Cheat Sheet for comprehensive Django

Project Setup

Creating a New Project

```
django-admin startproject projectname
```

Running the Development Server

```
python manage.py runserver
```

Creating an App

```
python manage.py startapp appname
```

Migrations

- **Creating Migrations:**

```
python manage.py makemigrations
```

- **Applying Migrations:**

```
python manage.py migrate
```

Models

Defining a Model

```
from django.db import models

class MyModel(models.Model):
    name = models.CharField(max_length=100)
    description = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)
```

Common Field Types

- `CharField(max_length=100)`
- `TextField()`
- `IntegerField()`

- `FloatField()`
- `BooleanField()`
- `DateTimeField(auto_now_add=True)`
- `ForeignKey(OtherModel, on_delete=models.CASCADE)`
- `ManyToManyField(OtherModel)`

Model Methods

- **String Representation:**

```
  def __str__(self):
      return self.name
```

- **Custom Query Methods:**

```
  @classmethod
  def get_popular_items(cls):
      return cls.objects.filter(views__gt=100)
```

Views

Function-Based Views

```
from django.http import HttpResponse

def my_view(request):
    return HttpResponse("Hello, World!")
```

Class-Based Views

```
from django.views.generic import ListView, DetailView
from .models import MyModel

class MyModelListView(ListView):
    model = MyModel
    template_name = 'mymodel_list.html'

class MyModelDetailView(DetailView):
    model = MyModel
    template_name = 'mymodel_detail.html'
```

Handling Forms

```
from django.shortcuts import render, redirect
from .forms import MyForm
```

```
def my_form_view(request):
    if request.method == 'POST':
        form = MyForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('success_url')
    else:
        form = MyForm()
    return render(request, 'my_form.html', {'form': form})
```

URLs

Basic URL Configuration

```
from django.urls import path
from . import views

urlpatterns = [
    path('mymodel/', views.MyModelListView.as_view(),
name='mymodel_list'),
    path('mymodel/<int:pk>/', views.MyModelDetailView.as_view(),
name='mymodel_detail'),
]
```

Including Other URL Configs

```
from django.urls import include, path

urlpatterns = [
    path('app/', include('appname.urls')),
]
```

Templates

Basic Template Structure

```
<!DOCTYPE html>
<html>
<head>
    <title>My Page</title>
</head>
<body>
    <h1>{{ title }}</h1>
    <ul>
    {% for item in items %}
        <li>{{ item.name }}</li>
```

```
    {% endfor %}
    </ul>
</body>
</html>
```

Template Tags and Filters

- **Tags:**

- `{% if condition %} ... {% endif %}`
- `{% for item in items %} ... {% endfor %}`
- `{% url 'view_name' arg1 arg2 %}`

- **Filters:**

- `{{ value|default:"nothing" }}`
- `{{ value|length }}`
- `{{ value|date:"Y-m-d" }}`

Forms

Defining a Form

```
from django import forms
from .models import MyModel

class MyForm(forms.ModelForm):
    class Meta:
        model = MyModel
        fields = ['name', 'description']
```

Form Fields

- `forms.CharField()`
- `forms.EmailField()`
- `forms.IntegerField()`
- `forms.ChoiceField(choices=CHOICES)`
- `forms.ModelChoiceField(queryset=MyModel.objects.all())`

Admin Interface

Registering Models

```
from django.contrib import admin
from .models import MyModel

admin.site.register(MyModel)
```

Customizing Admin

```
class MyModelAdmin(admin.ModelAdmin):
    list_display = ('name', 'description', 'created_at')
    search_fields = ('name',)
    list_filter = ('created_at',)

admin.site.register(MyModel, MyModelAdmin)
```

Authentication

User Authentication

```
from django.contrib.auth import authenticate, login, logout

def login_view(request):
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(request, username=username,
password=password)
        if user is not None:
            login(request, user)
            return redirect('home')
        else:
            # Handle invalid login
            pass
    return render(request, 'login.html')
```

User Logout

```
def logout_view(request):
    logout(request)
    return redirect('home')
```

Middleware

Custom Middleware

```
class MyMiddleware:
    def __init__(self, get_response):
        self.get_response = get_response

    def __call__(self, request):
        # Code to be executed before the view
        response = self.get_response(request)
```

```
        # Code to be executed after the view
        return response
```

Signals

Defining a Signal

```
from django.db.models.signals import post_save
from django.dispatch import receiver
from .models import MyModel

@receiver(post_save, sender=MyModel)
def my_model_post_save(sender, instance, created, **kwargs):
    if created:
        # Do something when a new instance is created
        pass
```

Testing

Writing Tests

```
from django.test import TestCase
from .models import MyModel

class MyModelTests(TestCase):
    def setUp(self):
        MyModel.objects.create(name="Test", description="Test
Description")

    def test_model_creation(self):
        obj = MyModel.objects.get(name="Test")
        self.assertEqual(obj.description, "Test Description")
```

Running Tests

```
python manage.py test
```

Deployment

Collecting Static Files

```
python manage.py collectstatic
```

Production Settings

- **Settings File:**

```
DEBUG = False
ALLOWED_HOSTS = ['yourdomain.com']
```

- **Database Configuration:**

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'mydatabase',
        'USER': 'mydatabaseuser',
        'PASSWORD': 'mypassword',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Tips and Tricks

Debugging

- **Print Statements:**

```
print(variable)
```

- **Django Debug Toolbar:**

```
pip install django-debug-toolbar
```

Performance Optimization

- **Caching:**

```
from django.core.cache import cache

cache.set('my_key', 'my_value', 300)
value = cache.get('my_key')
```

- **Database Indexing:**

```
class MyModel(models.Model):
    name = models.CharField(max_length=100, db_index=True)
```

Security Best Practices

- **Use `django.contrib.auth` for authentication.**
- **Always use `django.utils.crypto` for secure hashing.**
- **Keep Django and all dependencies up to date.**

Useful Commands

Shell

```
python manage.py shell
```

Creating a Superuser

```
python manage.py createsuperuser
```

Checking Project Status

```
python manage.py check
```

Additional Resources

- **Django Documentation:** docs.djangoproject.com
- **Django Packages:** djangopackages.org
- **Django Source Code:**
github.com/django/django

By Ahmed Baheeg Khorshid

ver 1.0