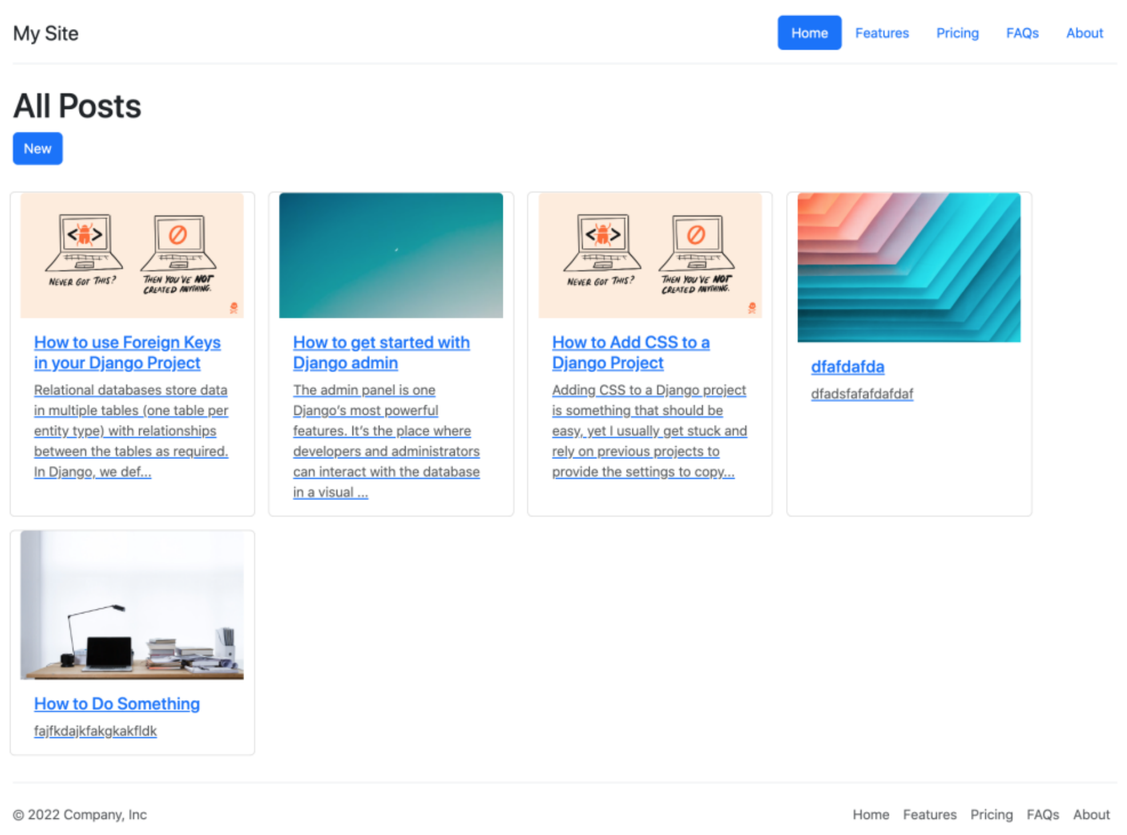
[(https://ctrlzblog.com)](https://ctrlzblog.com)

A Blog About Django & Web Development

BUILD A BETTER BLOG (HTTPS://CTRLZBLOG.COM/CATEGORY/DJANGO/BUILD-A-BETTER-BLOG/)

How To Upload Images In Django With Pillow (Blog Example)



All Posts

How to use Foreign Keys in your Django Project
Relational databases store data in multiple tables (one table per entity type) with relationships between the tables as required. In Django, we def...

How to get started with Django admin
The admin panel is one Django's most powerful features. It's the place where developers and administrators can interact with the database in a visual ...

How to Add CSS to a Django Project
Adding CSS to a Django project is something that should be easy, yet I usually get stuck and rely on previous projects to provide the settings to copy...

dfafdafda
dfadsfafafdafdaf

How to Do Something
fajfkdajkfakgkakfldk

© 2022 Company, Inc

By Alice Ridgway Updated October 30, 2022

blog (https://ctrlzblog.com/tag/blog/) django (https://ctrlzblog.com/tag/django/) images (https://ctrlzblog.com/tag/images/) pillow (https://ctrlzblog.com/tag/pillow/)

A common feature of Django projects is the ability to upload images. I'm going to show you an example where we add feature images to a blogging application.

For this tutorial, I'm going to assume you already have a project to add images to. If not, you can follow this tutorial by Django Girls (https://tutorial.djangogirls.org/en/) to create your own blog, or pull the code from this repository (GitHub (https://github.com/ctrlz-blog/basic-django-blog/tree/%2320-add-feature-image--before)).

We are going on work on the Post model and add the ability to upload images from a user form.

This tutorial will cover how to handle uploaded files **for development only.** When deploying your project, you will need to choose a different strategy to handle files uploaded in your Live environment (Django docs (https://docs.djangoproject.com/en/4.0/howto/static-files/deployment/)).

These are the steps we need to take:

1. Provide somewhere to store uploaded files
2. Update `urls.py`
3. Add an ImageField to the Post model
4. Update our form
5. Update the template
6. Update the view to process uploaded files

The code for this tutorial can be found here (GitHub (https://github.com/ctrlz-blog/basic-django-blog/tree/%2320-add-feature-image--after)).

1. Provide somewhere to store uploaded files

Before we can update our model, we need somewhere to put our uploaded files. You may need to add `import os` to the top of the file.

```
# settings.py

MEDIA_URL = "/media/"

MEDIA_ROOT = os.path.join(BASE_DIR, "media")
```

Update .gitignore

I recommend adding your media folder to your `.gitignore`. This is because uploaded files won't be needed on other environments with a separate database.

```
.vscode
venv
db.sqlite3
__pycache__
media
```

2. Update the URLs

We need to add a URL pattern to handle the upload of images. We only want to do it this way during development. In production, we won't want to store uploaded images that way. By stating `if settings.DEBUG`, we can ensure the URL pattern only gets used when running the application locally.

```
# projectconfig/urls.py

from django.conf import settings
from django.conf.urls.static import static
from django.contrib import admin
from django.urls import include, path



urlpatterns = [
    path("posts/", include("blog.urls")),
    path("admin/", admin.site.urls),
]

if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROOT)
```

3. Add an ImageField to the Post model

3a. Install Pillow

Pillow is a Python library (docs (https://pillow.readthedocs.io/en/stable/)) for processing images. Django does not allow you to use image fields in models without installing Pillow first. Trying to use this field without the package will get you this error:

```
(fields.E210) Cannot use ImageField because
Pillow is not installed.
        HINT: Get Pillow at
https://pypi.org/project/Pillow/ or run command
"python -m pip install Pillow".
```

To fix this, install Pillow:

```
$ pip install pillow
```

3b. Add the ImageField to the Post model

This is the code for my model. The `feature_image` field is added on the last line.

I have added `blank=True, null=True`, to make the field optional.

```
# blog/models.py

class Post(models.Model):

    STATUS_CHOICES = [
        ("draft", "Draft"),
        ("published", "Published"),
    ]

    title = models.CharField(max_length=255)
    slug = AutoSlugField(populate_from="title",
unique=True)
    status = models.CharField(max_length=20,
choices=STATUS_CHOICES, default="draft")

    body = models.TextField()

    created_date =
models.DateTimeField(auto_now_add=True)
    published_date =
models.DateTimeField(blank=True, null=True)
    updated_date =
models.DateTimeField(blank=True, null=True)
```

```
    tags = models.ManyToManyField(to=Tag,
related_name="posts", blank=True)
    category = models.ForeignKey(
        to=Category,
        related_name="posts",
        default=get_category_id,
        on_delete=models.SET_DEFAULT,
        blank=False,
        null=False,
    )

    feature_image =
models.ImageField(upload_to="uploads/",
null=True, blank=True)
```

4. Update the form

To make the image upload display on the form, add the name of the field to the list of `fields` in the `Meta` class.

```
# blog/forms.py

class PostForm(forms.ModelForm):
    class Meta:
        model = models.Post
        fields = ["title", "body", "category",
"tags", "feature_image"]

    tags = forms.ModelMultipleChoiceField(
        queryset=models.Tag.objects.all(),
widget=forms.CheckboxSelectMultiple
    )

    category = forms.ModelChoiceField(
        queryset=models.Category.objects.all(),
widget=forms.RadioSelect
    )
```

5. Update the template

5a. Update the add post form

We don’t need to make any changes to how the form is displayed but we do need to make changes about how the form is encoded (StackOverflow: What does enctype=’multipart/form-data’ mean?) (https://stackoverflow.com/questions/4526273/what-does-enctype-multipart-form-data-mean).

Adding `enctype="multipart/form-data"` is necessary to make your form upload files to the server. If you leave it out, images will be ignored when the form is submitted.

```
# templates/post_form.html

<form method="post" enctype="multipart/form-
data">
    {% csrf_token %}
    {{ form.as_p }}
    <input type="submit" class="btn btn-
primary" value="Save">
</form>
```

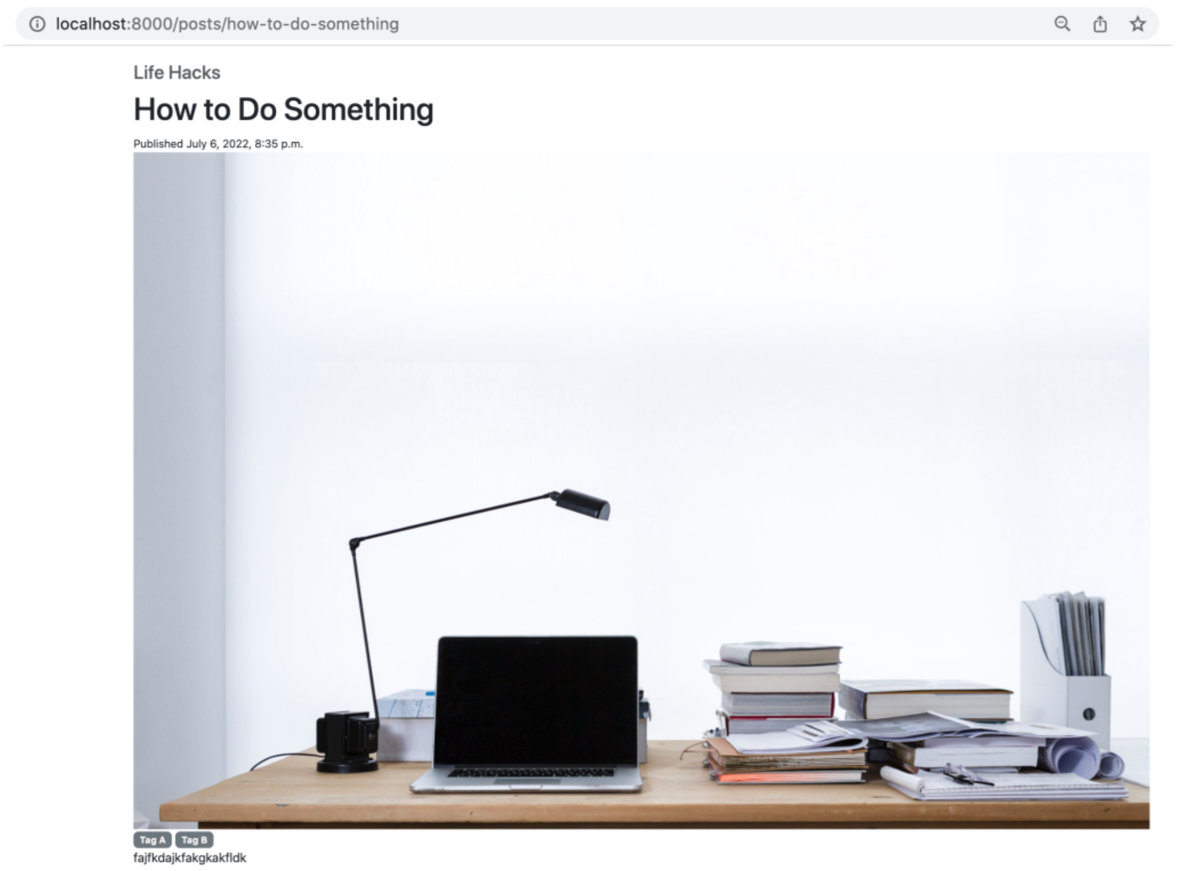
5b. Update the post detail template to display images

Here, I have updated the post detail template to display the images. The image field provides an attribute called `url` which provides the path to the image. This is the path to the file on the server, not a HTTP URL.

I am using Bootstrap, so adding a class of `img-fluid` will automatically size the image to fit the parent container.

```
# templates/post_detail.html

{% if post.feature_image %}
<img class="img-fluid" src="
{{post.feature_image.url}}" alt="
{{post.title}}">
{% endif %}
```


localhost:8000/posts/how-to-do-something

Life Hacks

How to Do Something

Published July 6, 2022, 8:35 p.m.

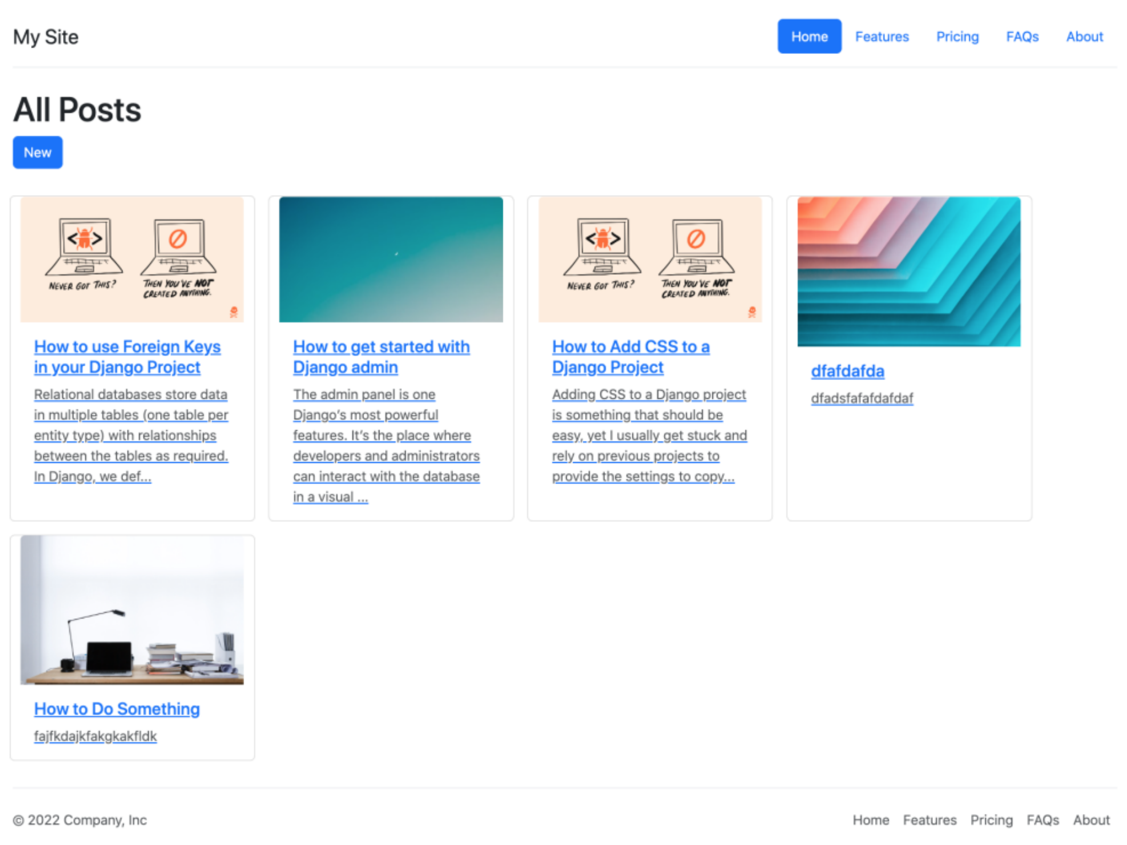
Tag A Tag B

fajfkdajkfakgkakfldk

5c. Update the post list template to show images

The final step is to update the post list.

```
# templates/index.html
{% if post.feature_image %}
    <img src="{{post.feature_image.url}}"
class="card-img-top" alt={{post.title}}>
{% else %}
    <img src="https://tinyurl.com/22jvzvfj"
class="card-img-top" alt={{post.title}}>
{% endif %}
```



My Site

Home Features Pricing FAQs About

All Posts

New

How to use Foreign Keys in your Django Project

Relational databases store data in multiple tables (one table per entity type) with relationships between the tables as required. In Django, we def...

How to get started with Django admin

The admin panel is one Django's most powerful features. It's the place where developers and administrators can interact with the database in a visual ...

How to Add CSS to a Django Project

Adding CSS to a Django project is something that should be easy, yet I usually get stuck and rely on previous projects to provide the settings to copy...

dfafdafda

dfadsfafafdafdaf

How to Do Something

fajfkdajkfakgkakfldk

© 2022 Company, Inc

Home Features Pricing FAQs About

6. Update the view to process uploaded files

The final step is to update the view.

The uploaded file is stored in `request.FILES`. The only change we need to make to the `add_post` and `edit_post` views is to instantiate the form with `request.FILES` as well as `request.POST`.

6a. Update the add_post view

```
# blog/views.py

def add_post(request: HttpRequest) ->
HttpResponse:

    if request.method == "POST":
        form = PostForm(data=request.POST,
files=request.FILES)

        if form.is_valid():
            # form.save() creates a post from
the form
            post = form.save()

            return redirect("post_detail",
slug=post.slug)

    else:
        form = PostForm()

    context = {"form": form, "edit_mode":
False}
```

```
    return render(request, "post_form.html",
context)
```

6b. Update the edit_post view

```
# blog/views.py

def edit_post(request: HttpRequest, slug: str)
-> HttpResponse:

    post = get_object_or_404(Post, slug=slug)

    form = PostForm(instance=post)

    if request.method == "POST":
        form = PostForm(data=request.POST,
files=request.FILES, instance=post)
        if form.is_valid():
            post = form.save()
            return redirect("post_detail",
slug=post.slug)

    context = {"form": form, "post": post,
"edit_mode": True}
    return render(request, "post_form.html",
context)
```

Related Posts

The Django Developer's Guide to Vite (https://ctrlzblog.com/the-django-developers-guide-to-vite/)

The easy way to add JavaScript to your Django projects.

Add Vue to your Django templates with Vite (https://ctrlzblog.com/add-vue-to-your-django-templates-with-vite/)

You don't have to build a separate app to use Vue with Django

How to Create Rows in the Database with Django ORM (https://ctrlzblog.com/how-to-create-rows-in-the-database-with-django-orm/)

Django ORM (Object Relational Mapper) allows your application to modify your database with Python. This means you won't have to write any SQL to manage

How Django Models Work – A Beginner's Guide (https://ctrlzblog.com/how-django-models-work-a-beginners-guide/)

Django models are used to manage your applications database without writing SQL.

User Registration with Django REST Framework (https://ctrlzblog.com/user-registration-with-django-rest-framework/)

User registration with Django REST Framework (DRF) can be challenging because you don't have the advantage of Django's built in user model. I am going

How to reset your Django Admin password when you can’t login (https://ctrlzblog.com/how-to-reset-your-django-admin-password-when-you-cant-login/)

How to quickly reset your Django Admin password without opening the browser.

(https://twitter.com/aliceridgway404)

Copyright Alice Ridgway 2023