**GeekBrains**

**Специальность – программист Python, цифровые профессии**

**Тема дипломного проекта: Разработка информационного**

**контроля сотрудников на предприятии**

**Студент: Некрасов Виктор Сергеевич**

Новосибирск - 2023 г.

СОДЕРЖАНИЕ

[Введение 3](#_Toc150651884)

[1. Создание базового проекта 4](#_Toc150651885)

[2. Работа с базой данных 11](#_Toc150651886)

[3. Оформление страниц сайта 17](#_Toc150651887)

[3.1. Главная страница 21](#_Toc150651888)

[3.2. Страница «О нас» 23](#_Toc150651889)

[3.3. Страница «Услуги» 24](#_Toc150651890)

[3.4. Страница «Тренеры» 25](#_Toc150651891)

[3.5. Страница «Политика конфиденциальности» 25](#_Toc150651892)

[4. Результат 27](#_Toc150651893)

[5. Заключение 31](#_Toc150651894)

# **Введение**

Современный мир требует от фитнес-залов активной онлайн-присутствия, чтобы привлечь больше клиентов и улучшить качество обслуживания. Создание сайта на Django является эффективным решением для решения этой проблемы, поскольку Django предоставляет множество инструментов для создания высококачественных и удобных сайтов.

Цель работы: создание сайта на Django для фитнес-зала.

В рамках работы будут рассмотрены следующие задачи: проектирование базы данных, разработка пользовательского интерфейса, другие аспекты, необходимые для создания полноценного сайта.

Стек технологий, который используется в работе:

* **Фреймворк Django** для разработки веб-приложений на языке Python.
* **HTML, CSS** для создания пользовательского интерфейса и взаимодействия с пользователем.
* **Bootstrap** - CSS-фреймворк, который позволяет быстро создавать адаптивные и стильные веб-страницы.
* **SQLite** - система управления базами данных (СУБД) для хранения информации.

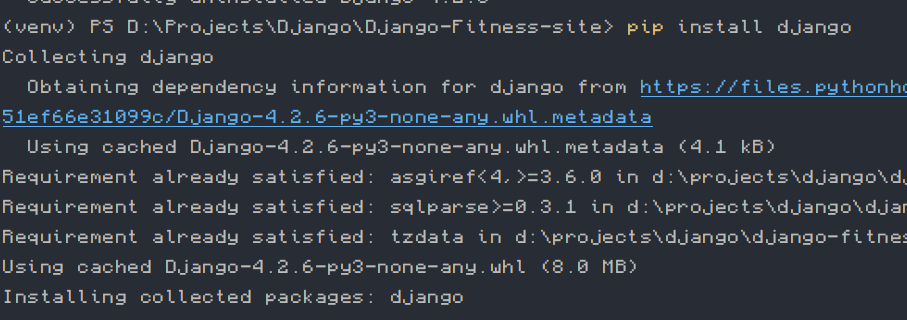
Работа состоит из следующих разделов: введение, содержание, процесс разработки сайта, результат и заключение.

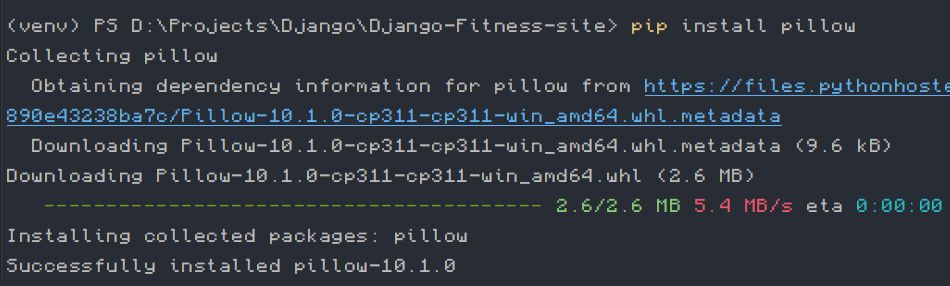
# **Создание базового проекта**

Установка необходимых библиотек (Django, tzdata, sqlparse, asgiref, pillow):

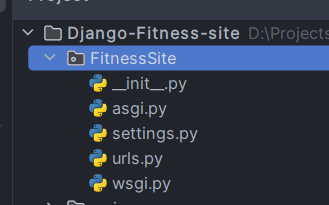
* Django - это высокоуровневый веб-фреймворк для разработки веб-приложений на языке Python. Он предоставляет инструменты и структуру для создания мощных и масштабируемых веб-проектов.
* tzdata - это база данных временных зон. Она содержит информацию о смещениях времени, правилах перехода на летнее/зимнее время и других параметрах для различных регионов мира.
* sqlparse - это библиотека для форматирования и анализа SQL-запросов в языке Python.
* asgiref - это библиотека, предоставляющая общий интерфейс для работы с асинхронными веб-серверами в Python.
* Pillow - это форк библиотеки Python Imaging Library (PIL), предоставляющей инструменты для работы с изображениями в Python.

Установка библиотек осуществляется вводом команды **`pip install название`** в командной строке.

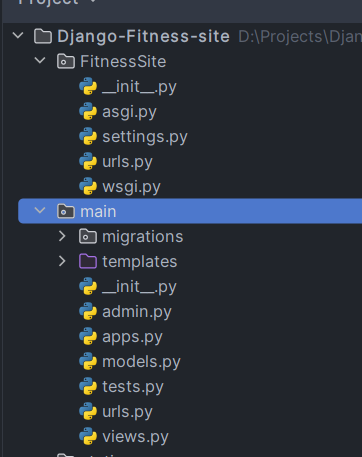




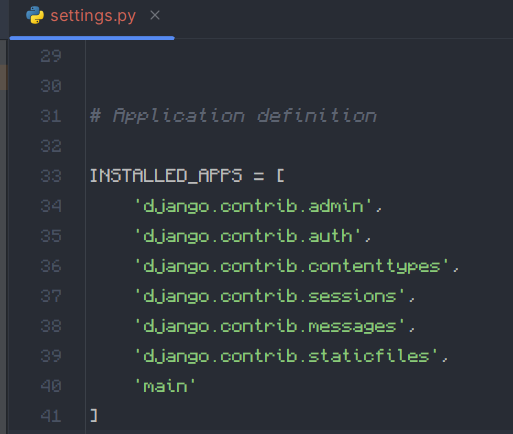
После установки всех необходимых библиотек, нужно ввести команду django-admin startproject FitnessSite, после чего создастся проект на Django с необходимыми папками и файлами.



Командой python manage.py startapp main создается новое приложение **“main”**

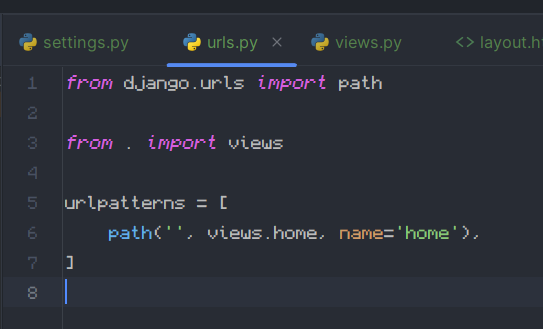


В файле **settings.py** нужно добавить в список **INSTALLED\_APPS** строку **‘main’** для подключения приложения к проекту

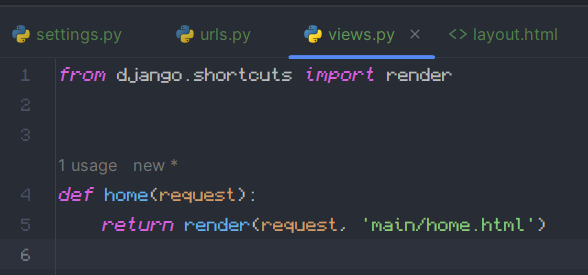


В приложении main создается файл **urls.py**, в котором будут храниться url адреса сайта.

Далее создается адрес на главную страницу home путем добавления ее в список urlpatterns.

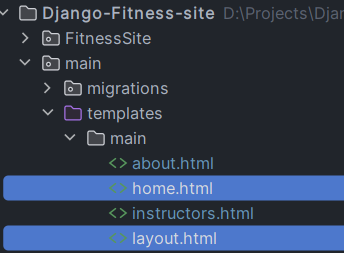


В файле views.py создается функция home(request), которая будет обрабатывать запросы при переходе на адрес. Возвращает функция html шаблон.

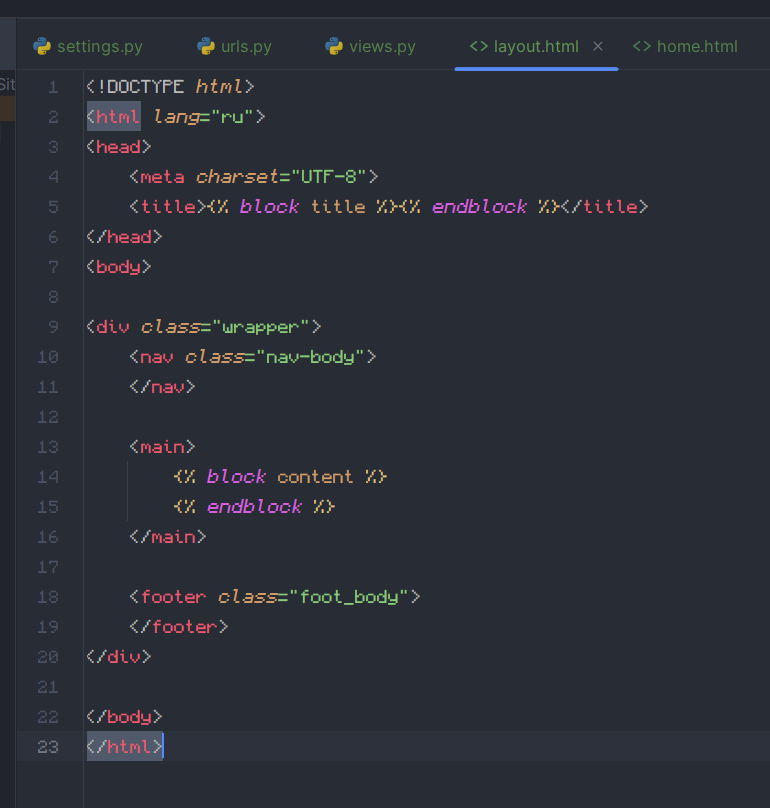


Необходимо создать в папке приложения папку templates, а в ней папку main, в которой будут находиться все html шаблоны.

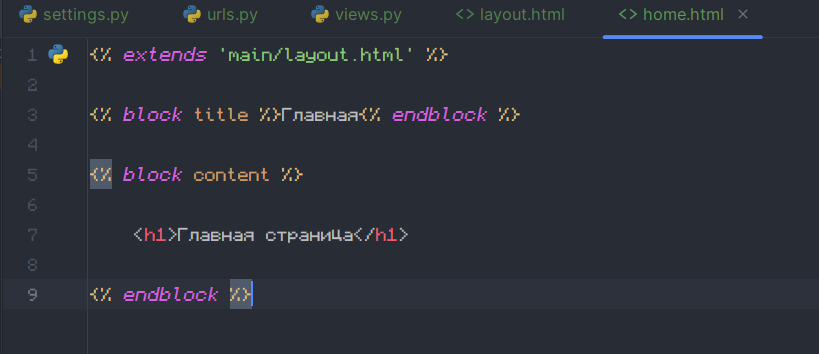
В созданную папку шаблонов добавляется файл layout.html, который будет базовым шаблоном, от которого все последующие шаблоны будут наследоваться, и home.html, который является шаблоном главной страницы



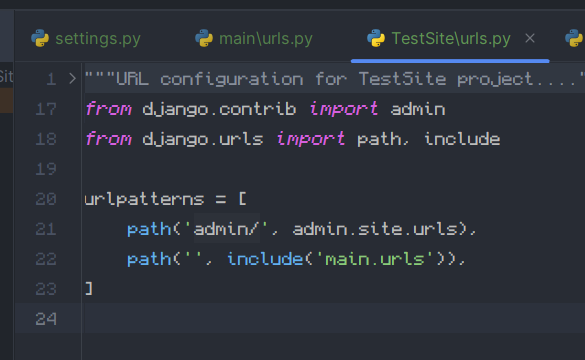
В файле layout.html нужно создать базовый шаблон для всех страниц с использованием шаблонизатора Jinja2, где **{% block title %}** и **{% block content %}** определяют блоки, содержимое в которых можно изменить в шаблонах, унаследованных от данного шаблона.



В файле home.html наследуется базовый шаблон layout.html строкой **{% extends 'main/layout.html' %}**, а затем в блоках вписывается нужная информация

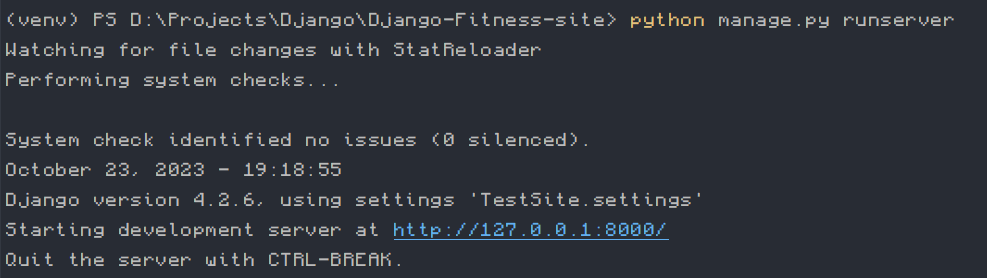


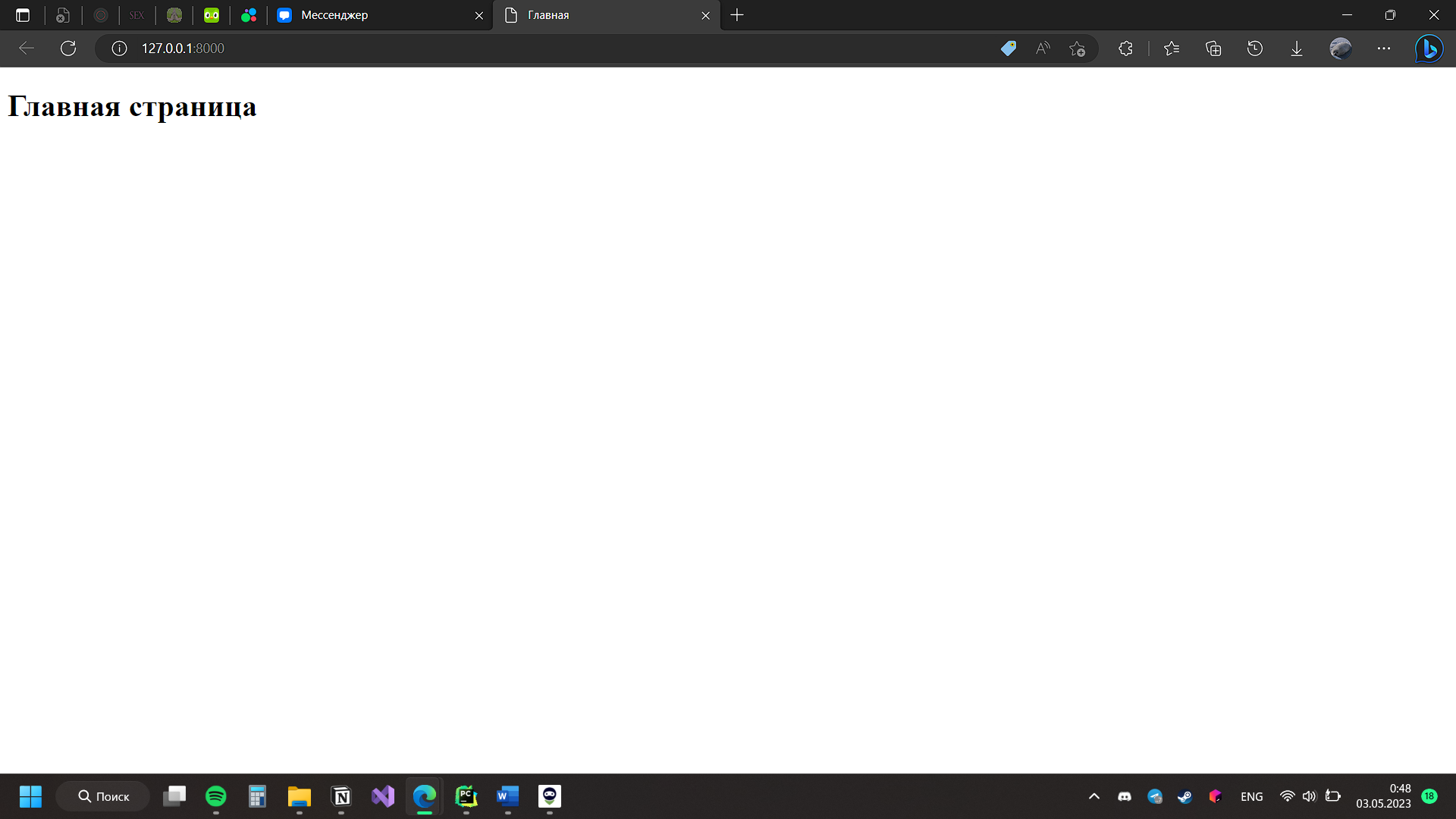
Также, в файле **urls.py** корневой папки добавляются все ссылки из файла **urls.py** приложения main.



Для первого запуска проекта требуется сделать базовые миграции перед первым запуском сайта командой **python manage.py migrate**:

Для запуска проекта, необходимо ввести команду python manage.py runserver и по ссылке <http://127.0.0.1:8000/> в браузере можно увидеть созданный проект, страница загружаются правильно





# **Работа с базой данных**

В файле **models.py** создается модель **Offers**, которая будет хранить в себе информацию о предложениях:

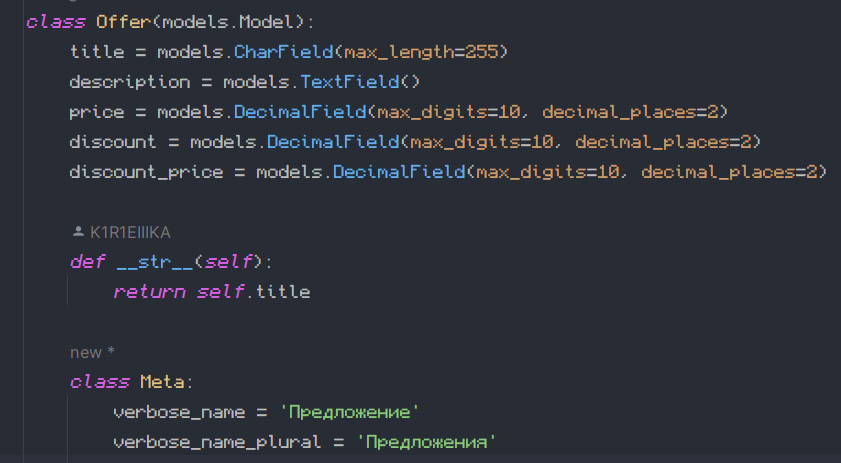
* title**:** Хранит заголовок предложения (CharField, максимум 255 символов).
* **description**: Содержит более длинное описание предложения (TextField).
* **price**: Сохраняет общую цену предложения (DecimalField, максимум 10 цифр, 2 знака после запятой).
* **discount**: Содержит размер скидки в процентах (DecimalField, максимум 10 цифр, 2 знака после запятой).
* **discount\_price**: Хранит цену предложения после применения скидки (DecimalField, максимум 10 цифр, 2 знака после запятой).
* Метод **\_\_str\_\_(self)**:

Возвращает строковое представление объекта. В данном случае, возвращает заголовок предложения, что полезно при отображении объектов в административном интерфейсе Django.

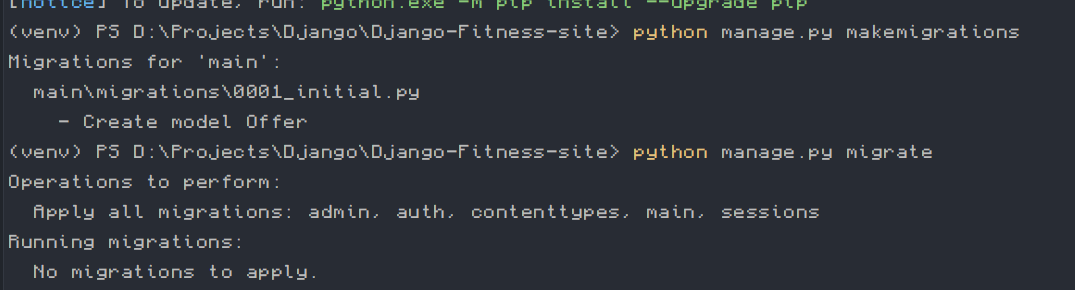
* Мета-класс **class Meta**:

Предоставляет метаданные модели.

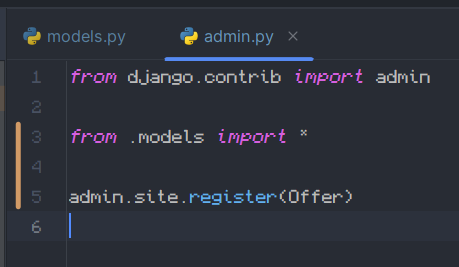
* + **verbose\_name**: Устанавливает читаемое имя модели в единственном числе.
  + **verbose\_name\_plural**: Устанавливает читаемое имя модели во множественном числе.



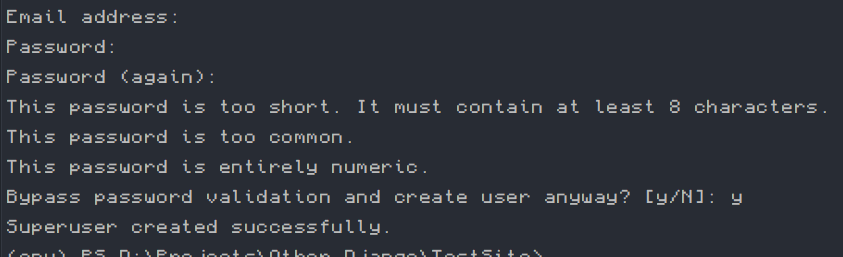
Также, необходимо ввести команды **python manage.py makemigrations** и **python manage.py migrate** для обновления базы данных и проведения миграций.



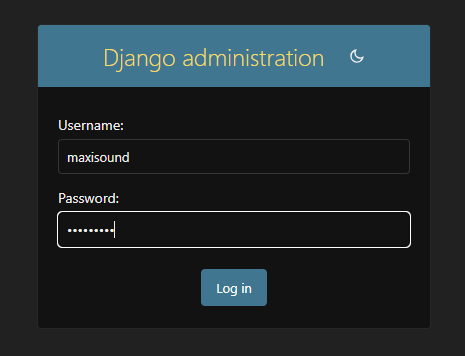
Для отображения модели в панели администратора, необходимо добавить ее в панель администратора через файл **admin.py**

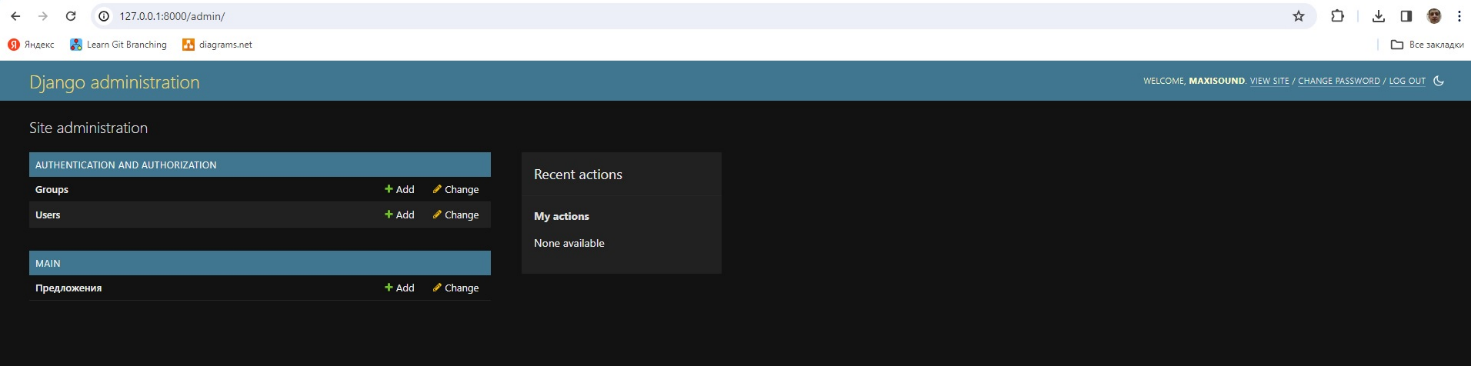


Для работы с панелей администратора необходимо создать суперпользователя командой **python manage.py createsuperuser**. Для создания достаточно указать только никнейм пользователя и пароль.



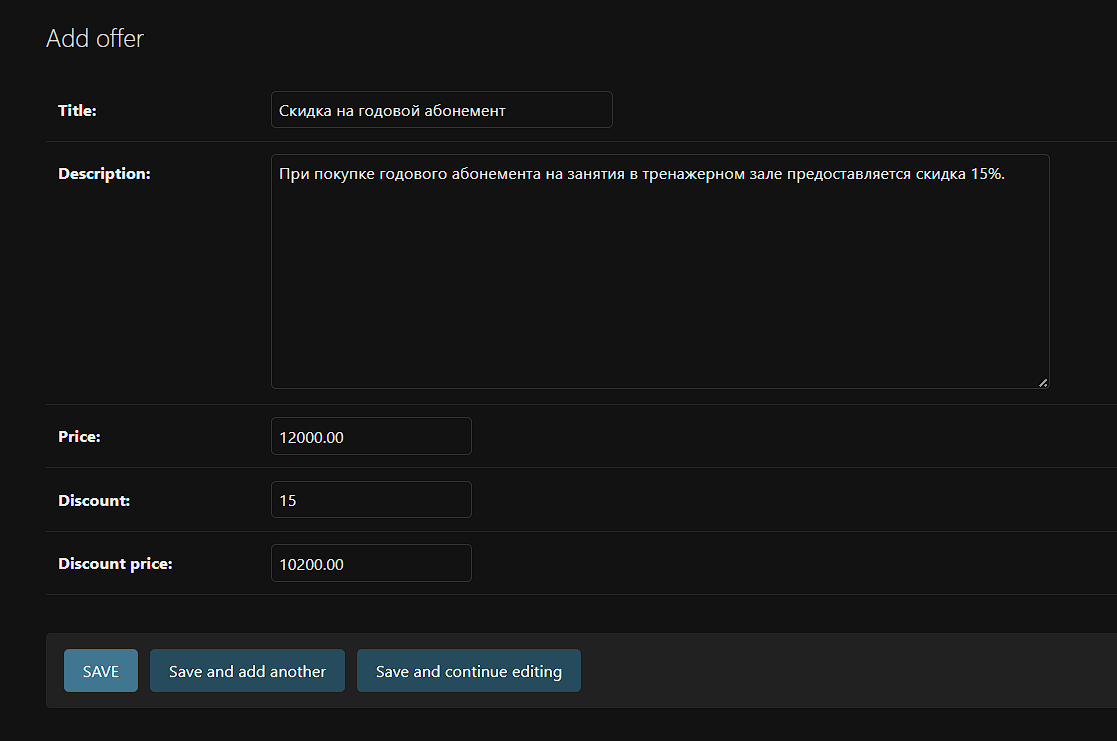
При переходе на страницу <http://127.0.0.1:8000/admin/>, нужно ввести данные от созданного суперпользователя, чтобы попасть в админ панель.

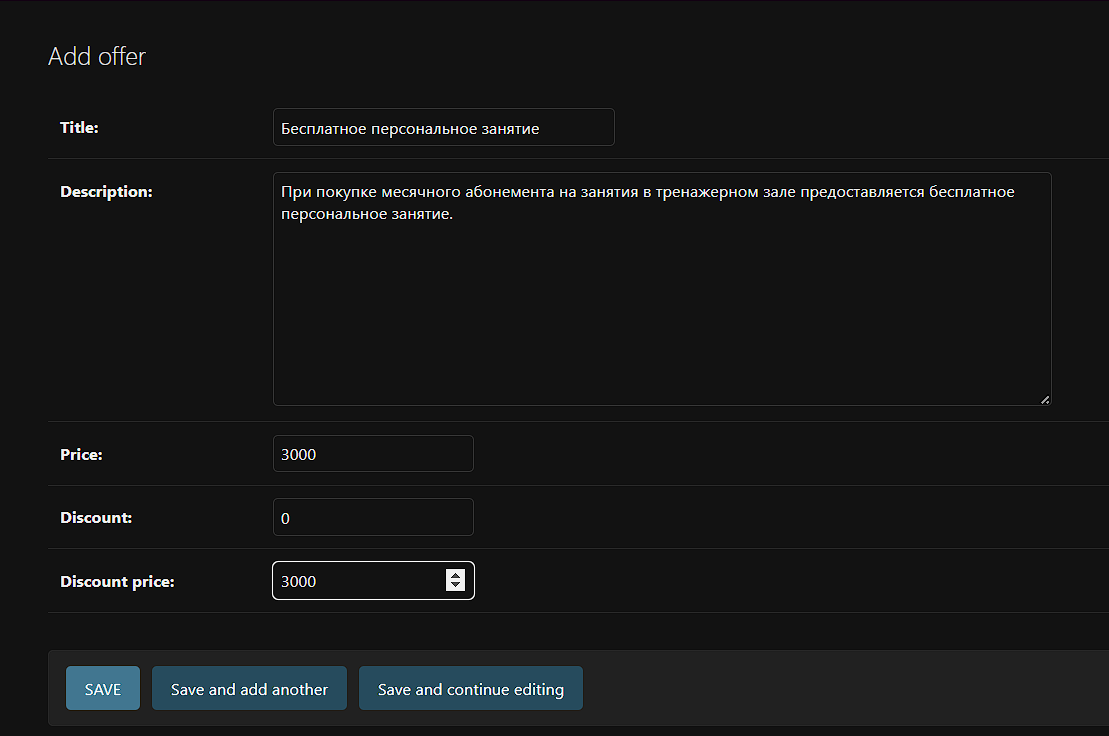


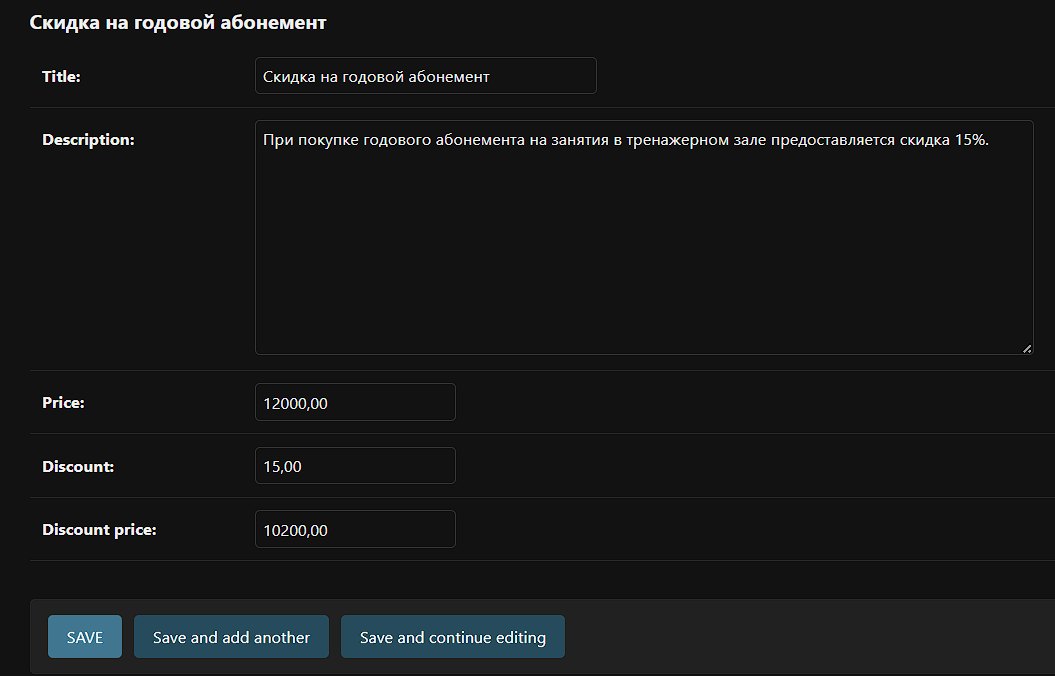


Чтобы добавить новые предложения (объекта класса **Offer**), нужно нажать на кнопку **Add offer** и заполнить форму.

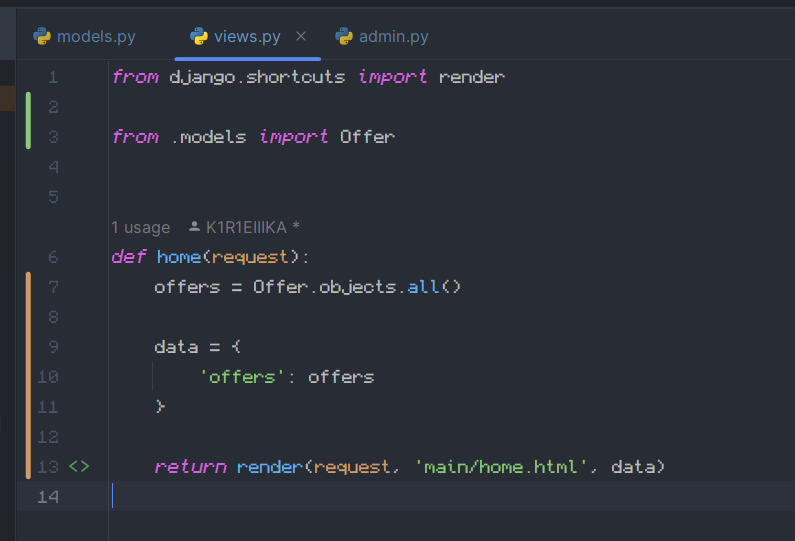
Добавление 3 предложений:



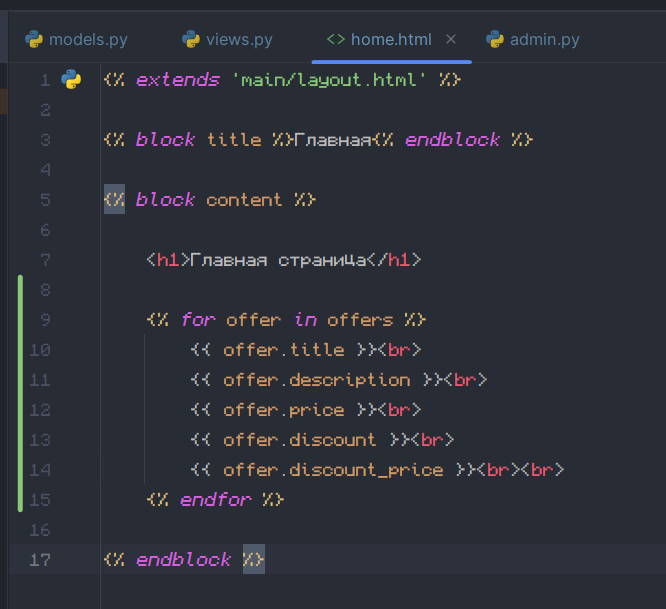




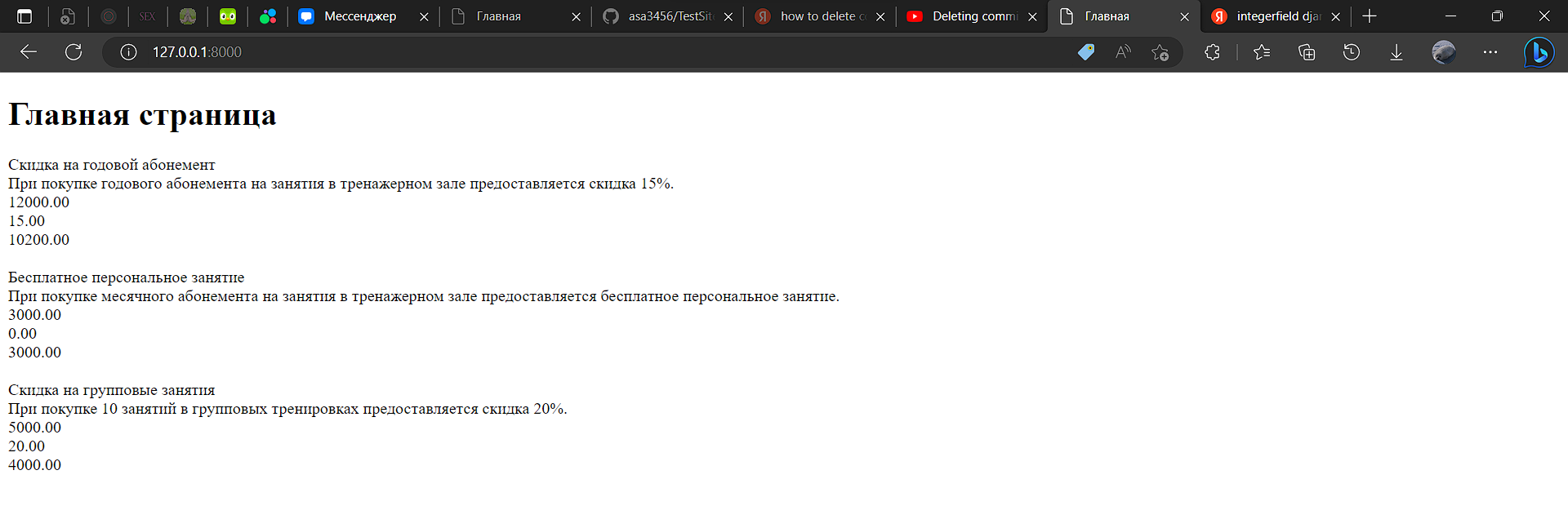
В файле **views.py** нужно импортировать таблицу Offers, чтобы на главной странице вывести все значения таблицы. Для получения всех предложений из таблицы, используется функция Offer.objects.all(). Эти предложения передаются в словарь, который отправляется на саму страницу.



В файле **home.html** можно вывести все значения таблицы при помощи перебора тегом **{% for offer in offers %}**



Результат



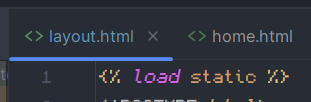
# **Оформление страниц сайта**

Для работы со статическими файлами необходимо создать в корневой папке проекта папку **static**, а в ней папки **css** и **images**. В них можно добавлять файлы со стилями и изображения.



В файл **layout.html**, который является основой будущих страниц, нужно добавить область навигации и нижнюю область с информацией (footer).

В 1 строке нужно импортировать папку со статическими объектами, чтобы можно было удобно загружать стили и изображения.



В блоке title необходимо создать блок **{% block title %}** для изменения заголовка, содержимое которого можно изменить в шаблонах, унаследованных от данного шаблона.

Также, нужно подключить стили из Bootstrap и созданный файл стилей при помощи ссылки шаблонного тега на статические файлы bootstrap.min.css и style.css: **{% static 'main/css/bootstrap.min.css' %}** и **{% static 'main/css/style.css' %}**

Далее, в теге body создается панель навигации в теге nav, которая состоит из ссылки на главную страницу сайта и списка ссылок на другие страницы.

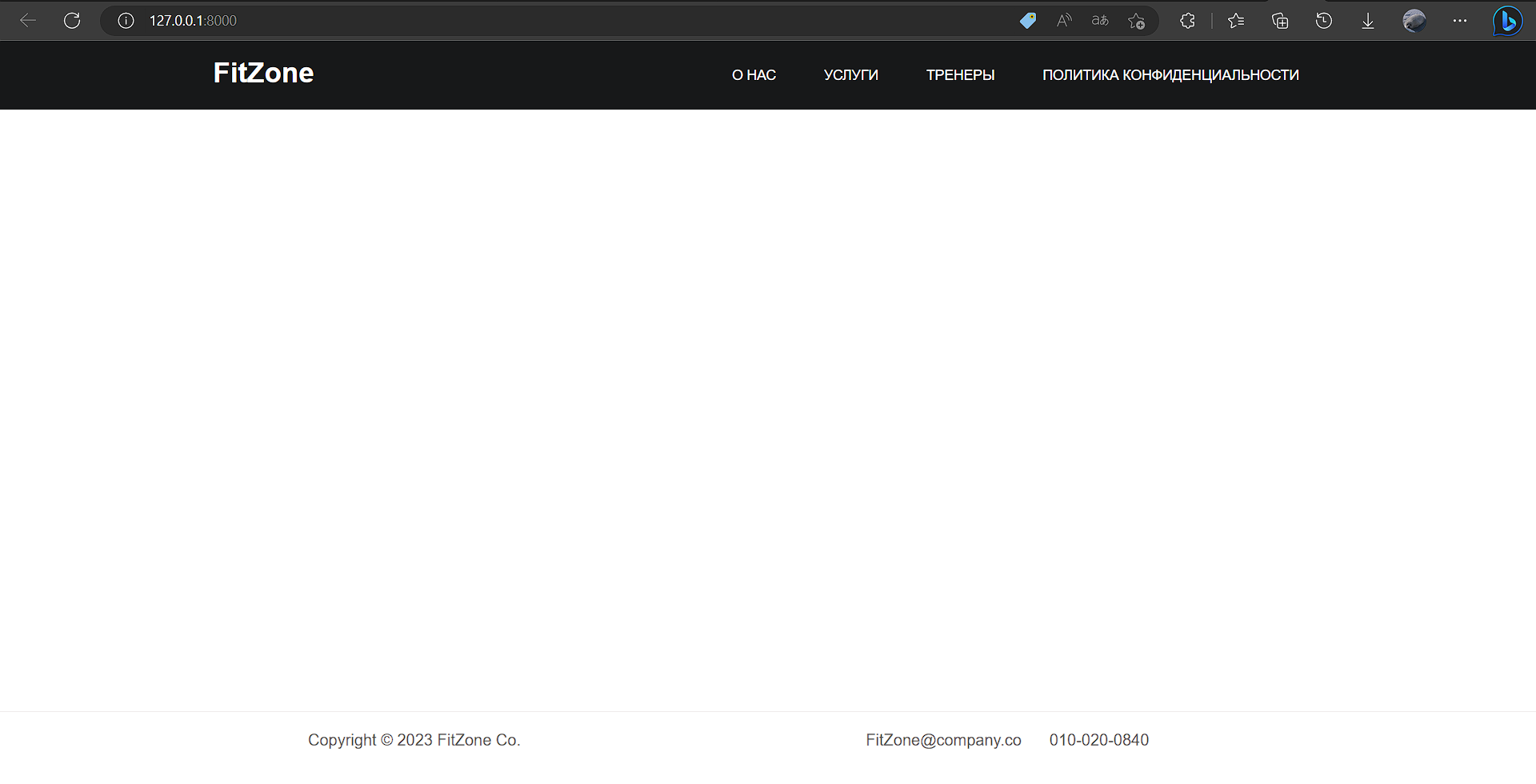
Все ссылки ведут на страницы, созданные в файле urls.py и указываются шаблонным тегом **{% url %}**.

Создается тег main, в который помещается блок с основным контентом тегом **{% block content %}**, который можно изменять в унаследованных шаблонах.

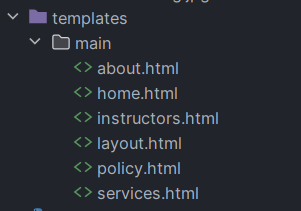
Наконец, необходимо создать тег footer c базовой информацией о сайте, которая всегда расположена снизу.

{% *load* static %}  
<!DOCTYPE *html*>  
<html *lang*="en">  
<head>  
  
 <title>{% *block* title %}{% *endblock* %}</title>  
  
 <meta *charset*="UTF-8">  
 <meta *http-equiv*="X-UA-Compatible" *content*="IE=Edge">  
 <meta *name*="viewport" *content*="width=device-width, initial-scale=1, maximum-scale=1">  
  
 <link *rel*="stylesheet" *href*="{% *static* 'css/bootstrap.min.css' %}">  
 <link *rel*="stylesheet" *href*="{% *static* 'css/style.css' %}">  
  
</head>  
<body>  
  
*<!-- MENU BAR -->*<div *class*="wrapper">  
  
 <nav *class*="navbar navbar-expand-lg fixed-top">  
 <div *class*="container">  
  
 <a *class*="navbar-brand" *href*="{% *url* 'home' %}">FitZone</a>  
  
 <div *class*="collapse navbar-collapse" *id*="navbarNav">  
 <ul *class*="navbar-nav ml-lg-auto">  
 <li>  
 <a *href*="{% *url* 'about' %}" *class*="nav-link">О нас</a>  
 </li>  
  
 <li>  
 <a *href*="{% *url* 'services' %}" *class*="nav-link">Услуги</a>  
 </li>  
  
 <li>  
 <a *href*="{% *url* 'instructors' %}" *class*="nav-link">Тренеры</a>  
 </li>  
  
 <li>  
 <a *href*="{% *url* 'policy' %}" *class*="nav-link">Политика конфиденциальности</a>  
 </li>  
 </ul>  
 </div>  
 </div>  
 </nav>  
  
 <main>  
 {% *block* content %}  
 {% *endblock* %}  
 </main>  
  
 *<!-- FOOTER -->* <footer *class*="site-footer">  
 <div *class*="container">  
 <div *class*="row">  
  
 <div *class*="ml-auto col-lg-4 col-md-5">  
 <p>Copyright &copy; 2023 FitZone Co.</p>  
 </div>  
  
 <div *class*="d-flex justify-content-center mx-auto col-lg-5 col-md-7 col-12">  
 <p *class*="mr-4">  
 <a *href*="#">FitZone@company.co</a>  
 </p>  
  
 <p>010-020-0840</p>  
 </div>  
  
 </div>  
 </div>  
 </footer>  
</div>  
  
</body>  
</html>

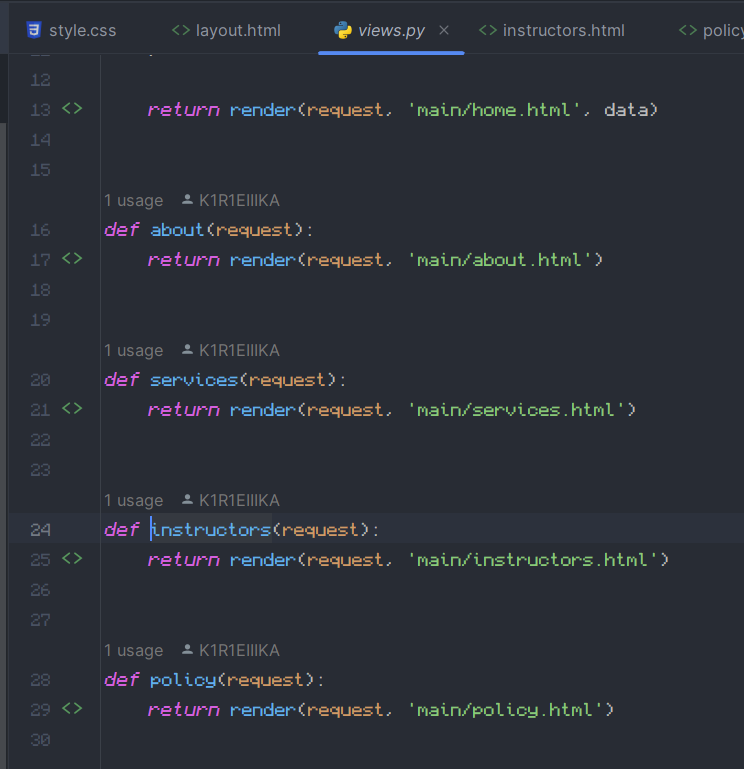
Вид шаблона layout.html без основной информации.



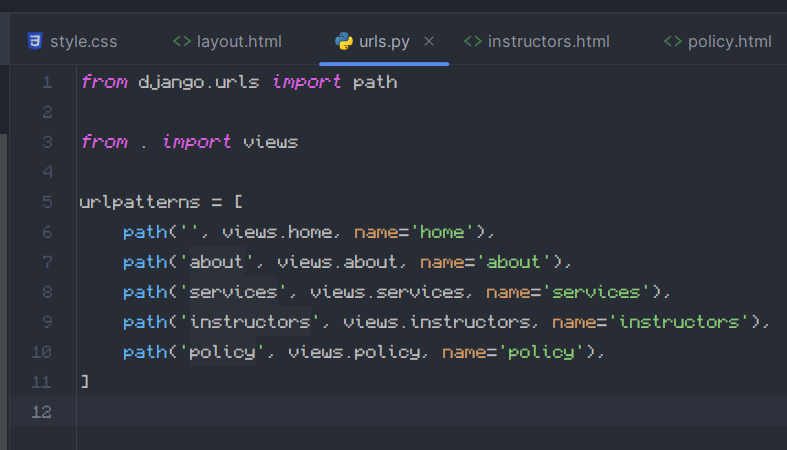
Необходимо создать пустые html страницы остальных страниц – **about**, **instructors**, **policy** и **services**.



В файле views.py обрабатываются запросы на страницы



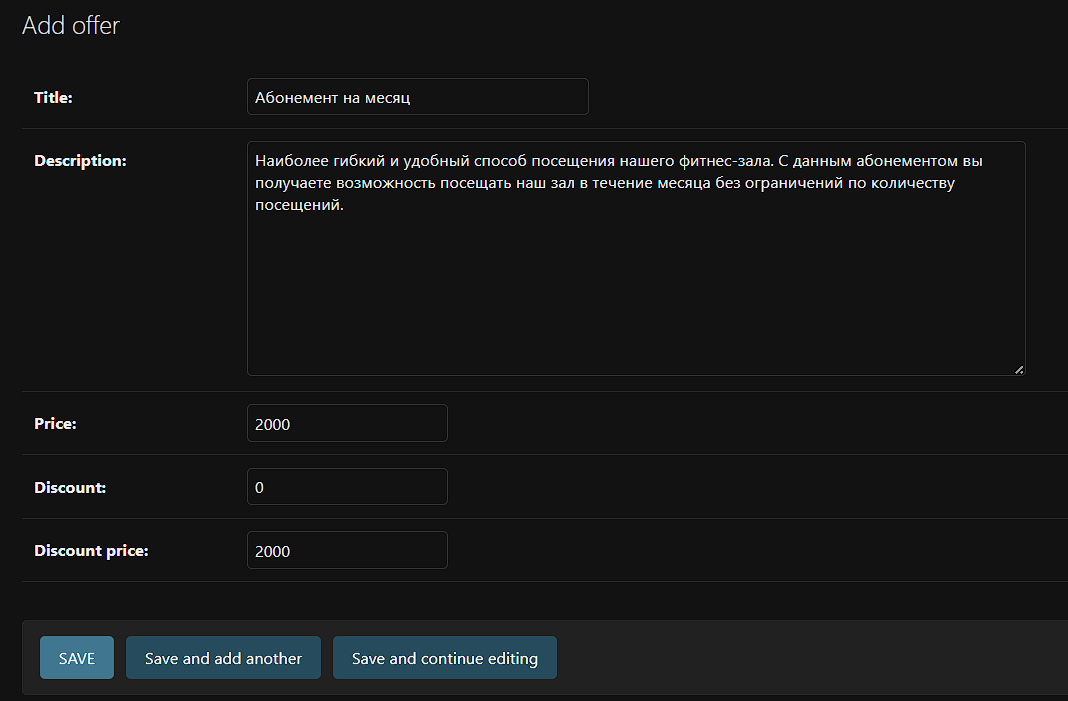
В файле urls.py нужно добавить адреса на страницы, а также назвать их, чтобы в html шаблонах можно было к ним обратиться через шаблонный тег.

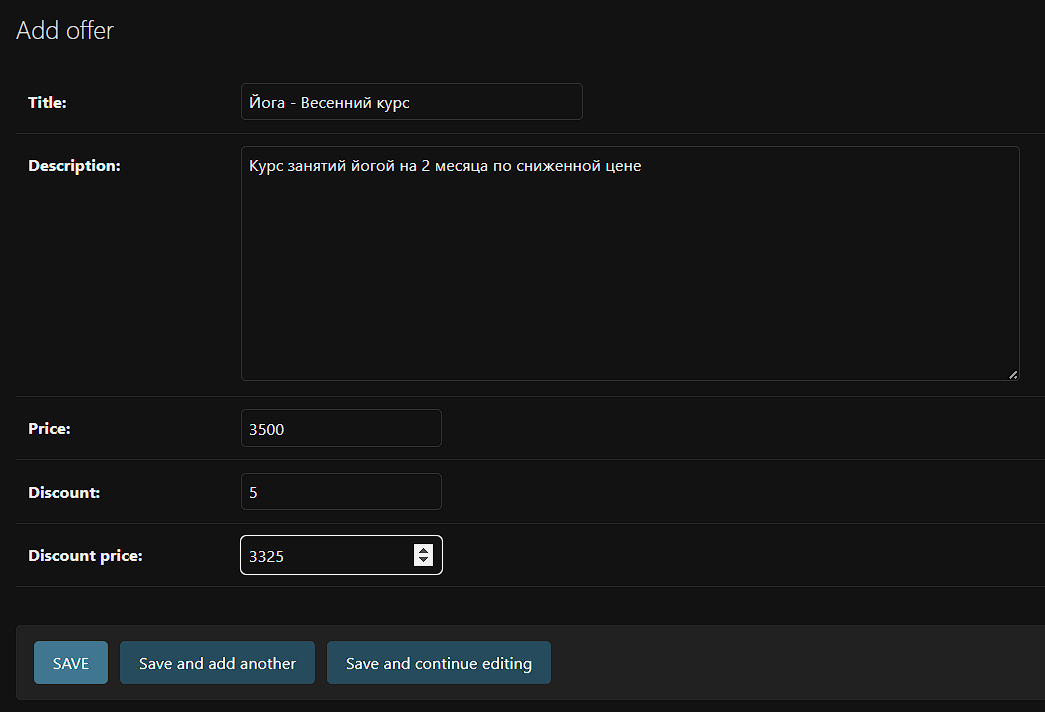


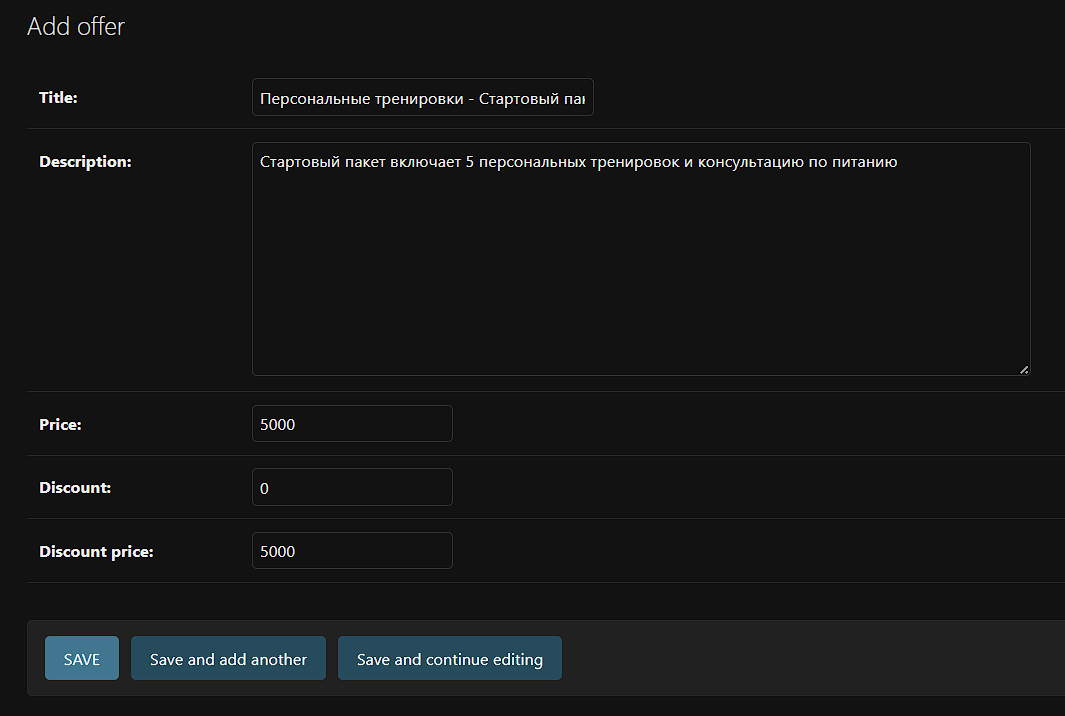
## **Главная страница**

На главной странице предусмотрено отображение трех наиболее выгодных предложений в порядке убывания, начиная с самого выгодного и заканчивая менее выгодными.

Чтобы расширить список услуг, можно добавить еще 3 предложения в базу данных:





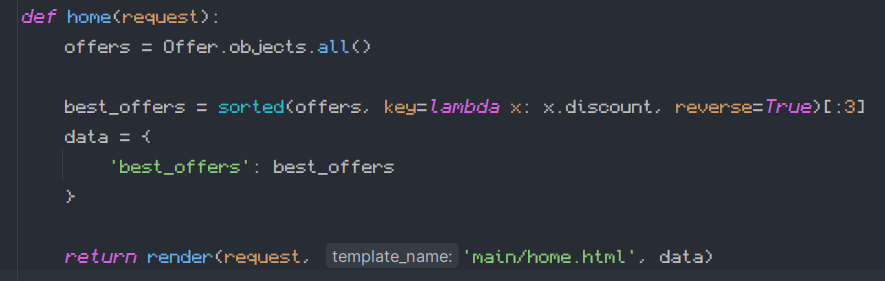


Для реализации вывода 3 наиболее выгодных предложений, в файле views.py можно воспользоваться методом sorted(). Этот метод позволяет отсортировать предложения в соответствии со значением скидки, а затем передать отсортированный список в шаблон для последующего отображения.

Для вывода предложений из базы данных используется цикл **{% for offer in offers %}**, который позволяет итерироваться по всем доступным предложениям. Для отображения информации о каждом предложении используются следующие теги:

* **{{ offer.title }}** - отображает заголовок предложения.
* **{{ offer.price|floatformat }}** - выводит цену предложения, применяя метод floatformat для удаления лишних нулей после запятой, что улучшает читаемость числа.
* **{{ offer.discount\_price|floatformat }}** - предоставляет цену предложения со скидкой, также с применением floatformat.
* **{{ offer.discount|floatformat }}** - отображает процент скидки на данное предложение, также с применением floatformat.
* **{{ offer.description }}** - содержит описание услуги, которое будет выведено на странице.

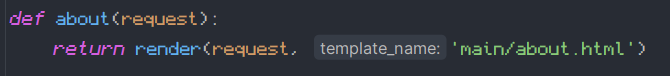
Эти теги позволяют динамически формировать информацию о каждом предложении, предоставляя пользователям подробные сведения о доступных услугах на сайте.



## **Страница «О нас»**

На странице «О нас» находятся необходимые детали о команде фитнес клуба в виде маркированного списка.

Также на странице находится информация об оборудовании, тренировках и прочих услугах клуба.



## **Страница «Услуги»**

На странице «Услуги» реализованы два основных блока, предоставляющих информацию о различных предложениях:

1. **Блок с предложениями со скидкой:**

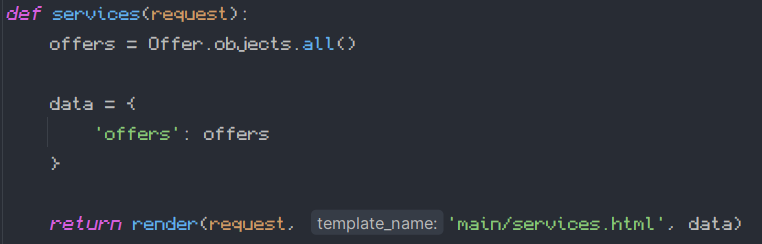
В этом блоке отображаются все доступные услуги, предоставляемые с скидкой не менее 20%. Визуальное представление предложений на странице осуществляется в порядке убывания величины скидки, начиная с самых выгодных и заканчивая менее выгодными вариантами.

Для реализации этой функциональности используется цикл **{% for offer in offers %}**, который проходит по всем предложениям из базы данных. Проверка на размер скидки осуществляется при помощи условия **{% if offer.discount >= 20 %}**, где отображаются только те предложения, у которых скидка соответствует или превышает 20%.

1. **Блок с информацией о всех предложениях:**

В этом блоке представлена общая информация обо всех предложениях, независимо от размера скидки. Предложения отображаются в порядке их добавления к услугам. Этот блок предоставляет полный обзор всех доступных вариантов услуг, что может быть полезным для пользователей, интересующихся разнообразием предоставляемых услуг.

Такая структура страницы услуг обеспечивает удобство в поиске и выборе услуг для пользователей, предоставляя информацию о скидках и общем перечне услуг на сайте.

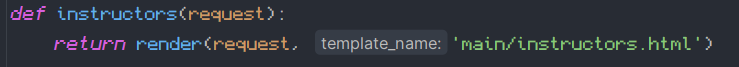


## **Страница «Тренеры»**

На странице "Тренеры" сайта представлена подробная информация о команде фитнес-клуба. Здесь пользователи могут ознакомиться с кратким описанием команды и получить доступ к списку всех тренеров, представленных с фотографиями и подробной информацией о каждом из них.

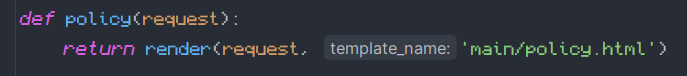
Для визуализации тренеров используется тег **{% static 'путь' %}**, который позволяет динамически загружать изображения тренеров на страницу. Каждый тренер представлен своей фотографией, что помогает пользователям лучше ознакомиться с членами команды.

Помимо визуального представления, краткая информация о каждом тренере также доступна на странице, например, специализация каждого тренера. Такая детализированная информация создает у пользователей уверенность в выборе тренера, отвечающего их требованиям.



## **Страница «Политика конфиденциальности»**

На странице "Политика конфиденциальности" представлена детальная и прозрачная политика, разработанная с целью обеспечения конфиденциальности и защиты данных пользователей.

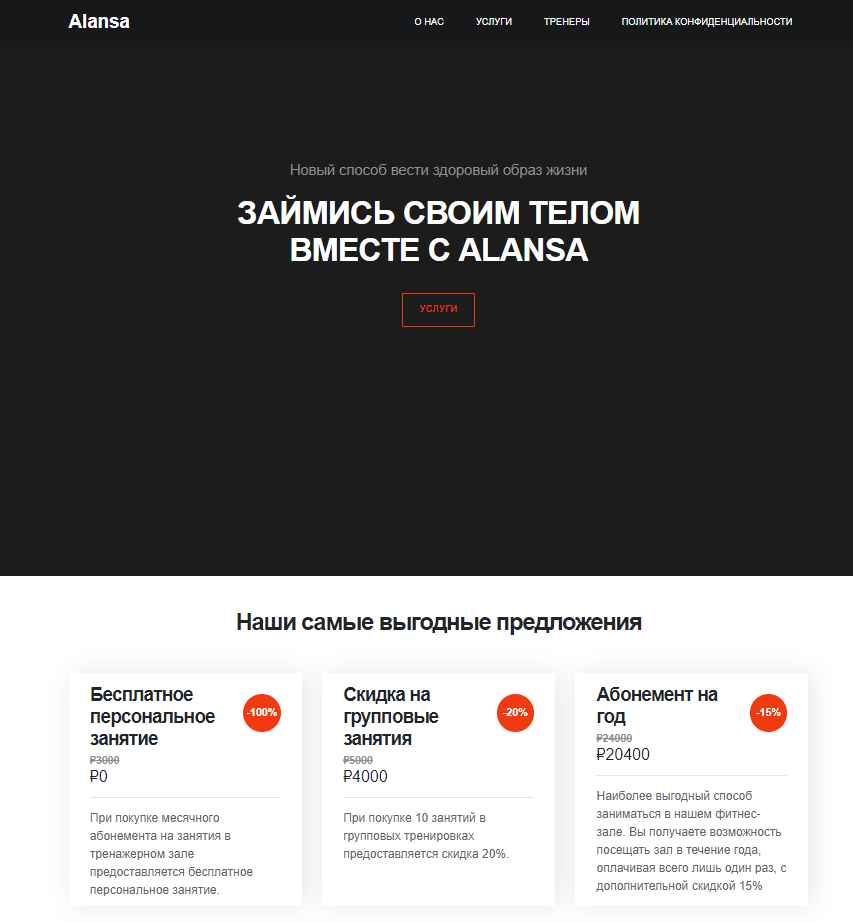
Цель страницы - обеспечить прозрачность и доверие в отношениях с пользователями, предоставляя им полную информацию о том, как обрабатываются и защищаются их личные данные.****

# **Результат**

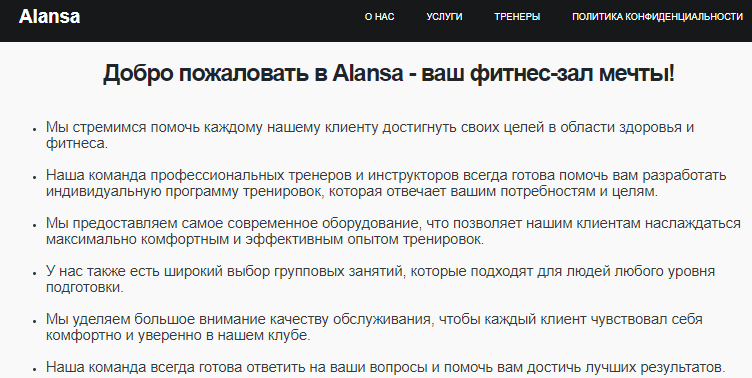
На сайте присутствуют 5 страниц:

* Главная страница
* Страница «О нас»
* Страница «Инструкторы»
* Страница «Тренеры»
* Страница «Политика конфиденциальности

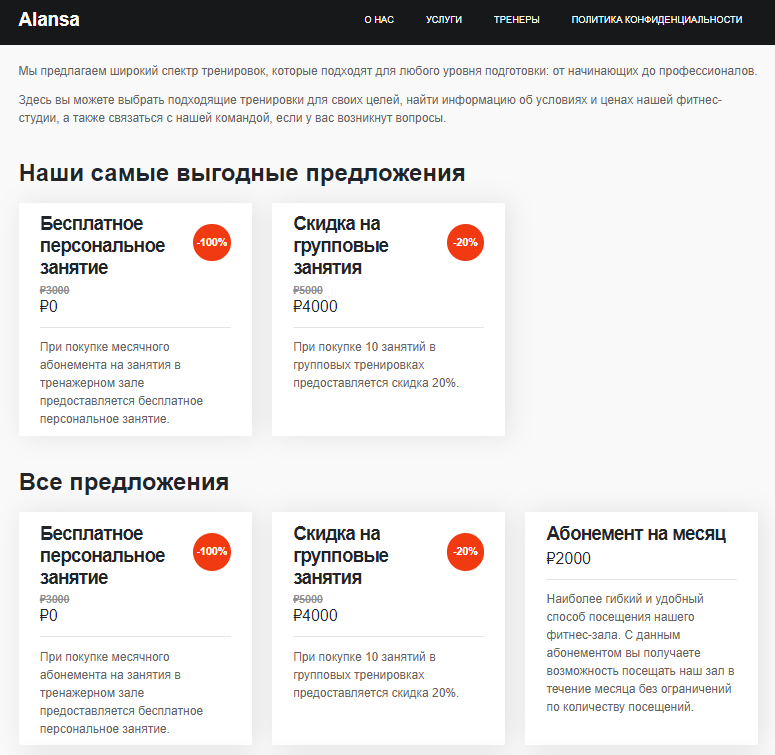
Главная страница:



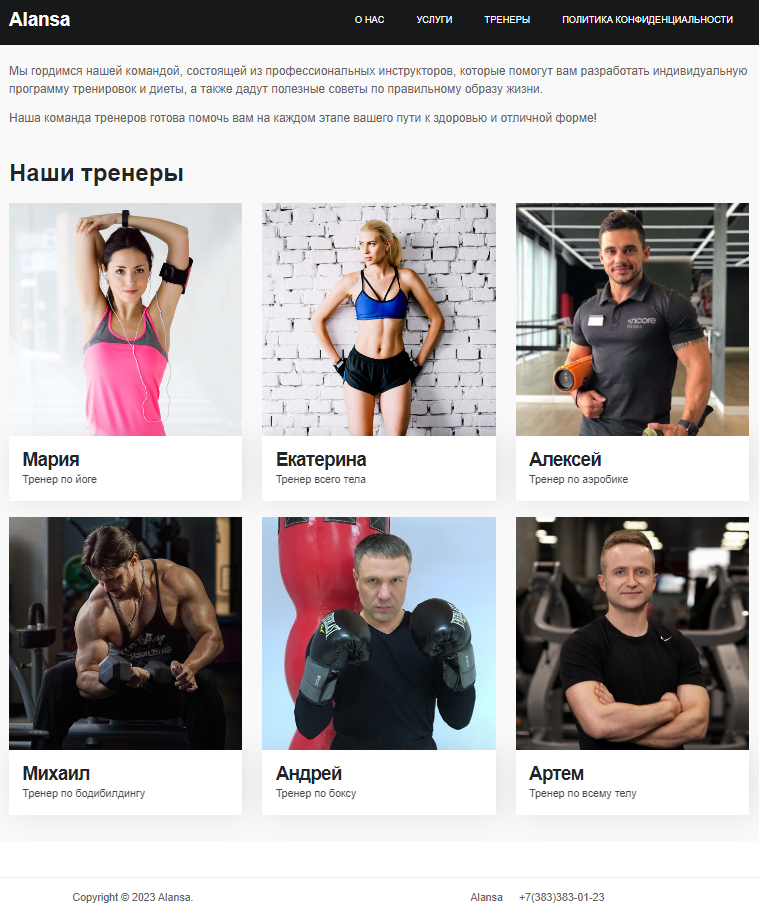
Страница «О нас»



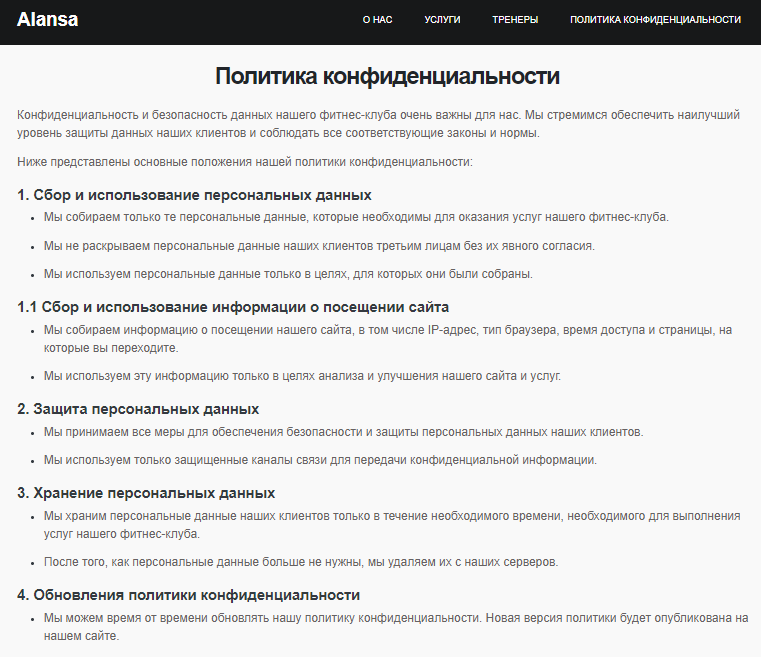
Страница «Услуги»



Страница «Тренеры»



Страница «Политика конфиденциальности»



# **Заключение**

В ходе выполнения дипломной работы был создан сайт на Django для компании, предоставляющей услуги фитнес, который включает в себя функционал для просмотра и бронирования услуг, получения информации о тренерах и организации, а также ознакомления с политикой конфиденциальности.

Была создана база данных со всеми услугами и предложениями, которая обеспечивает быстрый доступ к информации и удобство для пользователей. Главная страница сайта содержит общую информацию о компании, ее услугах и предложениях, что позволяет пользователям быстро найти нужную информацию.

Страницы с информацией о услугах, о тренерах и организации предоставляют более подробную информацию о компании и ее услугах, что помогает пользователям принимать более информированные решения при выборе услуг. Страница с политикой конфиденциальности обеспечивает безопасность и защиту данных пользователей.

Таким образом, созданный сайт на Django для фитнес-зала является удобным и функциональным инструментом для продвижения бизнеса в интернете и обеспечивает удобство и безопасность для пользователей. Разработка данного проекта на Django позволила сократить время на его создание и обеспечила высокую надежность и безопасность сайта.