**Тема занятия № 56: Модуль 33. Публикация web-сайта**

1. Подготовка веб-сайта к публикациИ

Разработка веб-сайта — процесс долгий и по-своему увлекательный. Но рано или поздно он подходит к концу. Сайт написан, проверен, возможно, наполнен какими-либо рабочими данными — и теперь его предстоит опубликовать в Сети.Перед публикацией веб-сайта предварительно нужно выполнить некоторые подготовительные работы.

**Написание шаблонов веб-страниц с сообщениями об ошибках**

Эти шаблоны будут применяться для генерирования страниц с сообщениями об ошибках при работе в эксплуатационном режиме:

□ 404.html — шаблон страницы с сообщением об ошибке с кодом статуса 404 (запрошенная страница отсутствует). Обычно такая страница содержит текст вида ’’Страница не найдена” и гиперссылку на главную страницу сайта.

Служебный контроллер, выводящий эту страницу, создает в контексте шаблона две переменные:

• request path — путь, выделенный из интернет-адреса, который был получен в составе запроса;

• exception — строка с текстом сообщения об отсутствии запрошенной страницы.

Помимо этого, шаблон 404.html имеет доступ ко всем переменным, добавленным в контекст шаблона зарегистрированными обработчиками контекста;

□ 500.html — шаблон страницы с сообщением об ошибке 500 (внутренняя ошибка сервера). Обычно такая страница содержит текст ’’Внутренняя ошибка сервера” и предложение попытаться обновить страницу спустя некоторое время.

Служебный контроллер, выводящий эту страницу, передает шаблонизатору пустой контекст шаблона без каких-либо переменных;

□ 403.html — шаблон страницы с сообщением об ошибке с кодом статуса 403 (доступ к запрошенной странице запрещен. В частности, эта ошибка возникает при обращении гостя к странице, к которой имеют доступ только зарегистрированные пользователи). Обычно такая страница содержит текст вида ’’Страница недоступна”, предложение выполнить процедуру входа на сайт и гиперссылки на страницу входа и главную страницу сайта.

Служебный контроллер, выводящий эту страницу, создает в контексте шаблона переменную exception, в которой хранится строка с текстом сообщения о недоступности запрошенной страницы;

□ 400.html — шаблон страницы с сообщением об ошибке 400 (клиентский запрос некорректно сформирован). Обычно такая страница содержит текст вида ’’Некорректный запрос”.

Служебный контроллер, выводящий эту страницу, передает шаблонизатору пустой контекст шаблона без каких-либо переменных.

Все эти шаблоны помещаются непосредственно в папку templates пакета приложения или в одну из папок, чей путь указан в параметре dirs настроек текущего шаблонизатора.

Внимание!

Стандартное приложение django.contrib.admin (административный веб-сайт) содержит в своем составе шаблоны 404.html и 500.html. Чтобы наш сайт использовал созданные нами шаблоны, а не принадлежащие этому приложению, мы можем прибегнуть к переопределению шаблонов.

Если какой-либо из упомянутых шаблонов отсутствует, то Django отправит клиенту пустой ответ с кодом статуса, соответствующим возникшей ошибке. В результате веб-обозреватель выведет встроенную в него страницу с описанием ошибки.

**Указание настроек эксплуатационного режима**

Следующий шаг — указание настроек проекта, которые будут действовать в эксплуатационном режиме:

□ debug — этому параметру, указывающему режим работы сайта, нужно присвоить значение False, задающее эксплуатационный режим;

□ allowed hosts — очень важный параметр, указывающий перечень хостов, с которых Django будет принимать отправленные клиентами данные. Если сайт получит данные с хоста, отсутствующего в этом перечне, то он возбудит исключение suspiciousoperation ИЗ модуля django.core.exceptions, ЧТО приведет к выдаче страницы с сообщением об ошибке 400 (некорректно сформированный запрос).

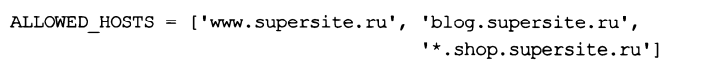
Перечень должен быть представлен в виде списка, элементами которого должны быть строки, указывающие разрешенные хосты. Эти строки могут быть:

• доменными именами;

• IP-адресами в формате ipv4 или ipv6;

• шаблонами доменных имен. В таких шаблонах можно применять специальный символ \* (звездочка), который обозначает произвольное количество любых знаков.

Пример:

****

Здесь в список разрешенных занесены хосты [www.supersite.ru,blog.supersite.ru](http://www.supersite.ru,blog.supersite.ru) и все хосты вида Произвольные ci£wa76i>.shop.supersite.ru (technics.

Shop.supersite.ru, furniture.shop.supersite.ru и т. П.).

Значения этого параметра по умолчанию:

• в отладочном режиме — список [' localhost', \*127.0.0.1’, ' [:: 1 ]' ] (т. е. локальный хост, представленный доменным именем и IP-адресами стандартов ipv4 и ipv6);

• в эксплуатационном режиме— ’’пустой” список. Поэтому перед запуском сайта в эксплуатацию этот параметр обязательно следует задать, иначе сайт работать не будет;

□ databases— необходимо указать параметры базы данных, которая будет использоваться сайтом в эксплуатационном режиме. Поскольку сайт, как правило, публикуется на компьютере, отличном от того, на котором он разрабатывался, параметры базы данных там, скорее всего, будут иными;

□ static root — возможно, понадобится изменить путь к папке, в которой хранятся статические файлы сайта;

□ media root — возможно, понадобится изменить путь к папке, в которой хранятся выгруженные файлы;

□ настройки подсистемы отправки электронных писем — следует изменить их на те, что будут использоваться сайтом в режиме эксплуатации;

□ caches — следует указать параметры подсистемы кэширования уровня сервера. Которая будет использоваться при эксплуатации сайта;

□ logging — понадобится задать окончательные настройки для подсистемы диагностики;

□ admins — здесь нужно задать перечень адресов электронной почты, принадлежащих администраторам;

□ managers — и адреса редакторов

□ secret key— не помешает удостовериться, что секретный ключ, задаваемый

Этим параметром, кроме данного сайта, не применяется более нигде.

**Удаление ненужных данных**

Часть данных, генерируемых работающим Django-сайтом, либо являются временными (устаревшие CAPTCHA, сессии и пр.), либо впоследствии могут быть созданы повторно (например, миниатюры). Перед публикацией сайта такие данные лучше удалить для уменьшения его объема, особенно если сайт будет переноситься на целевой компьютер по сети.

Вот список команд утилиты manage.py, служащих для удаления ненужных и временных данных:

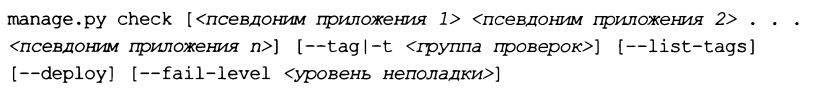
□ captcha clean —удаляет просроченные CAPTCHA из хранилища;

□ thumbnaii cieanup — удаляет файлы с миниатюрами: все или сгенерированные в течение указанного количества дней;

□ clearsessions —удаляет устаревшие сессии.

**Окончательная проверка веб-сайта**

По окончании подготовительных работ неплохо выполнить проверку, всё ли мы сделали как надо. Провести ее нам поможет команда check утилиты manage.py:

****

По умолчанию выполняется проверка всех приложений, имеющихся в проекте. Но можно задать проверку только приложений с заданными псевдонимами, перечислив их через пробел. Пример:



Поддерживаемые командные ключи:

□ —tag или -t — указывает группу проверок, которые необходимо провести. Доступны следующие группьс

• admin— всё связанное с административным веб-сайтом Django (редакторы, обычные и встроенные, действия и др.);

• caches — настройки подсистемы кэширования;

• compatibility— потенциальные проблемы при переходе на следующую версию Django;

• database —настройки используемых баз данных;

• models — объявления моделей, диспетчеров записей и наборов записей;

• security —настройки безопасности;

• signals —объявления сигналов и привязка к ним обработчиков;

• staticfiles — настройки подсистемы, обрабатывающей статические файлы;

• templates —настройки шаблонизаторов;

• uris —списки маршрутов.

Пример:



Можно указать произвольное количество групп проверок — каждую в отдельном ключе:



Если ключ не задан, выполняется проверка по всем группам, за исключением Database;

□ --list-tags —выводит список всех поддерживаемых групп проверок;

□ —deploy— выполняет дополнительные проверки, актуальные только для сайтов, предназначенных для публикации;

□ --fail-level — указывает уровень найденной неполадки, после которого проверка прекращается. Доступны уровни неполадок debug, info, warning, error и Critical. Если ключ не указан, то проверка завершается по выявлении неполадки уровня ERROR.

**Настройка веб-сайта для работы по протоколу HTTPS**

Любой Django-сайт может работать по защищенному протоколу HTTPS без какоголибо дополнительного конфигурирования. Однако настоятельно рекомендуется указать в модуле settings.py пакета конфигурации следующие настройки, затрагивающие безопасность сайта и защиту от сетевых атак:

□ secure ssl redirect — если True, то сайт при попытке доступа к нему по незащищенному протоколу HTTP станет выполнять перенаправление по тому же интернет-адресу, но с использованием протокола HTTPS (по умолчанию — False).

Если сайт должен работать исключительно по протоколу HTTPS, то необходимо установить этот параметр в True;

□ secure redirect exempt — задает перечень шаблонных путей, которые должны быть доступны по протоколу HTTP (соответственно, сайт не будет выполнять перенаправление с применением HTTPS при запросах по этим путям). Перечень задается в виде списка, каждый элемент которого указывает отдельный путь в виде регулярного выражения. Шаблонные пути не должны содержать начального слеша. Пример:



Принимается во внимание, только если параметру secure\_ssl\_redirect задано значение True.

Значение по умолчанию — ’’пустой” список;

□ secure ssl host — задает интернет-адрес хоста, на который сайт будет выполнять перенаправление с использованием HTTPS. Если None, перенаправление будет выполняться на изначальный хост.

Принимается во внимание, только если параметру secure ssl redirect задано значение True.

Значение по умолчанию — None;

□ secure hsts seconds — если задано значение, отличное от о, сайт будет вставлять в каждый отправляемый ответ заголовок формата:

Strict-Transport-Security: тах-аде=<время> одним фактом своего наличия в ответе такой заголовок сообщает веб-обозревателю, что сайт доступен исключительно по протоколу HTTPS, и при попытке получить к нему доступ по HTTP веб-обозревателю следует самостоятельно выполнить перенаправление с применением HTTPS.

Языковая конструкция max-age, содержащаяся в значении этого заголовка, задает время в секундах, в течение которого веб-обозреватель должен ’’помнить”, что сайт доступен исключительно по протоколу HTTPS. В качестве этого времени указывается значение параметра secure hsts seconds.

Значение по умолчанию: о (упомянутый ранее заголовок в ответы не вставляется).

Этот параметр нужно указывать, если сайт должен работать исключительно по протоколу HTTPS, чтобы усилить защиту от сетевых атак. Сначала в целях проверки работоспособности имеет смысл задать относительно небольшое значение, например 3600 (1 час), а потом, удостоверившись, что сайт полностью работоспособен, увеличить его, скажем, до 31536000 (1 года);

□ secure\_hsts\_include\_subdomains — если True, то сайт будет добавлять в значение

Заголовка Strict-Transport-Security конструкцию includesubdomains:

Strict-Transport-Security: тах-аде=<время> includesubdomains она предписывает веб-обозревателям блокировать доступ по HTTP также и к поддоменам.

Значение ПО умолчанию — False (не добавлять конструкцию includesubdomains).

Этот параметр принимается во внимание, если параметру secure hsts seconds задано значение, отличное от о;

□ securejists\_PRELOAD — если True, то сайт будет добавлять в значение заголовка Strict-Transport-Security конструкцию preload:



Она предписывает веб-обозревателю уведомлять веб-службы Google, что текущий сайт либо находится в статическом списке безопасных сайтов, поддерживаемом упомянутой ранее корпорацией, либо является кандидатом на включение туда.

Значение по умолчанию — False (не добавлять конструкцию preload).

Параметрам secure\_hsts\_include\_subdomains и secure\_hsts\_preload можно одновременно дать значение True. В таком случае в ответы будет добавляться заголовок:



□ SECURE\_CONTENT\_TYPE\_NOSNIFF-- если True, ТО СЯЙТ будет ВСТЯВЛЯТЬ В КЯЖДЫЙ отправляемый ответ заголовок:



Он запрещает веб-обозревателю определять тип загруженного файла по его содержимому, а, наоборот, предписывает всегда использовать тип, заданный в заголовке Content-Type полученного ответа.

Значение по умолчанию, начиная с Django 3.0 — True (в более старых версиях — False, указывающее не вставлять в ответы такой заголовок).

Установка значения True позволяет предотвратить некоторые типы сетевых атак, связанных с загрузкой клиентом небезопасных файлов (например, веб-страниц с вредоносными веб-сценариями), замаскированных под безопасные (например, изображения или архивы);

□ secure browser xss filter — если True, то сайт будет вставлять в каждый ответ заголовок:



Он указывает веб-обозревателю блокировать любой пришедший с сервера ответ, содержащий веб-сценарии, которые могут оказаться вредоносными (т. Е. Использоваться для атакass' — Cross-Site Scripting, межсайтовый скриптинг).

Значение по умолчанию — False (не вставлять такой заголовок);

□ secure referrer policy (начиная с Django 3.0)— если задано значение, отличное от None, то сайт будет помещать в каждый ответ заголовок Referrer-Policy, указывающий веб-обозревателю, вставлять ли при переходах на другую страницу в запросы заголовок Referrer с интернет-адресом предыдущей страницы.

Доступны следующие значения параметра:

• "no-referrer” — веб-обозреватель не должен вставлять в запросы заголовок Referrer;

• "no-referrer-when-downgrade" — вставлять этот заголовок только в том случае, если выполняется переход на сайт, работающий через HTTPS;

• "origin" — вставлять заголовок, но отправлять в нем интернет-адрес хоста, а не страницы;

• "origin-when-cross-origin" — вставлять заголовок, но отправлять в нем интернет-адрес страницы только при переходе на страницу того же сайта, в противном случае отправлять интернет-адрес хоста;

• "same-origin"— вставлять заголовок с интернет-адресом страницы только при переходе на страницу того же сайта, в противном случае не вставлять этот заголовок;

• "strict-origin" — вставлять заголовок, но отправлять в нем интернет-адрес хоста и только при переходе на сайт, работающий через HTTPS, в противном случае не вставлять заголовок;

• "strict-origin-when-cross-origin" — вставлять заголовок, отправлять в нем интернет-адрес страницы при переходе на страницу того же сайта, работающего через HTTPS, интернет-адрес хоста — при переходе на страницу другого сайта, также работающего через HTTPS, и вообще не вставлять заголовок в остальных случаях;

• "unsafe-uri" — всегда вставлять заголовок с интернет-адресом страницы;

• None — заголовок Referrer-Policy вообще не будет вставляться в ответы.

Например, при указании параметра:



Сайт будет отправлять в ответах заголовок:



В параметре можно указать сразу несколько значений— на тот случай, если какое-то из них не будет "знакомо" веб-обозревателю:

• либо в одной строке, перечислив их через запятую:



• либо в виде списка или кортежа:



Последнее значение из перечисленных будет трактоваться как предпочтительное.

Значение по умолчанию — None;

О csrf\_cookie\_secure — если True, то электронные жетоны в веб-формах для идентификации получаемых данных будут пересылаться в подписанных cookie (по умолчанию — False).

Этим параметрам также нужно задать значение True, чтобы обезопасить сайт и его посетителей от сетевых атак;

□ session\_cookie\_secure— ему нужно задать значение True, чтобы cookie сессий загружались только по протоколу HTTPS;

□ x frame options — указывает, разрешает ли сайт веб-обозревателям открывать свои страницы во фреймах. Доступны два строковых значения:

• "sameorigin" — разрешается открывать страницы только во фреймах, что находятся на страницах того же сайта;

• "deny" — полный запрет на открытие страниц текущего сайта во фреймах.

Значение по умолчанию, начиная с Django 3.0, — "deny" (в более старых версиях — "SAMEORIGIN").

Указывать значение "sameorigin" этого параметра следует только в случаях, если на страницах сайта не заносится какая-либо важная информация (например, номер кредитной карты);

□ secure proxy ssl header— задает пару "заголовок-значение", чье присутствие в запросе указывает на то, что запрос был выполнен по защищенному протоколу HTTPS. Значение параметра задается в виде кортежа из двух строковых элементов: первый элемент укажет заголовок (присутствующие в нем дефисы следует заменить подчеркиваниями, а сам заголовок— предварить символами нттр\_), а второй элемент — значение. Пример:



В этом случае присутствие в полученном запросе заголовка:



Сообщит Django о том, что запрос пришел по защищенному протоколу.

Значение по умолчанию: None (Django будет выяснять, пришел ли запрос по защищенному протоколу, проверяя, присутствует ли в начале интернет-адреса обозначение https://).

Этот параметр необходимо указывать, если Django-сайт соединяется с Интернетом через прокси-сервер. В таком случае Django не сможет определить, по какому протоколу клиент соединился с прокси-сервером: например, если сайт и прокси-сервер соединяются по протоколу HTTP, то даже при получении от клиента запроса по HTTPS Django будет ’’считать”, что соединение небезопасно.

Помимо указания параметра secure proxy ssl header, необходимо сконфигурировать прокси-сервер таким образом, чтобы, получив запрос по незащищенному протоколу, он вырезал из запроса указанный в параметре secure\_proxy\_ssl\_Header заголовок, а получив запрос по защищенному протоколу — добавлял его.

Однако в таком случае необходимо иметь доступ к прокси-серверу и возможность задавать его настройки.