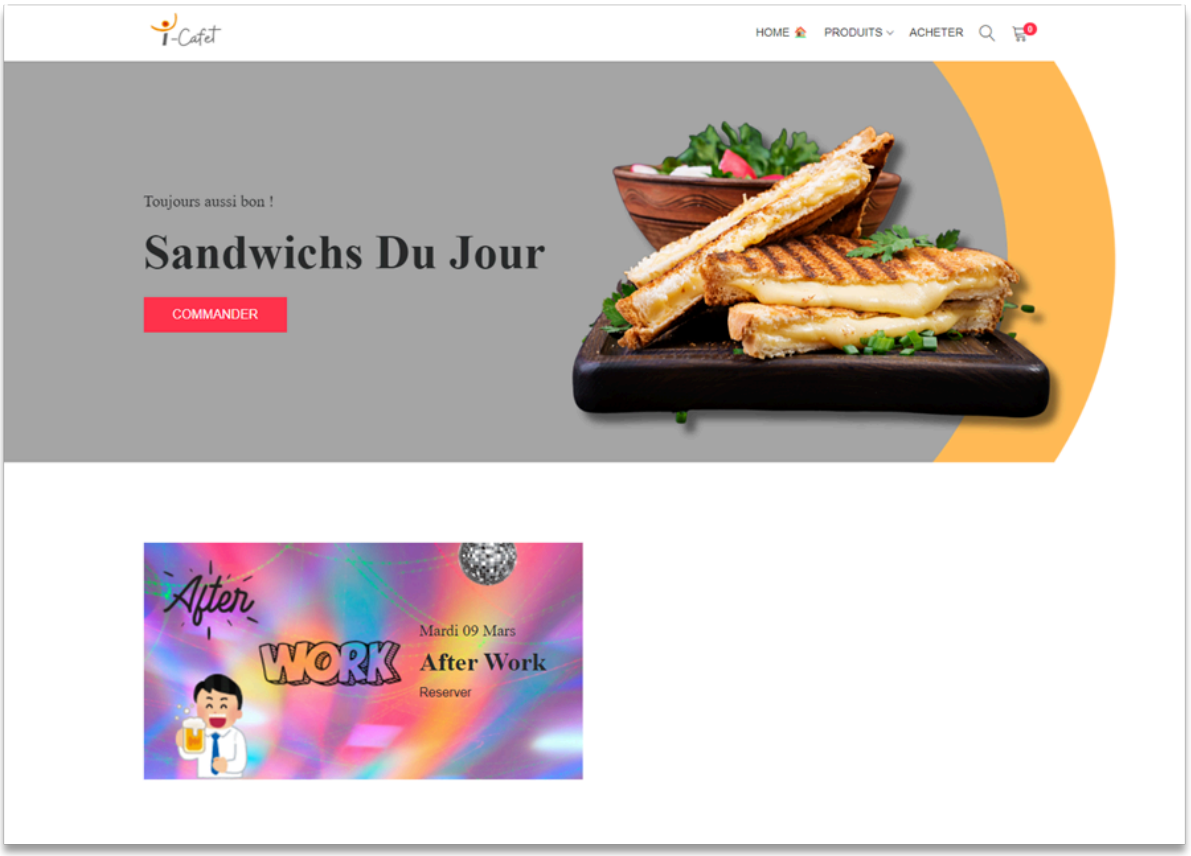
Documentation Technique
I-Cafet



Sommaire

Introduction

La documentation suivante décrit le fonctionnement et la structure du site web I-Cafet. Elle est destinée aux développeurs, aux administrateurs système et à toute personne travaillant sur le maintien ou l'évolution du site.

Architecture

Technologies Utilisées

- Frontend: HTML5, CSS3, JavaScript (Vue.js)
- Backend: DJango
- Base de données: SQLite

Installation Local

Préambule :

Installation python et PIP

Certains environnement et librairies sont requises pour faire fonctionner le site :
Environnement python https://www.python.org/downloads/
Installation PIP https://pypi.org/project/pip/

Installation librairies

Une fois Pip et Python installé executé ces commande:

```
python.exe -m pip install --upgrade pip
python.exe pip install django
python.exe pip install django-ckeditor
python.exe pip install Pillow
```

Installation du site

Récupération sur GIT

Cloner le repository depuis GitHub :
https://github.com/LouIcam2026/IcaMiam-SNI2026

Utilisation

Lancement server

Pour lancer le serveur localement, exécutez la commande suivante :
py manage.py runserver 80
L'application sera accessible à l'adresse http://127.0.0.1:80
L'application admin sera accessible à l'adresse http://127.0.0.1:80/admin
Pour créer son compte admin :
py manage.py createsuperuser

Fonctionnalités

- **Page d'accueil**: Vue d'ensemble des produits proposés par le BDE (Repas de midi et événements)
- **Page acheter**: Tous les produits proposés à la vente
- **Page panier**: Permet de consulter sa commande et de procéder au paiement
- **Page procéder au paiement**: Permet de renseigner ses coordonnées
- **Espace admin**: Modifier les événements et les produits(possible d'ajouter et de supprimer)

Support

Pour toute question ou problème, veuillez contacter l'équipe de développement à l'adresse suivante : https://github.com/LouIcam2026/IcaMiam-SNI2026/pulls

Structure du site

Loader.py et manage.py :

Ces fichiers sont obligatoires dans le bon fonctionnement du site il permettent le démarrage et mettre en oeuvre l'arborescence des fichiers

Templates

Base.html

Le fichier `base.html` constitue le gabarit principal de l'application Django. Il inclut les sections essentielles telles que les balises de métadonnées, les liens vers les fichiers CSS et JavaScript, ainsi que les blocs de contenu et de scripts spécifiques à chaque page.

Shop

Le dossier shop contient tous le formatage html des parties principales pour le site web il contient les fichiers : Cart,Checkout,index,page,shop_list,signle_product

Partials

Le dossier partials contient le formatage html de tout ce qui concerne les header et les footer du site ainsi que les mentions légales etc..
Le dossier contient : footer,header,pagination,product,slider,top_page

Staticfiles / Static

Le dossier `static_files` contient les fichiers statiques utilisés par l'application Django. Les fichiers statiques incluent généralement les fichiers CSS, JavaScript, les images, les polices et autres ressources qui ne changent pas souvent. Ces fichiers sont servis aux utilisateurs directement sans passer par le traitement de Django, ce qui améliore la performance de l'application.
Le dossier `static` dans une application Django est utilisé pour stocker les fichiers statiques comme les fichiers CSS, JavaScript, images, et autres ressources statiques. Ces fichiers sont servis directement aux utilisateurs sans nécessiter de traitement côté serveur, ce qui améliore la performance de l'application.

Shop

Urls

Le fichier `urls.py` dans une application Django est utilisé pour définir les routes URL (ou endpoints) que l'application reconnaît et les associe aux vues correspondantes. Cela permet à Django de répondre aux requêtes HTTP spécifiques en appelant les fonctions ou classes de vue appropriées.

Context_processors

Le fichier `context_processor.py` dans une application Django est utilisé pour définir des context processors personnalisés. Un context processor est une fonction qui reçoit la requête et retourne un dictionnaire de données qui sera automatiquement ajouté au contexte de chaque template rendu par Django.La fonction `site_settings` est un context processor qui récupère diverses configurations et données de la base de données et les ajoute au contexte des templates.

Apps

Le fichier `apps.py` dans une application Django est utilisé pour configurer l'application. Il définit une classe de configuration pour l'application qui peut être utilisée pour spécifier des configurations spécifiques de l'application, telles que le nom de l'application et les comportements par défaut.

Admin

Le fichier `admin.py` est utilisé pour enregistrer les modèles dans l'interface d'administration de Django et pour configurer leur apparence et comportement. Ce fichier personnalise la façon dont les différents modèles apparaissent et interagissent dans l'interface d'administration.

Models

Le dossier `models` dans le dossier `shop` contient les définitions des modèles de données utilisés pour représenter les entités du système de gestion de commerce électronique. Chaque modèle correspond à une table dans la base de données et définit les champs et les comportements des données.

Views

Les vues dans le dossier `shop` gèrent les requêtes HTTP et les réponses correspondantes pour les différentes pages et fonctionnalités du site e-commerce. Chaque vue est responsable de l'interaction avec les modèles, la préparation des données pour les modèles et la sélection des templates à utiliser pour générer les réponses HTML.

Media

Le répertoire `icamiam/media` est généralement utilisé dans les projets Django pour stocker les fichiers téléchargés par les utilisateurs, tels que les images, les documents, les vidéos, etc. Voici quelques remarques et recommandations pour l'utilisation et la gestion de ce répertoire dans un projet Django. Le répertoire contient les dossiers correspondants Slider,setting_images,product_images,nav_collections,category_images,carrier_images
NOTES: Les images des produits communs et toutes les images constantes devront être déplacées dans le dossier `icamiam/static/media` a l'avenir.

Databases

Le répertoire `icamiam/databases` contient des fichiers JSON utilisés par Django pour initialiser la base de données avec des données statiques lors du déploiement initial de l'application. Django gère cette section automatiquement en utilisant des scripts spécifiques pour charger ces données dans les modèles de base de données correspondants.

Config

Le fichier `settings.py` contient toutes les configurations nécessaires pour le projet Django `i-cafet`. Voici un guide détaillé pour les développeurs souhaitant comprendre ou modifier ces paramètres.
Le fichier `settings.py` définit les configurations essentielles pour l'application Django, telles que les paramètres de sécurité, les applications installées, les configurations de base de données, les répertoires de fichiers statiques, et bien plus encore.

Urls

Le fichier `urls.py` dans un projet Django est responsable de la gestion du routage des URLs vers les vues appropriées. Voici une description détaillée de son contenu et de son utilisation pour le projet `i-cafet`.
Ce fichier configure les URL pour l'ensemble du projet, y compris l'administration, les applications internes comme `shop` et la gestion des fichiers statiques en mode développement.

pycache

Le répertoire `_pycache_` est généré par Python pour stocker des fichiers bytecode compilés (.pyc) lors de l'importation des modules Python dans un projet. Voici ce que vous devez savoir à propos de ce répertoire dans le contexte de votre projet `i-cafet`.