровня системы, представляющая собой отображение логических связей между элементами базы данных в среде конкретной СУБД.

Исходными данными для даталогического проектирования является инфологическая модель предметной области.

Конечным результатом даталогического проектирования является логическая структура базы данных.

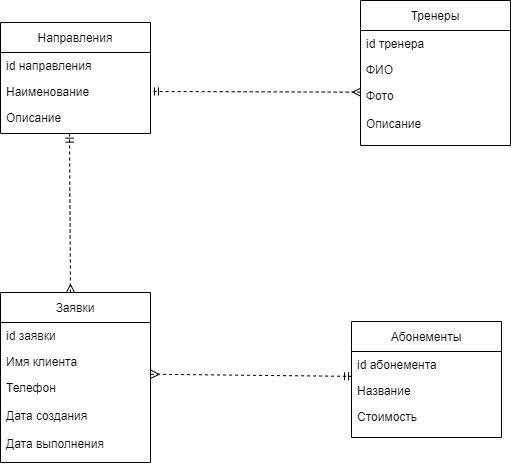


Рисунок 11 – Даталогическая модель

На рисунке 12 изображена ER-модель, позволяющая описывать концептуальные схемы предметной области.

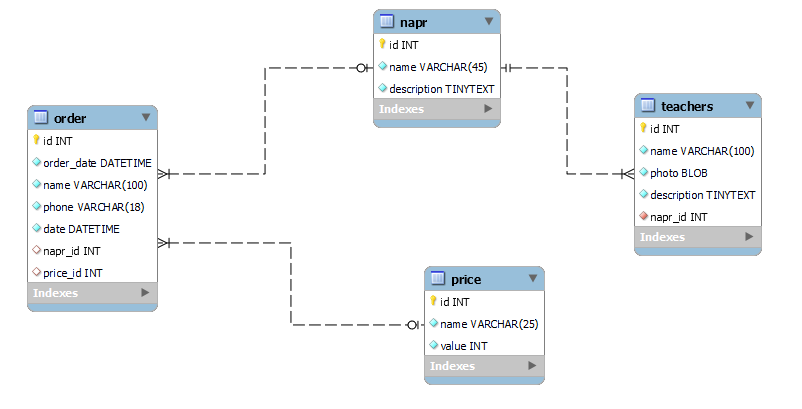


Рисунок 11 – ER-модель

На схеме базы данных представлены 4 таблицы: «order», «price», «napr» и «teachers». Поля id являются первичными ключами.

В таблицах 1, 2, 3, 4 представлены описания полей с типами данных для каждой таблицы данной ER-модели базы данных.

Таблица 1 – «order»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT | Уникальный идентификатор заявки |
| order\_date | DATETIME | Дата создания заявки |
| name | VARCHAR(100) | Имя клиента |
| phone | VARCHAR(18) | Номер телефона |
| date | DATETIME | Дата выполнения заявки |
| napr\_id | INT | Идентификатор направления (внешний ключ) |
| price\_id | INT | Идентификатор абонемента (внешний ключ) |

Таблица 2 – «napr»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT | Уникальный идентификатор направления |
| name | VARCHAR(45) | Наименование направления |
| description | TINYTEXT | Описание направления |

Таблица 3 – «price»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT | Уникальный идентификатор абонемента |
| name | VARCHAR(25) | Наименование абонемента |
| value | INT | Стоимость |

Таблица 3 – «price»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id | INT | Уникальный идентификатор тренера |
| name | VARCHAR(100) | Имя тренера |
| photo | BLOB | Фотография тренера |
| description | TINYTEXT | Описание тренера |
| napr\_id | INT | Идентификатор направления (внешний ключ) |

База данных приведена к 3-ой нормальной форме так как все поля, принимаемые больше одного значения, декомпозированы, и каждый не ключевой атрибут не транзитивно зависит от первичного ключа.

* 1. **Проектирование интерфейса**

Для проектирования интерфейса был выбран инструмент Figma – бесплатный онлайн-сервис для создания прототипов сайтов или приложений.

В результате проектирования интерфейса были созданы прототипы трёх страниц: «Главная» (рисунок 13), «Тренеры» (рисунок 14), «Цены» (рисунок 15).

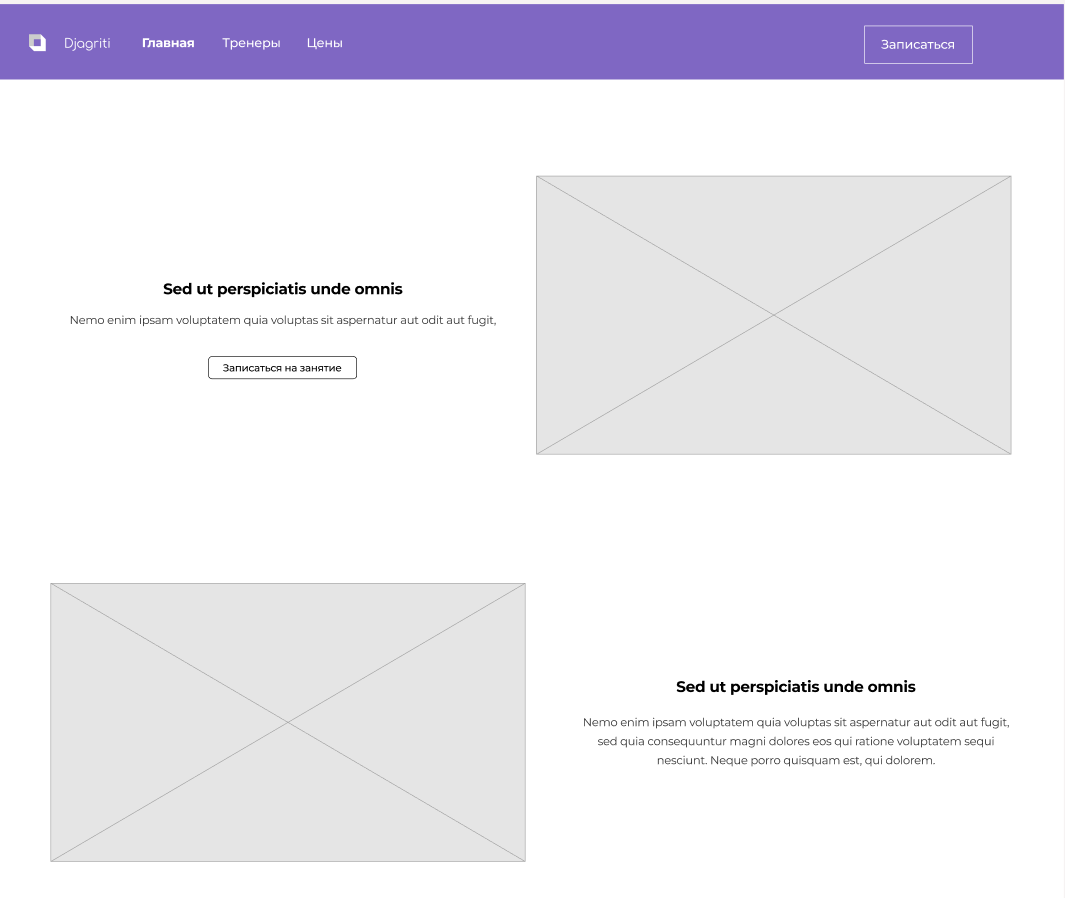


Рисунок 13 – Страница «Главная»

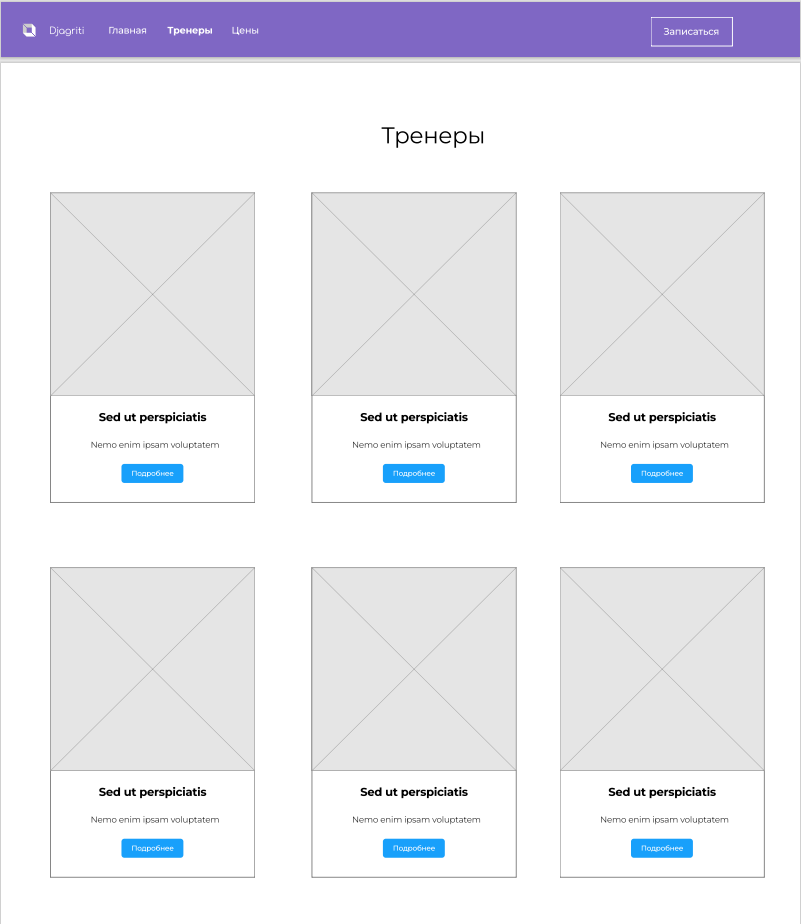


Рисунок 14 – Страница «Тренеры»

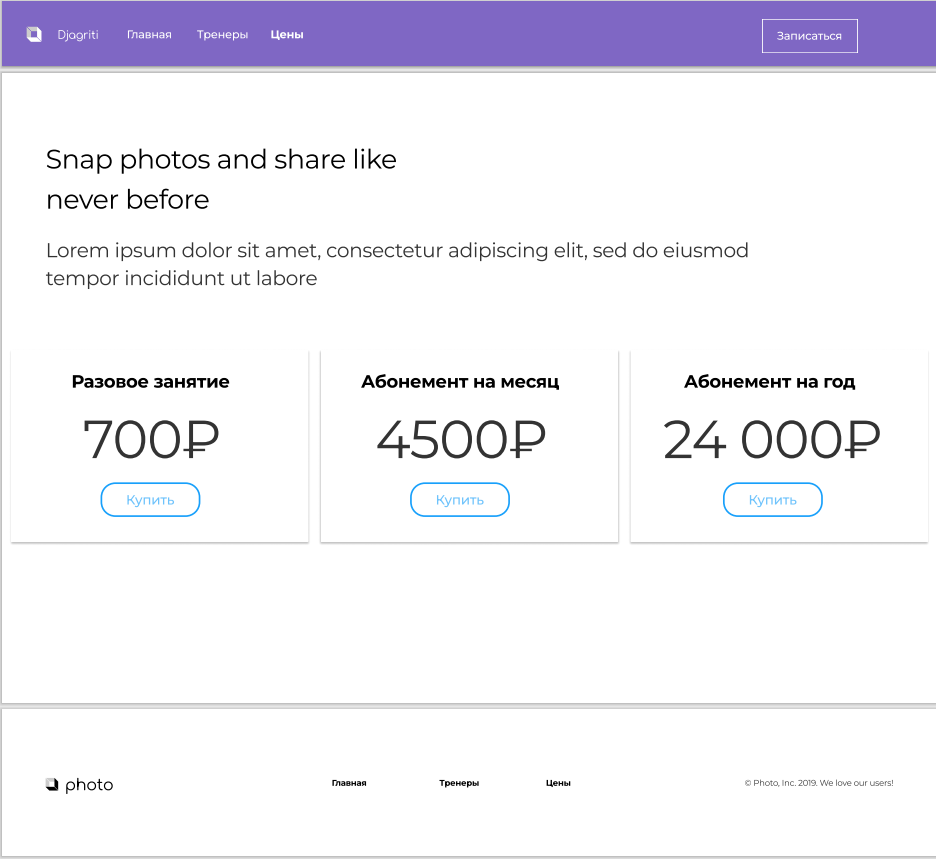


Рисунок 15 – Страница «Цены»

1. **Разработка ИС**
   1. **Разработка интерфейса ИС**

Для разработки пользовательского интерфейса были использованы такие инструменты как: HTML, CSS, Bootstrap.

В разрабатываемой информационной системе файлы, содержащие описание внешнего вида страниц, представлены в формате \*.html и \*.сss.

В фреймворке Django файлы формата \*.html называются «Шаблонами» и располагаются в каталоге «templates», а файлы формата \*.сss обычно располагаются в каталоге «static» и подключаются в html шаблонам посредством специального тега «{% load static %}»

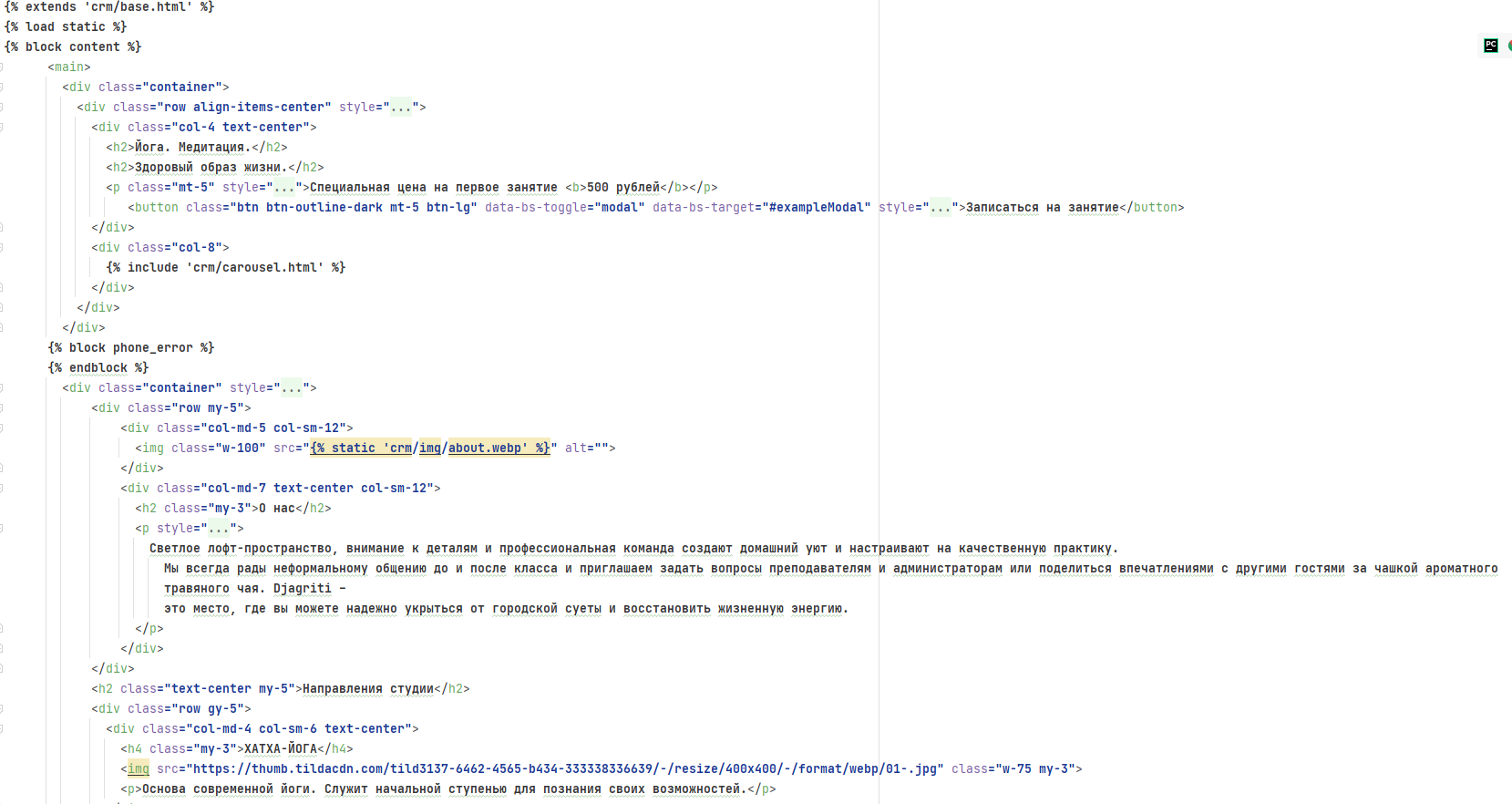


Рисунок 12 – HTML шаблон главной страницы

С помощью Bootstrap и CSS стилей были реализованы страницы: «Главная», «Тренеры», «Цены», «Авторизация», «Панель управления», а также форма для записи.

На рисунке 13 представлена страница «Главная», на которой отображается основная информация о работе Йога клуба.

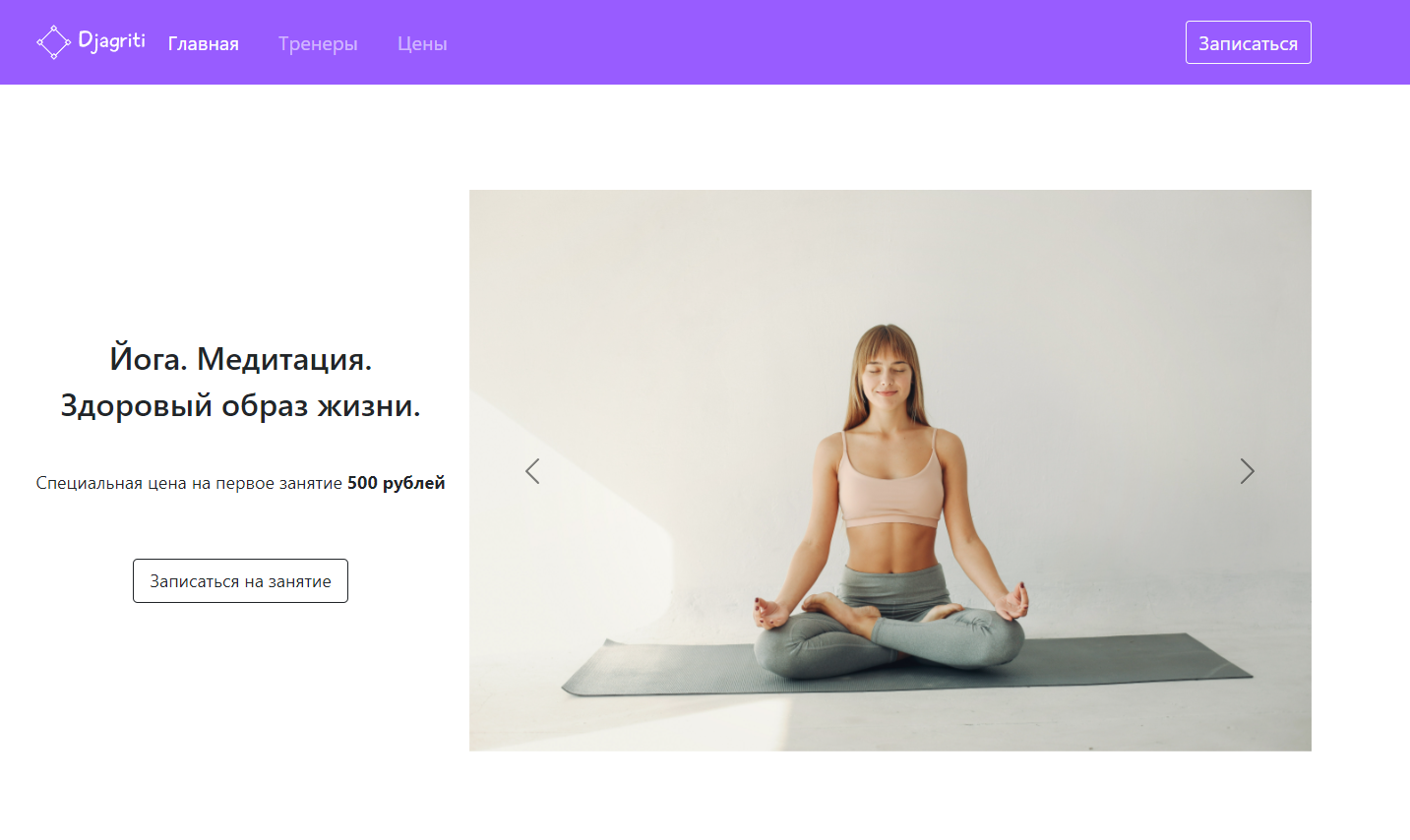


Рисунок 13 – Страница «Главная»

На рисунке 14 изображена страница «Тренеры», на которой представлена информация о работающих в йога клубе тренерах и их направлениях. А также предоставлена возможность чтения более подробной информации о каждом тренере.

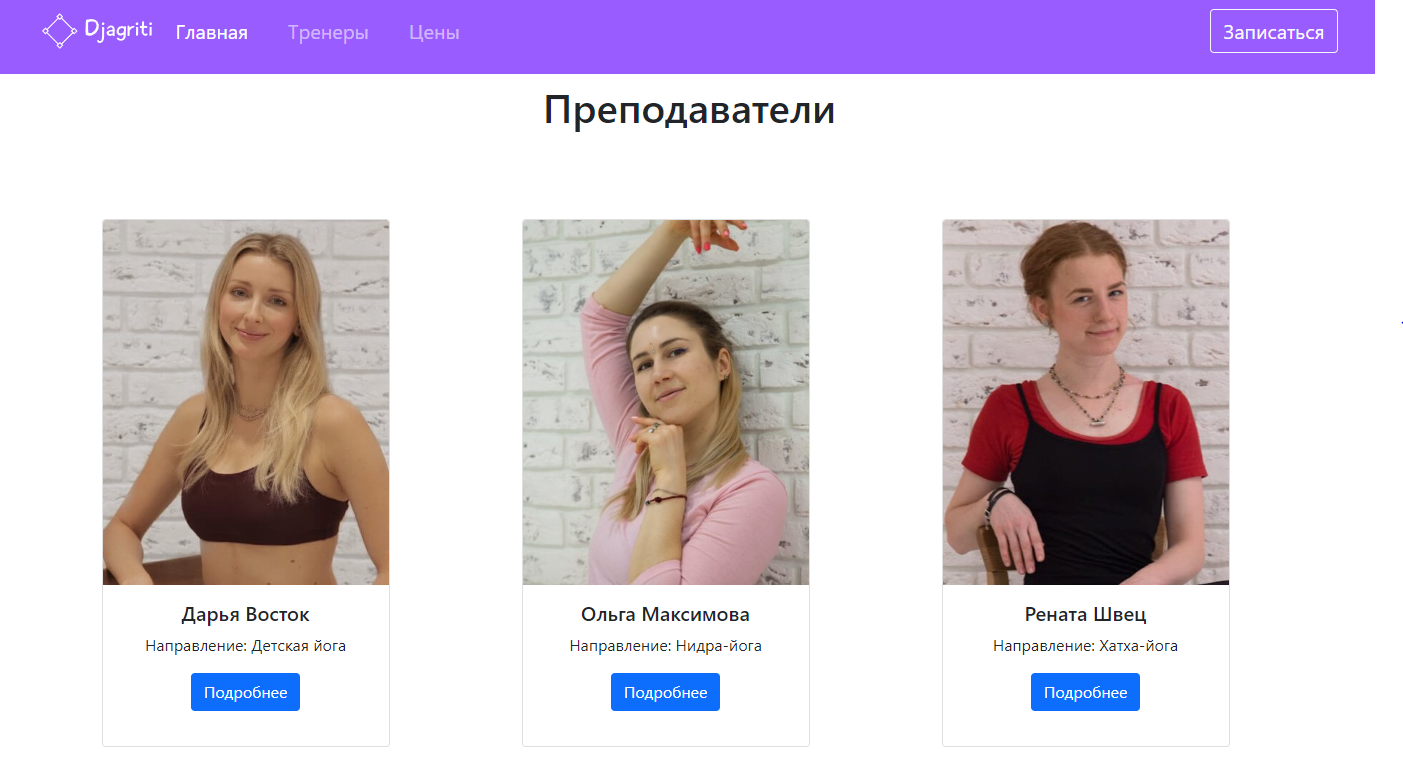


Рисунок 14 – Страница «Тренеры»

На странице «Цены» представлена информация о ценах на каждый существующий в йога клубе абонемент (рисунок 15).

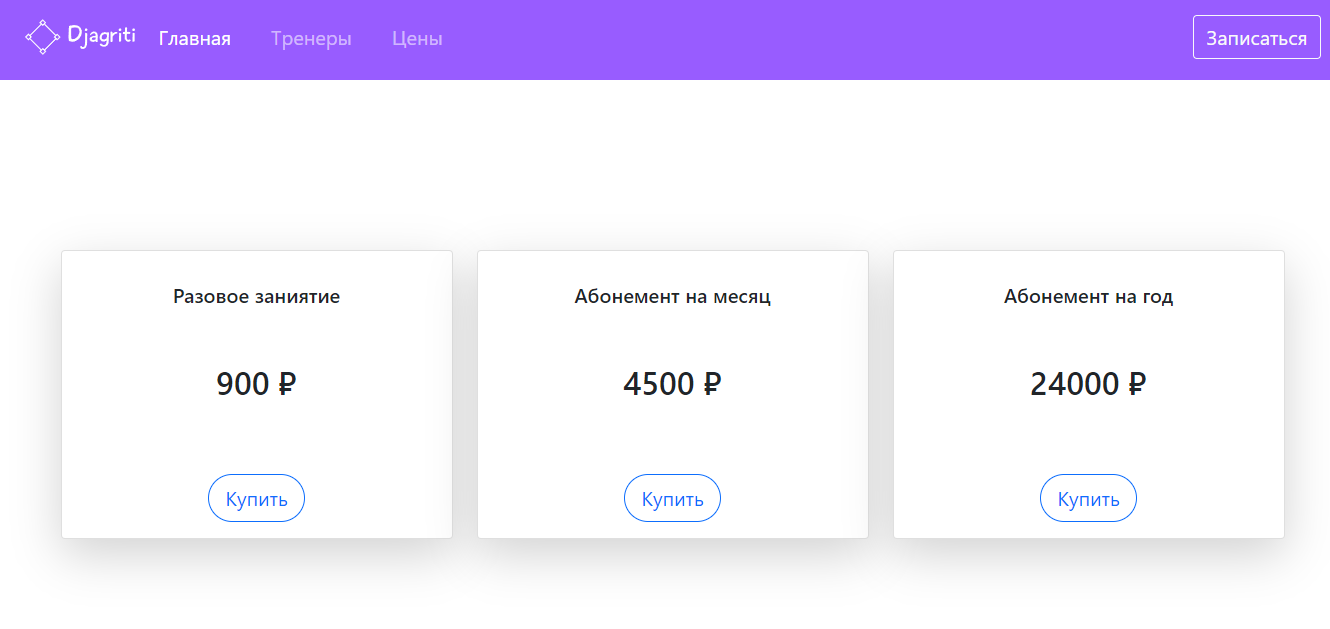


Рисунок 15 – Страница «Цены»

На странице «Авторизация» предоставлена возможность авторизации для входа на панель администрирования c помощью двух полей для ввода и кнопки «Войти» (рисунок 16).

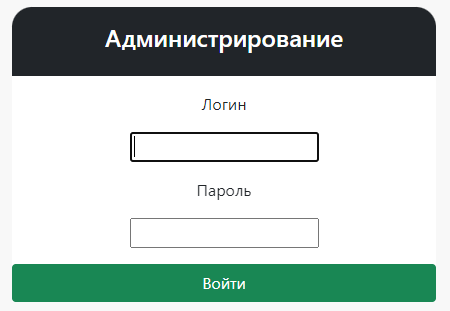


Рисунок 16 – Страница «Авторизации»

На рисунке 17 представлена панель управления, которая предоставляет возможность администратору информационной системы изменять и удалять существующие пользовательские заявки. А также изменять, удалять и добавлять информацию о тренерах, ценах, направлениях.

В информационной системе реализовано разграничение доступа. Существует два вида администраторов: администратор и суперадминистратор. При входе в учётную запись суперадминистратора появляется дополнительная вкладка «Администраторы», при переходе в которую появляется возможность добавления новых администраторов и удаления существующих.

Также на панели администрирования есть кнопка «Скачать отчёт» при нажатии на которую начинается загрузка файла формата \*.xlsx, в котором содержится отчёт работы йога клуба за текущий месяц.

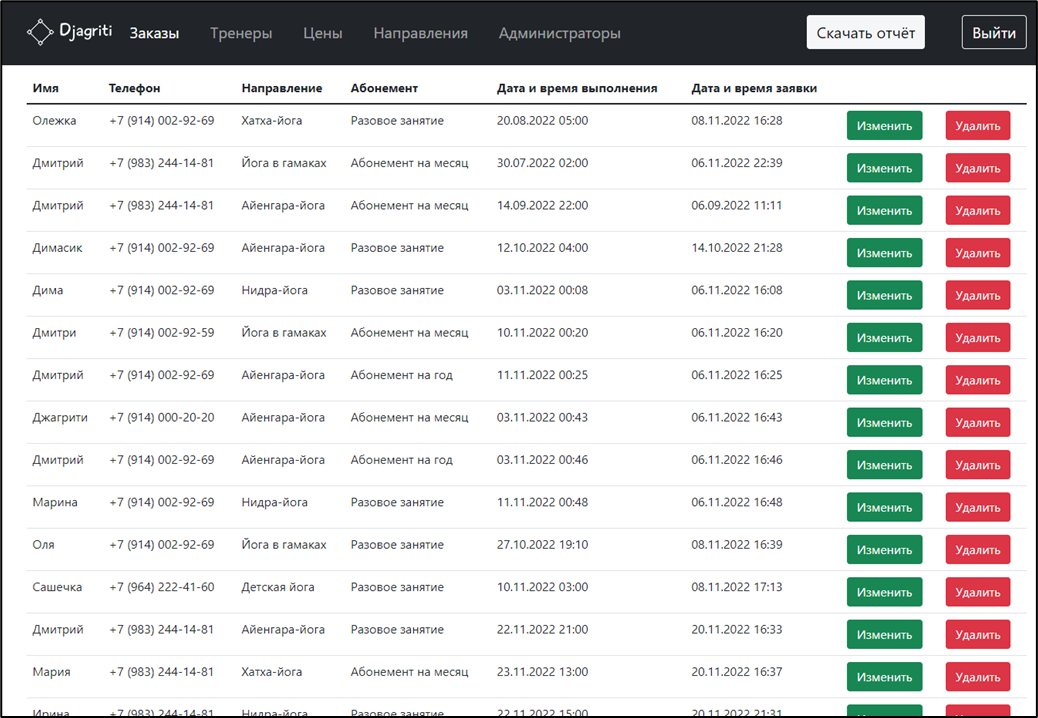


Рисунок 17 – Панель управления

На рисунке 18 изображена форма для записи, в которой клиент может записаться на занятие в йога клубе.

При записи на занятие клиент вводит своё имя, номер телефона, выбирает направление и абонемент, а также нужную ему дату и время. Клиент может не выбирать направление и абонемент, если не знает точно, куда он хочет. В этом случае, администратор свяжется с ним, чтобы предоставить рекомендации.

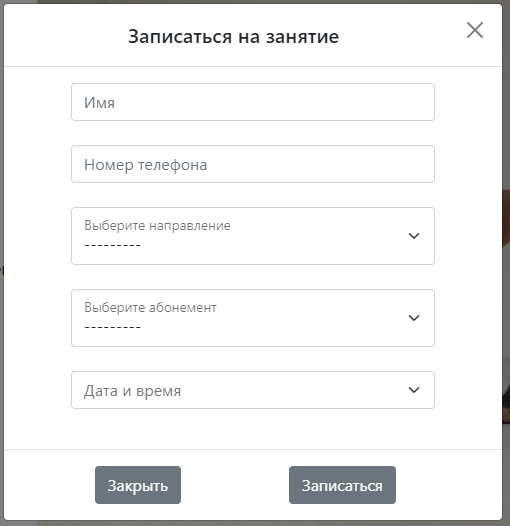


Рисунок 18 – Форма для записи

* 1. **Разработка базы данных ИС**

Разработка базы данных информационной системы «Йога клуб» реализовывалась в СУБД SQLite. База данных состоит из 4 таблиц.

В фреймворке Django используется ORM (объектно-реляционное отображение), которое позволяет представить описание таблиц базы данных в виде классов на языке Python. Все подобные классы, согласно архитектуре MVT, называются «моделями» и располагаются в файлах models.py и вместе образуют структуру базы данных.

Каждое свойство таких классов представляет собой описание соответствующего поля в таблице базы данных.

На рисунках 19-22 представлены модели, формирующие таблицы базы данных информационной системы.

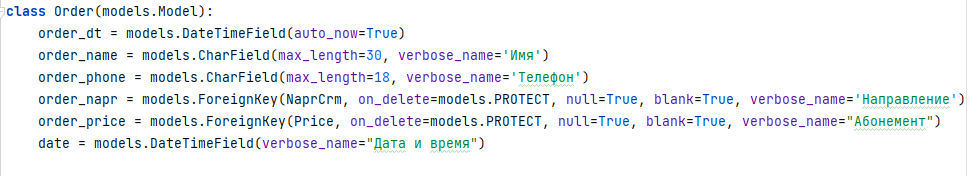


Рисунок 19 – Модель Order

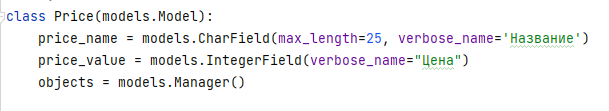


Рисунок 20 – Модель Price

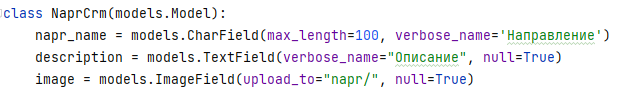


Рисунок 21 – Модель NaprCrm

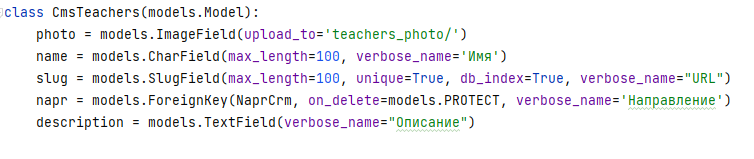


Рисунок 22 – Модель СmsTeacher

* 1. **Разработка ИС**

Подключение к базе данных SQLite осуществляется в конфигурационном файле settings.py (рисунок 23).

Переменная DATABASES содержит набор конфигураций подключений к базам данных в виде словаря. Ключи в этом словаре – названия подключений.

Конфигурация каждого подключения может состоять из ряда параметров. По умолчанию указываются только два параметра. Параметр ENGINE указывает на используемый движок для доступа к БД. В данном случае это встроенный пакет django.db.backends.sqlite3.

Второй параметр - NAME указывает на путь к базе данных. Для установки пути используется каталог из переменной BASE\_DIR, которая задана в начале файла:

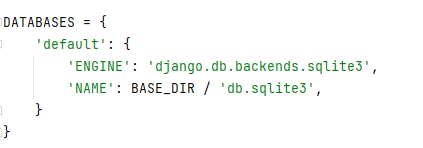


Рисунок 23 – Подключение к базе данных

Так как все таблицы базы данных в Django представлены в виде классов, то каждая отдельная запись в таблице является объектом, и поэтому запросы на выборку, добавление, удаление данных из базы осуществляются с помощью специальных методов, описанных в родительском классе Model.

В функции dashboard осуществлён запрос на выборку всех данных из таблицы Order c помощью метода all() (рисунок 24).

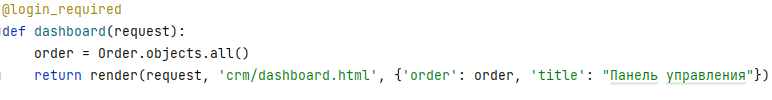


Рисунок 24 – Запрос к базе данных

Подобные функции в Django называются «представлениями», и служат для того, чтобы принимать запросы, и отправлять ответы в виде HTML страниц.

Все функции представления согласно архитектуре MVT располагаются в файле views.py, который представлен в приложении Б.

1. **Документирование программного продукта**
   1. **Руководство пользователя**

Для того, чтобы открыть информационную систему, необходимо запустить локальный веб-сервер. Для этого необходимо установить с официального сайта https://www.python.org/ интерпретатор Python версии 3.10

Далее необходимо установить виртуальное окружение для проекта. Для этого в PyСharm на верхней панели необходимо открыть вкладку «File», затем нажать на «Settings». Появится окно настройки проекта (рисунок 25).

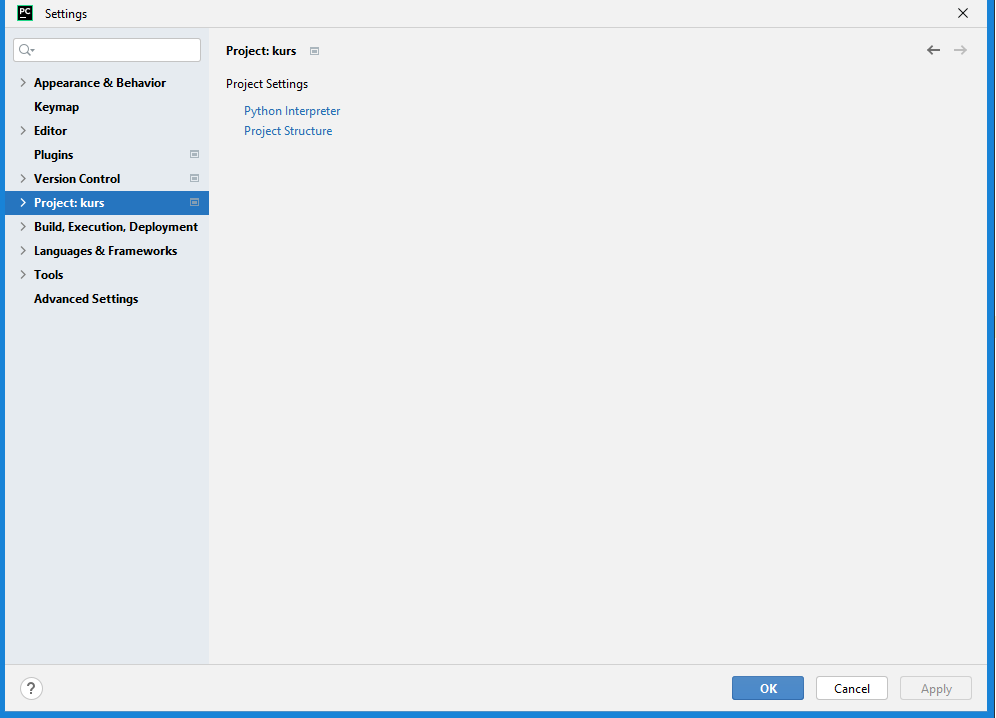


Рисунок 25 – Окно настройки проекта

Далее нужно перейти во вкладку «Project» и нажать на ссылку «Python Interpreter», появится окно выбора интерпретатора (рисунок 26).

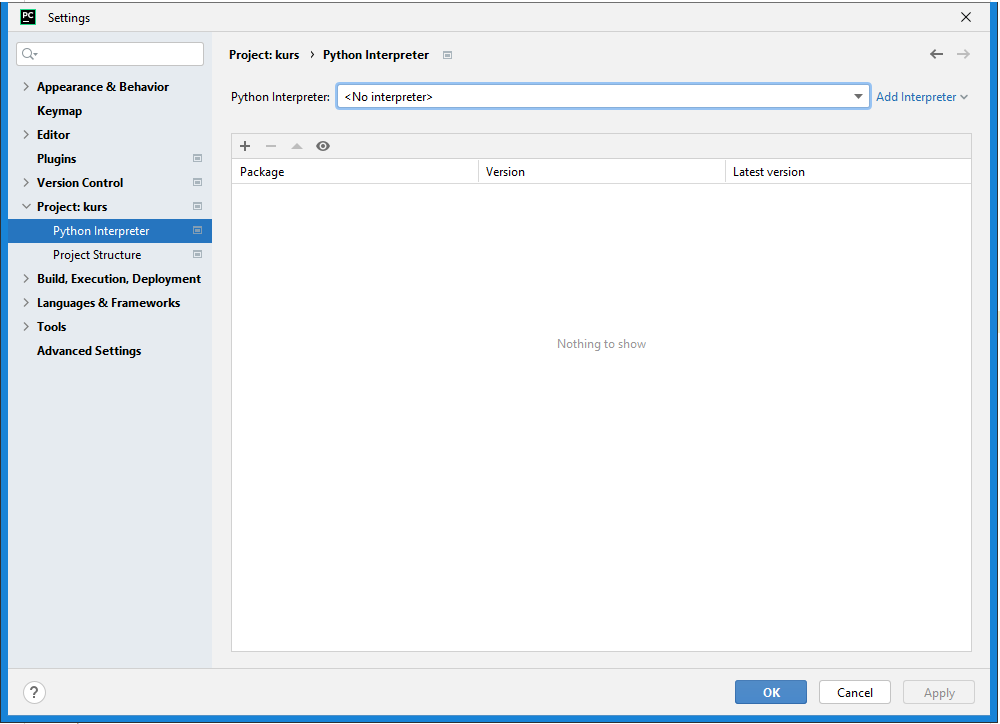


Рисунок 26 – Окна выбора интерпретатора.

Так как нет ни одного существующего виртуального окружения, необходимо его добавить, для этого нужно нажать на ссылку «Add Interpreter», появится окно добавления виртуального окружения (рисунок 27). На котором необходимо выбрать «New» и нажать кнопку «OK».

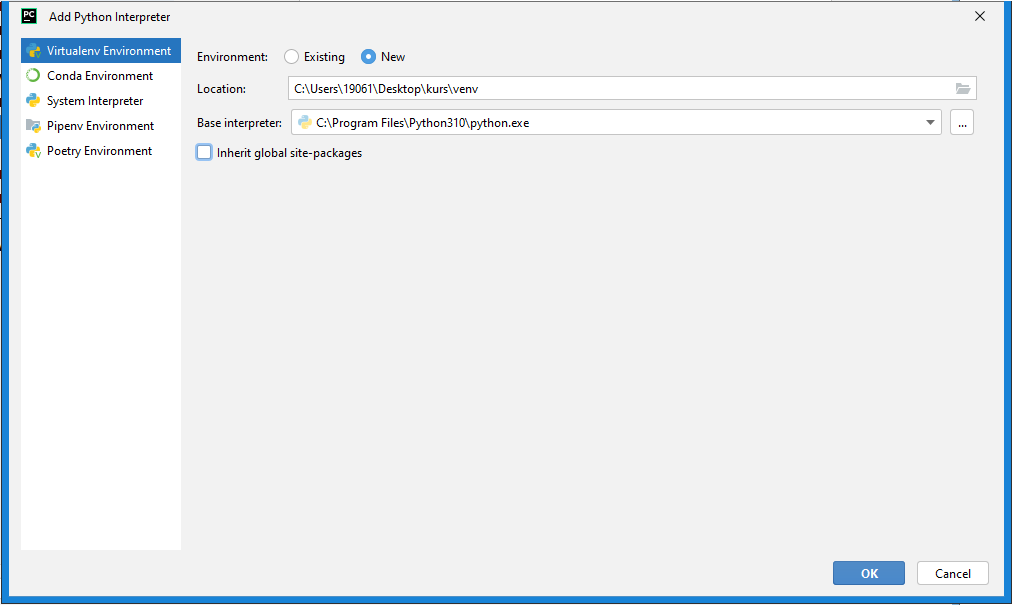


Рисунок 27 – Окно добавления виртуального окружения

После всех проделанных действий в корневой папке проекта появится папка «venv», в которой и будет храниться виртуальное окружение.

Далее необходимо установить в проект все нужные библиотеки и фреймворки. С этой целью был создан текстовый файл requirements.txt, в котором были прописаны все нужные библиотеки. Для того чтобы установить библиотеки из этого файла, в PyCharm необходимо открыть терминал и прописать команду pip install –r requirements.txt.

После того как все необходимые пакеты были установлены, можно приступать к запуску встроенного в Django тестового веб-сервера. Для этого необходимо в терминале прописать команду python manage.py runserver.

После прохождения всех предыдущих шагов проект можно будет открыть по адресу: <http://127.0.0.1:8000/>. При переходе по данной ссылке откроется стартовая страница, на которой представлена основная информация о йога клубе (рисунок 28).

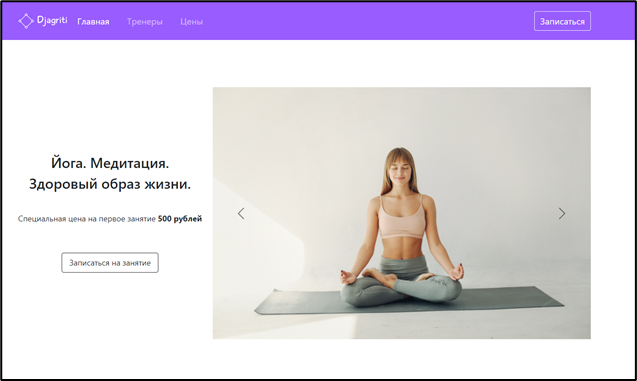


Рисунок 28 – Стартовая страница

Для того чтобы записаться на занятие на каждой веб-странице проекта существует кнопка «Записаться», при нажатии которую появится форма для ввода данных (рисунок 29).

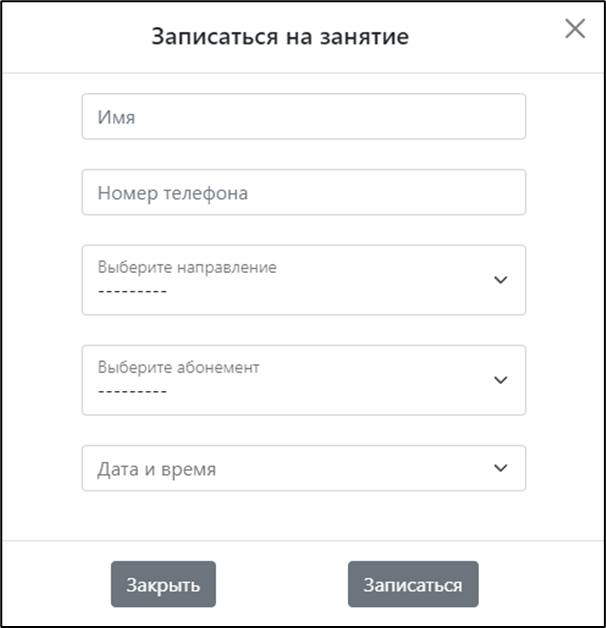


Рисунок 29 – Форма для записи на занятие

После того как данные заполнены, необходимо нажать на кнопку «Записаться» расположенные внизу формы

Поля «Имя», «Номер телефона» и «Дата и время» являются обязательными для заполнения, в том случае если какое-либо из этих полей было пропущено, появляется предупреждение (рисунок 29).

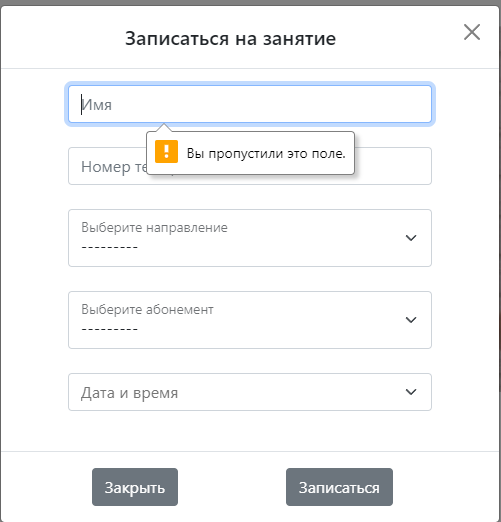


Рисунок 29 – Предупреждение о пропущенном поле

В случае если был указан несуществующий номер телефона, то появится сообщение об ошибке (рисунок 30).

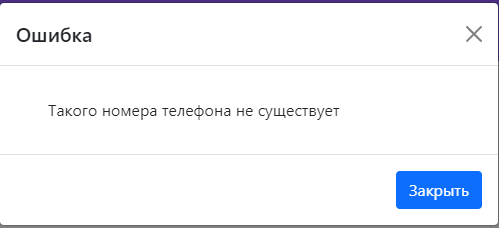


Рисунок 30 – Сообщение об ошибке при вводе несуществующего номера телефона

В случае если запись на занятие была произведена успешно, появится сообщение с благодарностью (рисунок 31), а также в чат в администраторов в мессенджере Telegram будет отправлено уведомление о новой заявке с сайта, содержащее информацию об имени и номера телефона клиента (рисунок 32).

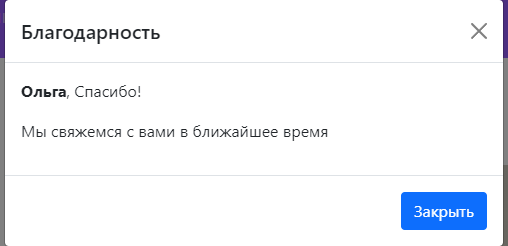


Рисунок 31 – Сообщение с благодарностью

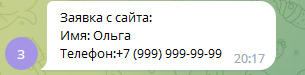


Рисунок 32 – Сообщение с благодарностью

Внизу каждой веб-страницы проекта расположена навигационная панель, в которой есть ссылка «Администратор», нажав на неё, появится страница авторизация для входа на панель администрирования (рисунок 33).

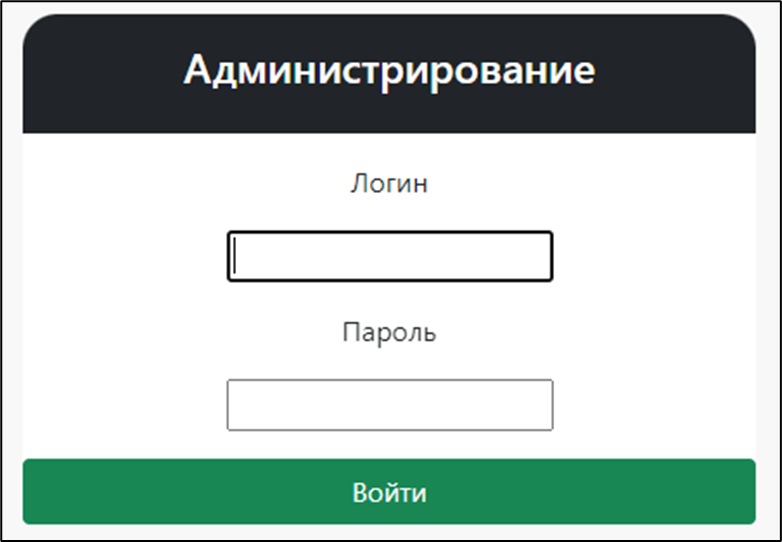


Рисунок 33 – Авторизация

В случае если данные были введены корректно, то при нажатии на кнопку «Войти» появляется страница панели администрирования (рисунок 33).

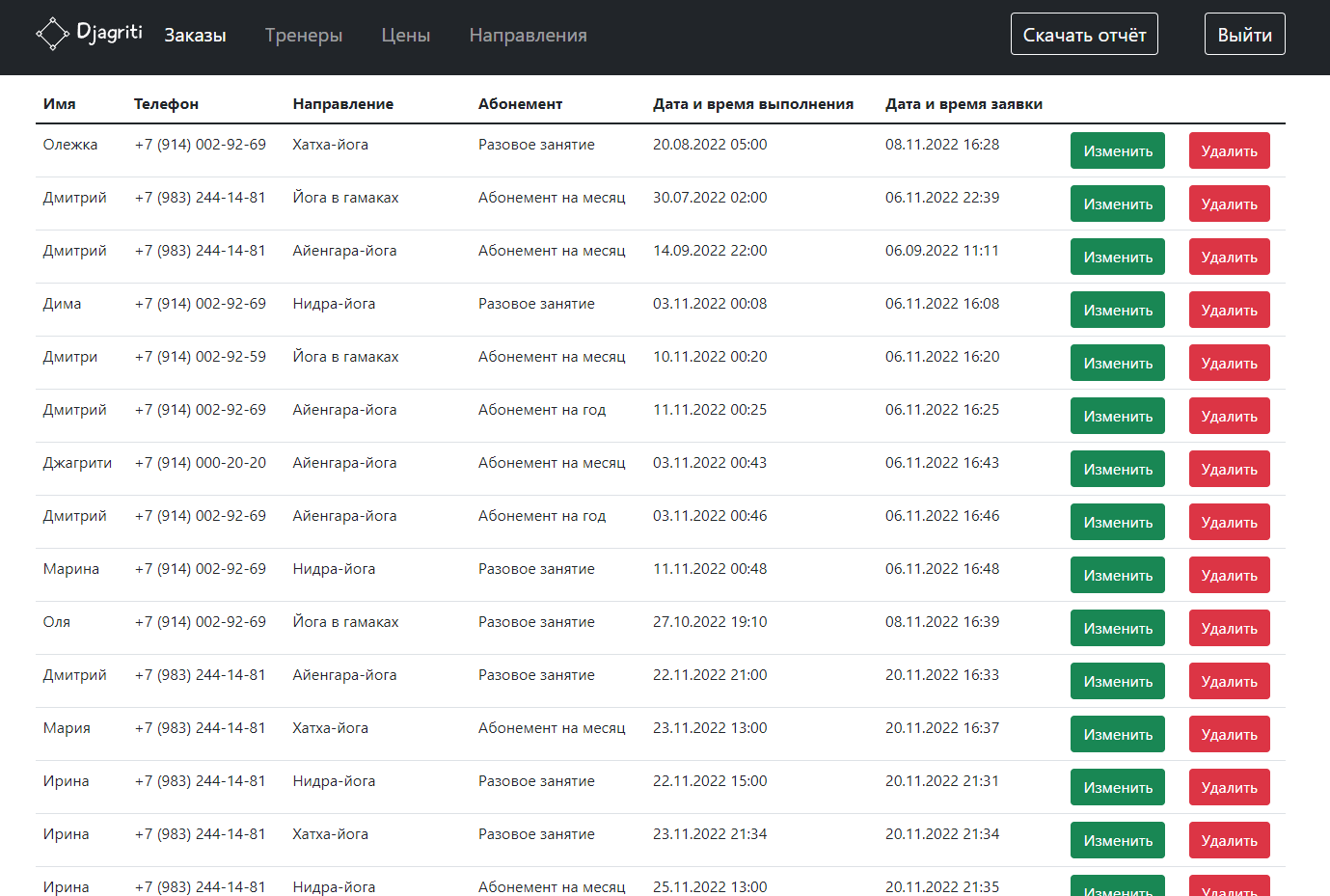


Рисунок 33 – Панель администрирования

При нажатии на кнопку «Удалить» удаляется выбранная запись из базы данных, при нажатии на кнопку «Изменить» появляется форма, на которой можно редактировать информацию, представленную в выбранной записи (рисунок 34), чтобы подтвердить изменения необходимо нажать на кнопку «Изменить» на форме редактирования.

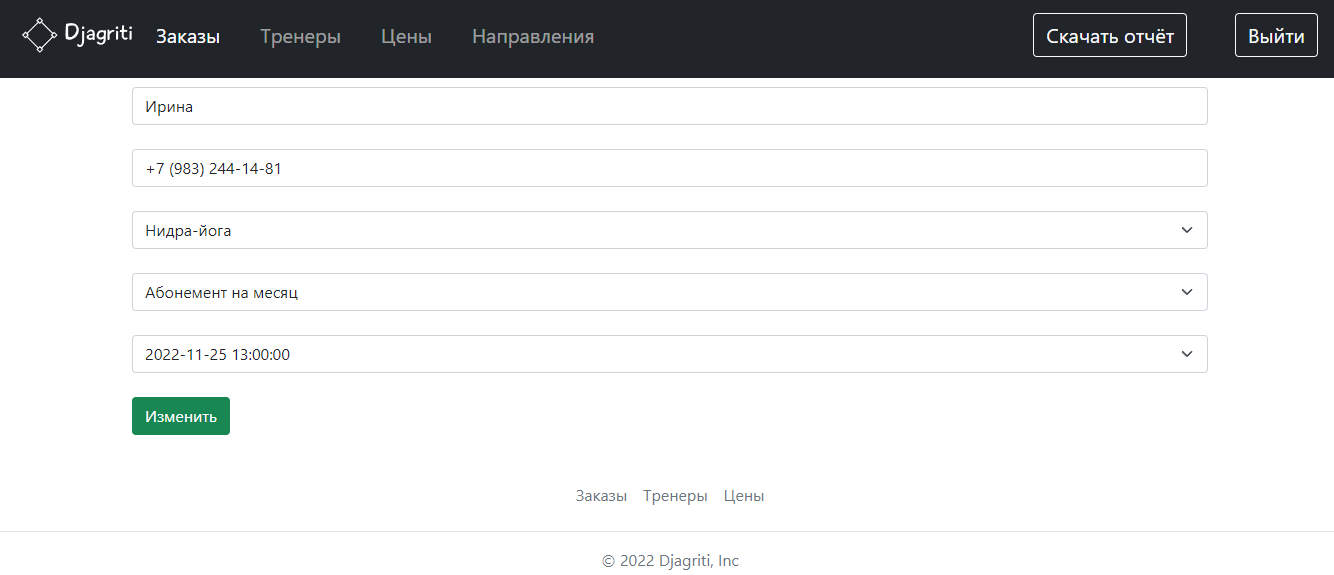


Рисунок 34 – Форма редактирования

При нажатии на кнопку «Скачать отчёт» в верхней панели, начинается загрузка документа Excel, в котором содержится отчёт за текущий месяц работы йога клуба, на котом представлены заявки за текущий месяц, количество заявок за текущий месяц, самое популярное направление, среди тех которые выбирали клиенты, а также популярный абонемент (рисунок 35).

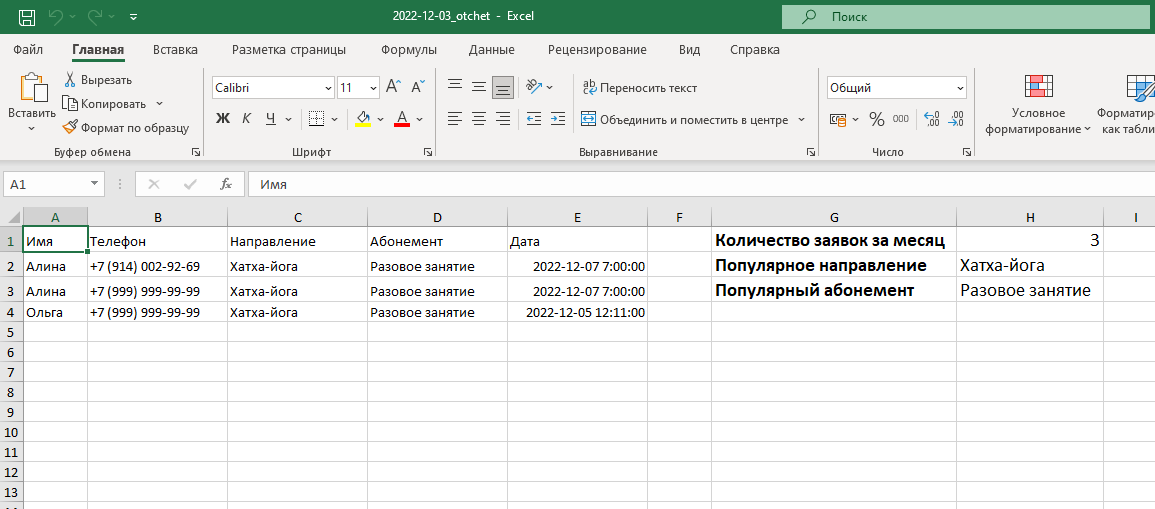


Рисунок 35 – Отчёт

**Заключение**

В ходе выполнения данного курсового проекта была разработана информационная система «Йога клуб», которая полностью удовлетворяет потребности клиента, и исключает дискомфорт и неудобство с посещением йога клуба. А также облегчает работу администраторов, позволяет оперативно узнавать о поступлении новых заявок на сайте, позволяет формировать отчёт работы йога клуба, собирая статистические данные.

Был определен и реализован следующий функционал информационной системы:

− создание заявок;

− просмотр заявок;

– редактирование заявок;

* редактирование и наполнение содержимого сайта;
* составление отчёта;
* авторизация

− отправка уведомлений в Telegram

Был разработан браузерный программный продукт, с интуитивным и современным интерфейсом и широким функционалом.

Основная сложность, которая была встречена во время разработки информационной системы – составление отчёта в Excel документе. Проблема была решена путём установки библиотеки «openpyxl», которая представляет функционал для работы с ячейками в документах Excel.

Все поставленные цели и задачи курсового проекта были успешно выполнены.

**Список используемых источников**

1. Уральский федеральный университет имени первого Президента России Б. Н. Ельцина: Введение в HTML. – URL: https://iit-web-lectures.readthedocs.io/ru/latest/www/html.html (дата обращения: 10.10.2022). – Текст: электронный.
2. Национальная библиотека им. Н. Э. Баумана: JavaScript. – URL: <https://ru.bmstu.wiki/JavaScript> (дата обращения: 27.09.2022). – Текст: электронный.
3. Timeweb Community: PyCharm: IDE для Python. – URL: https://clck.ru/r8pJu (дата обращения: 26.09.2022). – Текст: электронный.
4. METANIT.COM: Введение в SQLite. – URL: https://metanit.com/sql/sqlite/1.1.php (дата обращения: 29.09.2022). – Текст: электронный.
5. Django.fan: Документация по фреймворку и библиотекам Django. – URL: https://django.fun/ru/docs/ (дата обращения: 15.10.2022). – Текст: электронный.
6. Bootstrap: Начало работы. – URL: https://bootstrap-4.ru/docs/5.1/getting-started/introduction (дата обращения: 24.09.2022). – Текст: электронный.
7. Хакслет: почему Django — лучший фреймворк для разработки сайтов. – URL: https://clck.ru/32qfF2 (дата обращения: 15.10.2022). – Текст: электронный.
8. Timeweb Community: Язык программирования Python: применение, особенности и перспективы. – URL: https://timeweb.com/ru/community/articles/chto-takoe-python (дата обращения: 24.09.2022). – Текст: электронный.
9. Уральский федеральный университет имени первого Президента России Б. Н. Ельцина: Каскадные таблицы стилей. – URL: https://clck.ru/32qfJU (дата обращения: 15.10.2022). – Текст: электронный.
10. nationalteam.worldskills.ru: Проектирование диаграммы деятельности UML (Activity Diadram). – URL: https://clck.ru/32g7zj (дата обращения: 15.10.2022). – Текст: электронный.

# Приложение А Техническое задание

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное образовательное учреждение

Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

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

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ИНФОРМАЦИОННАЯ СИСТЕМА

«ЙОГА КЛУБ»

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Д.А. Гусаков) |

Иркутск 2022

**СОДЕРЖАНИЕ**

[1 Общие сведения 43](#_Toc54261854)

[2 Назначение и цели создания системы 43](#_Toc54261855)

[3 Требования к системе в целом 43](#_Toc54261856)

[3.1 Требования к структуре и функционированию системы 43](#_Toc54261857)

[3.2 Требования к надежности 44](#_Toc54261858)

[3.3 Требования к безопасности 44](#_Toc54261859)

[3.4 Требования к эксплуатации, техническому обслуживанию, ремонту и 44](#_Toc54261860)

[хранению компонентов системы 44](#_Toc54261861)

[4 Требования к документированию 44](#_Toc54261862)

[5 Состав и содержание работ по созданию системы 45](#_Toc54261863)

**1 Общие сведения**

Наименование работы: информационная система «Йога клуб».

Исполнитель: студент Иркутского Авиационного Техникума, группы ПКС-19-1, Гусаков Д.А.

Разработка информационной системы проходит в рамках курсового проекта по МДК.03.01 «Технология разработки программного обеспечения», на основании приказа №37-у от 12 сентября 2022 года.

Сроки разработки информационной системы с 12.09.2022 по 06.12.2022 года.

**2 Назначение и цели создания системы**

Целью курсового проекта является создание информационной системы «Йога клуб».

В данной информационной системе реализуется следующий функционал:

− создание заявок;

− просмотр заявок;

– редактирование заявок;

* редактирование и наполнение содержимого сайта;
* составление отчёта;
* авторизация

− отправка уведомлений в Telegram

# 3 Требования к системе в целом

## **3.1 Требования к структуре и функционированию системы**

Функции информационной системы:

1. раздел «Главная»;
   1. создание новой заявки;
   2. отображение информации о клубе
   3. отображение информации о направлениях.
2. раздел «Тренеры»;
   1. отображение информации о тренерах;
   2. регистрация нового тренера;
   3. удаление тренера.
3. раздел «Цены»;
   1. отображение информации о ценах;
   2. редактирование информации о ценах.

**3.2 Требования к надежности**

Для обеспечения надежности необходимо проверять корректность получаемых данных и реализовать валидность полей. Входные данные поступают в виде значений c клавиатуры. Эти значения отображаются в отдельных полях таблицы.

**3.3 Требования к безопасности**

Для обеспечения безопасности в информационной системе, необходимо реализовать разграничение прав доступа.

**3.4 Требования к эксплуатации, техническому обслуживанию, ремонту и**

**хранению компонентов системы**

Минимальные системные требования для рабочей станции:

1. Процессор: Pentium 4 с SSE2;
2. Оперативная память: 512 Мб;
3. Жёсткий диск: 350мб;
4. Операционная система: Windows, Linux, MacOS.

Требования к серверу:

1. Процессор: 8 ядер, тактовая частота 2.90 ГГц и выше
2. Платформа: 32-х или 64-х разрядная
3. Оперативная память: 10 ГБ и выше
4. Жесткий диск: Размер определяется объемом базы данных

**4 Требования к документированию**

Основным документом, регламентирующими использование информационной системы является руководство пользователя.

Основным документом, регламентирующими разработку информационной системы является техническое задание.

**5 Состав и содержание работ по созданию системы**

В таблице 1 представлены плановые сроки начала и окончания работы по созданию информационной системы

Таблица 1 – Плановые сроки по созданию информационной системы

|  |  |  |
| --- | --- | --- |
| **Наименование этапов курсового проекта** | **Срок** | **Объем** |
| Предпроектное исследование предметной области (выбор темы, постановка цели, задач, описание области применения, исследование предметной области) | 21.09.2022 | 10% |
| Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования) | 26.09.2022 | 20% |
| Проектирование программного обеспечения.  (разработка структурной и функциональной схемы ПО, проектирование базы данных (инфологическое, ER-модель, физическая модель) | 25.10.2022 | 50% |
| Разработка (программирование) и отладка программного продукта | 28.11.2022 | 90% |
| Составление программной документации (оформление ПЗ, руководство пользователя и презентации) | 06.12.2022 | 100% |
| Защита курсового проекта | 06.12 – 16.12.2022 | |

# Приложение Б Листинг views.py

def index(request):

slider\_list = CmsSlider.objects.all()

napr = NaprCrm.objects.all()

if request.method == 'POST':

form = OrderForm(request.POST)

name = request.POST['order\_name']

phone = request.POST['order\_phone']

if form.is\_valid():

send\_message(tg\_name=name, tg\_phone=phone)

form.save()

return render(request, 'crm/thanks\_page.html', {'slider\_list': slider\_list,'napr': napr, 'form': form, 'name': name})

else:

return render(request, 'crm/phone\_error.html', {'slider\_list': slider\_list,'napr': napr, 'form': form})

else:

form = OrderForm()

return render(request, 'crm/index.html',

{'slider\_list': slider\_list, 'form': form, 'napr': napr, 'title': 'Главная'})

def teachers(requst):

form = OrderForm()

posts = CmsTeachers.objects.all()

return render(requst, 'crm/teachers.html', {'form': form, 'posts': posts, 'title': 'Тренеры'})

def price(request):

form = OrderForm()

price\_site = PriceView.objects.all()

return render(request, 'crm/price.html', {'price\_site': price\_site, 'form': form, 'title': 'Цены'})

@login\_required

def dashboard(request):

order = Order.objects.all()

return render(request, 'crm/dashboard.html', {'order': order, 'title': "Панель управления"})

def show\_post(request, post\_slug):

form = OrderForm()

post = get\_object\_or\_404(CmsTeachers, slug=post\_slug)

return render(request, 'crm/teacher\_about.html', {'post': post, 'form': form, 'title': 'О тренерах'})

@login\_required

def delete\_order(request, id\_order):

order = Order.objects.get(pk=id\_order)

order.delete()

return redirect('dashboard')

@login\_required

def update\_order(request, id\_order):

order = Order.objects.get(pk=id\_order)

form = OrderForm(request.POST or None, instance=order)

if form.is\_valid():

form.save()

return redirect('dashboard')

return render(request, 'crm/update\_order.html', {'order': order, 'form': form})

@login\_required

def delete\_teacher(request, id\_teacher):

teacher = CmsTeachers.objects.get(pk=id\_teacher)

teacher.delete()

return redirect('admin\_teachers')

@login\_required

def update\_teacher(request, id\_teacher):

teacher = CmsTeachers.objects.get(pk=id\_teacher)

form = TeacherForm(request.POST or None, request.FILES or None, instance=teacher)

if form.is\_valid():

form.save()

return redirect('admin\_teachers')

return render(request, 'crm/update\_teacher.html', {'teacher': teacher, 'form': form})

@login\_required

def add\_teacher(request):

form = TeacherForm(request.POST or None, request.FILES or None)

if form.is\_valid():

form.save()

return redirect('admin\_teachers')

return render(request, 'crm/add\_teacher.html', {'form': form})

@login\_required

def delete\_price(request, id\_price):

price = Price.objects.get(pk=id\_price)

price.delete()

return redirect('admin\_price')

@login\_required

def update\_price(request, id\_price):

price = Price.objects.get(pk=id\_price)

form = PriceForm(request.POST or None, instance=price)

if form.is\_valid():

form.save()

return redirect('admin\_price')

return render(request, 'crm/update\_price.html', {'price': price, 'form': form})

@login\_required

def add\_price(request):

form = PriceForm(request.POST or None)

if form.is\_valid():

form.save()

return redirect('admin\_price')

return render(request, 'crm/add\_price.html', {'form': form})

@login\_required

def delete\_user(request, user\_name):

users = User.objects.get(username=user\_name)

users.delete()

return redirect('show\_users')

@login\_required

def add\_user(request):

form = UserForm(request.POST or None)

if form.is\_valid():

User.objects.create\_user(\*\*form.cleaned\_data)

return redirect('show\_users')

return render(request, 'crm/add\_user.html', {'form': form})

@login\_required

def admin\_teachers(request):

teacher = CmsTeachers.objects.all()

return render(request, 'crm/admin\_teachers.html', {'order': teacher, 'title': "Панель управления"})

@login\_required

def admin\_price(request):

price = Price.objects.all()

return render(request, 'crm/admin\_price.html', {'order': price, 'title': "Панель управления"})

@login\_required

def show\_users(request):

users = User.objects.all()

return render(request, 'crm/admin\_users.html', {'users': users, 'title': "Панель управления"})

@login\_required

def admin\_napr(request):

napr = NaprCrm.objects.all()

return render(request, 'crm/admin\_napr.html', {'napr': napr, 'title': "Панель управления"})

@login\_required

def delete\_napr(request, id\_napr):

napr = NaprCrm.objects.get(pk=id\_napr)

napr.delete()

return redirect('admin\_napr')

@login\_required

def update\_napr(request, id\_napr):

napr = NaprCrm.objects.get(pk=id\_napr)

form = NaprForm(request.POST or None, request.FILES or None, instance=napr)

if form.is\_valid():

form.save()

return redirect('admin\_napr')

return render(request, 'crm/update\_napr.html', {'napr': napr, 'form': form})

@login\_required

def add\_napr(request):

form = NaprForm(request.POST or None, request.FILES or None)

if form.is\_valid():

form.save()

return redirect('admin\_napr')

return render(request, 'crm/add\_napr.html', {'form': form})

def export\_to\_xlsx(request):

order = Order.objects.filter(date\_\_month=datetime.now().month)

count = order.count()

popular\_napr = Order.objects.filter(date\_\_month=datetime.now().month).values('order\_napr\_\_napr\_name').annotate(

total=Count("order\_napr")).order\_by("-total")[:1]

popular\_napr\_value = popular\_napr[0]

p = popular\_napr\_value['order\_napr\_\_napr\_name']

popular\_price = Order.objects.filter(date\_\_month=datetime.now().month).values('order\_price\_\_price\_name').annotate(

total=Count("order\_price")).order\_by("-total")[:1]

popular\_price\_value = popular\_price[0]

p\_price = popular\_price\_value['order\_price\_\_price\_name']

response = HttpResponse(

content\_type='application/vnd.openxmlformats-officedocument.spreadsheetml.sheet',)

response['Content-Disposition'] = 'attachment; filename={date}\_otchet.xlsx'.format(date=datetime.now().strftime('%Y-%m-%d'))

workbook = Workbook()

worksheet = workbook.active

worksheet.title = 'Отчёт'

columns = [

'Имя',

'Телефон',

'Направление',

'Абонемент',

'Дата',]

row\_num = 1

for col\_num, column\_title in enumerate(columns, 1):

cell = worksheet.cell(row=row\_num, column=col\_num)

cell.value = column\_title

for ord in order:

row\_num += 1

row = [ord.order\_name,ord.order\_phone,ord.order\_napr.napr\_name,

ord.order\_price.price\_name, ord.date,]

for col\_num, cell\_value in enumerate(row, 1):

cell = worksheet.cell(row=row\_num, column=col\_num)

cell.value = cell\_value

cell\_count\_name = worksheet.cell(row=1, column=7)

cell\_count\_name.value = "Количество заявок за месяц"

cell\_count\_name.font = Font(name='Calibri', bold=True, size=14)

column\_letter = get\_column\_letter(7)

column\_dimensions = worksheet.column\_dimensions[column\_letter]

column\_dimensions.width = 35

column\_letter = get\_column\_letter(8)

column\_dimensions = worksheet.column\_dimensions[column\_letter]

column\_dimensions.width = 21

cell\_count\_value = worksheet.cell(row=1, column=8)

cell\_count\_value.value = count

cell\_count\_value.font = Font(size=14)

cell\_count\_name = worksheet.cell(row=2, column=7)

cell\_count\_name.value = "Популярное направление"

cell\_count\_name.font = Font(name='Calibri', bold=True, size=14)

cell\_count\_value = worksheet.cell(row=2, column=8)

cell\_count\_value.value = p

cell\_count\_value.font = Font(size=14)

cell\_count\_name = worksheet.cell(row=3, column=7)

cell\_count\_name.value = "Популярный абонемент"

cell\_count\_name.font = Font(name='Calibri', bold=True, size=14)

cell\_count\_value = worksheet.cell(row=3, column=8)

cell\_count\_value.value = p\_price

cell\_count\_value.font = Font(size=14)

for i in range(2, len(columns) + 1):

column\_letter = get\_column\_letter(i)

column\_dimensions = worksheet.column\_dimensions[column\_letter]

column\_dimensions.width = 20

workbook.save(response)

return response