Mise en œuvre du projet de gestion des étudiants

Pour mettre en œuvre le projet de gestion des étudiants, nous suivrons une série d'étapes structurées, allant de la configuration initiale à la mise en production. Voici les éléments essentiels pour chaque étape du projet.

# 1. Planification et Préparation

- \*\*Définir les besoins et les spécifications\*\* : Clarifier les fonctionnalités nécessaires et les exigences du projet avec toutes les parties prenantes.

- \*\*Choisir les technologies\*\* :

- Backend : Django (Python)

- Base de données : MySQL

- Frontend : HTML, CSS, Bootstrap 5

- Contrôle de version : Git

- \*\*Configurer l'environnement de développement\*\* :

- Installer Python et Django

- Installer MySQL

- Configurer un dépôt Git pour le projet

# 2. Configuration du Projet Django

- \*\*Créer le projet Django\*\* :

```bash

django-admin startproject student\_management\_project

```

- \*\*Créer l'application Django\*\* :

```bash

cd student\_management\_project

python manage.py startapp student\_management\_application

```

- \*\*Configurer la base de données MySQL dans `settings.py`\*\* :

```python

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'student\_management\_db',

'USER': 'your\_mysql\_user',

'PASSWORD': 'your\_mysql\_password',

'HOST': 'localhost',

'PORT': '3306',

}

}

```

# 3. Définition des Modèles de Données

- \*\*Modèles dans `models.py`\*\* :

```python

from django.db import models

class Student(models.Model):

name = models.CharField(max\_length=100)

age = models.IntegerField()

sex = models.CharField(max\_length=10)

class Teacher(models.Model):

name = models.CharField(max\_length=100)

subject = models.CharField(max\_length=100)

class Course(models.Model):

name = models.CharField(max\_length=100)

teacher = models.ForeignKey(Teacher, on\_delete=models.CASCADE)

students = models.ManyToManyField(Student)

class Score(models.Model):

student = models.ForeignKey(Student, on\_delete=models.CASCADE)

course = models.ForeignKey(Course, on\_delete=models.CASCADE)

score = models.IntegerField()

```

# 4. Migration et Création des Tables

- \*\*Effectuer les migrations\*\* :

```bash

python manage.py makemigrations

python manage.py migrate

```

# 5. Insertion de Données Initiales

- \*\*Créer un fichier de script pour insérer les données\*\* ou utiliser l'interface d'administration Django :

```python

from student\_management\_application.models import Student, Teacher, Course, Score

teacher1 = Teacher.objects.create(name='John Doe', subject='Mathematics')

student1 = Student.objects.create(name='Alice Smith', age=20, sex='F')

course1 = Course.objects.create(name='Algebra', teacher=teacher1)

course1.students.add(student1)

Score.objects.create(student=student1, course=course1, score=95)

```

# 6. Développement des Vues et Templates

- \*\*Vues dans `views.py`\*\* :

```python

from django.shortcuts import render

from .models import Student, Teacher, Course, Score

def student\_list(request):

students = Student.objects.all()

return render(request, 'student\_management\_application/student\_list.html', {'students': students})

def course\_list(request):

courses = Course.objects.all()

return render(request, 'student\_management\_application/course\_list.html', {'courses': courses})

```

- \*\*Templates HTML\*\* :

- `student\_list.html` :

```html

{% extends 'base.html' %}

{% block content %}

<h2>Liste des étudiants</h2>

<ul>

{% for student in students %}

<li>{{ student.name }} - {{ student.age }} ans</li>

{% endfor %}

</ul>

{% endblock %}

```

# 7. Création des Formulaires

- \*\*Formulaires dans `forms.py`\*\* :

```python

from django import forms

from .models import Student, Course, Score

class StudentForm(forms.ModelForm):

class Meta:

model = Student

fields = ['name', 'age', 'sex']

```

- \*\*Vue pour ajouter un étudiant\*\* :

```python

from django.shortcuts import render, redirect

from .forms import StudentForm

def add\_student(request):

if request.method == 'POST':

form = StudentForm(request.POST)

if form.is\_valid():

form.save()

return redirect('student\_list')

else:

form = StudentForm()

return render(request, 'student\_management\_application/add\_student.html', {'form': form})

```

- \*\*Template pour le formulaire\*\* :

- `add\_student.html` :

```html

{% extends 'base.html' %}

{% block content %}

<h2>Ajouter un étudiant</h2>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit">Ajouter</button>

</form>

{% endblock %}

```

# 8. Interface Utilisateur et Design

- \*\*Créer un template principal `base.html`\*\* :

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/5.0.0-alpha1/css/bootstrap.min.css">

<title>Gestion des Étudiants</title>

</head>

<body>

<nav class="navbar navbar-expand-lg navbar-light bg-light">

<a class="navbar-brand" href="#">Gestion des Étudiants</a>

<div class="collapse navbar-collapse">

<ul class="navbar-nav mr-auto">

<li class="nav-item"><a class="nav-link" href="{% url 'student\_list' %}">Étudiants</a></li>

<li class="nav-item"><a class="nav-link" href="{% url 'course\_list' %}">Cours</a></li>

</ul>

</div>

</nav>

<div class="container">

{% block content %}

{% endblock %}

</div>

</body>

</html>

```

# 9. Tests et Débogage

- \*\*Effectuer des tests unitaires\*\* pour vérifier la validité des modèles et des vues.

- \*\*Utiliser les outils de débogage\*\* fournis par Django pour identifier et corriger les erreurs.

# 10. Déploiement

- \*\*Configurer un serveur web\*\* (par exemple, Apache ou Nginx) pour servir l'application Django.

- \*\*Configurer le serveur MySQL\*\* pour être accessible par l'application.

- \*\*Déployer le code sur un serveur de production\*\* et configurer les variables d'environnement nécessaires.

# 11. Documentation et Formation

- \*\*Rédiger la documentation utilisateur\*\* pour expliquer comment utiliser l'application.

- \*\*Former les utilisateurs finaux\*\* à l'utilisation de l'application pour assurer une adoption réussie.

En suivant ces étapes, nous mettrons en œuvre une application de gestion des étudiants robuste et efficace, prête à être utilisée dans un environnement académique.