Flask vs Django: Давайте выберем ваш следующий фреймворк Python

Описание

Вы взвешиваете варианты для своего нового сайта или приложения и пытаетесь выбрать между Flask и Django? Flask и Django – два чрезвычайно мощных и популярных Python-фреймворка, но даже профессиональным веб-разработчикам бывает трудно определить, какой из них лучше подходит для их конкретных требований. Это руководство по сравнению Flask-vs-Django поможет вам принять решение, рассмотрев основные различия между ними.

Что такое Flask?

Flask – это микрофреймворк, который не требует внешних библиотек для реализации своих функций. Он был разработан в 2011 году Армином Ронахером, который пришел к нему во время разработки решения, объединяющего Werkzeug (серверный фреймворк) и Jinja2 (библиотека шаблонов). Flask поставляется с большим количеством инструментов, технологий и библиотек, необходимых для разработки веб-приложений. Flask предлагает валидацию форм и другие расширения для объектно-реляционного отображения, открытой аутентификации, загрузки файлов и др. Среди известных компаний, использующих Flask, – Samsung, Netflix, Lyft, Reddit и Zillow.

Что такое Django?

Django – это бесплатная веб-платформа с открытым исходным кодом, впервые выпущенная в 2005 году Адрианом Холовати и Саймоном Уиллисоном. Django – это сложный фреймворк на основе Python с полным набором настроек разработки, таких как макеты шаблонов, запросы и устранение неполадок, cookies, валидация форм, модульные тесты, настройки таблиц и другие функциональные возможности, которые разработчики используют для создания динамических веб-приложений.

Django следует архитектурному шаблону Model-View-Template (MVT), который помогает разработчикам эффективно выполнять рутинные или сложные задачи с минимальным вмешательством протоколов, управления и систем при создании высокоинтенсивных приложений и веб-сайтов, управляемых базами данных. Низкая связанность, подключаемость, возможность повторного использования и простой код – вот некоторые из основных принципов Django. Неудивительно, что такие мировые компании, как Instagram, National Geographic, Spotify, Mozilla, Quora и Discovery Channel используют Django для своих проектов.

Ключевые особенности Flask

Особенности любого фреймворка могут указать на приложения, для которых он наиболее полезен. Flask обладает некоторыми передовыми возможностями для создания и оптимизации веб-приложений. Вот несколько основных особенностей Flask, которые вы должны знать.

Легкий вес

Flask не использует внешние библиотеки в процессе разработки. Это делает его легким. Он основан на Python в меньшем масштабе без какого-либо набора специфических инструментов. У него также нет слоя базы данных или механизмов для утверждения формы и использования расширений.

Система модульного тестирования

Встроенная система модульного тестирования Flask помогает ускорить отладку и стимулирует надежную разработку.

Поддержка расширений

Flask поддерживает внедрение сторонних расширений, которые добавляют функциональность вашему приложению. Среди множества доступных расширений Flask – Flask SQLAlchemy, добавляющий поддержку работы с базами данных, и WTForms, поддерживающий работу с веб-формами. Среди других расширений – те, которые отвечают за безопасность. Другими словами, там, где Flask не дотягивает, он компенсирует это поддержкой расширений.

Совместимость с WSGI

Werkzeug, на котором основан Flask, представляет собой библиотеку утилит Web Server Gateway Interface (WSGI). Это делает приложения Flask совместимыми с любым веб-сервером, совместимым с WSGI. Существует множество веб-серверов WSGI, но в производственной среде требуются надежные серверы, такие как Nginx или Apache. К счастью, эти популярные серверы можно настроить на поддержку WSGI-функциональности Flask.

Ключевые особенности Django

Фреймворк Django предлагает целый ряд интересных возможностей, идеально подходящих для компаний, которые нанимают разработчиков, стремящихся сдавать проекты в срок. Давайте рассмотрим некоторые из них.

Надежная безопасность

Подход Django к безопасности помогает пользователям бороться с такими угрозами, как SQL-инъекции, подделка межсайтовых запросов (CSRF), межсайтовый скриптинг (XSS), clickjacking и др. Django автоматически устраняет уязвимости, которые бэкенд-разработчик может непреднамеренно упустить при создании веб-приложения. Кроме того, вы можете безопасно работать с учетными записями пользователей и паролями с помощью функции активной аутентификации пользователей Django.

Адаптация к проектам с высокой нагрузкой

Django может работать с большим объемом трафика, например, со сложными социальными сетями и системами управления контентом. Поскольку оптимизация времени загрузки и производительности может быть сложной задачей при работе с такими системами баз данных, как MongoDB и Redis, Django может быстро обрабатывать все данные, сокращая время загрузки. Кроме того, Django гарантирует безошибочные результаты независимо от сложности кодирования.

Философия “Не повторяйся”

Создание веб-приложения подразумевает большое количество повторений, например, использование бизнес-логики и HTML-шаблонов. Подобная избыточность приводит к логическим ошибкам и проблемам с обслуживанием. Следуя принципу Don’t Repeat Yourself (DRY), Django гарантирует, что информация не будет дублироваться. Логика, поддерживающая группу данных, должна быть четкой и надежной. Поэтому вы можете сосредоточиться на написании оригинального кода, повторно используя существующий. Это помогает ускорить разработку веб-приложений.

SEO-дружественный

Dajngo облегчает отображение сайта и имеет возможность генерировать динамические ссылки и URL на основе целевых ключевых слов. Это поможет вам улучшить рейтинг вашего сайта в поисковых системах.

Разнообразие пакетов

Django поддерживает множество пакетов для разработки полнофункционального приложения. Он не позволяет вам полагаться на сторонние решения, предоставляя наборы практически для всего. Например, вы можете использовать фреймворк Django REST для создания API или пакет Django CORS Headers для создания кросс-оригинальных запросов. Django Debug Toolbar помогает в устранении неполадок в приложениях, а Django Allauth обеспечивает социальную и локальную аутентификацию.

Плюсы и минусы использования Flask

Хотя Flask обладает широкими функциональными возможностями и известен простотой использования, у него есть несколько недостатков, о которых стоит упомянуть. Ниже перечислены преимущества и недостатки использования Flask.

Преимущества

- Гибкость с полным доступом к управлению
- Отлично подходит для небольших проектов
- Масштабируемый
- Легко интегрировать базы данных
- Адаптируется к новым технологиям
- Легко и быстро создавать прототипы
- Расширяемый
- Небольшая кодовая база с простыми функциями
- Простая маршрутизация URL через Werkzueg
- Позволяет экспериментировать с архитектурой и библиотеками

Недостатки

- Нет поддержки многостраничных приложений

- Отсутствие функций безопасности
- Нет встроенного сайта администратора
- Сложное обслуживание
- Отсутствие объектно-реляционного отображения без расширений

Плюсы и минусы использования Django

Как вы знаете, ничто интересное никогда не бывает полностью односторонним, и Django не является исключением. Как и Flask, Django имеет множество плюсов и несколько препятствий.

Преимущества

- Полнофункциональный фреймворк
- Концепция, основанная на батарейках
- Высокая масштабируемость
- Исчерпывающая документация
- SEO-дружественный
- Кросс-платформенная совместимость
- Быстрая разработка и быстрая обработка
- Высокий уровень безопасности
- Отличное управление CDN

Недостатки

- Не подходит для небольших проектов
- Отсутствие конвенций
- Монолитная рабочая среда
- Ограниченное количество компонентов и вариантов дизайна
- Трудно освоить и имеет крутую кривую обучения
- Не позволяет одновременно обрабатывать несколько запросов

Flask vs Django: Сравнение с ног на голову

Теперь мы подошли к битве фреймворков. Давайте сравним Flask с Django и выясним, какое решение лучше для вас. Готовы? Приступаем.

Экономическая эффективность

И Flask, и Django – это бесплатные Python-фреймворки с открытым исходным кодом. С точки зрения бизнеса, использование Flask может быть дороже, чем Django, поскольку он имеет меньше встроенных функций и требует большего обслуживания, особенно с точки зрения времени разработки и безопасности. В любом случае, стоимость использования того или иного фреймворка будет определяться вашими конкретными требованиями. Если вам нужны дополнительные встроенные функции или вы предпочитаете более комплексный фреймворк, Django может стать лучшим выбором для вас.

Победитель: Django

Время разработки

Flask и Django призваны помочь разработчикам работать умнее и экономить много времени. Тем не менее, один из них отстает от другого. Несмотря на то, что Flask позволяет сократить общее время разработки, для большинства сайтов он обычно использует подход “сборки с нуля”, что делает разработку полноценного веб-приложения довольно трудоемкой. Если вы продвинутый пользователь, то сроки разработки могут быть замедлены из-за ограниченных возможностей встроенной разработки. Flask также использует шаблонизатор Jinja2, который прост в использовании, но отнимает время при разработке. В отличие от него, Django способствует быстрой разработке, особенно если у вас сжатые сроки, благодаря различным встроенным функциям и минимальному использованию кода.

Например, разработчики могут создать минимально жизнеспособный продукт (MVP) быстрее с Django, чем с Flask. Именно поэтому Django идеально подходит для запуска крупных веб-сайтов. Хотя Flask также позволяет быстро создавать MVP, есть очевидная разница, когда речь идет о фронтенде. С Django быстрее добавить фронтенд и получить обратную связь. Встроенный движок шаблонов Django ускоряет разработку по сравнению с движком Flask. Более того, Django оснащен множеством стандартных библиотек, которые помогают разработчикам в создании широко используемых функций и решении общих проблем разработки.

Победитель: Django

Простота использования/обучение

Если вы новичок, но уже знакомы с Python, вам будет легко разобраться с минималистичной структурой Flask. При реализации моделей машинного обучения Flask гораздо удобнее для новичков, чем Django. Кроме того, он требует меньше строк кода. Django – это более сложное веб-решение, которое требует обширных знаний для создания сложного приложения, таких как команды терминала, продвинутые навыки программирования на Python, переменные, циклы и функции. Богатая документация по Flask особенно полезна для новых пользователей, которые хотят начать работу с этим фреймворком.

Она имеет менее крутую кривую обучения, чем Django, предполагает тщательное объяснение и охватывает все, начиная с установки и заканчивая развертыванием. Django, однако, имеет гораздо больше документации, чем Flask, потому что это полнофункциональный фреймворк, а не минималистский. Начните с Flask, независимо от того, является ли изучение Django или Flask вашей основной целью. Он отлично подходит для понимания принципов веб-разработки, эталонов и основных компонентов веб-фреймворков.

Победитель: Flask

Возможности трудоустройства

Разработчики Python пользуются большим спросом в связи с растущей потребностью в Python в основных отраслях промышленности. Знание Flask или Django, или и того, и другого, повысит ваши карьерные перспективы. По данным talent.com, средний разработчик Python Flask в США зарабатывает $126 750 в год или $60,94 в час. Наиболее опытные программисты зарабатывают до $171 000 в год, в то время как начальный уровень начинается от $107 250. Средняя зарплата разработчика Python Django составляет $120 000 в год или $57,69 в час. Зарплата начального уровня начинается от $100 000 в год, а самые опытные сотрудники зарабатывают до $152 655.

Победитель: Flask

Особенности

За исключением нескольких встроенных возможностей, Flask в основном полагается

на расширения для своей функциональности. Будь то интеграция баз данных или реализация дополнительных мер безопасности – практически для всего, что вам нужно, найдется расширение. В отличие от Flask, Django поставляется в комплекте с полезными функциями и несколькими доступными расширениями. С точки зрения разнообразия функций, Flask выходит вперед. Но всегда полезно иметь большинство необходимых функций, встроенных в фреймворк, и в этом отношении Django лидирует.

База данных

Flask не ставит ограничений на хранение данных, и поэтому существует широкий спектр доступных библиотек и расширений специально для этого. Django, с другой стороны, имеет свой собственный простой, но мощный ORM, который поддерживает ряд реляционных баз данных, таких как SQLite, PostgreSQL, MariaDB и Oracle. ORM предлагает поддержку для создания и управления миграциями баз данных.

Авторизация/аутентификация

Большинство веб-приложений требуют авторизации и аутентификации. Flask обеспечивает поддержку только сессий на основе cookies, но вы можете получить расширения для управления учетными записями, авторизации и аутентификации. В отличие от Flask, Django поставляется с уже встроенными функциями авторизации и аутентификации. Он также поставляется с управлением учетными записями и поддержкой сессий.

Администратор

Flask не имеет собственной панели администратора, которая представляет собой веб-приложение, предоставляющее пользовательский интерфейс для управления данными на основе ваших моделей. Вместо этого доступно расширение под названием Flask-Admin, которое предлагает все эти и другие функции. Django здесь снова сияет, поскольку он поставляется с функциональной панелью администратора, которая позволяет вам быстро выполнять CRUD-операции над вашей моделью по мере создания приложения, не требуя никакого дополнительного кода.

Маршрутизация и представления

И Flask, и Django позволяют сопоставлять URL-адреса с представлениями. Они также поддерживают представления на основе функций и классов. Для Flask расширение Werkzeug реализует маршрутизацию URL. Оно также обрабатывает запросы и ответы приложения. Django маршрутизирует шаблон URL к представлению, ссылаясь на объект запроса, который содержит информацию о запросе для вызова требуемого представления.

Формы

Чтобы работать с формами, Flask интегрирует расширение Flask-WTF с WTForms. Впоследствии WTForms-Alchemy использует модели SQLAlchemy для создания форм. По умолчанию установка Django предоставляет формы; которые модели данных создают с помощью ModelForms, который обрабатывает ввод данных и валидацию клиентов и серверов. Эти формы также решают вопросы безопасности, такие как CSRF, SQL инъекции и XSS.

Движок шаблонов

Flask использует Jinja2 в качестве движка шаблонов, в то время как Django использует свой собственный движок шаблонов для добавления данных на существующую HTML-страницу вашего приложения. Однако Django также может использовать Jinja2. Оба шаблонизатора имеют сходство в синтаксисе и возможностях.

Производительность и скорость

Будучи основанным на Werkzeug, Flask стремится быть как можно более простым, но при этом расширяемым, чтобы обеспечить легкое будущее расширение проекта. Именно эта простота, гибкость и легкость позволяют разработчикам создавать небольшие приложения в более короткие сроки. Для тех, кто хочет быстро и без усилий создавать более сложные проекты, подойдет Django. Django способен обеспечить быстрый результат и плавно провести приложения от концепции до завершения. Пока что гонка за производительностью остается жесткой. Тем не менее, Flask вырывается вперед благодаря своим небольшим требованиям, поскольку вы можете построить целый проект, используя вдвое меньшее количество строк кода и меньшее потребление ресурсов с Flask, чем с Django.

Победитель: Flask

Масштабируемость

Flask является масштабируемым, поскольку может обрабатывать большое количество запросов каждый день. Микрофреймворк модулирует весь код и позволяет разработчикам работать над независимыми частями по мере роста кодовой базы. Обычно приложения Flask разрабатываются локально с использованием WSGI-сервера Werkzeug, но развертываются публично с использованием WSGI-сервера, такого как Gunicorn или uWSGI, с надежным сервером, таким как Nginx, в качестве публичного прокси.

Django – это компонентный фреймворк, который обеспечивает полную масштабируемость. Каждый слой разрабатываемого приложения является независимым, что означает, что вы можете масштабировать приложение на любом уровне. Django также использует балансировку нагрузки и кластеризацию для запуска приложения на различных серверах. Другими словами, вы можете масштабировать свое веб-приложение без ущерба для общей производительности или времени загрузки. Django может быть лучшим выбором, если вам нужно разработать высокомасштабируемое веб-приложение.

Победитель: Django

Безопасность

Flask поставляется с несколькими встроенными функциями безопасности, такими как защита от CSRF. Но есть несколько расширений безопасности, которые вы можете добавить. Несмотря на то, что Flask имеет меньше требований к кодированию, чем Django, что делает его менее восприимчивым к киберугрозам, он все еще сильно зависит от сторонних расширений, что означает, что безопасность приложения сильна лишь настолько, насколько силен его самый слабый плагин. Это повышает ответственность команды разработчиков за поддержание безопасности в актуальном состоянии путем оценки и отслеживания сторонних библиотек и расширений.

С другой стороны, большая кодовая база Django открывает его для большего количества вредоносных действий. Тем не менее, Django по умолчанию имеет высокий уровень безопасности и включает в себя множество интегрированных

средств защиты, таких как скрипты, хеширование паролей пользователей, CSRF-токены, а также модули аутентификации и авторизации. Это позволяет программистам не допускать распространенных ошибок безопасности и выполнять контрольный список безопасности перед развертыванием. Кроме того, команда разработчиков Django оперативно и проактивно сообщает о любых выявленных недостатках безопасности. Таким образом, Django легче обеспечить безопасность на начальном этапе и поддерживать ее на протяжении всей жизни вашего приложения.

Победитель: Django

Поддержка сообщества

Для разработчика очень важно иметь солидное сообщество пользователей выбранного вами фреймворка. Эти сообщества могут оказать вам большую помощь, отвечая на любые ваши вопросы. Они также могут стимулировать разработку новых приложений, связанных с фреймворком. Flask и Django имеют большие, знающие и увлеченные сообщества. У Django группа больше, поскольку он существует на 5 лет дольше, чем Flask. Но, несмотря на меньший размер, сообщество Flask довольно активно.

Победитель: Django

Flask против Django: Подводя итоги

Итак, что же значит для вас это сравнение Flask vs Django? Чтобы помочь вам принять решение, эта таблица подводит итог.

	Flask	**Django**
Экономическая эффективность	Бесплатный с открытым исходным кодом, будет стоить дороже за обслуживание и безопасность.	Бесплатный с открытым исходным кодом, подходит для малобюджетных проектов.

Время разработки	Процесс разработки занимает больше времени, чем Django.	Отлично подходит для быстрой разработки и проектов с жесткими сроками.
Простота использования/обучение	Простой в освоении и понятный. Лучше всего подходит для начинающих.	Более сложный и требует углубленного знания Python. Лучше всего подходит для опытных пользователей.
Возможности трудоустройства	Пользуется спросом, средняя зарплата немного выше, чем у Django.	Более популярны, чем Flask. Пользуется спросом, средняя зарплата чуть ниже, чем у Flask.
Характеристики	Flask в основном полагается на сторонние расширения для дополнительной функциональности.	Django поставляется с широким набором встроенных функций.
Производительность	Flask является легким и более простым, что позволяет создавать проекты в более короткие сроки и с меньшим количеством кодирования.	Больше подходит для быстрого и легкого создания больших проектов.
Масштабируемость	Ограниченная масштабируемость. Идеально подходит для небольших проектов.	Высокая масштабируемость. Рекомендуется для крупных, сложных проектов.

Безопасность	Короткая кодовая база делает его более защищенным от угроз, но ограниченным в интегрированных функциях безопасности.	Длинная кодовая база, что делает ее более подверженной ошибкам и атакам, но по умолчанию включает множество функций безопасности.
Скорость	Легкий, более гибкий и быстрый, чем Django.	Полнофункциональное решение. Менее гибкое, чем Flask.

Flask vs Django: Что выбрать?

Если сравнивать эти два фреймворка по различным критериям, то ни Flask, ни Django не могут быть признаны лучшими во всех отношениях. Flask, как правило, проще, в то время как Django предпочтительнее для крупномасштабных проектов с быстрой разработкой и предлагает больше встроенных функций. Оба фреймворка отлично подходят для создания веб-приложений, и выбор лучшего для вас будет зависеть от размера проекта, потребностей и существующих требований.

Заключение

В этой битве между Flask и Django, явным победителем для вас будет тот, который действительно отвечает вашим потребностям. Некоторые программисты даже используют оба фреймворка! Если вы не знаете, с чего начать, мы рекомендуем начать с Flask. Flask практичен, потому что с ним легче начать работу, не чувствуя себя перегруженным. Django – это фантастическое веб-решение для создания крутых приложений, но менее подходящее для новичка. Вы профессиональный разработчик? Благодаря широкому набору функций Django вам или вашей команде придется принимать меньше решений. Это позволит вам быстрее разработать свой проект.

Дата Создания
13.06.2023