**Тема занятия № 3: Создание и настройка проекта**

1. **Подготовка к работе**

Перед началом разработки сайта на Django, даже перед созданием его проекта, следует выполнять действия, перечисленные далее.

1. Проверить версии установленного ПО: исполняющей среды Python и серверных СУБД, если используемые проектом базы данных хранятся на них.

Установить сам фреймворк Django. Сделать это можно, указав в командной строке следующую команду:



Установить клиентские программы и дополнительные библиотеки для работы с СУБД.

Создать базу данных, в которой будут храниться данные сайта. Процедура ее создания зависит от формата базы.

Также Django поддерживает работу с базами данных Oracle, Microsoft SQL Server, Firebird, IBM DB2 и механизмом ODBC.

1. **Создание проекта Django**

Новый проект Django создается командой startproject утилиты django-admin, отдаваемой в следующем формате:

Django-admin startproject <имя проекта> [<путь к папке проекта>]

Если путь к папке проекта не указан, папка проекта будет создана в текущей папке и получит то же имя, что и сам проект. В противном случае папкой проекта станет папка с указанным в команде путем.

Папку проекта можно впоследствии переместить в любое другое место файловой системы, а также переименовать. Никакого влияния на работу проекта эти действия не оказывают.

1. **Настройки проекта**

Настройки проекта указываются в модуле settings.py пакета конфигурации. Значительная их часть имеет значения по умолчанию, оптимальные в большинстве случаев.

Настройки хранятся в обычных переменных Python. Имя переменной и есть имя соответствующего ей параметра.

1. **Основные настройки**

Основные настройки:

□ base dir — путь к папке проекта. По умолчанию вычисляется автоматически:

□ debug — режим работы сайта: отладочный (значение True) или эксплуатационный (False). По умолчанию — False (эксплуатационный режим), однако сразу при создании проекта для этого параметра указано значение True (т. Е. Сайт для облегчения разработки сразу же вводится в отладочный режим).

Если сайт работает в отладочном режиме, то при возникновении любой ошибки в коде сайта Django выводит веб-страницу с детальным описанием этой ошибки.

В эксплуатационном режиме в таких случаях выводятся стандартные сообщения веб-сервера наподобие ’’Страница не найдена” или ’’Внутренняя ошибка сервера”. Помимо того, в эксплуатационном режиме действуют более строгие настройки безопасности:

□ default charset — кодировка веб-страниц по умолчанию. По умолчанию: utf-8;

□ root urlconf — путь к модулю, в котором записаны маршруты уровня проекта, в виде строки. Значение этого параметра указывается сразу при создании проекта:

□ secret key — секретный ключ в виде строки с произвольным набором символов.

Используется программным ядром Django и подсистемой разграничения доступа для шифрования важных данных.

Значение параметра по умолчанию— пустая строка. Однако непосредственно при создании проекта ему присваивается секретный ключ, сгенерированный утилитой django-admin.

Менять этот секретный ключ без особой необходимости не стоит. Также его следует хранить в тайне, в противном случае он может попасть в руки злоумышленникам, которые используют его для атаки на сайт.

1. **Параметры баз данных**

Все базы данных, используемые проектом, записываются в параметре databases. Его значением должен быть словарь Python. Ключи элементов этого словаря задают псевдонимы баз данных, зарегистрированных в проекте. Можно указать произвольное число баз данных. Если при выполнении операций с моделями база данных не указана явно, то будет использоваться база с псевдонимом default.

В качестве значений элементов словаря также указываются словари, хранящие, собственно, параметры соответствующей базы данных. Каждый элемент вложенного словаря указывает отдельный параметр.

Значение параметра databases по умолчанию — пустой словарь. Однако при создании проекта ему дается следующее значение:

Оно указывает единственную базу данных, применяемую по умолчанию. База записывается в формате sqlite и хранится в файле db.sqlite3 в папке проекта.

Вот параметры баз данных, поддерживаемые Django:

□ engine — формат используемой базы данных.

Указывается как путь к модулю, реализующему работу с нужным форматом баз данных, в виде строки. Доступны следующие значения:

• django.db.backends.sqlite3— sqlite;

• dj ango.db.backends.mysql — mysql;

• dj ango.db.backends.postgresql —postgresql;

• dj ango. Db. Backends. Oracle — Oracle;

□ name — путь к файлу базы данных, если используется sqlite, или имя базы данных в случае серверных СУБД;

□ time zone — временная зона для значений даты и времени, хранящихся в базе.

Используется в том случае, если формат базы данных не поддерживает хранение значений даты и времени с указанием временной зоны. Значение по умолчанию — None (значение временной зоны берется из одноименного параметра проекта}.

Следующие параметры используются только в случае серверных СУБД:

□ host — интернет-адрес компьютера, на котором работает СУБД;

□ port — номер TCP-порта, через который выполняется подключение к СУБД. По

Умолчанию — пустая строка (используется порт по умолчанию);

□ user— имя пользователя, от имени которого Django подключается к базе данных;

□ password — пароль пользователя, от имени которого Django подключается к базе;

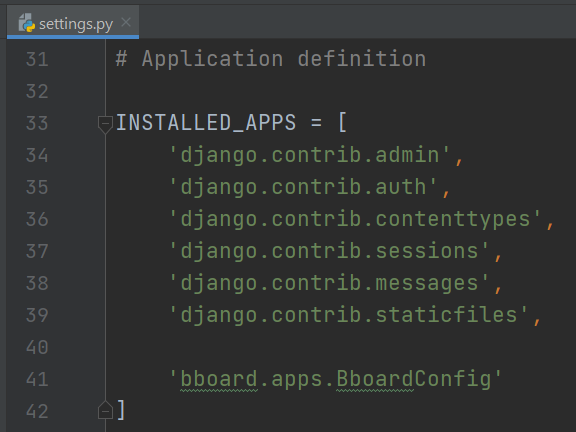
□ conn max age— время, в течение которого соединение с базой данных будет открыто, в виде целого числа в секундах.

Если задано значение 0, соединение будет закрываться сразу после обработки запроса. Если задано значение None, соединение будет открыто всегда. По умолчанию — 0;

□ options — дополнительные параметры подключения к базе данных, специфичные для используемой СУБД. Записываются в виде словаря, в котором каждый элемент указывает отдельный параметр. По умолчанию — пустой словарь.

1. **Список зарегистрированных приложений**

Список приложений, зарегистрированных в проекте, задается параметром installed apps. Все приложения, составляющие проект (написанные самим разработчиком сайта, входящие в состав Django и дополнительных библиотек), должны быть приведены в этом списке.



Он содержит такие стандартные приложения:

□ dj ango. Contrib. Admin —административный веб-сайт Django;

□ django.contrib.auth— встроенная подсистема разграничения доступа. Используется административным сайтом (приложением django.contrib.admin);

□ django.contrib.contenttypes —хранит список всех моделей, объявленных во всех приложениях сайта. Необходимо при создании полиморфных связей между моделями, используется административным сайтом, подсистемой разграничения доступа (приложениями django. Contrib. Admin И dj ango. Contrib. Auth);

□ django.contrib. Sessions— обрабатывает серверные сессии. Требуется при задействовании сессий и используется административным сайтом (приложением dj ango. Contrib. Admin);

□ django. Contrib. Messages — выводит всплывающие сообщения. Требуется для обработки всплывающих сообщений и используется административным сайтом (приложением django. Contrib.admin);

□ django.contrib.staticfiles— обрабатывает статические файлы.

Необходимо, если в составе сайта имеются статические файлы.

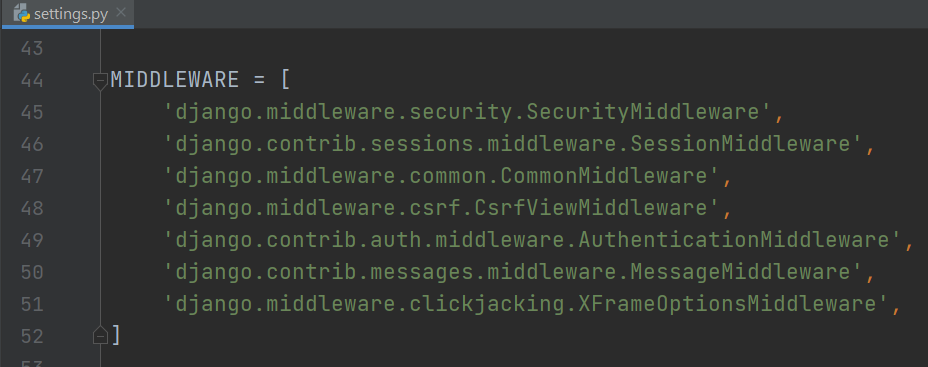
Если какое-либо из указанных приложений не нужно для работы сайта, его можно удалить из этого списка. Также следует убрать используемые удаленным приложением посредники (о них мы скоро поговорим).

1. **Список зарегистрированных посредников**

Посредник (middleware) Django — это программный модуль, выполняющий предварительную обработку клиентского запроса перед передачей его контроллеру и окончательную обработку ответа, сгенерированного контроллером, перед его отправкой клиенту.

Список посредников, зарегистрированных в проекте, указывается в параметре middleware. Все посредники, используемые в проекте (написанные разработчиком сайта, входящие в состав Django и дополнительных библиотек), должны быть приведены в этом списке.

Значение параметра middleware по умолчанию — пустой список. Однако сразу при создании проекта ему присваивается следующий список:



В нем перечислены стандартные посредники:

□ django.middleware.security.securitymiddleware — реализует дополнительную защиту сайта от сетевых атак;

□ django.contrib.sessions.middleware.sessionmiddleware — обрабатывает Серверные сессии на низком уровне. Используется подсистемами разграничения доступа, сессий и всплывающих сообщений (приложениями django.contrib.auth, django. Contrib.sessions И django. Contrib.messages);

□ django.middleware.common.commonmiddleware— участвует в предварительной обработке запросов;

□ django.middleware.csrf.csrfviewmiddleware— осуществляет защиту ОТ межсайтовых атак при обработке данных, переданных сайту HTTP-методом POST;

□ django.contrib.auth.middleware.authenticationmiddleware— добавляет В объект запроса атрибут, хранящий текущего пользователя. Через этот атрибут в контроллерах и шаблонах можно выяснить, какой пользователь выполнил вход на сайт и выполнил ли вообще. Используется административным сайтом и подсистемой разграничения доступа (приложениями django.contrib.admin И django. Contrib. Auth);

□ django. Contrib.messages.middleware.messagemiddleware — обрабатывает всплывающие сообщения на низком уровне. Используется административным сайтом И подсистемой всплывающих Сообщений (приложениями django. Contrib.admin и dj ango.contrib.messages);

□ django.middleware, clickjacking.xframeoptionsmiddleware— реализует дополнительную защиту сайта от сетевых атак.

Если какой-либо из этих посредников не нужен, его можно удалить из списка. Исключение составляют посредники:

Dj ango.middleware.security.securitymiddleware,

Dj ango .middleware. Common. Commonmiddleware,

Dj ango.middleware.clickj acking.xframeoptionsmiddleware,

Dj ango.middleware.csrf.csrfviewmiddleware — их удалять не стоит.

1. **Языковые настройки**

□ language code — код языки, на котором будут выводиться системные сообщения и страницы административного сайта, в виде строки. По умолчанию: ”en-us” (американский английский). Для задания русского языка следует указать:

LANGUAGE\_CODE = ’ru'

□ use\_i18n— если True, будет активизирована встроенная в Django система автоматического перевода на язык, записанный в параметре language code, после чего все системные сообщения и страницы административного сайта будут выводиться на этом языке. Если False, автоматический перевод выполняться не будет, и сообщения и страницы станут выводиться на английском языке. По умолчанию---True.

□ use\_l18n — если True, числа, значения даты и времени при выводе будут форматироваться по правилам языка из параметра language code. Если False, все эти значения будут форматироваться согласно настройкам, заданным в проекте. По умолчанию — False, однако при создании проекта ему дается значение True.

□ time zone— обозначение временной зоны в виде строки. По умолчанию — "America/Chicago". Однако сразу же при создании проекта ему присваивается значение "итс" (всемирное координированное время). Список всех доступных временных зон можно найти по интернет-адресу https://en.wikipedia.org/wiki/List\_of\_tz\_database\_time\_zones.

□ use tz — если True, Django будет хранить значения даты и времени с указанием временной зоны, в этом случае параметр time zone указывает временную зону по умолчанию. Если False, значения даты и времени будут храниться без отметки временной зоны, и временную зону для них укажет параметр time zone. По умолчанию — False, однако при создании проекта ему дается значение True.

Следующие параметры будут приниматься Django в расчет только в том случае, если отключено автоматическое форматирование выводимых чисел, значений даты и времени (параметру use\_lisn дано значение False):

□ decimal separator — символ-разделитель целой и дробной частей вещественных чисел. По умолчанию: ”. ” (точка);

□ number grouping — количество цифр в числе, составляющих группу, в виде целого числа. По умолчанию — 0 (группировка цифр не используется);

□ thousand separator — символ-разделитель групп цифр в числах. По умолчанию:

" г" (запятая);

□ use thousands separator — если True, числа будут разбиваться на группы, если False —не будут. По умолчанию — False;

□ short date format — "короткий" формат значений даты. По умолчанию: "m/d/Y”

(<месяц>/<число>/<год из четырех цифр>}. Для указания формата <число>.<месяц>.<год из четырех цифр> следует добавить в модуль settings.py выражение:

SHORT\_DATE\_FORMAT = ' j . M. Y'

□ short datetime format — ’’короткий” формат временных отметок (значений даты и времени). По умолчанию: "m/d/Y р” (<месяц>/<число>/<год из четырех цифр> <часы в 12-часовом формате>). Задать формат <число>.<месяц>.<год из четырех цифр> <часы в 24-часовом формате>.<минуты>\<секунды> можно, добавив в модуль settings.py выражение:

SHORT\_DATETIME\_FORMAT = ’j.m.Y H:i:s’

То же самое, только без секунд:

SHORT\_DATETIME\_FORMAT = ’j.m.Y H:i’

□ date format — ’’полный” формат значений даты. По умолчанию: ”N j, Y” (^название месяца по-английски> <число>, <год из четырех цифр>). Чтобы указать формат <число> <название месяца> <год из четырех цифр>, следует добавить в модуль settings.py выражение:

DATE\_FORMAT = ’j Е Y'

□ datetime format — ’’полный” формат временных отметок. По умолчанию: "N j,ч., р” (<название месяца по-английски> <число>, <год из четырех цифр> <часы в 12-часовом форматер. Для указания формата <число> <название месяца><год из четырех цифр> <часы в 24-часовом формате>:<минуты>\<секунды> нужно добавить в модуль settings.py выражение:

DATETIME\_FORMAT = 'j Е Y H:i:s’

То же самое, только без секунд:

DATETIME\_FORMAT = 'j Е Y H:i’

□ time format - формат значений времени. По умолчанию: пр” (только часы в 12-часовом формате). Для задания формата <часы в 24-часовом формате>\<минуты>:<секунды> достаточно добавить в модуль settings.py выражение:

TIME\_FORMAT = ’H:i:s’

То же самое, только без секунд:

TIME\_FORMAT = ’H:i'

□ month day format— формат для вывода месяца и числа. По умолчанию: "f, j” (<название месяца по-английски>. <число>);

□ year month format— формат для вывода месяца и года. По умолчанию: ”f y” (<название месяца по-английски> <год из четырех цифр>).

В значениях всех этих параметров применяются специальные символы, используемые в фильтре шаблонизатора date;

□ date input formats — список форматов, в которых посетителям допускается заносить значения даты в поля ввода. Получив из веб-формы значение даты в виде строки, Django будет последовательно сравнивать его со всеми форматами, имеющимися в этом списке, пока не найдет подходящий для преобразования строки в дату.

Значение по умолчанию — довольно длинный список форматов, среди которого, к сожалению, нет формата <число>.<месяц>.<год из четырех цифр>. Чтобы указать его, следует записать в модуле settings.py выражение:

DATE\_INPUT\_FORMATS = [ ’ %d. %m. %Y' ]

□ datetime input formats — список форматов, в которых посетителям допускается заносить временные отметки в поля ввода. Получив из веб-формы временную отметку в виде строки, Django будет последовательно сравнивать его со всеми форматами из списка, пока не найдет подходящий для преобразования строки в значение даты и времени.

Значение по умолчанию — довольно длинный список форматов, среди которого, к сожалению, нет формата <число>.<месяц>.<год из четырех цифр> <часы в 24-часовом формате>\<минуты>.<секунды>. Чтобы указать его, следует записать в модуле settings.py выражение:

DATETIME\_INPUT\_FORMATS = ['%d.%m.%Y %H:%M:%S']

То же самое, только без секунд:

DATETIME\_INPUT\_FORMATS = [ ' %d. %m. %Y %H: %M’ ]

□ time input formats — список форматов, в которых посетителям допускается заносить значения времени в поля ввода. Получив из веб-формы значение времени в виде строки, Django будет последовательно сравнивать его со всеми форматами из списка, пока не найдет подходящий для преобразования строки в значение времени.

Значение по умолчанию: [’%H:%M:%s\ ’%H:%M:%s.%f’%н:%м’] (в том числе форматы <часы в 24-часовом формате>\<минуты>\<секунды> и <часы в 24-часовом формате>\<минуты>}.

В значениях всех этих параметров применяются специальные символы, поддерживаемые функциями strftimeo и strptimeo Python (их перечень приведен в документации по Python и на странице https://docs.python.Org/3/Iibrary/datetime.html#strftime-strptime-behavior).

□ first day of week— номер дня, с которого начинается неделя, в виде целого числа от о (воскресенье) до 6 (суббота). По умолчанию: о (воскресенье).

Здесь описаны не все параметры проекта, доступные для указания. Остальные параметры, затрагивающие работу отдельных подсистем и касающиеся функционирования сайта в эксплуатационном режиме, мы рассмотрим в последующем.

1. **Создание, настройка и регистрация приложений**

Приложения реализуют отдельные части функциональности сайта Django. Любой проект должен содержать по крайней мере одно приложение.

1. **Создание приложений**

Создание приложение выполняется командой startapp утилиты manage.ру, записываемой в формате:

Manage.ру startapp <имя приложения> [<путь к папке пакета приложения>]

Если путь к папке пакета приложения не указан, то папка пакета приложения с заданным именем будет создана в папке проекта. В противном случае в пакет приложения будет преобразована папка, расположенная по указанному пути.

1. **Настройка приложений**

Модуль apps.py пакета приложения содержит объявление конфигурационного класса. Хранящего настройки приложения. Код такого класса, задающего параметры приложения bboard



Он является подклассом класса appconfig ИЗ модуля django.apps и содержит набор

Атрибутов класса, задающих немногочисленные параметры приложения:

□ name —полный путь к пакету приложения, записанный относительно папки проекта, в виде строки. Единственный параметр приложения, обязательный для указания. Задается утилитой manage.py непосредственно при создании приложения, и менять его не нужно;

□ label — псевдоним приложения в виде строки. Используется в том числе для указания приложения в вызовах утилиты manage.py. Должен быть уникальным в пределах проекта. Если не указан, то в качестве псевдонима принимается последний компонент пути из атрибута name;

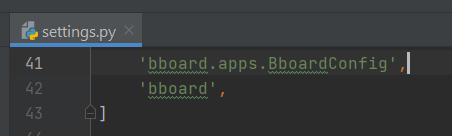
□ verbose name — название приложения, выводимое на страницах административного сайта Django. Если не указан, то будет выводиться псевдоним приложения;

□ path— файловый путь к папке пакета приложения. Если не указан, то Django

Определит его самостоятельно.

1. **Регистрация приложения в проекте**

\*\*\*Чтобы приложение успешно работало, оно должно быть зарегистрировано в списке приложений installed apps, который находится в параметрах проекта. Зарегистрировать его можно двумя способами.



1. **Отладочный веб-сервер Django**

Запуск отладочного веб-сервера Django выполняется утилитой manage.py при получении ею команды runserver, записываемой в формате:

Manage.py runserver [[<интернет-адрес>:]<порт>] [—noreload]

[—nothreading] [—ipv6] [-6]

Отладочный веб-сервер Django самостоятельно отслеживает изменения в программных модулях, при их сохранении сам выполняет перезапуск. Нам об этом беспокоиться не следует.

По умолчанию отладочный сервер доступен с локального хоста через ТСР-порт № 8000 — по интернет-адресам http://localhost:8000/ и http://127.0.0.1:8000/.

Можно указать другой порт и, при необходимости, интернет-адрес. Например, так

Задается использование порта № 4000:



Задаем использование интернет-адреса 1.2.3.4 и порта № 4000:

Manage.py runserver 1.2.3.4:4000

Команда runserver поддерживает следующие дополнительные ключи:

□ —noreload — отключить автоматический перезапуск при изменении программного кода;

□ —nothreading— принудительный запуск отладочного сервера в однопоточном режиме (если не указан, сервер запускается в многопоточном режиме);

□ —ipv6 или -6— работать через протокол ipv6 вместо ipv4. В этом случае по умолчанию будет использоваться интернет-адрес ::1.