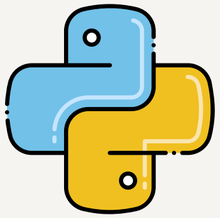
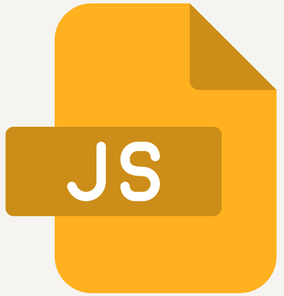




STRAIGHT TO THE POINT WITH

django

Presented By

SUETONE CARNEIRO

WHAT WILL WE **LEARN TODAY?**

INTRODUCTION
Basic architecture of a Django project

PREPARING THE ENVIRONMENT AND OUR FIRST PROJECT
Virtual environment
django admin startproject ***project_name***

MODULARIZATION - CREATING DJANGO APPS
python3 manage.py startapp ***app_name***

DATA LAYER
Models and migrations

DJANGO ADMIN

STEP ZERO
INTRODUCTION

django

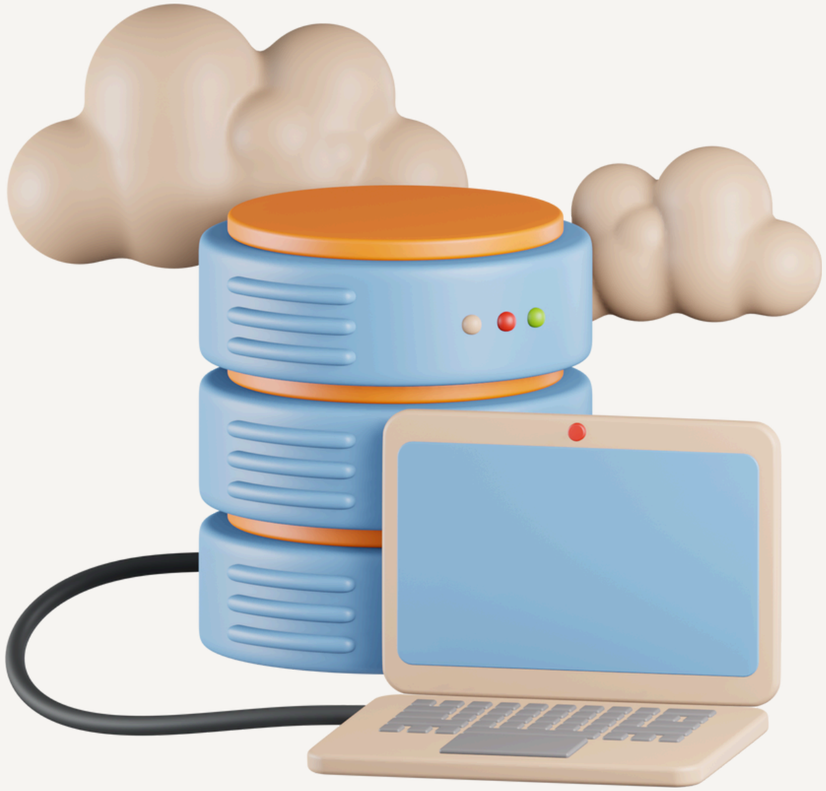
Model → DataBase issues

View → Request handler/ Validations

Template → FrontEnd issues

Model

django



SQL ❌

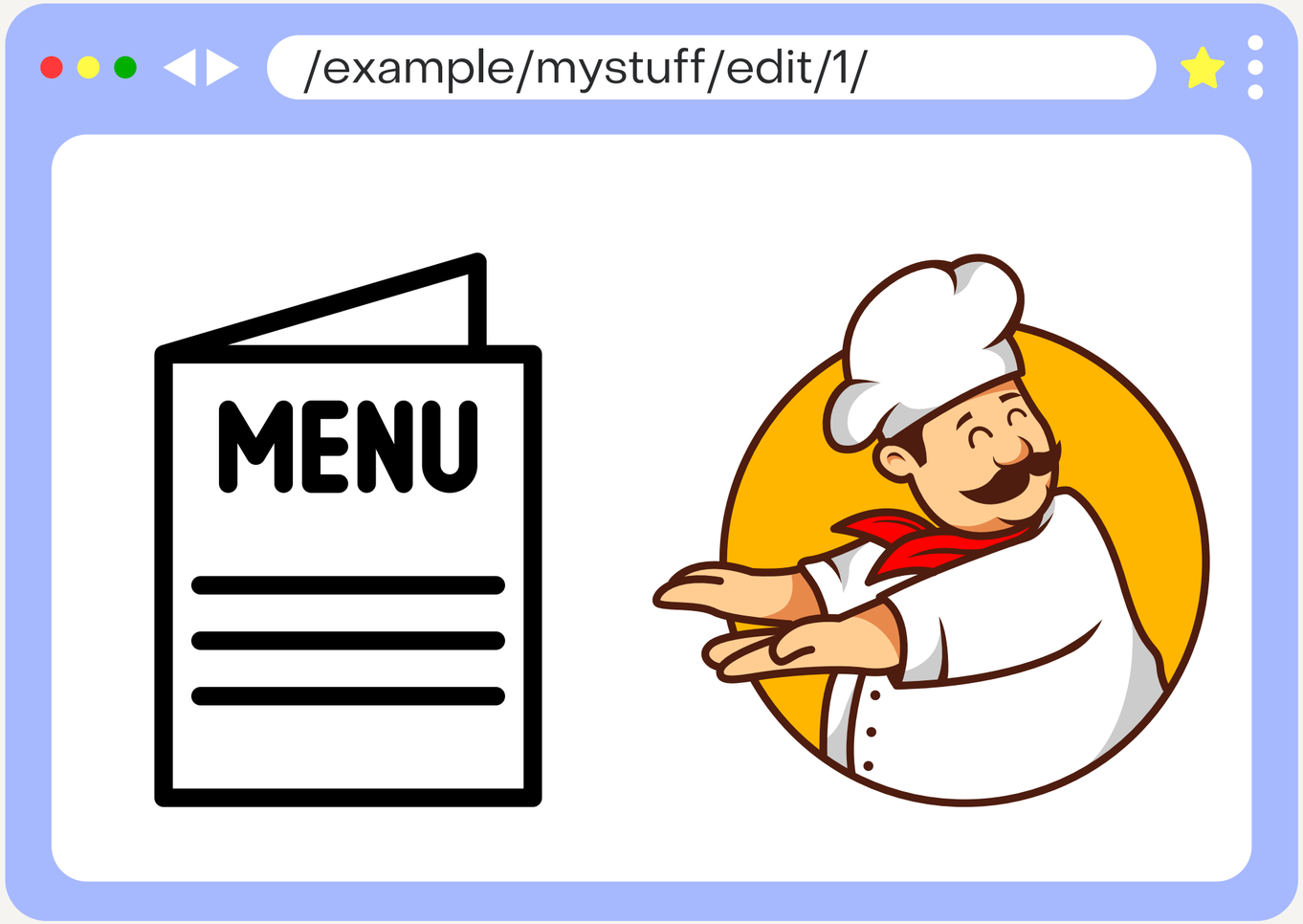
ORM ✅

(Object-Relational Mapping)

View

Deals with requests.
Organizes routes(URLs)

django



+ Some kind of validation, if needed

Template

django

PREPARING THE **ENVIRONMENT**

```
code@home:~$ python3 -m venv .venv

code@home:~$ source .venv/bin/activate

code@home:~$ pip install django

code@home:~$ django-admin startproject [...]
                                        name it

code@home:~$ python3 manage.py runserver
```

Reminder: add a directory to keep static files :)

Straight to the point with Django

TALKING ABOUT **MODULARIZATION**

HOW TO KEEP THINGS **ORGANIZED?**

django

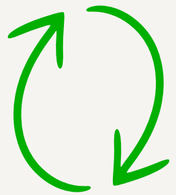
> Django comes with a utility that automatically generates the basic directory structure of an app, so you can focus on writing code rather than creating directories.

>> Django Documentation

HOW TO KEEP THINGS **ORGANIZED?**

django

- Step 1: python3 manage.py startapp *app_name*
- *Step* 2: create file for APP URLs (urls.py)
- Step 3: create a view
- Step 4: create a URL to this view
- Step 5: ‘notify’ the project that we have created new routes and a new app (settings.py -> installed apps)
- Step 6: work, work, work...



*After configuring the app

Reminder: add a directory to keep your templates :)

templates/app_name

DJANGO
CLASS BASED VIEWS

app_name/views.py

```
from django.views.generic import TemplateView

class IndexView(TemplateView):
        template_name = 'app_name/index.html'
```

AFTER CREATING A VIEW...
SET YOUR ROUTES!

app_name/urls.py

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

urlpatterns = [
        path(‘ ’, views.IndexView.as_view(
                        template_name=’app_name/index.html’
                ), name="index"),
]
```

ALSO CONFIGURE ON **PROJECT URLS!**

*Only required when creating the app

project_name/urls.py

```
from django.contrib import admin
from django.urls import include, path


urlpatterns = [
        path('admin/', admin.site.urls),
        path(' ', include('app_name.urls')),
]
```

ALSO CONFIGURE ON **PROJECT SETTINGS!**

*Only required when creating the app

django

project_name/settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.messages',
    'django.contrib.staticfiles',
     'app_name.apps.App_nameConfig',
]
```

FINALLY, OUR
TEMPLATE :)

django

app_name/templates/index.html

HTML

Straight to the point with Django

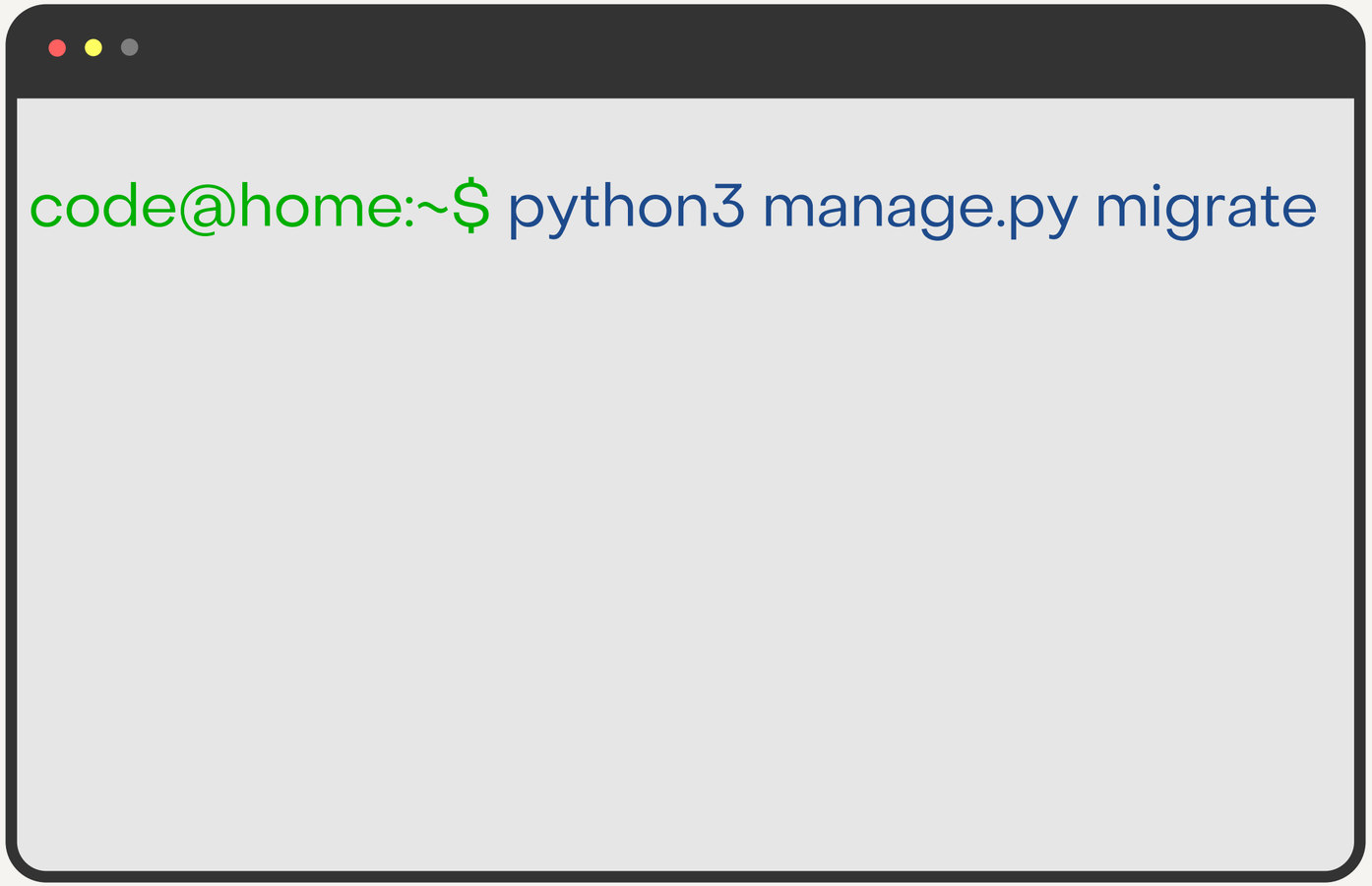
DATA LAYER

DATA LAYER

django

Shall we do the first test?

This action will create the tables that come with Django by default.



WHAT HAPPENS WHEN I CREATE **MY OWN TABLES?**

django

```
code@home:~$ python3 manage.py makemigrations

code@home:~$ python3 manage.py migrate
```

-> With these two commands, Django will create your tables in the database :)

Let’s see how it works 

WHAT HAPPENS WHEN I CREATE **MY OWN TABLES?**

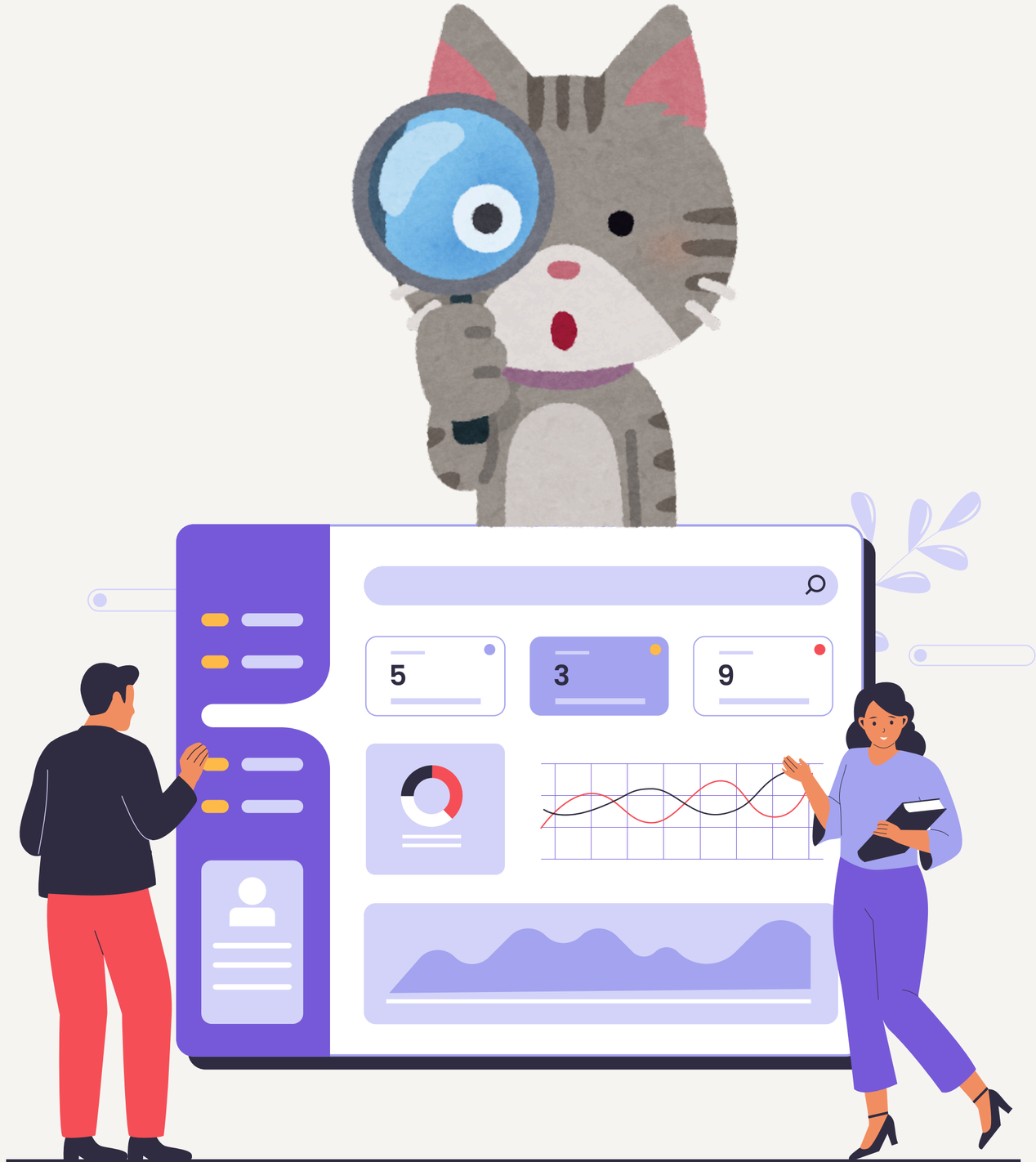
```
from django.db import models
from django.contrib.auth.models import User


class Usuario(models.Model):
    usuario = models.OneToOneField(User, on_delete=models.CASCADE)
    email = models.EmailField(unique=True, null=False)
    nome_completo = models.CharField(
        max_length=255, verbose_name='Nome Completo')
    endereco = models.CharField(max_length=255, verbose_name='Endereço')
    telefone = models.CharField(max_length=22, verbose_name='Telefone')

    def __str__(self):
        return f'{self.nome_completo} - {self.email}'
```

Straight to the point with Django

DJANGO ADMIN



DJANGO ADMIN

Django has a built-in administrative panel that can be customized (wow).

Okay. But... how to access it?
There is a route that we have probably already seen...

The route is: **admin/**

DJANGO **ADMIN**

Creating a super user

Let’s see the documentation :)

Creating an admin user ¶

First we'll need to create a user who can login to the admin site. Run the following command:

```
$ python manage.py createsuperuser
```

Enter your desired username and press enter.

```
Username: admin
```

You will then be prompted for your desired email address:

```
Email address: admin@example.com
```

The final step is to enter your password. You will be asked to enter your password twice, the second time as a confirmation of the first.

```
Password: **********
Password (again): *********
Superuser created successfully.
```

STRAIGHT TO THE POINT WITH

django

Presented By
SUETONE CARNEIRO