Сайт с базой данных общеобразовательных учреждений Москвы

Актуальность: в Москве есть множество школ. Некоторые из них участвуют в московских проектах, углубленно изучают различные предметы, сотрудничают с разными вузами. Официальные сайты школ содержат одинаковую по структуре информацию. Но поиск по таким сайтам с целью выбора учебного заведения займет большое количество времени и крайне неудобен. Наш проект поможет структурировать все эти данные по категориям, отсортировать их по рейтингу и баллам ЕГЭ. Это упростит школьникам и их родителям выбор учебного заведения.

Цель работы: создать сайт с базой данных общеобразовательных учреждений города Москвы

Задачи: изучить информацию о московских школах и собрать ее в единую базу данных; создать сайт

**Сравнение фреймворков Django и Flask**

*Python* является одним из наиболее популярных языков программирования для веб-разработки. Это объясняется рядом причин:

* Python – интерпретируемый язык, поэтому он не зависит от платформы. К тому же, программы, написанные на интерпретируемых языках более гибкие, они используют динамическую типизацию (переменная связывается с типом не при объявлении переменной, а в момент присваивания значения; благодаря этому в различных участках программы переменная может принимать значения разных типов) и имеют небольшой размер.
* Python – объектно-ориентированный язык программирования. Это упрощает обновление кода, делает программу читабельнее, позволяет эффективнее работать в команде над одним проектом
* Простотой синтаксис, большое количество библиотек для разных задач (например, для подключения базы данных или выполнения сложных расчетов)
* Совместимость с различными фреймворками

Для создания сайтов часто используют фреймворки.

Фреймворк (англ. framework – «каркас») – платформа для упрощения создания и работы над программным проектом. Формирует структуру программной системы. Во фреймворке реализованы классы, определены переменные и константы, созданы готовые решения различных проблем (обработка запросов, работа с базами данных, формами).

Существует множество различных фреймворков на Python, но остановимся на двух наиболее популярных: *Django* и *Flask*.

**Django** - высокоуровневый веб-фреймворк с открытым исходным кодом. Используется популярными веб-приложениями и сайтами, такими как Mozilla, Instagram, Dropbox, Pinterest, Spotify.

**Flask** – микрофреймворк (упрощенный фреймворк) с минималистичными функциями. Следующие сайты компаний написаны на *Flask*: Red Hat, Reddit, Netflix, Airbnb, RackSpace, Lyft.

**Возможности фреймворков**

**Формы**

Форма – раздел HTML-документа, в котором хранится введенная пользователем информация для последующей обработки.

*Django* предлагает инструменты для работы с формами. Поддерживается создание форм, проверка данных и CSRF-токенов (уникальные значения веб-приложений, предотвращающие подделку межсайтовых запросов). Формы отображаются при помощи встроенных шаблонов.

*Flask* не имеет встроенных инструментов для работы с формами. Для использования форм существует библиотека Flask-WTF.

**Обработка http-запросов и маршрутизация**

Http-запрос – сообщение, отправляемое клиентом (пользователем) на сервер, для вызова определенных действий на сайте. Обработка запросов – ключевой момент в любом веб-приложении.

В *Django* обработка запросов происходит отдельно от определения маршрутов (адресов страниц). Когда пользователь запрашивает страницу, *Django* проходит по всем шаблонам URL, сопоставляет их с заданным адресом и вызывает представление страницы, находящееся в другом файле.

Объект request, в котором хранится запрос, подается в *Django* каждый раз в качестве аргумента. В этом случае контекст запроса отделен от самого веб-приложения, но передача параметра каждый раз, когда нужно получить доступ к запросу, может быть неудобной.

Во *Flask* маршруты объявлены с помощью декораторов в одном месте.

Объект request объявлен как локальная переменная потока и доступ к нему можно получить просто импортировав из библиотеки. Значит, передавать запрос не нужно, что значительно облегчает работу.

**Шаблоны**

Шаблон – инструмент, упрощающий работу с разметкой сайта и связывающий ее с данными.

В *Flask* используется шаблонизатор Jinja2. *Django* имеет собственный шаблонизатор Django Template Engine и допускает использование других языков шаблонов.

Оба фреймворка также предоставляют возможность использовать статические файлы в шаблонах.

**ORM**

ORM – инструмент, связывающий базу данных с объектно-ориентированным языком. Т.е. это технология, преобразующая данные в таблице в классы, а классы – в данные.

В *Django* есть ORM, подходящий для большинства случаев, кроме некоторых сложных запросов. В этом случае можно использовать SQLAlchemy. Поддерживается миграция моделей (добавление изменений моделей в базу данных).

Во *Flask* ORM нет, но можно использовать SQLAlchemy. К тому же, *Flask* дает свободу в управлении базами данных, в отличии от *Django*, где многое сделано автоматически.

**Безопасность и производительность**

*Django* поддерживает защиту от XSS, CSRF, SQL-инъекций и кликджекинга.

*Flask* поддерживает защиту от XSS.

*Flask* ввиду своей легкости и малого количества дополнительных функций быстрее *Django*, но на практике эти различия не сильно заметны.

**Вывод:** *Django* имеет больший встроенный функционал, нежели *Flask*. Поэтому в нашем проекте будет использован фреймворк Django.

**Описание базы данных**

Таблица schools

Содержит данные о школах. Описание полей таблицы:

* id – идентификатор школы
* short\_name – короткое название школы
* full\_name – полное название школы
* educational\_services – предоставляемые формы обучения
* institution\_addresses – строка идентификаторов адресов (из таблицы addresses)
* legal\_organization – организационно-правовая норма
* chief\_name – ФИО руководителя школы
* public\_phone – номер телефона школы
* email – почта школы
* web\_site – сайт школы

Таблица address

Содержит информацию о зданиях школ. Описание полей таблицы:

* id – идентификатор здания
* adm\_area – административный округ
* district – район
* address – адрес
* public\_phone – номер телефона
* available\_k – доступность объекта для инвалидов-колясочников
* available\_o – доступность объекта для инвалидов-опорников
* available\_z – доступность объекта для инвалидов по зрению
* available\_s – доступность объекта для инвалидов по слуху