17.02.2024

Charm . – чтобы открыть в пайчарм через терминал

mkdir

cd

python3 -m venv venv

. venv/bin/activate

pip install Django

django-admin startproject nameproject .

django-admin startapp appname

python3 manage.py runserver

19.02.2024 второй урок по django

Strtfime()

20.02.2024 третий урок по django

Дорогой дневник сегодня я ощутил такой позор который не описать словами я не смог ответить на вопрос как создают Django приложении

Shell

Открывает консоль проекта

From models import News

News.objects.all()

Возвращает querset

News.objects.all().count()

News.objects.get()

News.objects(name=’News2’)

News.filter()

News.exclude(name= ‘News’)

Возврашает те данные где нет совпадение с условаиями

News.first()

Возвращает первый попавший элемент

21.02.2024 четвертый урок по Django

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [os.path.join(BASE\_DIR, 'templates')],

'APP\_DIRS': True,

'OPTIONS': {

'context\_processors': [

'django.template.context\_processors.debug',

'django.template.context\_processors.request',

'django.contrib.auth.context\_processors.auth',

'django.contrib.messages.context\_processors.messages',

],

},

},

]

os.path.join(BASE\_DIR, ‘templates’)

22.02.2024 пятый урок по django

from django.urls import path

from django.conf import settings

from django.urls import path

from django.conf.urls.static import static

urlpatterns = [

    path("about/", TemplateView.as\_view(template\_name="about.html")),

    path("contact/", TemplateView.as\_view(template\_name="contact.html")),

]

STATIC\_URL = 'static/'

STATICFILES\_DIRS = [os.path.join(BASE\_DIR, 'static')]

STATIC\_ROOT = os.path.join(BASE\_DIR, 'static', 'static\_root')

MEDIA\_URL = 'media/'

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

urlpatterns += static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

\_\_\_-----------------------------------------------\_\_\_\_

------------mamaRishan1970---------------

22.02.2024 шестой урок по django

* ForeignKey – для связей Many to One (поля отношений);
* ManyToManyField – для связей Many to Many (многие ко многим);
* OneToOneField – для связей One to One (один к одному).

Класс ForeignKey принимает два обязательных аргумента:

* ссылка или строка класса модели, с которой происходит связывание (в нашем случае это класс Category – модели для категорий);
* опция on\_delete, накладывающая ограничения при удалении внешней записи (в нашем примере – это удаление из таблицы Category).

В свою очередь, опция on\_delete может принимать следующие значения:

* models.CASCADE – при удалении записи из первичной модели (у нас это таблица Category) происходит удаление всех записей из вторичной модели (Women), связанных с удаляемой категорией;
* models.PROTECT – запрещает удаление записи из первичной модели, если она используется во вторичной (выдает исключение);
* models.SET\_NULL – при удалении записи первичной модели устанавливает значение foreign key в NULL у соответствующих записей вторичной модели;
* models.SET\_DEFAULT – то же самое, что и SET\_NULL, только вместо значения NULL устанавливает значение по умолчанию, которое должно быть определено через класс ForeignKey;
* models.SET() – то же самое, только устанавливает пользовательское значение;
* models.DO\_NOTHING – удаление записи в первичной модели не вызывает никаких действий у вторичных моделей.

напиши описание перечисленных команд Для первой загрузки git init git add . git commit -m "first commit" git branch -M main git remote add origin https://github.com/prolabagency/example.git git push -u origin main Загрузка изменений на github git add . git commit -m "first commit" git push Загрузка изменений из github git push Загрузка целого проекта из github git clone

ChatGPT

**ChatGPT**

Команды Git представляют собой инструменты для управления версиями кода и сотрудничества в разработке программного обеспечения. Вот описание перечисленных команд:

1. **git init**: Создает новый репозиторий Git. Это выполняется один раз для проекта. Команда создает скрытую папку **.git**, которая содержит всю информацию о версионном контроле проекта.
2. **git add .**: Добавляет все изменения в рабочем каталоге в индекс (staging area) для последующего коммита. Точка здесь означает "все файлы и папки". Можно также указать конкретные файлы для добавления.
3. **git commit -m "first commit"**: Создает коммит, который представляет собой зафиксированное изменение. Флаг **-m** позволяет добавить сообщение о коммите, обычно описывающее изменения, внесенные в этот коммит.
4. **git branch -M main**: Переименовывает текущую ветку в **main**. Это часто используется в новых репозиториях вместо стандартной ветки **master**.
5. **git remote add origin https://github.com/prolabagency/example.git**: Связывает локальный репозиторий Git с удаленным репозиторием на GitHub. **origin** - это просто имя, которое обычно используется для ссылки на удаленный репозиторий.
6. **git push -u origin main**: Отправляет локальные коммиты на удаленный репозиторий GitHub в ветку **main**. Флаг **-u** устанавливает отслеживание, что позволяет в будущем использовать просто **git push** без указания места назначения и имени ветки.
7. **git push**: Отправляет локальные коммиты на удаленный репозиторий. Если ветка уже отслеживается, то Git отправит коммиты в соответствующую ветку на удаленном репозитории.
8. **git clone**: Клонирует удаленный репозиторий Git в новый локальный репозиторий. Это позволяет загрузить целый проект с GitHub (или другого хоастинга) на ваш компьютер для работы с ним локально.

Эти команды помогают в управлении процессом разработки и обмена кодом между разработчиками.

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/Amantur1develooper/my\_own\_site2.git

git push -u origin main

from .models import News, Category, Tag, AdditionalNewsInfo

from django.utils.safestring import mark\_safe

@admin.register(News)

class NewsAdmin(admin.ModelAdmin):

list\_display = ('id', 'name', 'category', 'date', 'is\_published', 'get\_image')

list\_display\_links = ('id', 'name')

search\_fields = ('name', 'description', 'content')

list\_filter = ('category', 'date', 'is\_published', 'tags')

readonly\_fields = ('date', 'get\_full\_image')

# filter\_vertical = ('tags',)

filter\_horizontal = ('tags',)

# raw\_id\_fields = ('category',)

@admin.display(description='Изображение')

def get\_image(self, news: News):

return mark\_safe(f'<img src="{news.image.url}" width="150px">')

@admin.display(description='Изображение')

def get\_full\_image(self, news: News):

return mark\_safe(f'<img src="{news.image.url}" width="50%">')

@admin.register(Category)

class CategoryAdmin(admin.ModelAdmin):

list\_display = ('id', 'name')

list\_display\_links = ('id', 'name')

search\_fields = ('name',)

@admin.register(Tag)

class CategoryAdmin(admin.ModelAdmin):

list\_display = ('id', 'name')

list\_display\_links = ('id', 'name')

search\_fields = ('name',)

@admin.register(AdditionalNewsInfo)

class CategoryAdmin(admin.ModelAdmin):

list\_display = ('id', 'news\_name', 'email', 'rating')

list\_display\_links = ('id', 'news\_name')

search\_fields = ('news\_\_name', 'news\_\_description', 'news\_\_content')

list\_filter = ('news', 'news\_\_category', 'news\_\_date', 'news\_\_tags')

readonly\_fields = ('news\_name',)

@admin.display(description='Название')

def news\_name(self, info: AdditionalNewsInfo):

return f'{info.news.name}'

валидатор

rating = models.IntegerField(validators=[MinValueValidator(1), MaxValueValidator(5)], verbose\_name='оценка')

\*\*\*\*\*

Оценка от 1 до 5

12.03.24

from django.utils.safestring import mark\_safe

class NewsAttributeStackedInline(admin.StackedInline):

model = NewsAttribute

extra = 1

class AdditionalNewsInfoTabularInline(admin.TabularInline):

model = AdditionalNewsInfo

@admin.register(News)

class NewsAdmin(admin.ModelAdmin):

list\_display = ('id', 'name', 'category', 'date', 'is\_published', 'get\_image')

list\_display\_links = ('id', 'name')

search\_fields = ('name', 'description', 'content')

list\_filter = ('category', 'date', 'is\_published', 'tags')

readonly\_fields = ('date', 'get\_full\_image')

# filter\_vertical = ('tags',)

filter\_horizontal = ('tags',)

# raw\_id\_fields = ('category',)

inlines = (NewsAttributeStackedInline, AdditionalNewsInfoTabularInline)

@admin.display(description='Изображение')

def get\_image(self, news: News):

return mark\_safe(f'<img src="{news.image.url}" width="150px">')

@admin.display(description='Изображение')

def get\_full\_image(self, news: News):

return mark\_safe(f'<img src="{news.image.url}" width="50%">')

Q параметры https:?key=value&+

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

MEDIA\_URL = '/media/'

academy2019