Níže najdeš podrobný přehled projektu „Webové stránky“, vysvětlený krok za krokem pro někoho, kdo se s ním setkává poprvé.

## 1) Struktura projektu

webove\_stranky/ # Kořen repozitáře  
template: # Nastavení Django šablon  
db.sqlite3 # SQLite databáze (testovací)  
manage.py # Spouštěcí skript Django  
django\_project/ # Složka s hlavními nastaveními (settings, urls, asgi, wsgi)  
website/ # Aplikace 'website', logika webu  
├─ admin.py # Správa modelů v Django admin (zatím prázdné)  
├─ apps.py # Registrace aplikace v projektu  
├─ models.py # Definice databázových modelů (zatím prázdné)  
├─ views.py # Funkce, které reagují na HTTP požadavky  
group: migrations/ # Automaticky generované migrace databáze  
├─ urls.py # URL směrování specifické pro appku  
├─ forms.py # Definice HTML formulářů a validace  
└─ tests.py # Testy pro aplikaci (zatím prázdné)  
static/ # CSS, JS, obrázky pro front-end  
templates/website/ # HTML šablony pro jednotlivé stránky

## 2) Nastavení projektu (settings.py)

* **BASE\_DIR**: kořenová složka, odkud Django hledá templates/, static/ a další
* **INSTALLED\_APPS**: seznam nainstalovaných aplikací (zde: website + defaultní Django moduly)
* **MIDDLEWARE**: vrstva mezi requestem a view, např. bezpečnostní hlavičky, správa session
* **TEMPLATES**: cesta k HTML šablonám (templates/website/)
* **DATABASES**: SQLite databáze (db.sqlite3), jednoduché pro vývoj
* **STATICFILES\_DIRS**: cesta k static/, kde jsou CSS, JS, obrázky
* **SECRET\_KEY**: tajný klíč pro šifrování; v produkci by měl být v proměnné prostředí

## 3) Směrování (URL routing)

### a) Kořenové urls.py

from django.contrib import admin  
from django.urls import path, include  
  
urlpatterns = [  
 path('admin/', admin.site.urls), # Django admin  
 path('', include('website.urls')), # všechny ostatní URL směruj do appky  
]

### b) website/urls.py

from django.urls import path  
from . import views  
  
urlpatterns = [  
 path('', views.home, name='home'), # úvodní stránka  
 path('o-nas/', views.o\_nas, name='o\_nas'), # stránka "O nás"  
 path('jobs/', views.jobs\_list, name='jobs'), # seznam pracovních pozic  
 path('kontakt/', views.poptat\_dopravu, name='kontakt'),  
]

Každá cesta (path) volá příslušnou funkci ve views.py a má jméno (name=) pro reverzní generování URL.

## 4) View funkce (views.py)

views.py obsahuje HTTP kontrolery:

from django.shortcuts import render  
from .forms import PoptavkaForm  
  
# 1) Úvodní stránka  
def home(request):  
 return render(request, 'website/home.html')  
  
# 2) O nás  
def o\_nas(request):  
 return render(request, 'website/o\_nas.html')  
  
# 3) Seznam pozic  
def jobs\_list(request):  
 jobs = [ # statický seznam dictů, ve skutečnosti by přišel z DB  
 {'title': 'Ekonomický analytik', 'location': 'Praha'},  
 {'title': 'Datový analytik', 'location': 'Brno'},  
 ]  
 return render(request, 'website/jobs\_list.html', {'jobs': jobs})  
  
# 4) Kontaktní formulář  
def poptat\_dopravu(request):  
 if request.method == 'POST':  
 form = PoptavkaForm(request.POST)  
 if form.is\_valid():  
 # TODO: uložit do DB / poslat e-mail  
 return render(request, 'website/diky.html', {'data': form.cleaned\_data})  
 else:  
 form = PoptavkaForm()  
 return render(request, 'website/kontakt.html', {'form': form})

* render(request, 'šablona.html', context): vykreslí HTML šablonu s daty context.
* Formulář (PoptavkaForm) se zpracuje metodou POST, validuje se a při úspěchu zobrazí stránka “Díky”.

## 5) Formulář (forms.py)

from django import forms  
  
class PoptavkaForm(forms.Form):  
 name = forms.CharField(label='Jméno', max\_length=100)  
 email = forms.EmailField(label='E-mail')  
 message = forms.CharField(label='Zpráva', widget=forms.Textarea)

* **CharField**: textové pole s omezením délky
* **EmailField**: kontroluje, že to je validní e-mail
* **Textarea**: vícenásobný řádek pro zprávu
* Při form.is\_valid() Django zkontroluje povinná pole a typy

## 6) Šablony (templates/website/)

* home.html, o\_nas.html, jobs\_list.html, kontakt.html, diky.html
* Každá šablona dědí z base.html (hlavička, patička, odkazy na CSS/JS)
* Vkládají se bloky jako {% block content %}
* V jobs\_list.html se iteruje:
* {% for job in jobs %}  
   <div>{{ job.title }} – {{ job.location }}</div>  
  {% endfor %}

## 7) Statické soubory (static/)

* **CSS**: vzhled stránek
* **JS**: případné interaktivní skripty (aktuálně prázdné)
* **img**: obrázky (logo apod.)

Django je servíruje v debug režimu automaticky.

## 8) Migrace (migrations/)

* Obsahuje historii změn modelů (zatím žádné modely)
* Při přidání modelů se tu vytvoří soubory 0001\_initial.py atd.

### Co dál?

* **Úpravy view**: přidat skutečnou logiku (např. číst/zapisovat do DB)
* **Modely**: definovat website/models.py třídy pro uložení poptávek či jobů
* **Admin**: zaregistrovat modely v admin.py, správa přes Django Admin
* **Testy**: doplnit tests.py o jednotkové testy view a formulářů
* **Deployment**: připravit nastavení pro produkci (ENV, DEBUG=False)

## 9) Lokalizace a ladění CSS stylů

Když na stránce vidíš, že něco nefunguje nebo se zobrazuje jinak, než očekáváš, použij nástroje pro vývojáře ve svém prohlížeči (např. Chrome DevTools nebo Firefox Developer Tools):

1. **Otevři Inspektor**
   * Klikni pravým tlačítkem na prvek, který chceš zkoumat, a zvol „Inspect”/„Prozkoumat“.
   * V levé části se zvýrazní HTML kód; prvek bude obalený v tagu, např. <div class="header">.
2. **Zkoumej CSS pravidla**
   * V pravé části (panel Styles) uvidíš seznam CSS selektorů a vlastností, které se na prvek vztahují.
   * Pořadí pravidel odpovídá kaskádě: od nejvyšší priority nahoře po nižší dole. Přepsaná vlastnost (overridden) je přeškrtnutá.
   * Vedle každé vlastnosti je odkaz (filename:line), kde je styl definovaný, např. styles.css:45. Kliknutím se dostaneš přímo do zdrojového souboru.
3. **Uprav styly na živo**
   * V panelu Styles klikni na hodnotu či selektor a změň ji – uvidíš okamžitý efekt na stránce.
   * Můžeš přidat nové pravidlo nebo zakomentovat existující vlastnost.
4. **Najdi, odkud styl přichází**
   * Pokud je pravidlo složené (např. .header .title), zkontroluj strukturu HTML v horní části inspektoru.
   * Můžeš vyhledávat v CSS souborech: v DevTools v panelu Sources (Zdrojové soubory) hledej název třídy nebo ID.
5. **Vyzkoušej změnu v kódu**
   * Jakmile najdeš správné pravidlo a úpravy fungují, otevři lokálně soubor v static/css/ a aplikuj stejné změny.
   * Ulož a znovu načti stránku (Ctrl+R) pro ověření finální podoby.

Tip: Pokud používáš více CSS souborů nebo framework (např. Bootstrap), může se pravidlo ukrývat ve více místech. Klíčové je sledovat pořadí načítání (<link> v base.html) a prioritu selektorů.

S tímto postupem rychle zjistíš, které CSS styly ovlivňují konkrétní prvek, a kde je najít v projektu.

* **Úpravy view**: přidat skutečnou logiku (např. číst/zapisovat do DB)
* **Modely**: definovat website/models.py třídy pro uložení poptávek či jobů
* **Admin**: zaregistrovat modely v admin.py, správa přes Django Admin
* **Testy**: doplnit tests.py o jednotkové testy view a formulářů
* **Deployment**: připravit nastavení pro produkci (ENV, DEBUG=False)