МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГАОУ ВО «ПЕРМСКИЙ ГОСУДАРСТВЕННЫЙ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»

ЛИЦЕЙ С УГЛУБЛЕННЫМ ИЗУЧЕНИЕМ ОТДЕЛЬНЫХ

УЧЕБНЫХ ПРЕДМЕТОВ

**ИНСТРУМЕНТ ДЛЯ ПРОВЕДЕНИЯ УЧЕБНЫХ МЕРОПРИЯТИЙ И РАБОТ**

Индивидуальный проект

Ученика

технологического класса

Турова А. Д.

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

*подпись*

Научные руководители:

Преподаватель  
Мерзляков. А. Ф.

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

*подпись*

Преподаватель

Почтоева. О. Д.

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

*подпись*

Пермь 2023

**ОГЛАВЛЕНИЕ**

[Введение 4](#_Toc154105847)

[1. ТЕХНИЧЕСКОЕ ОПИСАНИЕ ИНСТРУМЕНТА 6](#_Toc154105848)

[1.1. Функционал инструмента для пользователей 6](#_Toc154105849)

[1.1.1. Типы пользователей 6](#_Toc154105850)

[1.1.2. Регистрация и авторизация 6](#_Toc154105851)

[1.1.3. Функционал для организатора мероприятий 6](#_Toc154105852)

[1.1.4. Функционал для участника мероприятий 7](#_Toc154105853)

[1.1.5. Функционал для пользователей 7](#_Toc154105854)

[1.2. Мероприятия 8](#_Toc154105855)

[1.2.1. Виды мероприятия 8](#_Toc154105856)

[1.2.1.1. Моментальный вывод правильных ответов 8](#_Toc154105857)

[1.2.1.2. “Секретные” правильные ответы 9](#_Toc154105858)

[1.2.1.3. Вывод правильных ответов после прохождения мероприятия 9](#_Toc154105859)

[1.2.2. Тест 9](#_Toc154105860)

[1.2.3. Контрольная работа 9](#_Toc154105861)

[1.2.4. Смешанный тип мероприятия 10](#_Toc154105862)

[1.3. Дерево сайта 10](#_Toc154105863)

[1.4. Используемый стек технологий 15](#_Toc154105864)

[1.5. Дизайн 15](#_Toc154105865)

[1.6. Описание базы данных 15](#_Toc154105866)

[2. ЧТО ТАКОЕ САЙТ? КАКИЕ БЫВАЮТ САЙТЫ? 17](#_Toc154105867)

[2.1. Виды сайтов по функциональности 17](#_Toc154105868)

[2.2. Виды сайтов по отзывчивости дизайна 18](#_Toc154105869)

[2.3. Виды сайтов по способу формирования контента 18](#_Toc154105870)

[2.4. Виды сайтов по типу контента 19](#_Toc154105871)

[3.ОБЗОР НА ФРЕЙМВОРК DJANGO 20](#_Toc154105872)

[3.1. Что же представляет из себя Django? 20](#_Toc154105873)

[3.2. История появления? В чем основные плюсы Django? 21](#_Toc154105874)

[3.3. Проекты и приложения 21](#_Toc154105875)

[3.4. Главные компоненты веб-фреймворка 22](#_Toc154105876)

[3.5. Архитектура 23](#_Toc154105877)

[3.6. Некоторые возможности Django 25](#_Toc154105878)

[4. ОБЗОР НА BOOTSTRAP 28](#_Toc154105879)

[4.1. История появления Bootstrap 28](#_Toc154105880)

[4.2. Из каких компонентов состоит Bootstrap? 29](#_Toc154105881)

[4.3. Преимущества и недостатки Bootstrap 29](#_Toc154105882)

[5. ЭТАПЫ РАЗРАБОТКИ 32](#_Toc154105883)

[5.1. Frontend-разработка 32](#_Toc154105884)

[5.2. Backend-разработка 33](#_Toc154105885)

[5.3. Deploy проекта 35](#_Toc154105886)

[5.3.1. Nginx 35](#_Toc154105887)

[5.3.2. Gunicorn 35](#_Toc154105888)

[5.4. Запуск проекта в режиме разработки 36](#_Toc154105889)

[Заключение 39](#_Toc154105890)

[Список источников 40](#_Toc154105891)

# Введение

В современном образовании учебные мероприятия проводятся не так эффективно, как могли бы. Создателям образовательных мероприятий приходится тратить много времени на проверку ответов обучающихся, составление рейтинга, анализ результатов. При наличии некоторого инструмента, который решает проблемы, перечисленные выше, организаторы могут значительно сэкономить время на работу, которую можно автоматизировать, и сосредоточится на чем-то более важном (например, на содержании мероприятия или работы, проверки наличия ошибок в работе).

Существуют подобные на мой проект образовательные инструменты. Самые известные из них это СДАМ ГИА — образовательный портал для подготовки к экзаменам [14], Kahoot — это игровая обучающая платформа [16], а также Stepik — многофункциональная и гибкая платформа для создания образовательных материалов [15]. В каждом из этих трёх инструментов есть как плюсы, так и минусы. Например, на сайте СДАМ ГИА отсутствует возможность создавать свои уникальные задания, вместо этого организаторы могут брать задания из огромного банка заданий, а также нельзя отслеживать рейтинг в реальном времени. На сайтах Stepik, Kahoot нельзя создавать мероприятия, в которых содержатся задания, вид которых отличается от стандартных, а также нет возможности создать уникальные правила мероприятия.

Три самых важных преимущества нашего инструмента, которые отличают его от других похожих проектов – это:

1. Составление рейтинга, согласно правилам выбранного мероприятия
2. Гибкость настройки мероприятия
3. Возможность создавать гибкие задания

**Цель проекта:** разработать эффективный образовательный инструмент для создания и проведения учебных мероприятий.

**Задачи:**

1. Составить техническое описание проекта
2. Изучить классификацию сайтов
3. Провести исследование стека технологий для написания бэкенда и фронтенда сайта
4. Реализовать проект в виде сайта

**Объект:** сайт

**Предмет:** разработка сайта

Проект имеет практическое значение, так как может быть использован на занятиях в школах и других учебных заведениях, а также при проведении внеурочных мероприятий.

В работе использованы теоретические и эмпирические методы. К первым можно отнести анализ данных, их обобщение, моделирование. Ко вторым – анализ и обработка данных, сравнение. Методология включает в себя разработку и программирование сайта при помощи выбранных фреймворков, а также его тестирование и внедрение в работу.

# 1. ТЕХНИЧЕСКОЕ ОПИСАНИЕ ИНСТРУМЕНТА

## 1.1. Функционал инструмента для пользователей

Инструмент состоит из следующих основных функциональных блоков:

1. Регистрация, аутентификации, авторизация;
2. Функционал для организатора мероприятий;
3. Функционал для участника мероприятий;
4. Функционал для пользователей, которые в данный момент времени не являются ни организаторами, ни участниками.

### 1.1.1. Типы пользователей

Инструмент предусматривает два вида пользователей: организатор, участник. Тип пользователя может меняться в зависимости от роли в мероприятии. Например, если, пользователь создает мероприятие, то он организатор, но после проведения этого мероприятия он сам может принять участие в другом мероприятии и тогда он станет участником.

### 1.1.2. Регистрация и авторизация

При регистрации аккаунта запрашиваются следующие поля:

* username – обязательное поле;
* email – обязательное поле;
* пароль – обязательное поле;
* проверка пароля – обязательное поле.

При авторизации аккаунта запрашиваются следующие поля:

* username – обязательное поле;
* пароль – обязательное поле.

### 1.1.3. Функционал для организатора мероприятий

Организатор обладает следующим функционалом:

* Выбор шаблона мероприятия из предложенных (тест, контрольная, смешанный тип) и создание его с гибкой настройкой функционала.
* Отображение рейтинга участников мероприятия, как в реальном времени, так и после окончания мероприятия, а также возможность ставить таймер, после истечения которого участники не смогут видеть рейтинг, это нужно, чтобы личность победителя оставалась тайной.
* После окончания мероприятия, организатор может оставить или убрать доступ на просмотр заданий и их решений (если есть) всем участникам мероприятия, а также всем пользователям сайта.
* При создании закрытого мероприятия генерируется специальный секретный ключ, по которому остальным пользователям (в будущем участникам) можно присоединиться к мероприятию.

### 1.1.4. Функционал для участника мероприятий

Участник обладает следующим функционалом:

* Вход в закрытое мероприятие по секретному ключу.
* Возможность принять участие в закрытом мероприятие, при наличии секретного ключа.
* Регистрация на мероприятие, после которой пользователь может войти в мероприятия и выполнить предложенные задания.
* Возможность видеть, как текущий рейтинг участников, так и рейтинг участников после завершения мероприятия.
* Если мероприятие предполагает моментальный вывод правильного или неправильного ответа, то он выводится.

### 1.1.5. Функционал для пользователей

После аутентификации пользователь имеет доступ к своему личному кабинету. В личном кабинете реализованы следующие функциональные блоки:

* Редактирование данных профиля.
* Смена пароля.
* Просмотр образовательных мероприятий, в которых пользователь принимал участие, со списками задач, решениями (если есть), своими ответами.

## 1.2. Мероприятия

Мероприятие состоит из заданий, на которые участники дают свой ответ. В зависимости от его типа вид вопроса и ответа меняется.

В инструменте множество категорий: тест, контрольная и смешанный тип (контрольная + тест).

Любое мероприятие вне зависимости от его типа обладает следующим функционалом:

* К любому заданию мероприятия можно прикрепить не более 3 фотографий.
* После прохождения мероприятия выводится количество верных ответов, а также процент верных ответов и оценка.
* Если задание предполагает ручную проверку, то его проверку осуществляет непосредственно организатор мероприятия, иначе проверка происходит автоматически.

### 1.2.1. Виды мероприятия

Любое мероприятие делится на три подтипа в зависимости от предполагаемого функционала:

#### 1.2.1.1. Моментальный вывод правильных ответов

После ответа на вопрос происходит один из двух возможных сценария:

1. Если ответ верен, то он подсвечивается зелёным цветом и выводится комментарий (если есть).
2. Если ответ неверен, то он подсвечивается красным цветом и выводится комментарий (если есть).
3. Если ответ предполагает ручной проверки, то выводится сообщение «Ваш ответ был отправлен организаторам олимпиады. Дождитесь его проверки».

#### 1.2.1.2. “Секретные” правильные ответы

Ни во время прохождения мероприятия, ни после участник не узнает правильные ответы. Он может видеть только количество верно выполненных заданий и сами задания.

#### 1.2.1.3. Вывод правильных ответов после прохождения мероприятия

Только после прохождения мероприятия будет видны те задания, в которых допущены ошибки.

### 1.2.2. Тест

Тест обладает следующей особенностью – к вопросу предложен список ответов, из которых надо выбрать один или несколько верных.

Тест предполагает следующие функционал:

* Количество ответов может быть не более 1000;
* Проверка ответов происходит автоматически.

### 1.2.3. Контрольная работа

Контрольная работа обладает следующей особенностью – ниже вопроса есть поля для ввода ответа, в который нужно ввести точный ответ или прикрепить файл.

Контрольная работа предполагает следующие функциональные блоки:

* Если вопрос требует ответа, состоящего из слов, то проверка проходит автоматически.
* Если вопрос требует развернутого ответа, то нужно прикрепить файл. В данном случае проверка происходит вручную организатором мероприятия.

### 1.2.4. Смешанный тип мероприятия

Смешанный тип мероприятия (тест + контрольная) – это мероприятие, включающее в себя совокупность всех особенностей вышеперечисленных мероприятий. Для каждого задания в отдельности предложено выбрать формат его представления.

## 1.3. Дерево сайта

|  |  |  |
| --- | --- | --- |
| Название страницы | URL страницы | Описание страницы |
| Главная страница | https://tcse.ru/ | На этой странице находятся обзор сайта |
| Домашняя страница | https://tcse.ru/home | На этой странице находятся мероприятия, созданные пользователем, и мероприятия, в которых пользователь принимал участие |
| Обратная связь | https://tcse.ru/feedback | На этой странице находится форма с отзывом о сайте, а также поле для ввода пожеланий, например, предложение какую новую категорию мероприятий добавить |
| О сайте | https://tcse.ru/about | На этой странице находится основная информация о сайте |
| Авторизация | https://tcse.ru/login | На этой странице происходит авторизация пользователя по его логину и паролю |
| Регистрация | https://tcse.ru/signup | На этой странице происходит регистрация пользователя по следующим полям: логин, электронная почта, пароль |
| Выход из аккаунта | https://tcse.ru/logout | На этой странице происходит выход пользователя из аккаунта |
| Профиль пользователя | https://tcse.ru/user/username | На этой странице показывается информация об пользователе, а именно фамилия, имя, логин, почта |
| Смена пароля | https://tcse.ru/user/username/ change\_password | На этой странице отображается форма для смены пароля |
| Редактирование данных пользователя | https://tcse.ru/user /username/ change | На этой странице отображается форма для редактирования, данных пользователя, а именно фамилии, имени, почты и логина |
| Просмотр мероприятий, в которых участвовал пользователь | https://tcse.ru/user /username/ events | На этой странице отображаются список мероприятий, в которых пользователь принимал участие, в виде таблицы |
| Просмотр мероприятия, в котором участвовал пользователь | https://tcse.ru/user /username/ events/ name\_of\_event | На этой странице отображается описание мероприятия, в котором принимал участие пользователь, а также результат участия |
| Просмотр ответов пользователя на мероприятие | https://tcse.ru/user /username/ events/ name\_of\_event/  number\_of\_task | На этой странице отображается ответ, данный пользователем на данное задание, а также правильный ответ, если его наличие предусмотрено правилами мероприятия |
| Мероприятия | https://tcse.ru/ events | На этой странице отображается список мероприятий по категориям |
| Категория мероприятия | https://tcse.ru/ events/ name\_of\_category | На этой странице отображается описание категории мероприятия |
| Регистрация и вход в мероприятие | https://tcse.ru/events/ name\_of\_category/ name\_of\_event | На этой странице пользователь видит описание мероприятия и регистрируется на него. Если мероприятие приватное, то пользователь сначала вводит секретный ключ, а потом регистрируется и участвует в нём |
| Прохождение мероприятия | https://tcse.ru/events/ name\_of\_category/ name\_of\_event/ number\_of\_task | На этой странице пользователь выполняет само задания, а также переключается между заданиями мероприятия |
| Рейтинг мероприятия | https://tcse.ru/events/ name\_of\_category/ name\_of\_event/ rating | На этой странице пользователь видит рейтинг мероприятия в виде таблицы |
| Выбор категории мероприятия | https://tcse.ru/events/create | На этой странице пользователь выбирает категорию мероприятия, которое он будет создавать |
| Создание мероприятия | https://tcse.ru/events/create/ name\_of\_category | На этой странице пользователь создает мероприятие с гибкой настройкой и заданиями типа выбранной категории |
| *Таблица 1. Описание дерева сайта* | | |

В таблице 1 в столбце под названием URL таблицы следующие слова означают:

* username – логин пользователя;
* name\_of\_event – название мероприятия (поле slug из таблицы event в БД);
* number\_of\_task – номер текущего задания;
* name\_of\_category – название категории (поле slug из таблицы category в БД).

Если пользователь не авторизован, то страницы: обратная связь, выход из аккаунта, смена пароля, редактирование данных пользователя, просмотр мероприятий, в которых участвовал пользователь, просмотр мероприятия, в котором участвовал пользователь, просмотр ответов пользователя мероприятие, регистрация и вход в мероприятие, прохождение мероприятия, выбор категории мероприятия, создание мероприятия – автоматически редиректятся на страницу авторизации пользователя.

|  |
| --- |
|  |
| *Рис 1. Дерево сайта* |

## 1.4. Используемый стек технологий

* Backend:
  + Язык Python (Django);
  + СУБД SQLite,
* Frontend:
  + HTML;
  + CSS;
  + JS;
  + Bootstrap.
* Production:
  + NGINX;
  + GUNICORN.

## 1.5. Дизайн

Минималистичный, интуитивно понятный. Белый фон, с элементами тёмно-синего цвета на странице. В рамках проекта разработан логотип инструмента, который присутствует в «шапке» сайта. Сайт адаптивный и динамический.

## 1.6. Описание базы данных

На рисунке 2 представлена ERD (entity relationship diagram) базы данных.

|  |
| --- |
|  |
| *Рис. 2. ER диаграмма базы данных* |

# 2. ЧТО ТАКОЕ САЙТ? КАКИЕ БЫВАЮТ САЙТЫ?

Сайт — это совокупность связанных между собой веб-страниц, которые доступны в интернете через протоколы HTTP/HTTPS. Совокупность всех общедоступных веб-сайтов называют всемирной паутиной или сетью. [7]

Сайт — совокупность электронных документов (файлов) частного лица или организации в компьютерные сети, объединённых под одним адресом (доменным именем или IP-адрес). [3]

Все сайты в совокупности составляют Всемирную паутину, где коммуникация (паутина) объединяет сегменты информации мирового сообщества в единое целое — базу данных и коммуникации планетарного масштаба. Для прямого доступа клиентов к сайтам на серверах был специально разработан протокол HTTP. [3]

## 2.1. Виды сайтов по функциональности

Сайты в интернете можно разделить по функциональности. Считают, что веб-сайты с наилучшим дизайном являются наиболее сложными с функциональной стороны. Однако, зачастую ситуация обратная. Возьмем, к примеру, гиганта интернет-торговли Amazon. Сложно назвать Amazon визуально привлекательным сайтом, но он спроектирован для реализации широкого круга задач. [1], [2], [17]

Существует девять основных видов сайтов:

* Сайты-визитки;
* Сайты электронной коммерции;
* Интернет-порталы (в том числе и магазины);
* Социальные платформы;
* Вики-сайты;
* Web-приложения;
* Персональный сайт;
* Одностраничный (информационный) сайт;
* Корпоративные информационные сайты.

Мой проект относится к web-приложение.

Web-приложениями являются разнообразные программные решения, доступ к которым предоставляется через веб-браузер. Они имеют ряд достоинств, и почти все крупные компании используют их как часть своих пользовательских предложений.

## 2.2. Виды сайтов по отзывчивости дизайна

С учетом того, насколько эффективно веб-сайт адаптирован для разных электронных устройств, можно выделить четыре группы макетов и шаблонов сайта:

* Статический;
* жидкий (подвижный);
* отзывчивый (responsive);
* адаптивный макет.

Мой проект относится к адаптивному макету.

В адаптивной разработке дизайна, макет определяет ширину экрана и с применением медиа-запросов будет корректироваться соответственно.

Адаптивный макет включает модули с фиксированными размерами, но их количество обеспечивает совместимость с любыми экранами. CSS медиа-запрос распознает ширину браузера и автоматически подбирает для него наиболее подходящий вариант размера. [1]

## 2.3. Виды сайтов по способу формирования контента

Сайты могут быть разделены на две категории в зависимости от способа формирования и обновления контента:

* Статические сайты

Эти сайты являются наиболее простыми и возникли в интернете раньше всех других. Содержимое статических сайтов не изменяется в зависимости от действий пользователя и обновляется нечасто, только с привлечением контент-менеджера. Такие сайты создаются с помощью базового HTML-кода и выполняют информационную функцию.

* Динамические сайты

Динамические сайты позволяют отображать различный контент при каждом посещении. К ним относятся блоги, сайты электронной коммерции и в целом любой сайт, который регулярно обновляется с активным участием пользователей. Такие сайты могут быть настроены для показа разного контента разным пользователям, учитывая время суток и другие факторы. Динамические сайты предоставляют более индивидуализированный и интерактивный интерфейс для пользователей, однако их создание является более сложным и дорогостоящим по сравнению со статическими сайтами.

Мой проект относится к динамическому виду сайта. [1]

## 2.4. Виды сайтов по типу контента

Когда речь идёт о классификации веб-сайтов по типу контента, возможно, бесчисленное количество вариантов. Вот несколько самых популярных направлений для создания контента на сайтах, хотя стоит учесть, что между ними могут быть пересечения и схожие элементы: [1], [19]

* блоги;
* корпоративные сайты;
* краудфандинговые платформы;
* сайты электронной коммерции;
* образовательные ресурсы;
* новостные сайты;
* социальные медиа;
* сайты ТВ или потокового видео.

Мой проект относится к образовательным ресурсам.

# 3.ОБЗОР НА ФРЕЙМВОРК DJANGO

При выборе инструмента для написания бэкенда сайта были рассмотрены следующие фреймворки на языке Python: Django, Flask, FastAPI.

Django является одним из самых популярных и проверенных временем фреймворков для разработки веб-приложений на Python. В отличие от Flask, который имеет меньший размер кодовой базы и больше подходит для небольших проектов, Django обеспечивает более полный и структурированный подход к разработке сложных веб-приложений с акцентом на управлении базами данных.

FastAPI, будучи самым современным и передовым фреймворком, предоставляет высокую производительность и поддержку асинхронных запросов. Однако, выбор Django более предпочтителен для нашего проекта, так как стабильность, многолетний опыт использования и обширное сообщество разработчиков имеют большее значение.

В целом, выбор фреймворка Django обоснован его проверенным качеством, богатым функционалом и поддержкой сложных веб-приложений с акцентом на управлении базами данных.

## 3.1. Что же представляет из себя Django?

Django — мощный веб-фреймворк на языке Python, предоставляющий обширные возможности для создания сложных сайтов и веб-приложений. Он состоит из набора компонентов, которые решают распространенные задачи веб-разработки, и эти компоненты слабо сцеплены между собой, что позволяет управлять ими независимо и разделять обязанности разных слоев веб-фреймворка. В результате, слой базы данных не знает о том, как данные отображаются на странице, а система шаблонов не знает о веб-запросах и т. д. Эффективность использования Django подтверждается его популярностью среди разработчиков и успешными примерами веб-приложений, созданных на его основе.

## 3.2. История появления? В чем основные плюсы Django?

Django был создан между 2003 и 2005 годами группой разработчиков, работавших над веб-сайтами газет. После разработки нескольких сайтов они начали повторно использовать общий код и шаблоны, что привело к созданию веб-фреймворка Django, который стал проектом с открытым исходным кодом в июле 2005 года. [18]

С тех пор Django продолжает расти и улучшаться, включая последние обновления в версии 4. Основные преимущества Django включают поддержку различных типов баз данных, шаблонизаторов и кэширования, а также добавление общих функций и классов, что упрощает разработку для программистов. В настоящее время Django является процветающим проектом с открытым исходным кодом, в который вовлечены тысячи пользователей и участников, и он способен создавать веб-сайты любого типа.

## 3.3. Проекты и приложения

Один из основных принципов фреймворка — DRY (don't repeat yourself). Веб-системы на Django строятся из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из заметных архитектурных отличий этого фреймворка от некоторых других.

В Django проектом считается установленный веб-фреймворк Django с несколькими настроечными параметрами.

Приложение — это группа моделей, шаблонов, URL-адресов и представлений. Приложения взаимодействуют с веб-фреймворком с целью обеспечения определенных функциональностей, и их можно реиспользовать в разных проектах.

Проект можно трактовать как свой собственный веб-сайт, содержащий несколько приложений, которые другие проекты Django тоже могут использовать.

На рисунке 3 показана структура проекта Django

|  |
| --- |
|  |
| *Рис. 3. Структура проекта/приложений Django* |

## 3.4. Главные компоненты веб-фреймворка

Архитектура Django похожа на MVC (Model - Views - Controller). Контроллер классической модели MVC примерно соответствует уровню, который в Django называется Представление (View), а презентационная логика Представления реализуется в Django уровнем Шаблонов (Templates). Из-за этого уровневую архитектуру Django часто называют «Модель-Шаблон-Представление» (MTV). [5]

Но все же сложно сказать, что Django следует именно модели MVC, так как Controller обрабатывается в самой среде разработки, а все самое интересное в этом фреймворке происходит в моделях, шаблонах и представлениях. Из вышесказанного стоит отметить, что Django использует модель MTV (Model – Template - View).

Обязанности в шаблоне архитектурного дизайна MTV Django распределены следующим образом:

* Модель – определяет логическую структуру данных и является обработчиком данных между базой данных и их представлением;
* Шаблон – это слой представления. В Django используется система текстовых шаблонов, в которой хранится все, что браузер прорисовывает на страницах;
* Представление – взаимодействует с базой данных через модель и передает данные в шаблон для их прорисовки и просмотра.

Сам веб-фреймворк выступает в качестве контроллера. Он отправляет запрос в надлежащее представление в соответствии с конфигурацией URL-адреса.

## 3.5. Архитектура

На рисунке 4 демонстрируется процесс обработки запросов Django включая различные ключевые компоненты системы, такие как модели, шаблоны, URL-адреса и представления, которые управляют циклом запроса/ответа.

|  |
| --- |
|  |
| *Рис. 4. Архитектура Django* |

Вот как Django оперирует HTTP-запросами и генерирует ответы:

1. Веб-браузер запрашивает страницу по ее URL-адресу, и веб-сервер передает HTTP-запрос веб-фреймворку Django.
2. Django просматривает свои сконфигурированные шаблоны URL-адресов и останавливается на первом, который совпадает с запрошенным URL-адресом.
3. Django исполняет представление, соответствующее совпавшему шаблону URL-адреса.
4. Представление потенциально использует модели данных, чтобы извлекать информацию из базы данных.
5. Модели данных обеспечивают определение данных и их поведение. Они используются для запроса к базе данных.
6. Представление прорисовывает шаблон (обычно с использованием HTML), чтобы отображать данные на странице, и возвращает их вместе с HTTP-ответом.

## 3.6. Некоторые возможности Django

* ORM, API доступа к БД с поддержкой транзакций: Django предоставляет высокоуровневый ORM, который позволяет работать с базами данных, используя Python-код, без необходимости писать SQL-запросы. Он поддерживает транзакции для обеспечения целостности данных.
* Встроенный интерфейс администратора, с уже имеющимися переводами на многие языки: Django включает в себя готовый к использованию интерфейс администратора, который позволяет управлять данными вашего приложения прямо из браузера. Интерфейс администратора имеет переводы на разные языки и может быть легко настроен.
* Диспетчер URL на основе регулярных выражений: Django предоставляет мощный механизм маршрутизации URL, который позволяет определить пути к вашим представлениям на основе регулярных выражений. Это упрощает процесс создания SEO-оптимизированных URL-адресов.
* Расширяемая Инструмент шаблонов с тегами и наследованием: Django использует собственный язык шаблонов, который позволяет упростить процесс создания HTML-страниц. Язык шаблонов предоставляет теги и фильтры для обработки данных и поддерживает наследование шаблонов, что облегчает повторное использование кода.
* Инструмент кэширования: Django предлагает различные способы кэширования для улучшения производительности вашего веб-приложения. Можно кэшировать отдельные представления, части шаблона или результаты запросов к базе данных.
* Интернационализация: Django поддерживает многоязычность и позволяет легко создавать интернационализированные веб-приложения, используя встроенные инструменты для перевода текстов и форматирования даты и времени.
* Подключаемая архитектура приложений, которые можно устанавливать на любые Django-сайты: Django позволяет создавать модульные приложения, которые можно легко интегрировать в другие проекты на Django. Это облегчает повторное использование кода и расширение функциональности веб-приложения.
* «generic views» — шаблоны функций контроллеров: Django предоставляет набор предопределенных представлений, которые облегчают создание стандартных функций контроллеров, таких как отображение списка объектов или обработка форм.
* Авторизация и аутентификация. Django имеет встроенную систему аутентификации и авторизации, которая позволяет управлять пользователями и их правами доступа. Вы также можете подключить сторонние модули аутентификации, такие как LDAP или OpenID.
* Инструмент фильтров («middleware») в Django позволяет создавать дополнительные обработчики запросов. Встроенные фильтры включают кэширование, сжатие, нормализацию URL и поддержку анонимных сессий. Middleware — это система "плагинов" для глобального изменения входных и выходных данных Django. Каждый компонент middleware отвечает за выполнение определенной функции, например, аутентификацию пользователей через сессии.
* Библиотека для работы с формами в Django облегчает создание и наследование форм на основе существующих моделей базы данных. Это позволяет ускорить процесс разработки, так как не требуется создавать формы с нуля.
* В Django имеется встроенная автоматическая документация по тегам шаблонов и моделям данных, доступная через административное приложение. Это удобный инструмент для быстрого доступа к информации о структуре и возможностях Django. [4], [6], [8], [9]

# ОБЗОР НА BOOTSTRAP

Bootstrap — это набор инструментов для разработки веб-сайтов и веб-приложений. Инструментарий включает в себя HTML, CSS и JavaScript шаблоны и компоненты, такие как меню, формы, кнопки и т.д. Bootstrap разработан, чтобы ускорить процесс создания веб-сайтов и дать возможность разработчикам создавать современные и отзывчивые веб-сайты. Bootstrap также предоставляет возможность создания адаптивных веб-сайтов, которые оптимально отображаются на разных мобильных устройствах, планшетах и настольных компьютерах.

## 4.1. История появления Bootstrap

Bootstrap был разработан дизайнерами и разработчиками Twitter и стал одним из наиболее популярных фреймворков пользовательского интерфейса и проектов с открытым исходным кодом.

Созданный в 2010 году Марком Отто и Джейкобом Торнтоном, Bootstrap первоначально назывался Twitter Blueprint. После проведения первой недели Hack в Twitter проект получил массовое развитие, и разработчики всех уровней присоединились к нему. За год до официального выпуска Bootstrap использовался для создания внутренних инструментов в компании и по-прежнему пользуется большой популярностью.

Первая версия Bootstrap была выпущена 19 августа 2011 года, и с тех пор было более 20 релизов, включая два крупных обновления с версиями 2 и 3. Во второй версии была добавлена адаптивная функциональность ко всему фреймворку, а в третьей версии библиотека была полностью переписана, чтобы по умолчанию поддерживать мобильные устройства.

В четвертой версии проекта было сделано два ключевых архитектурных изменения: переход на Sass и на CSS flexbox. Это было сделано для продвижения новых свойств CSS, уменьшения зависимостей и внедрения новых технологий в современных браузерах.

Версия Bootstrap 5 направлена на улучшение кодовой базы v4 с минимумом критических изменений. Разработчики улучшили существующие функции и компоненты, удалили поддержку старых браузеров, отказались от jQuery в пользу JavaScript и внедрили новые технологии, такие как настраиваемые свойства CSS. [11]

## 4.2. Из каких компонентов состоит Bootstrap?

Bootstrap включает в себя следующие компоненты:

* Инструменты для создания макетов, такие как обёрточные контейнеры, мощная система сеток, гибкие медиа-объекты и адаптивные утилитные классы.
* Классы для стилизации основного контента, включая текст, изображения, код, таблицы и элементы figure.
* Готовые компоненты, такие как кнопки, формы, горизонтальные и вертикальные навигационные панели, слайдеры, выпадающие списки, аккордеоны, модальные окна и всплывающие подсказки.
* Утилитные классы для решения традиционных задач, которые часто возникают перед веб-разработчиками, такие как выравнивание текста, отображение и скрытие элементов, задание цвета, фона, margin и padding отступов и т.д. [10]

## 4.3. Преимущества и недостатки Bootstrap

Плюсы использования Bootstrap для фронтенд-разработки сайтов включают следующие аспекты: [10], [11]

* Быстрота создания качественной адаптивной верстки благодаря готовым классам и компонентам, разработанным профессионалами, что упрощает работу даже для начинающих веб-разработчиков.
* Кроссбраузерность и кроссплатформенность, обеспечивающие корректное отображение и функционирование сайта в разных браузерах и операционных системах.
* Большое количество готовых, хорошо продуманных компонентов, которые прошли проверку большим сообществом веб-разработчиков на разных устройствах.
* Возможность настройки фреймворка под свой проект с помощью изменения SCSS переменных и использования миксинов, что позволяет менять количество колонок, цвета, радиусы скругления, отступы и многое другое.
* Низкий порог вхождения, так как для работы с Bootstrap не нужны глубокие знания HTML, CSS и JavaScript - достаточно знать основы этих технологий.
* Единый стиль дизайна и согласованность между различными компонентами, что обеспечивает гармоничный внешний вид.
* Огромное количество сообществ и учебных материалов, которые помогут не только разобраться с фреймворком, но и найти ответы на возникающие вопросы.

Однако, Bootstrap не подходит для всех ситуаций. Например, его нецелесообразно использовать для:

* Создания уникального дизайна фронтенда проектов.
* Верстки проектов, если у вас достаточно времени и знаний в HTML, CSS и JavaScript для самостоятельной разработки.

Bootstrap, как и большинство подобных фреймворков, имеет недостатки. Среди них можно отметить следующие: [10]

* Больший размер итоговых CSS и JS-файлов проекта по сравнению с самостоятельной разработкой, так как стили фреймворка и его JS-код содержат универсальный код для множества ситуаций, из которых в конкретном проекте может потребоваться только часть.
* Сложность использования Bootstrap для верстки сайтов с уникальным дизайном, так как в этом случае придется значительно переписывать код, и простая настройка переменных Bootstrap не будет достаточной.

Bootstrap был выбран в качестве инструмента для разработки фронтенда проекта, поскольку, несмотря на некоторые недостатки, этот фреймворк предлагает ряд значительных преимуществ. Основные причины выбора Bootstrap включают адаптивность, широкий набор компонентов, возможность стилизации и кастомизации тем, а также обширную документацию и активное сообщество разработчиков. Благодаря этим преимуществам, можно быстро и эффективно разрабатывать высококачественные веб-страницы и современные пользовательские интерфейсы, что играет ключевую роль в разработке нашего проекта.

# ЭТАПЫ РАЗРАБОТКИ

## 5.1. Frontend-разработка

Frontend — это та часть веб-сайта, которую пользователь видит и использует. Она отвечает за отображение контента и взаимодействие с пользователем прямо в браузере. Хотя frontend не занимается обработкой данных или выполнением сложных вычислений, он все равно играет важную роль в проекте. В конечном итоге именно от него зависит внешний вид и пользовательский опыт. Для реализации внешнего вида сайта используется HTML для структуры страницы, CSS для стилизации элементов и Bootstrap для создания адаптивного и современного дизайна.

Все страницы сайта разделены на несколько блоков:

* Aside – расположена слева. При маленьком разрешении экрана она скрывается. Сверху расположены логотип и название, далее вниз идут ссылки на страницы. Боковую панель можно закрыть или открыть при нажатии на кнопку стрелочки.
* Header – расположен сверху, но правее боковой панели. На нем расположены кнопки «Войти/Регистрация», а если текущий пользователь авторизован, то кнопка «Выйти» и ссылка на аккаунт пользователя.
* Footer – расположен снизу, но правее боковой панели. На нем расположены ссылки на страницы сайта, а также контактная информация.
* Main – расположен между Header и Footer. При изменении ширины Aside, т.е. при его скрытии, ширина Main, Header, Footer также меняется динамически. В Main в зависимости от текущей страницы, расположена различная информация.

Чтобы обеспечить работу сайта на любых устройствах, были использованы специальные возможности языка стилей CSS, а именно media-запросы и container queries, учитывающие размер экрана устройства пользователя при применении стилей. С помощью media-запросов все блоки сайта корректно отображаются на любом устройстве.

## 5.2. Backend-разработка

В ходе создания бэкенд части веб-сайта мы максимально использовали возможности фреймворка Django. При работе с Django важно разделять любой проект на отдельные приложения в соответствии с их функциональностью. Также стоит отметить, что каждый проект автоматически включает в себя базовое приложение, которое управляет настройками проекта, и его название совпадает с названием самого проекта.

Проект состоит из шести приложений: предустановленное – «project», и собственные – «about», «users», «events», «feedback», «homepage». В приложении «events» содержится вся логика работы мероприятий. Приложение «users» отвечает за авторизацию и регистрацию пользователей на сайте, а также хранение информации о пользователях в базе данных.

Файл «views.py» в папке приложения отвечает за отображение данных из базы данных в шаблонах страниц. В Django для создания представлений можно использовать два подхода: FBV (функциональные представления) и CBV (классы-представления). FBV обычно используется для создания простых многостраничных сайтов, в то время как CBV удобны для проектов, где требуется более гибкая архитектура.

Выбор CBV для разработки данного сайта был обоснован несколькими преимуществами:

* Возможность повторного использования кода: CBV позволяют создавать классы представлений, которые могут быть унаследованы и изменены для различных целей, обеспечивая гибкость и экономию времени.
* DRY (Don't Repeat Yourself) - использование CBV способствует уменьшению дублирования кода и повышению его сопровождаемости.
* Расширяемость кода: CBV можно легко расширить с использованием Mixins, что делает их более универсальными и функциональными.
* Структурирование кода: CBV на основе классов позволяют лучше организовать представления, обрабатывая различные HTTP-запросы с помощью методов класса, что способствует ясности и читаемости кода.
* Встроенные общие классы-представления Django: Django предоставляет набор готовых общих классов-представлений (например, ListView, DetailView и другие), которые упрощают работу с различными типами данных и запросами.

Модели (конструкторы), на основе которых создаются объекты в базе данных, отвечают за добавление определенной информации. Все эти модели хранятся в файлах «models.py». В каждой модели представлен класс, который содержит поля, обязательные для всех объектов данной модели, а также может содержать дополнительные методы для выполнения специфических функций. Каждая модель в проекте имеет метод «\_\_str\_\_», определяющий, какое поле объекта будет выведено при обращении к нему. Более того, практически все модели включают метод «get\_absolute\_url», необходимый для определения URL-пути на странице с объектом данной модели в файле «urls.py».

В «urls.py» содержится информация о том, какие URL-пути соответствуют различным страницам сайта. Эти URL-пути связаны с различными представлениями, определенными в «views.py». Кроме того, они могут использовать специальный метод «get\_absolute\_url» модели, чтобы включить уникальные детали объекта в URL-адресе.

Файл «admin.py» используется для настройки административной панели Django, предоставляя возможности для настройки отображения объектов модели. Это включает в себя сортировку объектов, выбор количества отображаемых объектов на странице, а также определение полей, которые можно использовать для перехода к редактированию объекта, автоматического заполнения полей на основе других и многое другое.

## 5.3. Deploy проекта

Последним этапом разработки стала публикация сайта в интернете. Для этого мы приобрели доменное имя, выбрали подходящий хостинг для размещения сайта, оплатили аренду сервера и установили необходимые библиотеки, после чего перенесли все файлы на сервер.

### 5.3.1. Nginx

Nginx — это веб-сервер и прокси-сервер, который может использоваться для обработки HTTP-запросов. С помощью него можно раздавать статические файлы, управлять нагрузкой, выполнять кэширование, обеспечивать безопасность соединений по SSL и многое другое.

Он известен своей высокой скоростью и эффективностью, способен обрабатывать огромное количество запросов одновременно. Более того, его функциональность можно расширить при помощи разнообразных плагинов и модулей, которые позволяют настраивать его в соответствии с вашими потребностями.

Для использования Nginx необходимо его установить на сервер и настроить соответствующие конфигурационные файлы. Например, его можно сконфигурировать для пересылки запросов на другие серверы или для кэширования статических файлов. Для удобства также существуют готовые дистрибутивы Nginx, уже настроенные для быстрого развертывания на сервере.

### 5.3.2. Gunicorn

Gunicorn (Green Unicorn) — это WSGI (Web Server Gateway Interface) HTTP-сервер для Python, который обеспечивает высокую производительность и надежность веб-приложений. Он используется для запуска Python-приложений, написанных с использованием фреймворков, таких как Django, Flask, Pyramid и других.

Gunicorn работает в качестве прокси-сервера, который принимает запросы от клиентов и направляет их к соответствующему приложению на Python, которое обрабатывает эти запросы и возвращает ответы. Важно отметить, что этот сервер способен обрабатывать несколько запросов одновременно и масштабироваться автоматически, чтобы поддерживать большое количество пользователей.

Кроме того, у Gunicorn имеется множество функций, обеспечивающих безопасность, надежность и управляемость приложений Python. Он поддерживает различные протоколы, включая HTTP, HTTPS и Unix-сокеты. Также предоставляет широкий спектр настроек, таких как количество рабочих процессов, таймауты и логирование.

В целом, Gunicorn является важным компонентом инфраструктуры веб-приложений Python, обеспечивающий высокую производительность, надежность и управляемость.

## 5.4. Запуск проекта в режиме разработки

Проект расположен в github репозитории https://github.com/Le0ks/TCSE.

Чтобы запустить этот проект в режиме разработки, вам нужно следовать этой инструкции.

Все команды следует запускать в командной строке в каталоге, где вы хотите разместить проект.

Первое, что нужно сделать, это клонировать репозиторий:

> git clone https://github.com/Le0ks/TCSE.git

> cd TCSE

Создайте виртуальную среду для установки зависимостей и активируйте ее:

> python -m venv venv

> venv\Scripts\activate

Затем установите все зависимости:  
(venv) > pip install -r requirements.txt

После того как pip закончит загрузку зависимостей, переименуйте файл ex.env в .env и откройте его.

В этом файле находятся следующие строки:

SECRET\_KEY=YOUR\_SECRET\_KEY

DEBUG=YOUR\_DEBUG

ALLOWED\_HOSTS=YOUR\_ALLOWED\_HOST

INTERNAL\_IPS=YOUR\_INTERNAL\_IPS

USER\_IS\_ACTIVE=YOUR\_USER\_IS\_ACTIVE

LOGIN\_ATTEMPS= YOUR\_LOGIN\_ATTEMPS

USE\_SMTP=YOUR\_USE\_SMTP

EMAIL\_HOST=YOUR\_EMAIL\_HOST

EMAIL\_PORT=YOUR\_EMAIL\_PORT

EMAIL\_USE\_TLS=YOUR\_EMAIL\_USE\_TLS

EMAIL\_USE\_SSL=YOUR\_EMAIL\_USE\_SSL

EMAIL\_HOST\_USER=YOUR\_EMAIL\_HOST\_USER

EMAIL\_HOST\_PASSWORD=YOUR\_EMAIL\_HOST\_PASSWORD

SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_KEY=YOUR\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_KEY

SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_SECRET=YOUR\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_SECRET

SOCIAL\_AUTH\_YANDEX\_OAUTH2\_KEY=YOUR\_SOCIAL\_AUTH\_YANDEX\_OAUTH2\_KEY=SOCIAL\_AUTH\_YANDEX\_OAUTH2\_SECRET=YOUR\_SOCIAL\_AUTH\_YANDEX\_OAUTH2\_SECRET

Измените присваиваемые значения на ваши. Например, вот так:

SECRET\_KEY=secret

DEBUG=True

ALLOWED\_HOSTS="\*"

INTERNAL\_IPS=127.0.0.1

USER\_IS\_ACTIVE=True

LOGIN\_ATTEMPS=5

USE\_SMTP=False

EMAIL\_HOST=smtp.yandex.ru

EMAIL\_PORT=465

EMAIL\_USE\_TLS=False

EMAIL\_USE\_SSL=True

EMAIL\_HOST\_USER=secret

EMAIL\_HOST\_PASSWORD=secret

SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_KEY=secret

SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_SECRET=secret

SOCIAL\_AUTH\_YANDEX\_OAUTH2\_KEY=secret

SOCIAL\_AUTH\_YANDEX\_OAUTH2\_SECRET=secret

Измените каталог на project и запустите сервер django в режиме разработчика:

(venv) > cd project

(venv) > python manage.py runserver

Перейдите на сайт <http://127.0.0.1:8000/>.

# Заключение

В ходе работы мы изучили и успешно применили основы разработки веб-сайтов на практике. Мы познакомились с инструментами, которые применяются для разработки вебсайтов: фреймворками, базами данных и различными языками программирования. На данный момент веб-сайт готов и протестирован для использования на различных устройствах. Проект можно продолжать развивать, добавлять новые типы мероприятий и вопросов, улучшать пользовательский интерфейс, а также оптимизировать работу сайта. Таким образом, мы успешно достигли цели работы, решив поставленные задачи.

# Список источников

**Литературные материалы**

1. Дронов В. А. Django 3.0. Практика создания веб-сайтов на Python: учебное пособие: "БХВ", 2021. 704 с.
2. Постолит А. В. Python, Django и PyCharm для начинающих: учебное пособие: «БХВ», 2021. 464 с.
3. Django for Beginners: Build Websites with Python and Django (Welcome to Django) / ed. by William S. Vincent: WelcomeToCode, 2020. 317 p.

**Электронные документы:**

1. Какие бывают сайты? Виды сайтов по контенту, дизайну и функциональности // URL: <https://sdvv.ru/articles/testovyy-razdel/kakie-byvayut-sayty-vidy-saytov-po-kontentu-dizaynu-i-funktsionalnosti/> (дата обращения - 20.11.2022)
2. Типы сайтов // URL: <https://avada-media.ua/tipy-sajtov/> (дата обращения - 13.10.2022)
3. Академик // URL: <https://dic.academic.ru/dic.nsf/ruwiki/1135216> (дата обращения - 21.10.2022)
4. Сложные IT-проекты. Автоматизация бизнеса // URL: <https://web-creator.ru/articles/django> (дата обращения - 17.12.2022)
5. Особенности и плюсы использования Django в web'е // URL: <https://habr.com/ru/sandbox/156526/> (дата обращения - 02.10.2022)
6. Django documentation // URL: <https://www.djangoproject.com/community/> (дата обращения - 17.11.2022)
7. SEONews // URL: <https://clck.ru/334Dhf> (дата обращения - 13.12.2022)
8. Документации Django // URL: <https://django.fun/docs/django/ru/4.0/> (дата обращения - 23.10.2022)
9. Django girl // URL: <https://djangogirls.org/ru/resources/> (дата обращения - 15.12.2022)
10. ИТШеф // URL: https://itchief.ru/bootstrap/introduction/ (дата обращения – 12.02.2023)
11. Документация Bootstrap // URL: <https://bootstrap-4.ru/docs/5.0/about/overview/> (дата обращения – 12.02.2023)
12. СДАМ ГИА // URL: <https://sdamgia.ru/> (дата обращения - 13.05.2023)
13. Stepik // URL: <https://stepik.org/learn>/ (дата обращения – 13.05.2023)
14. Kahoot // URL: <https://kahoot.com/> (дата обращения – 13.05.2023)
15. Какие бывают сайты // URL: https://vebrost.ru/blog/kakie-byvayut-sayty/ (дата обращения – 20.11.2022)
16. What types of websites exist on the internet // URL: https://wayup.in/ru/blog/what-types-of-websites-exist-on-the-internet/ (дата обращения – 20.11.2022)
17. Виды сайтов // URL: https://gb.ru/blog/vidy-internet-sajtov/ (дата обращения – 20.11.2022)
18. История фреймворка Django // URL: https://clck.ru/34QXo7 (дата обращения - 20.12.2022)
19. Инструмент для проведение учебных мероприятий и работ // URL: https://tcse.ru (дата обращения – 25.12.2023)