LAVEILLE
MATHIS

DOCUMENTATION TECHNIQUE STAGE-DJANGO

NOTRE DAME DE LA
PROVIDENCE AVRANCHES

📘 Documentation Technique – Stage-Django

1. Introduction

Le projet **Stage-Django** est une application web développée avec le framework Django. Elle permet au service informatique de l'entreprise STURNO de gérer efficacement les informations relatives aux stages : entreprises partenaires, stagiaires, offres de stage, etc. Ce document détaille l'architecture technique, les choix technologiques, les modèles de données et les fonctionnalités principales de l'application.

---

2. Architecture Technique

Technologies utilisées

- **Backend** : Django 5.2.1 (Python 3.12)
- **Base de données** : SQLite (en développement)
- **Frontend** :
    - Django Templates
    - Bootstrap 3.3.7
    - Font Awesome 6.0.0
- **Modules Django** :
    - `django-tables2` : gestion des tableaux de données
    - `django-cors-headers` : gestion des CORS
    - `asgiref` ≥ 3.8.1
    - `sqlparse` ≥ 0.3.1

Organisation du projet

arduino

```
Stage-Django/
├── django-web-app/
│   ├── app/
│   │   ├── home/
│   │   │   ├── templates/
│   │   │   ├── models.py
│   │   │   ├── views.py
│   │   │   ├── forms.py
│   │   │   └── tables.py
│   │   ├── locale/
│   │   ├── static/
│   │   └── manage.py
└── env/
```

Légende des fichiers principaux :

- `models.py` : définition des modèles de données (entreprises, stagiaires, offres)
- `views.py` : logique des vues (affichage, traitement des requêtes)
- `forms.py` : gestion des formulaires utilisateur
- `tables.py` : configuration des tableaux de données avec `django-tables2`
- `templates/` : fichiers HTML pour le rendu des pages
- `static/` : fichiers statiques (CSS, JS, images)

---

3. Modèles de Données

Modèles principaux

```
# models.py

class Entreprise(models.Model):
    nom = models.CharField(max_length=100)
    adresse = models.CharField(max_length=255)
    secteur = models.CharField(max_length=100)
    telephone = models.CharField(max_length=20)
    email = models.EmailField()

class Stagiaire(models.Model):
    nom = models.CharField(max_length=100)
    prenom = models.CharField(max_length=100)
    formation = models.CharField(max_length=100)
    entreprise = models.ForeignKey(Entreprise,
on_delete=models.CASCADE)

class OffreStage(models.Model):
    intitulé = models.CharField(max_length=100)
    description = models.TextField()
    entreprise = models.ForeignKey(Entreprise,
on_delete=models.CASCADE)
```

Schéma du Modèle Conceptuel de Données (MCD)

---

4. Routes et Vues

Endpoints principaux

```
# urls.py

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.accueil, name='accueil'),
    path('entreprises/', views.entreprise_list,
name='entreprise_list'),
    path('stagiaires/', views.stagiaire_list,
name='stagiaire_list'),
    path('offres/', views.offre_list, name='offre_list'),
]
```

Vues associées

```
# views.py

def entreprise_list(request):
    entreprises = Entreprise.objects.all()
    return render(request, 'entreprises/list.html', {'entreprises':
entreprises})

def stagiaire_list(request):
    stagiaires = Stagiaire.objects.all()
    return render(request, 'stagiaires/list.html', {'stagiaires':
stagiaires})

def offre_list(request):
    offres = OffreStage.objects.all()
    return render(request, 'offres/list.html', {'offres': offres})
```

---

5. Fonctionnalités Avancées

Templates et Vues

- **Templates de base** :
  - `base.html` : template principal
  - `entreprises/list.html` : liste des entreprises
  - `stagiaires/list.html` : liste des stagiaires
  - `offres/list.html` : liste des offres de stage

Internationalisation

- Support multilingue (français par défaut)
- Fichiers de traduction dans le répertoire `locale/`

Exportation des données

- Possibilité d'exporter les données au format PDF et Excel
- Filtrage et tri des données via `django-tables2`

---

6. Sécurité et Authentification

- Utilisation du système d'authentification intégré de Django
- Accès restreint aux pages de gestion des données aux utilisateurs authentifiés
- Protection CSRF activée pour tous les formulaires

---

7. Tests et Validation

- Tests manuels effectués pour chaque fonctionnalité développée
- Vérification de la cohérence des données et du bon fonctionnement des vues
- Prévision de l'ajout de tests unitaires avec `pytest` pour les futures versions

---

8. Installation du Projet

Prérequis

- Python 3.12 ou version supérieure
- pip (gestionnaire de paquets Python)

Étapes d'installation

```
# 1. Cloner le dépôt
git clone https://github.com/MathisLaveille/Stage-Django.git
cd Stage-Django/django-web-app

# 2. Créer l'environnement virtuel
python -m venv env

# 3. Activer l'environnement
# Sous Windows :
env\Scripts\activate
# Sous macOS/Linux :
source env/bin/activate

# 4. Installer les dépendances
pip install -r requirements.txt

# 5. Effectuer les migrations
python manage.py makemigrations
python manage.py migrate

# 6. Créer un superutilisateur
python manage.py createsuperuser

# 7. Lancer le serveur de développement
python manage.py runserver
```

---

9. Conclusion

L'application **Stage-Django** offre une solution efficace pour la gestion des stages au sein de l'entreprise STURNO. Grâce à une architecture modulaire et à l'utilisation des bonnes pratiques de développement Django, elle est facilement maintenable et extensible.

Améliorations futures

- Intégration de l'authentification JWT pour une API REST sécurisée
- Optimisation des requêtes vers la base de données
- Ajout de tests unitaires automatisés
- Migration vers PostgreSQL pour un environnement de production