

Azure Blob Storage

Django App

Initial Setup (Windows):

- Create a Project Folder: e.g azurestoragedemo

```
D:\>md azurestoragedemo  
  
D:\>cd azurestoragedemo  
  
D:\azurestoragedemo>|
```

- Create and activate a virtual environment: `python -m venv venv`

```
D:\azurestoragedemo>python -m venv venv  
  
D:\azurestoragedemo>venv\Scripts\activate  
  
(venv) D:\azurestoragedemo>|
```

- Install the required packages:
 >pip install azure-storage-blob django

```
D:\azurestoragedemo>venv\Scripts\activate  
  
(venv) D:\azurestoragedemo>pip install django azure-storage-blob  
Collecting django  
  Using cached Django-5.1.7-py3-none-any.whl.metadata (4.1 kB)  
Collecting azure-storage-blob  
  Using cached azure_storage_blob-12.25.0-py3-none-any.whl.metadata (26 kB)  
Collecting asgiref<4,>=3.8.1 (from django)  
  Using cached asgiref-3.8.1-py3-none-any.whl.metadata (9.3 kB)  
Collecting sqlparse>=0.3.1 (from django)
```

Create a Django Project e.g. blobstoreapp

- Create a Project Folder: e.g blobWrite

```
(venv) D:\azurestoragedemo>django-admin startproject blobWrite

(venv) D:\azurestoragedemo>cd blobWrite

(venv) D:\azurestoragedemo\blobWrite>python manage.py startapp blobstoreapp

(venv) D:\azurestoragedemo\blobWrite>dir
Volume in drive D is New Volume
Volume Serial Number is CEC6-C47A

Directory of D:\azurestoragedemo\blobWrite

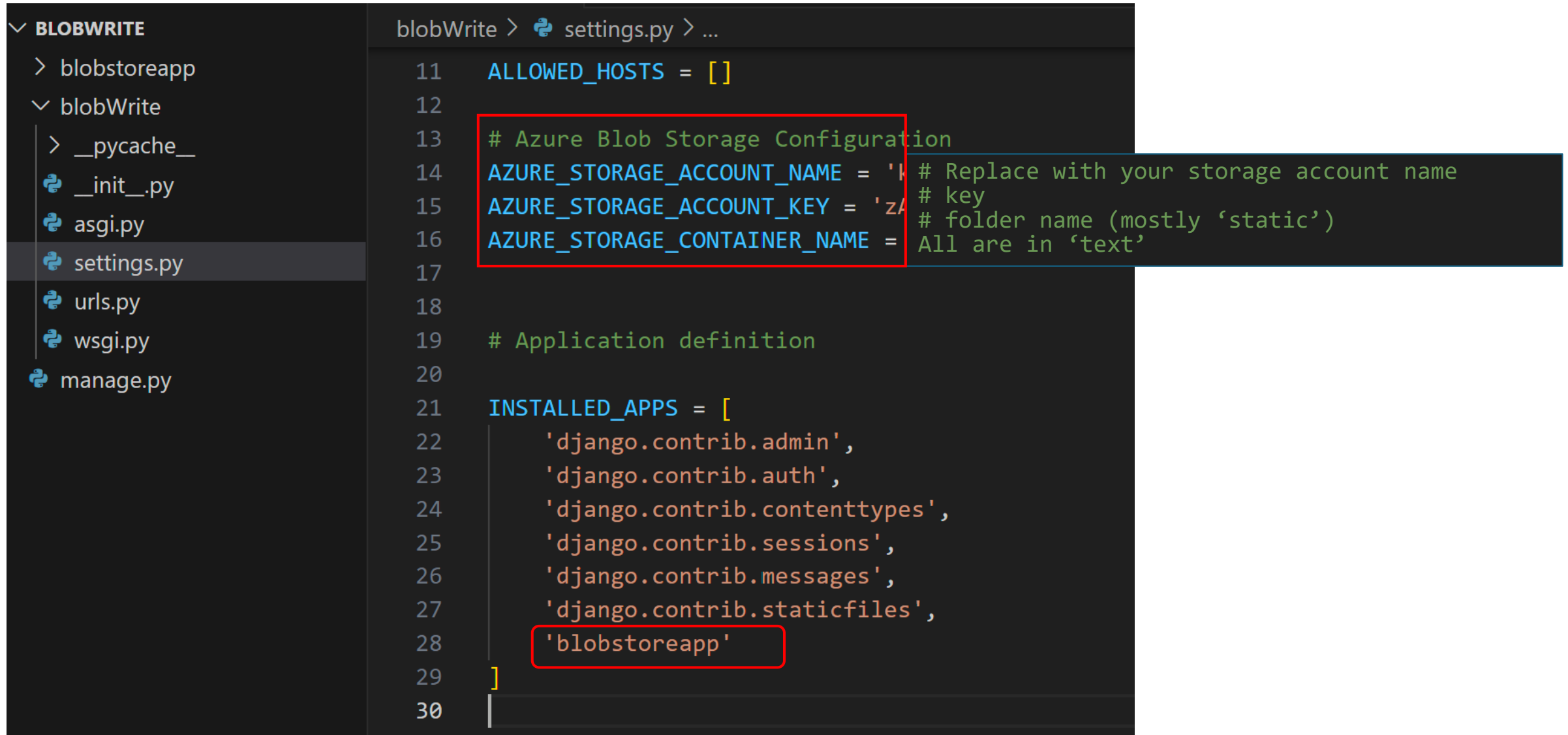
25-03-2025  17:08    <DIR>          .
25-03-2025  17:08    <DIR>          ..
25-03-2025  17:08    <DIR>          blobstoreapp
25-03-2025  17:08    <DIR>          blobWrite
25-03-2025  17:08                687 manage.py
                1 File(s)                687 bytes
                4 Dir(s)  298,607,767,552 bytes free

(venv) D:\azurestoragedemo\blobWrite>
```

- Open your code editor:

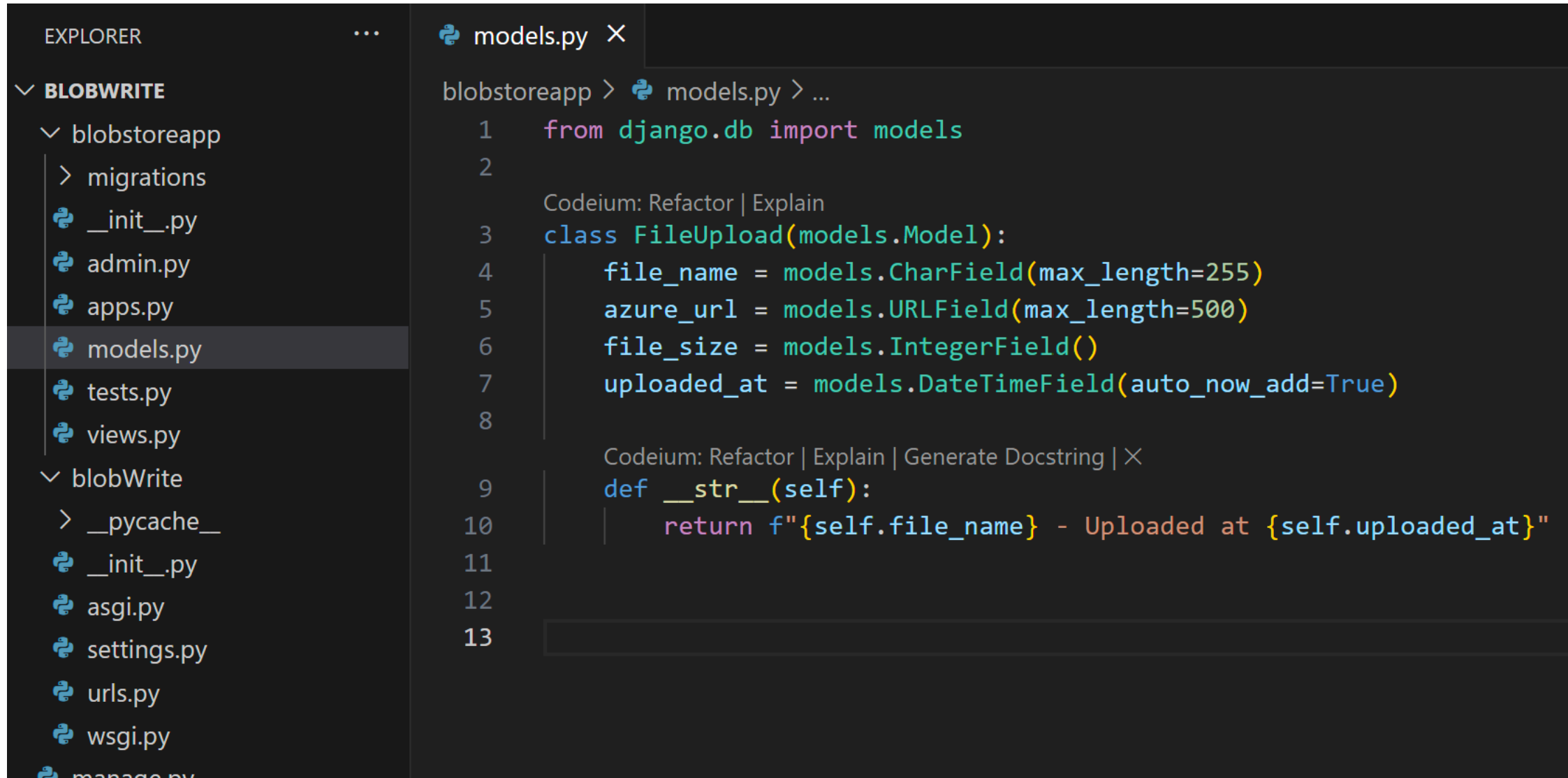
```
(venv) D:\azurestoragedemo\blobWrite>code .
```

azurestoragedemo/blobWrite/blobWrite/settings.py



```
11  ALLOWED_HOSTS = []
12
13  # Azure Blob Storage Configuration
14  AZURE_STORAGE_ACCOUNT_NAME = 'k' # Replace with your storage account name
15  AZURE_STORAGE_ACCOUNT_KEY = 'zA' # key
16  AZURE_STORAGE_CONTAINER_NAME = 'static' # folder name (mostly 'static')
17                                     # All are in 'text'
18
19  # Application definition
20
21  INSTALLED_APPS = [
22      'django.contrib.admin',
23      'django.contrib.auth',
24      'django.contrib.contenttypes',
25      'django.contrib.sessions',
26      'django.contrib.messages',
27      'django.contrib.staticfiles',
28      'blobstoreapp'
29  ]
30
```

azurestroragedemo/blobWrite/blobstoreapp\models.py

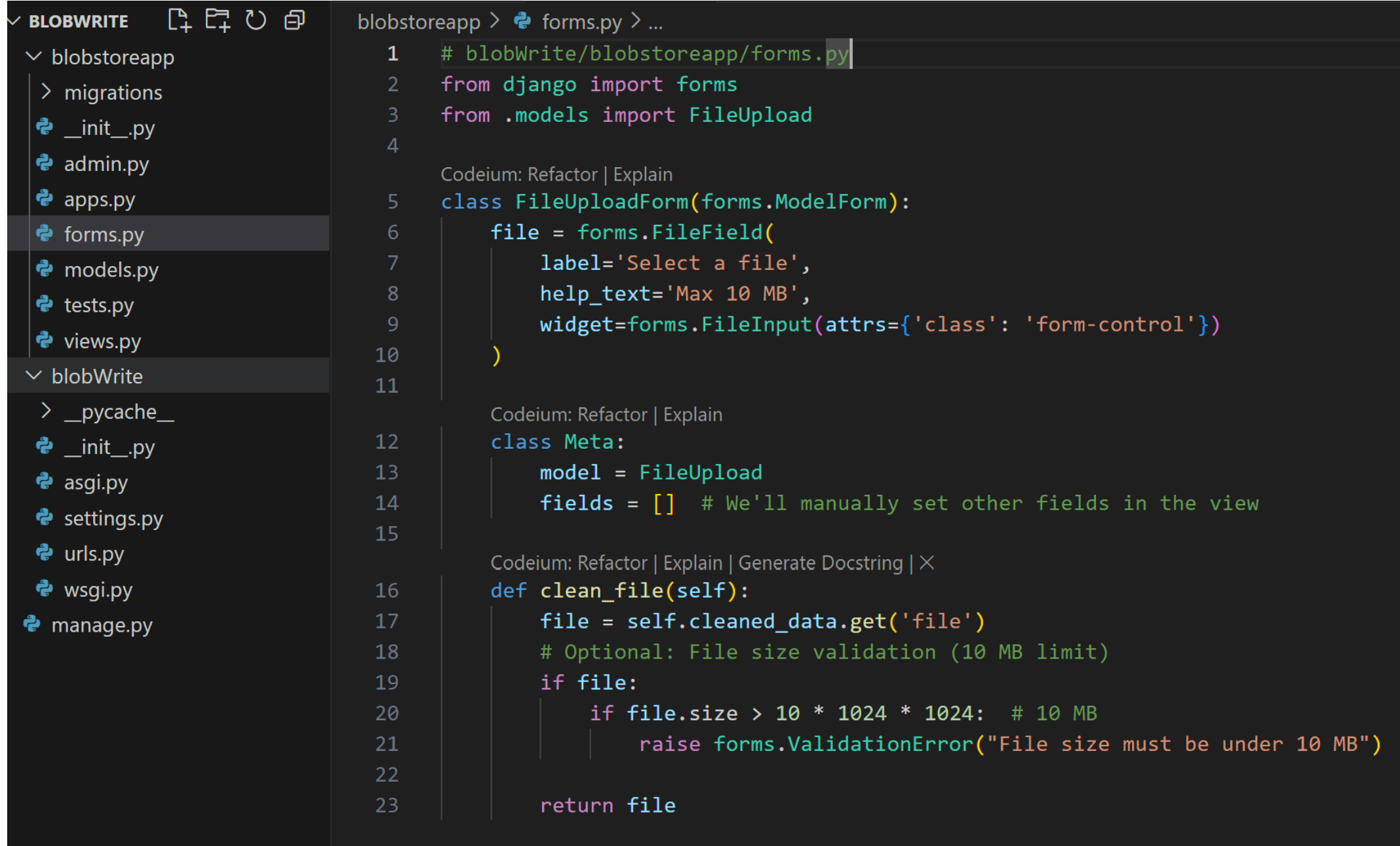


The image shows a code editor interface with a dark theme. On the left is the 'EXPLORER' sidebar showing a file tree. The tree is expanded to show the 'blobWrite' directory, which contains a subdirectory 'blobstoreapp' and several Python files. The 'models.py' file in 'blobstoreapp' is selected and highlighted. The main editor area shows the content of 'models.py'. The code starts with an import from 'django.db' and defines a 'FileUpload' model class. The class has four fields: 'file_name' (CharField), 'azure_url' (URLField), 'file_size' (IntegerField), and 'uploaded_at' (DateTimeField). A string representation method '__str__' is also defined, returning a formatted string with the file name and upload date. The editor includes a breadcrumb 'blobstoreapp > models.py > ...' and a Codeium toolbar with options like 'Refactor', 'Explain', 'Generate Docstring', and a close button.

```
EXPLORER
  ...
  ▼ BLOBWRITE
    ▼ blobstoreapp
      > migrations
      __init__.py
      admin.py
      apps.py
      models.py
      tests.py
      views.py
    ▼ blobWrite
      > __pycache__
      __init__.py
      asgi.py
      settings.py
      urls.py
      wsgi.py
      manage.py

models.py X
blobstoreapp > models.py > ...
1  from django.db import models
2
   Codeium: Refactor | Explain
3  class FileUpload(models.Model):
4      file_name = models.CharField(max_length=255)
5      azure_url = models.URLField(max_length=500)
6      file_size = models.IntegerField()
7      uploaded_at = models.DateTimeField(auto_now_add=True)
8
   Codeium: Refactor | Explain | Generate Docstring | X
9  def __str__(self):
10     return f"{self.file_name} - Uploaded at {self.uploaded_at}"
11
12
13
```

azurestoragedemo/blobWrite/blobstoreapp/forms.py

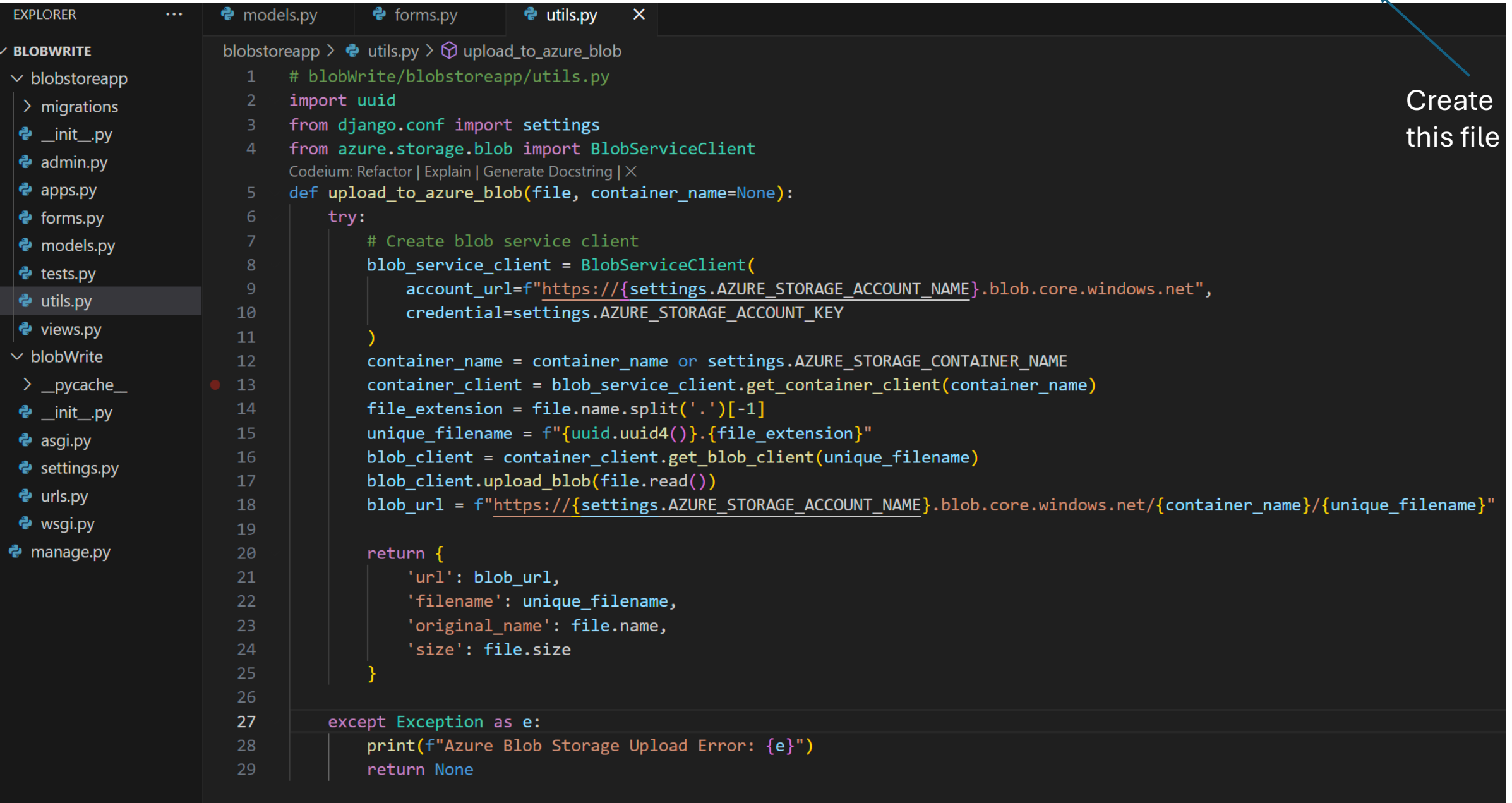


The image shows a code editor interface. On the left is a file explorer with a tree view. The tree is expanded to show the 'blobWrite' directory, which contains a subdirectory '__pycache__' and several Python files: '__init__.py', 'asgi.py', 'settings.py', 'urls.py', 'wsgi.py', and 'manage.py'. The 'forms.py' file is highlighted. The main editor area on the right shows the code for 'forms.py'. The code defines a Django form class 'FileUploadForm' that inherits from 'forms.ModelForm'. It has a single field 'file' of type 'FileField' with a label 'Select a file', a help text 'Max 10 MB', and a widget 'FileInput' with a class attribute 'form-control'. A 'Meta' class is also defined with 'model = FileUpload' and an empty 'fields' list. A 'clean_file' method is defined to validate the file size, ensuring it is under 10 MB, and raising a 'ValidationError' if it is not. The code is as follows:

```
1  # blobWrite/blobstoreapp/forms.py
2  from django import forms
3  from .models import FileUpload
4
5  class FileUploadForm(forms.ModelForm):
6      file = forms.FileField(
7          label='Select a file',
8          help_text='Max 10 MB',
9          widget=forms.FileInput(attrs={'class': 'form-control'})
10     )
11
12     class Meta:
13         model = FileUpload
14         fields = [] # We'll manually set other fields in the view
15
16     def clean_file(self):
17         file = self.cleaned_data.get('file')
18         # Optional: File size validation (10 MB limit)
19         if file:
20             if file.size > 10 * 1024 * 1024: # 10 MB
21                 raise forms.ValidationError("File size must be under 10 MB")
22
23         return file
```

Create
this file

azurestoragedemo/blobWrite/blobstoreapp/Utils.py



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows the project structure, with 'blobstoreapp' expanded. The code editor shows the content of 'utils.py', which defines a function 'upload_to_azure_blob' that uploads a file to Azure Blob Storage. An arrow points from the title 'azurestoragedemo/blobWrite/blobstoreapp/Utils.py' to the file 'utils.py' in the file explorer.

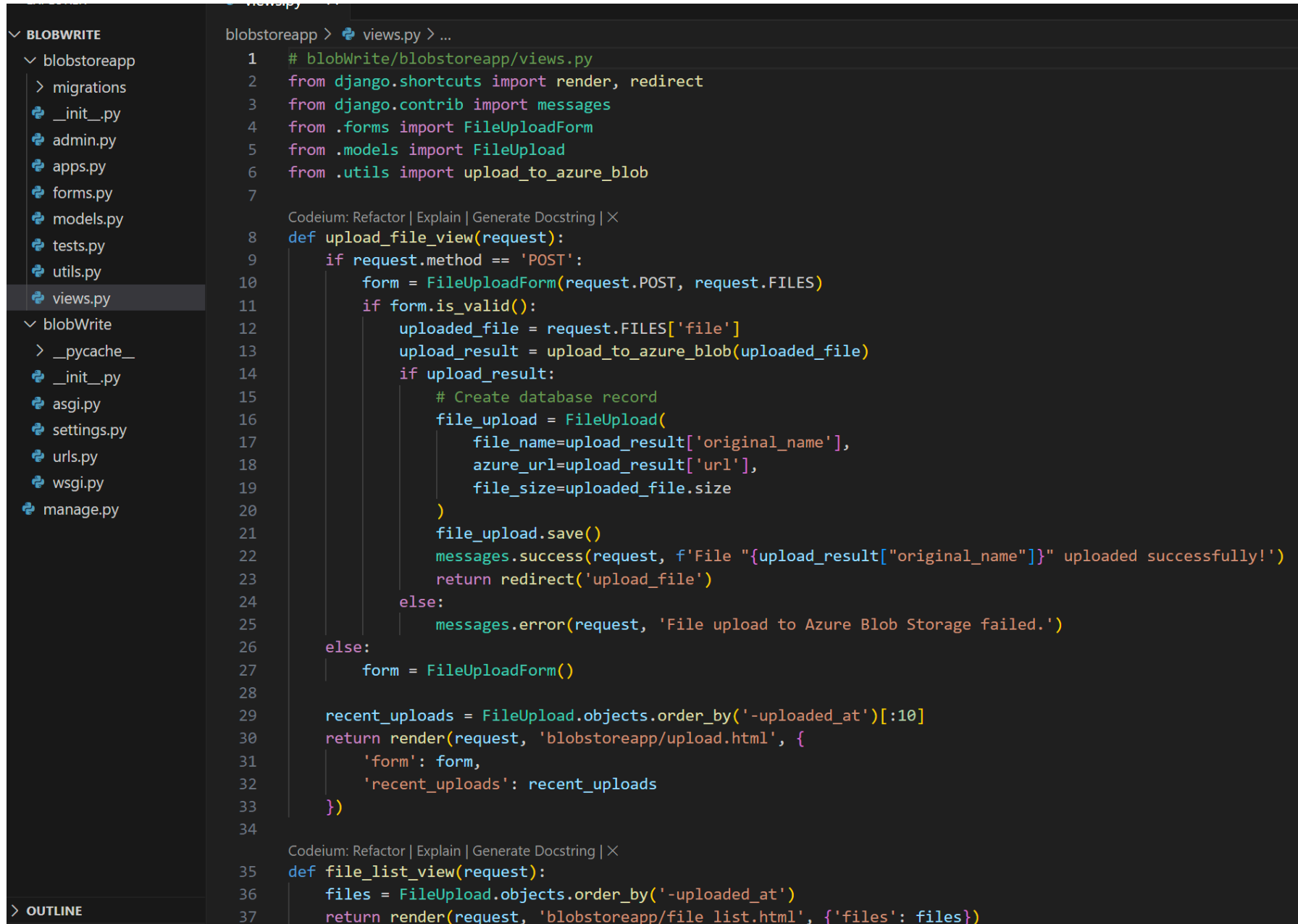
```
EXPLORER
...
models.py
forms.py
utils.py X

BLOBWRITE
  blobstoreapp
    migrations
    __init__.py
    admin.py
    apps.py
    forms.py
    models.py
    tests.py
    utils.py
    views.py
  blobWrite
    __pycache__
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
    manage.py

blobstoreapp > utils.py > upload_to_azure_blob
1 # blobWrite/blobstoreapp/utils.py
2 import uuid
3 from django.conf import settings
4 from azure.storage.blob import BlobServiceClient
5 def upload_to_azure_blob(file, container_name=None):
6     try:
7         # Create blob service client
8         blob_service_client = BlobServiceClient(
9             account_url=f"https://{settings.AZURE_STORAGE_ACCOUNT_NAME}.blob.core.windows.net",
10             credential=settings.AZURE_STORAGE_ACCOUNT_KEY
11         )
12         container_name = container_name or settings.AZURE_STORAGE_CONTAINER_NAME
13         container_client = blob_service_client.get_container_client(container_name)
14         file_extension = file.name.split('.')[-1]
15         unique_filename = f"{uuid.uuid4()}.{file_extension}"
16         blob_client = container_client.get_blob_client(unique_filename)
17         blob_client.upload_blob(file.read())
18         blob_url = f"https://{settings.AZURE_STORAGE_ACCOUNT_NAME}.blob.core.windows.net/{container_name}/{unique_filename}"
19
20         return {
21             'url': blob_url,
22             'filename': unique_filename,
23             'original_name': file.name,
24             'size': file.size
25         }
26
27     except Exception as e:
28         print(f"Azure Blob Storage Upload Error: {e}")
29         return None
```

Create this file

azurestoragedemo/blobWrite/blobstoreapp\views.py



```
blobstoreapp > views.py > ...
1  # blobWrite/blobstoreapp/views.py
2  from django.shortcuts import render, redirect
3  from django.contrib import messages
4  from .forms import FileUploadForm
5  from .models import FileUpload
6  from .utils import upload_to_azure_blob
7
8  def upload_file_view(request):
9      if request.method == 'POST':
10         form = FileUploadForm(request.POST, request.FILES)
11         if form.is_valid():
12             uploaded_file = request.FILES['file']
13             upload_result = upload_to_azure_blob(uploaded_file)
14             if upload_result:
15                 # Create database record
16                 file_upload = FileUpload(
17                     file_name=upload_result['original_name'],
18                     azure_url=upload_result['url'],
19                     file_size=uploaded_file.size
20                 )
21                 file_upload.save()
22                 messages.success(request, f'File "{upload_result["original_name"]}" uploaded successfully!')
23                 return redirect('upload_file')
24             else:
25                 messages.error(request, 'File upload to Azure Blob Storage failed.')
26         else:
27             form = FileUploadForm()
28
29         recent_uploads = FileUpload.objects.order_by('-uploaded_at')[:10]
30         return render(request, 'blobstoreapp/upload.html', {
31             'form': form,
32             'recent_uploads': recent_uploads
33         })
34
35 def file_list_view(request):
36     files = FileUpload.objects.order_by('-uploaded_at')
37     return render(request, 'blobstoreapp/file_list.html', {'files': files})
```


azurestoragedemo\blobWrite\blobstoreapp\urls.py

```

BLOBWRITE
  blobstoreapp
    migrations
    __init__.py
    admin.py
    apps.py
    forms.py
    models.py
    tests.py
    urls.py

blobstoreapp > urls.py > ...
1  # blobWrite/blobstoreapp/urls.py
2  from django.urls import path
3  from .views import upload_file_view, file_list_view
4
5  urlpatterns = [
6      path('', upload_file_view, name='upload_file'),
7      path('files/', file_list_view, name='file_list'),
8  ]
9

```

Create
this file

azurestoragedemo\blobWrite\blobWrite\urls.py

```

BLOB...  [Icons]
  blobstoreapp
  blobWrite
    __pycache__
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py

blobWrite > urls.py > ...
1  # blobWrite/blobWrite/urls.py
2  from django.contrib import admin
3  from django.urls import path, include
4
5  urlpatterns = [
6      path('admin/', admin.site.urls),
7      path('', include('blobstoreapp.urls')),
8  ]

```

Create folders and files for frontend

```
(venv) D:\azurestoragedemo\blobWrite>md blobstoreapp\templates\blobstoreapp
```



```
D:\AZURESTORAGEDEMO\BLOBWRITE\BLOBSTOREAPP
├── migrations
├── templates
│   └── blobstoreapp
```

```
D:\AZURESTORAGEDEMO\BLOBWRITE\BLOBSTOREAPP\TEMPLATES
```

```
├── blobstoreapp
│   ├── file_list.html
│   └── upload.html
```

```
# blobWrite/blobstoreapp/templates/blobstoreapp/file_list.html
<!DOCTYPE html>
<html lang="en">
<head> ...
</head>
<body>
  <h1>All Uploaded Files</h1>

  <table>
    <thead>
      <tr>
        <th>File Name</th>
        <th>Size</th>
        <th>Uploaded At</th>
        <th>Azure URL</th>
      </tr>
    </thead>
    <tbody>
      {% for file in files %}
        <tr>
          <td>{{ file.file_name }}</td>
          <td>{{ file.file_size|filesizeformat }}</td>
          <td>{{ file.uploaded_at|date:"F d, Y H:i" }}</td>
          <td><a href="{{ file.azure_url }}" target="_blank">View</a></td>
        </tr>
      {% empty %}
        <tr>
          <td colspan="4">No files uploaded yet</td>
        </tr>
      {% endfor %}
    </tbody>
  </table>

  <p><a href="{% url 'upload_file' %}">Back to Upload</a></p>
</body>
</html>
```

```
# blobWrite/blobstoreapp/templates/blobstoreapp/upload.html
<!DOCTYPE html>
<html lang="en">
<head> ...
</head>
<body>
  <h1>Upload File to Azure Blob Storage</h1>

  {% if messages %}
    <div class="messages">
      {% for message in messages %}
        <p class="{{ message.tags }}">{{ message }}</p>
      {% endfor %}
    </div>
  {% endif %}

  <form method="post" enctype="multipart/form-data">
    {% csrf_token %}
    {{ form.file }}
    <button type="submit">Upload</button>
  </form>

  <div class="recent-uploads">
    <h2>Recent Uploads</h2>
    <ul>
      {% for upload in recent_uploads %}
        <li>
          <a href="{{ upload.azure_url }}" target="_blank">{{ upload.file_name }}</a>
          ({{ upload.file_size|filesizeformat }}) -
          {{ upload.uploaded_at|date:"F d, Y H:i" }}
        </li>
      {% empty %}
        <li>No recent uploads</li>
      {% endfor %}
    </ul>
  </div>

  <p><a href="{% url 'file_list' %}">View All Files</a></p>
</body>
</html>
```

Create database migrations

```
python manage.py makemigrations blobstoreapppython
manage.py migrate
```

Run the Application

```
python manage.py runserver
```

127.0.0.1:8000

blobWrite/blobstoreapp/templates/blobstoreapp/upload.html

Upload File to Azure Blob Storage

Choose File

No file chosen

Upload

Recent Uploads

- [Pj_Healthishta_K1_Kolkata_MGD_Female_15-05-2023docx.pdf](#) (38.9 KB) - March 25, 2025 11:23

[View All Files](#)

Upload

Change access level

Refresh

Delete

Authentication method: Access key [\(Switch to Microsoft Entra user ac](#)
Location: static

Search blobs by prefix (case-sensitive)

Add filter

Name	Modified
<input type="checkbox"/> <no name>	
<input type="checkbox"/> css	
<input type="checkbox"/> font	
<input type="checkbox"/> image	
<input type="checkbox"/> js	
<input type="checkbox"/> media	
<input type="checkbox"/> pipeline	
<input type="checkbox"/> wordfiles	
<input type="checkbox"/> 58ca184b-3d00-428a-a309-07d6bdc...	3/25/2025, 4:53:16 PI
<input type="checkbox"/> blob	10/14/2024, 5:48:33 .

End