In Django, you can save a base64-encoded image in a `FileField` while also including file size, file type, and other information. To achieve this, you'll need to follow these steps:

1. **Decode Base64 Data**: Decode the base64-encoded image data to its binary form.
2. **Create a Django File**: Create a Django `File` object from the decoded binary data, which can be saved in a `FileField`.
3. **Set Model Attributes**: Set the necessary attributes of your model, such as file size, file type, etc.
4. **Save Model Instance**: Save the model instance with the file data and attributes.

Here's an example of how you could implement this:

Assuming you have a model named `ImageModel` with a `FileField` named `image_file`, and you want to save the file size and file type:

```
import base64
from django.db import models
from django.core.files.base import ContentFile

class ImageModel(models.Model):
    image_file = models.FileField(upload_to='images/')
    file_size = models.PositiveIntegerField()
    file_type = models.CharField(max_length=50)

    @classmethod
    def create_from_base64(cls, base64_data):
        data = base64.b64decode(base64_data)
        file_size = len(data)
        file_type = "image/png"  # Replace with the actual file type

        image = cls(image_file=None, file_size=file_size,
file_type=file_type)
        image.image_file.save('image.png', ContentFile(data))
        return image
```

In this example, the `create_from_base64` method decodes the base64 data, calculates the file size, sets the file type (you need to determine the correct file type), and then saves the `File` object into the `image_file` field using the `save` method.

Usage:

```
base64_data = "..."  # Your base64-encoded image data here
image = ImageModel.create_from_base64(base64_data)
image.save()
```

Make sure to replace `'images/'` with the desired upload path for your `FileField`. Also, ensure that you have properly configured your Django settings to handle file uploads.

Please note that this example is simplified and may need further adjustments depending on your specific requirements and configurations. Additionally, consider error handling and validation for production use.