



**ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR**
Membre de
HONORIS UNITED UNIVERSITIES

PROJET DE FIN D'ANNÉE

3ème Année en Ingénierie Informatique et Réseaux

**Conception et réalisation d'une
application e-commerce**

Réalisé par :
TAHDI LOKMAN
KNOUZI AHMED

Encadrant :
DAOUDI IMANE

Année universitaire : 2024/2025

DÉDICACE

À ma mère et mon père,

Aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consentis pour mon instruction et mon bien-être. Je vous remercie pour tout le soutien et l'amour que vous me portez depuis mon enfance et j'espère que votre bénédiction m'accompagnera toujours. Que ce modeste travail soit l'exaucement de vos voeux tant formulés, le fruit de vos innombrables sacrifices, bien que je ne vous en acquitterai jamais assez. Puisse Dieu, le Très Haut, vous accorder santé, bonheur et longue vie et faire en sorte que jamais je ne vous déçoive.

À mes frères,

Pour tous les moments passés ensemble, tout le bonheur que vous avez pu m'apporter, tous les moments difficiles qu'on a pu surmonter ensemble. Votre soutien et vos encouragements ont été essentiels tout au long de ce parcours. Merci d'avoir été là pour moi à chaque étape.

À mes amis,

À mes professeurs,

À tous ceux que j'aime.

Sans vous, ma vie n'aurait pas eu son charme. Je vous aime et je vous dédie ce travail.

Remerciements

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet de fin d'année.

Avant tout, je remercie **Dieu Tout-Puissant** de m'avoir donné la force, la patience et la motivation nécessaires pour mener à bien ce travail.

Je remercie chaleureusement **mon encadrant pédagogique**, [Nom de l'encadrant], pour ses conseils avisés, sa disponibilité et son accompagnement tout au long de ce projet.

Mes remerciements vont également à **l'ensemble du corps enseignant de l'EMSI**, pour la qualité de l'enseignement et leur engagement tout au long de ma formation.

Je n'oublie pas ma **famille**, qui m'a toujours soutenu avec amour, encouragements et confiance.

Enfin, je remercie mes **amis et collègues** pour leur entraide, leur soutien moral et les échanges constructifs qui m'ont permis de progresser tout au long de cette aventure.

Table des matières

DÉDICACE	1
Remerciements	2
1 Contexte général du projet	7
1.1 Introduction	7
1.2 Contexte du projet	7
1.3 Périmètre du projet	8
1.4 Problématique	8
1.5 Objectifs du projet	8
1.6 Benchmark	9
1.7 Méthodologies de gestion de projet	9
1.7.1 Méthodologie adoptée : la méthode agile (Scrum simplifié)	10
1.7.2 Étapes principales du projet	10
1.8 Diagramme de Gantt	11
1.9 Conclusion	11
2 Analyse et conception	13
2.1 Introduction	13
2.2 Besoins fonctionnels	13
2.2.1 Visiteur (non connecté)	13
2.2.2 Client connecté	13
2.2.3 Administrateur	14
2.3 Besoins non fonctionnels	14
2.3.1 Performance	14
2.3.2 Sécurité	14
2.3.3 Compatibilité	14
2.3.4 Fiabilité	14
2.3.5 Extensibilité et maintenabilité	15
2.3.6 Utilisabilité	15
2.4 Diagramme de cas d'utilisation	15
2.4.1 Acteurs	15
2.4.2 Cas d'utilisation pour chaque acteur	15
2.4.3 Relations entre les éléments	16
2.5 Diagramme de séquence (Passer une commande)	17
2.5.1 Participants	17
2.5.2 Séquence d'interactions	17
2.6 Diagramme de classes	18
2.6.1 Classes principales	18
2.6.2 Relations	19
2.7 Diagramme d'activité – Passer une commande	20
2.7.1 Activités	20
2.8 Conclusion	21

3 Réalisation	24
3.1 Introduction	24
3.2 Architecture de la solution	24
3.3 Outils et technologies utilisés	24
3.3.1 Langages de programmation	24
3.3.2 Framework	25
3.3.3 Système de gestion de base de données	25
3.3.4 Outils de développement	26
3.4 Interfaces de l'application	27
3.4.1 Interface d'accueil	27
3.4.2 Interface de catalogue/produits	28
3.4.3 Interface de panier	29
3.4.4 Interface de commande	30
3.4.5 Interfaces d'authentification	31
3.4.6 Interface des commandes	32
3.4.7 Interface d'administration (réservée au personnel)	33
3.4.8 Dashboard	34
3.5 Conclusion	35

Table des figures

1.1 Diagramme de Gantt du projet	11
2.1 Diagramme de cas d'utilisation	16
2.2 Légende des symboles UML	17
2.3 Diagramme de séquence - Passer une commande	18
2.4 Diagramme de classes	22
2.5 Diagramme d'activité - Passer une commande	23
3.1 Logo HTML	24
3.2 Logo CSS	25
3.3 Logo JavaScript	25
3.4 Logo Python	25
3.5 Logo Django	26
3.6 Logo SQLite	26
3.7 Logo Visual Studio Code	26
3.8 Logo GitHub	27
3.9 Interface d'accueil	27
3.10 Interface catalogue/produits	28
3.11 Interface panier	29
3.12 Interface commande	30
3.13 Interface d'authentification	31
3.14 Interface historique des commandes	32
3.15 Interface d'administration	33
3.16 Dashboard	34

Liste des tableaux

1.1 Comparaison des solutions e-commerce	9
--	---

Chapitre 1

Contexte général du projet

1.1 Introduction

À l'ère du numérique, le commerce en ligne s'impose comme un levier incontournable de croissance pour les entreprises. Que ce soit pour vendre des produits physiques ou proposer des services numériques, les plateformes e-commerce jouent un rôle central dans la transformation des habitudes de consommation. Face à cette évolution, la maîtrise des outils et des technologies nécessaires à la création d'une boutique en ligne devient une compétence stratégique, tant sur le plan technique que fonctionnel.

Ce projet s'inscrit dans cette dynamique en proposant la **conception et la réalisation d'une application e-commerce complète**, allant de l'analyse des besoins jusqu'au déploiement de la solution. L'objectif est de mettre en œuvre une plateforme intuitive, sécurisée et responsive, permettant à des utilisateurs de parcourir un catalogue de produits, de passer commande, et de gérer leur compte en ligne.

Pour ce faire, des technologies modernes ont été mobilisées : **Python et Django** pour la logique côté serveur, **SQLite** pour la gestion des données, et **HTML/CSS/Bootstrap** pour une interface utilisateur fluide et attrayante. Ce projet constitue ainsi une mise en pratique concrète des principes de développement web, de conception orientée objet et de modélisation de base de données, tout en répondant à de réels besoins fonctionnels du e-commerce.

1.2 Contexte du projet

Le commerce électronique connaît une croissance exponentielle depuis plusieurs années, bouleversant les modes traditionnels d'achat et de vente. L'accessibilité croissante à Internet, le développement des moyens de paiement en ligne, et la généralisation des smartphones ont contribué à faire du e-commerce un secteur clé de l'économie numérique.

Dans ce contexte, de nombreuses entreprises, quelle que soit leur taille, cherchent à se doter de plateformes de vente en ligne pour élargir leur clientèle, optimiser leurs ventes et renforcer leur présence sur le marché. Concevoir une application e-commerce ne se limite pas à proposer des produits en ligne ; il s'agit de mettre en place un système complet et fiable qui prend en compte l'expérience utilisateur, la sécurité des transactions, la gestion des stocks, le suivi des commandes et la personnalisation du service.

C'est dans cette perspective que s'inscrit ce projet, dont l'objectif est de développer une application web de type e-commerce. Le but est de simuler un environnement professionnel complet dans lequel l'utilisateur peut naviguer, consulter des fiches produits, s'inscrire, passer commande, et suivre ses achats. Le projet offre ainsi une opportunité d'appliquer de manière concrète les compétences acquises en développement web, en bases de données et en conception logicielle, à travers un cas d'usage réaliste et représentatif des besoins du marché actuel.

1.3 Périmètre du projet

Le périmètre de ce projet englobe l'ensemble des fonctionnalités essentielles au bon fonctionnement d'une plateforme e-commerce de base. Le développement se concentre sur les aspects techniques, fonctionnels et visuels nécessaires à la mise en ligne d'un site marchand simple, tout en respectant les bonnes pratiques de sécurité, d'ergonomie et de performance.

- Affichage d'un **catalogue de produits** classés par catégories.
- Affichage des **détails de chaque produit** (image, description, prix, disponibilité).
- **Inscription et authentification** des utilisateurs (clients).
- Gestion du **panier d'achat** (ajout, modification, suppression d'articles).
- **Passage de commande** avec validation du panier.
- Espace **client** pour consulter l'historique des commandes.
- Interface d'administration (via Django admin) pour :
 - Ajouter, modifier ou supprimer des produits.
 - Gérer les utilisateurs et les commandes.
 - Gérer les catégories de produits.

1.4 Problématique

Dans un environnement de plus en plus digitalisé, où les habitudes de consommation évoluent vers des solutions rapides, accessibles et personnalisées, **comment concevoir et développer une application e-commerce fiable, ergonomique et sécurisée**, capable de répondre aux besoins des utilisateurs tout en respectant les contraintes techniques et fonctionnelles d'un projet web moderne ?

Cette problématique soulève plusieurs enjeux :

- Comment assurer une **navigation fluide** et intuitive pour l'utilisateur ?
- Comment organiser efficacement le **catalogue de produits** et le processus d'achat ?
- Comment gérer de manière sécurisée les **informations utilisateurs** et les **transactions** ?
- Comment garantir la **scalabilité** et la **maintenabilité** de l'application pour une éventuelle évolution future ?

L'objectif du projet est donc de répondre à cette problématique en mettant en œuvre une solution web complète, développée à l'aide du framework Django, qui couvre toutes les étapes essentielles d'une expérience e-commerce de base.

1.5 Objectifs du projet

Le projet a pour objectif principal de **concevoir et développer une application web e-commerce fonctionnelle**, permettant à des utilisateurs de naviguer dans un catalogue de produits, de gérer un panier, de passer des commandes, et d'accéder à un espace personnel. Pour cela, plusieurs objectifs spécifiques ont été définis :

- **Modéliser** une base de données adaptée à une structure e-commerce (produits, catégories, utilisateurs, commandes).
- Concevoir une **interface utilisateur intuitive et responsive** pour faciliter la navigation et l'achat en ligne.
- Mettre en place un **système d'authentification sécurisé** pour la gestion des comptes clients.
- Implémenter les fonctionnalités essentielles : **consultation des produits, gestion du panier, validation des commandes, consultation de l'historique**.
- Créer une interface d'**administration** permettant la gestion des produits, des commandes et des utilisateurs.
- Tester l'application afin d'assurer sa **fiabilité**, sa **cohérence fonctionnelle** et son bon **comportement en conditions d'utilisation réelle**.

1.6 Benchmark

Avant d'entamer la conception de l'application e-commerce, il est essentiel d'étudier les solutions existantes sur le marché afin d'identifier les bonnes pratiques, les fonctionnalités courantes et les limites potentielles. Ce **benchmark** présente une comparaison entre trois types de plateformes e-commerce populaires : **Shopify**, **WooCommerce**, et une **application e-commerce développée sur mesure avec Django** (comme dans notre projet).

TABLE 1.1 – Comparaison des solutions e-commerce

Critère	Shopify	WooCommerce (WordPress)
Type	Solution SaaS	Plugin CMS open-source
Facilité de mise en œuvre	Très facile	Moyennement facile
Personnalisation	Limitée aux options proposées	Moyenne à élevée
Fonctionnalités natives	Très complètes	Dépend des extensions
Coût	Abonnement mensuel	Gratuit (hors hébergement)
Contrôle sur le code source	Non	Partiel
Sécurité	Gérée par la plateforme	Dépend des plugins

- Les solutions comme **Shopify** ou **WooCommerce** sont idéales pour des utilisateurs sans connaissances techniques, mais elles offrent une **personnalisation limitée**.
- En revanche, une solution **développée sur mesure avec Django** permet de construire une plateforme **adaptée aux besoins spécifiques du projet**, avec un **contrôle total sur l'architecture, la logique métier et la sécurité**.

1.7 Méthodologies de gestion de projet

Pour mener à bien le développement de l'application e-commerce, il est essentiel de s'appuyer sur une méthodologie de gestion de projet adaptée. Celle-ci permet de structurer les différentes phases du projet, de planifier les tâches, de gérer les ressources et de suivre l'avancement du travail de manière organisée et efficace.

1.7.1 Méthodologie adoptée : la méthode agile (Scrum simplifié)

Dans ce projet, une approche **agile** a été privilégiée, inspirée de la méthode **Scrum**, mais adaptée à une petite équipe ou à un travail individuel. Cette méthodologie a été choisie pour sa flexibilité, sa capacité à s'adapter aux changements, et sa focalisation sur la livraison itérative de fonctionnalités opérationnelles.

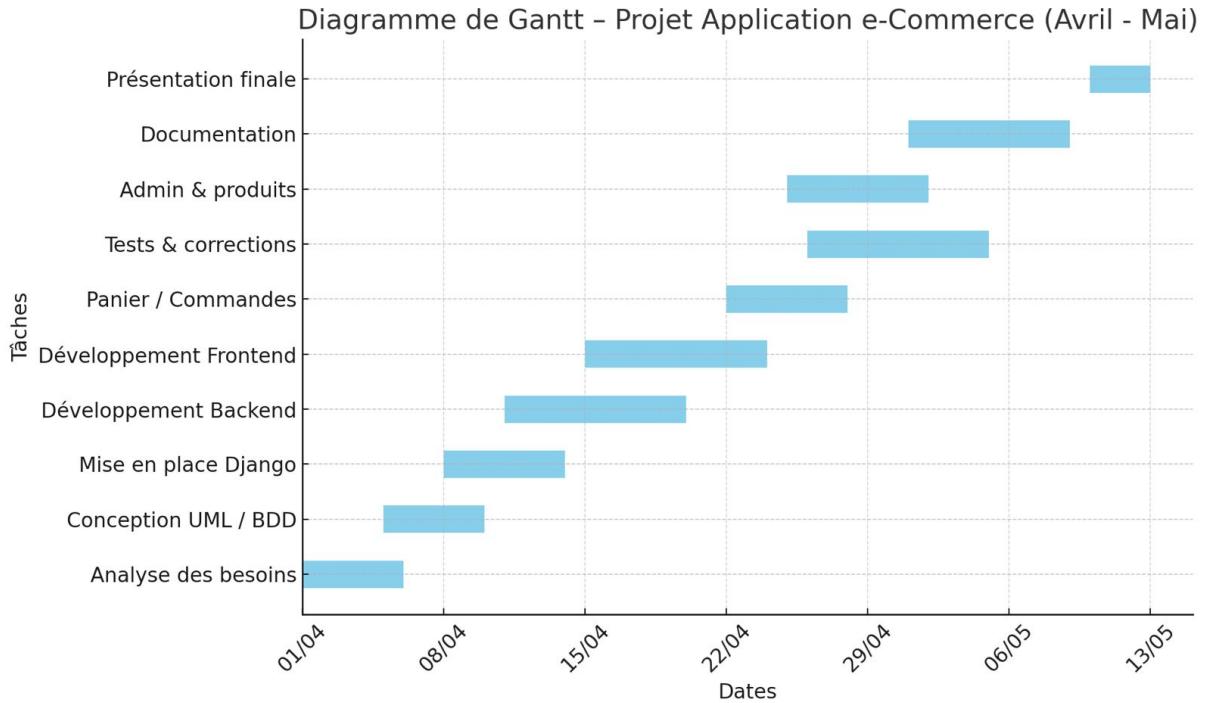
Les grands principes suivis :

- **Découpage du projet en sprints** (cycles courts de développement de 1 à 2 semaines).
- À la fin de chaque sprint, une ou plusieurs **fonctionnalités sont testées et livrables**.
- Des **réajustements** sont possibles après chaque étape, en fonction des retours ou des problèmes rencontrés.
- Suivi de l'avancement via un **tableau Kanban** (physique ou numérique).

1.7.2 Étapes principales du projet

- **Analyse des besoins** : Identification des fonctionnalités principales, des profils utilisateurs, des contraintes techniques.
- **Modélisation** : Conception de la base de données et des diagrammes UML (cas d'utilisation, classes, etc.).
- **Conception technique** : Définition de l'architecture de l'application (structure Django, routes, vues, templates, etc.).
- **Développement** :
 - Backend (modèles, vues, logique métier)
 - Frontend (interfaces, navigation, intégration Bootstrap)
 - Fonctionnalités spécifiques (panier, commandes, espace client)
- **Tests fonctionnels** : Vérification du bon fonctionnement des fonctionnalités développées.
- **Documentation et déploiement** : Rédaction des guides d'utilisation, installation et mise en ligne éventuelle.

1.8 Diagramme de Gantt



Le diagramme de Gantt est un outil visuel de gestion de projet qui montre les tâches sur une échelle de temps, illustrant leur durée et séquence. Chaque tâche est représentée par une barre horizontale, facilitant la planification et le suivi des délais. Il permet de visualiser les dépendances et les chevauchements entre les tâches.

FIGURE 1.1 – Diagramme de Gantt du projet

1.9 Conclusion

La conception et la réalisation d'une application e-commerce constituent un projet complet, mobilisant à la fois des compétences en analyse, en modélisation, en développement web, ainsi qu'en gestion de projet.

À travers les différentes étapes – de l'identification des besoins jusqu'aux tests finaux – ce travail a permis de mettre en pratique des notions essentielles telles que l'architecture MVC avec Django, la structuration d'une base de données relationnelle, l'authentification des utilisateurs, la gestion dynamique des contenus, et l'implémentation de fonctionnalités clés comme le panier, les commandes ou l'espace client.

L'approche méthodologique adoptée, basée sur une gestion agile simplifiée, a facilité l'organisation du travail et l'évolution itérative du projet. L'utilisation d'outils de planification comme le diagramme de Gantt ou de suivi comme Git a contribué à un développement structuré et progressif.

Ce projet constitue une base solide pour une boutique en ligne fonctionnelle, évolutive et personnalisable. Des pistes d'amélioration sont envisageables pour des versions futures,

telles que l'intégration d'un module de paiement en ligne, un système de gestion des livraisons ou une interface mobile dédiée.

En somme, cette expérience a permis de mieux comprendre les enjeux techniques et fonctionnels liés à la mise en place d'un site e-commerce, tout en consolidant les compétences pratiques en développement web fullstack avec Django.

Chapitre 2

Analyse et conception

2.1 Introduction

Avant de débuter la phase de développement d'une application e-commerce, il est indispensable de passer par une étape d'analyse et de conception. Cette phase joue un rôle fondamental dans la réussite du projet, car elle permet de comprendre en profondeur les besoins des utilisateurs, de définir les fonctionnalités à implémenter, et d'imaginer une architecture claire, cohérente et évolutive.

L'analyse vise à identifier les différents acteurs du système (utilisateur, administrateur), leurs interactions avec l'application, ainsi que les processus métier à modéliser (consultation des produits, gestion du panier, validation de commande, etc.). À partir de cette compréhension, la phase de conception permet de représenter ces éléments sous forme de modèles graphiques (cas d'utilisation, diagrammes de classes, maquettes d'interfaces), en vue de guider efficacement le développement technique.

Cette section présente donc les résultats de l'analyse fonctionnelle et technique du projet, ainsi que les choix de conception retenus pour assurer la fiabilité, la modularité et la maintenabilité de l'application.

2.2 Besoins fonctionnels

Les besoins fonctionnels correspondent aux différentes **fonctions que le système doit être capable d'accomplir** afin de répondre aux attentes des utilisateurs. Pour cette application e-commerce, ces besoins sont identifiés à partir des rôles principaux : **visiteur, client inscrit, et administrateur**.

2.2.1 Visiteur (non connecté)

- Consulter le catalogue des produits.
- Visualiser les détails d'un produit (description, prix, image, stock).
- S'inscrire via un formulaire de création de compte.
- Ajouter des produits au panier (temporairement).

2.2.2 Client connecté

- Se connecter à son compte.
- Ajouter/supprimer des produits dans son panier.
- Passer une commande (validation, etc.).
- Consulter l'historique de ses commandes.

2.2.3 Administrateur

- Se connecter à l'espace d'administration.
- Ajouter, modifier ou supprimer des produits.
- Gérer les catégories de produits.
- Gérer les comptes utilisateurs.
- Consulter et suivre les commandes des clients.
- Mettre à jour le stock des produits.

2.3 Besoins non fonctionnels

Les besoins non fonctionnels définissent les **contraintes de qualité** et les exigences techniques que le système doit respecter, indépendamment des fonctionnalités métier. Ils garantissent la fiabilité, la performance, la sécurité et la maintenabilité de l'application.

2.3.1 Performance

- L'application doit charger les pages rapidement (moins de 3 secondes en moyenne).
- Le système doit pouvoir gérer un nombre raisonnable d'utilisateurs simultanés sans dégradation visible (ex. 20-50 utilisateurs pour un projet académique).

2.3.2 Sécurité

- Les mots de passe doivent être stockés de manière sécurisée (hachage).
- Les zones sensibles (panier, commandes, administration) doivent être protégées par authentification.
- Les données utilisateurs doivent être protégées contre les accès non autorisés.
- Mise en place de protections contre les attaques courantes (CSRF, injections SQL, XSS).

2.3.3 Compatibilité

- L'application doit être fonctionnelle sur les principaux navigateurs (Chrome, Firefox, Edge).
- L'interface doit être responsive, adaptée aux écrans d'ordinateurs, tablettes et smartphones (grâce à Bootstrap ou CSS Media Queries).

2.3.4 Fiabilité

- Le système doit permettre de récupérer gracieusement en cas d'erreur (ex : erreur 404, produit non disponible).
- Des validations doivent être présentes côté client (HTML/JavaScript) et côté serveur (Django) pour éviter les incohérences.

2.3.5 Extensibilité et maintenabilité

- Le code doit être structuré et commenté pour faciliter les évolutions futures.
- L'application doit permettre l'ajout de nouvelles fonctionnalités sans refonte majeure (ex. système de paiement, gestion des livraisons).

2.3.6 Utilisabilité

- L'interface doit être claire, intuitive et facile à prendre en main pour l'utilisateur final.
- Les formulaires doivent être simples, et les messages d'erreurs ou de confirmation explicites.

2.4 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est un outil de modélisation visuelle utilisé en ingénierie logicielle pour décrire les interactions entre les utilisateurs et un système. Il représente les différentes actions que les utilisateurs peuvent effectuer et comment ces actions sont réalisées à travers les fonctionnalités du système. C'est une méthode efficace pour visualiser les exigences fonctionnelles d'un système logiciel.

2.4.1 Acteurs

- **Visiteur** : personne non connectée.
- **Client** : utilisateur ayant un compte.
- **Administrateur** : personne gérant le système.

2.4.2 Cas d'utilisation pour chaque acteur

Visiteur

- Consulter le catalogue.
- Rechercher un produit.
- Afficher les détails d'un produit.
- Créer un compte.

Client

- Se connecter.
- Gérer son panier (ajouter, retirer).
- Passer une commande.
- Consulter l'historique de commandes.

Administrateur

- Se connecter à l'espace admin.
- Gérer les produits (ajouter, modifier, supprimer).
- Gérer les utilisateurs.
- Gérer les commandes.
- Mettre à jour les stocks.

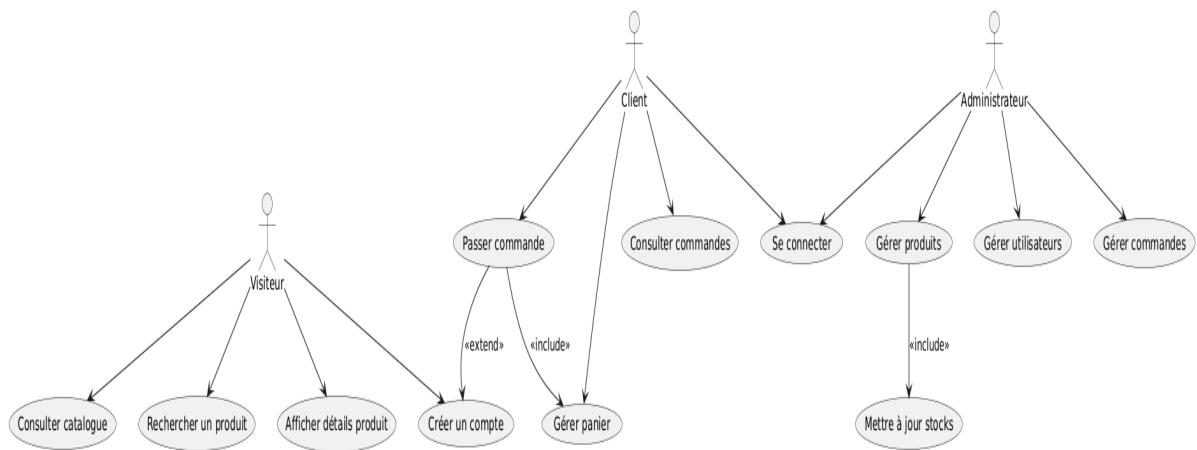


FIGURE 2.1 – Diagramme de cas d'utilisation

2.4.3 Relations entre les éléments

- Le client **doit se connecter** pour passer une commande.
- Passer une commande **inclus** la gestion du panier.
- Gérer les produits **inclus** la mise à jour du stock.

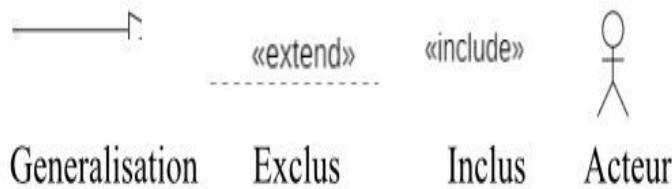


FIGURE 2.2 – Légende des symboles UML

2.5 Diagramme de séquence (Passer une commande)

Le diagramme de séquence est un outil fondamental de la modélisation dynamique qui permet de représenter l’interaction entre les différents acteurs et objets du système dans le temps. Il montre la chronologie des échanges de messages nécessaires pour réaliser une fonctionnalité particulière. Ce type de diagramme est particulièrement utile pour analyser les processus métiers et valider la logique des opérations avant la phase de développement.

2.5.1 Participants

- Client
- Page produit
- Panier
- Service de commande
- Base de données

2.5.2 Séquence d’interactions

1. Le client consulte un produit.
2. Il ajoute le produit au panier.
3. Il valide le panier.
4. Le système passe la commande.
5. Le service commande enregistre la commande en base de données.
6. Le client reçoit une confirmation.

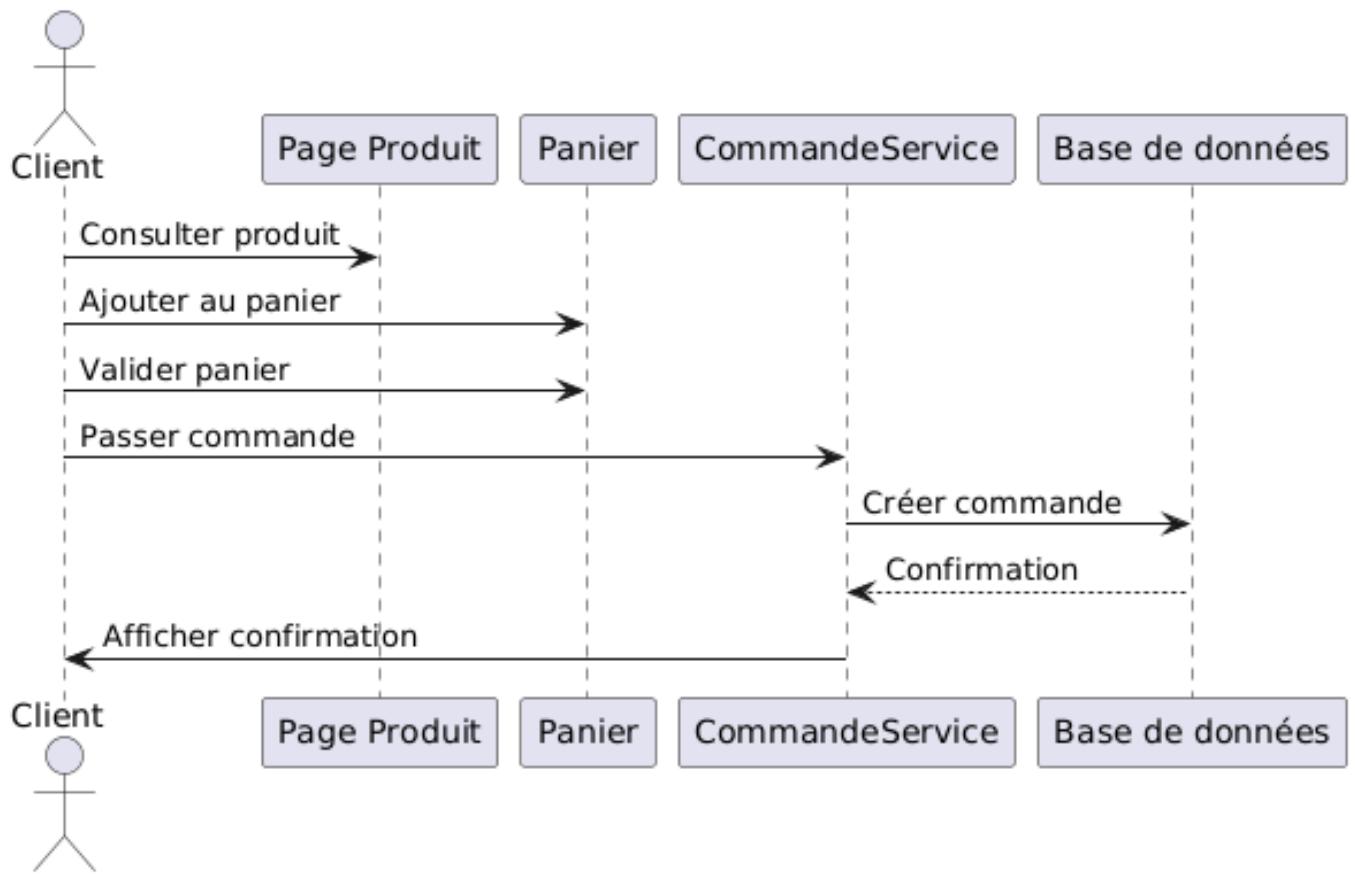


FIGURE 2.3 – Diagramme de séquence - Passer une commande

2.6 Diagramme de classes

Le diagramme de classe est un type de diagramme UML (Unified Modeling Language) qui représente la structure statique d'un système logiciel en illustrant les classes du système, leurs attributs, leurs méthodes, ainsi que les relations entre ces classes. Il est utilisé pour modéliser la structure conceptuelle d'une application ou d'un système, en mettant en évidence les objets et leurs interactions dans le cadre du développement logiciel orienté objet.

2.6.1 Classes principales

- **Utilisateur**
 - Attributs : id, nom, email, mot de passe
 - Méthodes : seConnecter(), seDéconnecter()
 - **Client** (hérite de Utilisateur)
 - Attributs : adresse

- Méthodes : passerCommande()
- **Administrateur** (hérite de Utilisateur)
 - Méthodes : gérerProduits(), gérerUtilisateurs()
- **Produit**
 - Attributs : id, nom, description, prix, stock
 - Méthodes : mettreAJourStock()
- **Panier**
 - Attributs : id, produits[]
 - Méthodes : ajouterProduit(), retirerProduit(), calculerTotal()
- **Commande**
 - Attributs : id, date, statut
 - Méthodes : confirmer()

2.6.2 Relations

- Un client a un panier.
- Un client peut passer plusieurs commandes.
- Une commande contient plusieurs produits.
- Un panier peut contenir plusieurs produits.

2.7 Diagramme d'activité – Passer une commande

Le **diagramme d'activités** est un type de diagramme UML (Unified Modeling Language) qui permet de **modéliser le déroulement d'un processus** ou d'un enchaînement de tâches dans un système. Il représente les **étapes successives**, les **conditions**, les **décisions**, et les **boucles** que peut suivre un acteur ou un système pour accomplir une activité.

Il est souvent comparé à un **organigramme**, mais avec des éléments plus riches pour la **logique métier**.

2.7.1 Activités

1. Consulter le catalogue.
2. Ajouter un produit au panier.
3. Se connecter ou s'inscrire.
4. Vérifier si le panier est vide.
 - Si oui : afficher un message d'erreur.
 - Sinon : continuer.
5. Valider le panier.
6. Saisir l'adresse de livraison.
7. Choisir le mode de paiement.
8. Confirmer la commande.
9. Enregistrer la commande.
10. Afficher la confirmation.

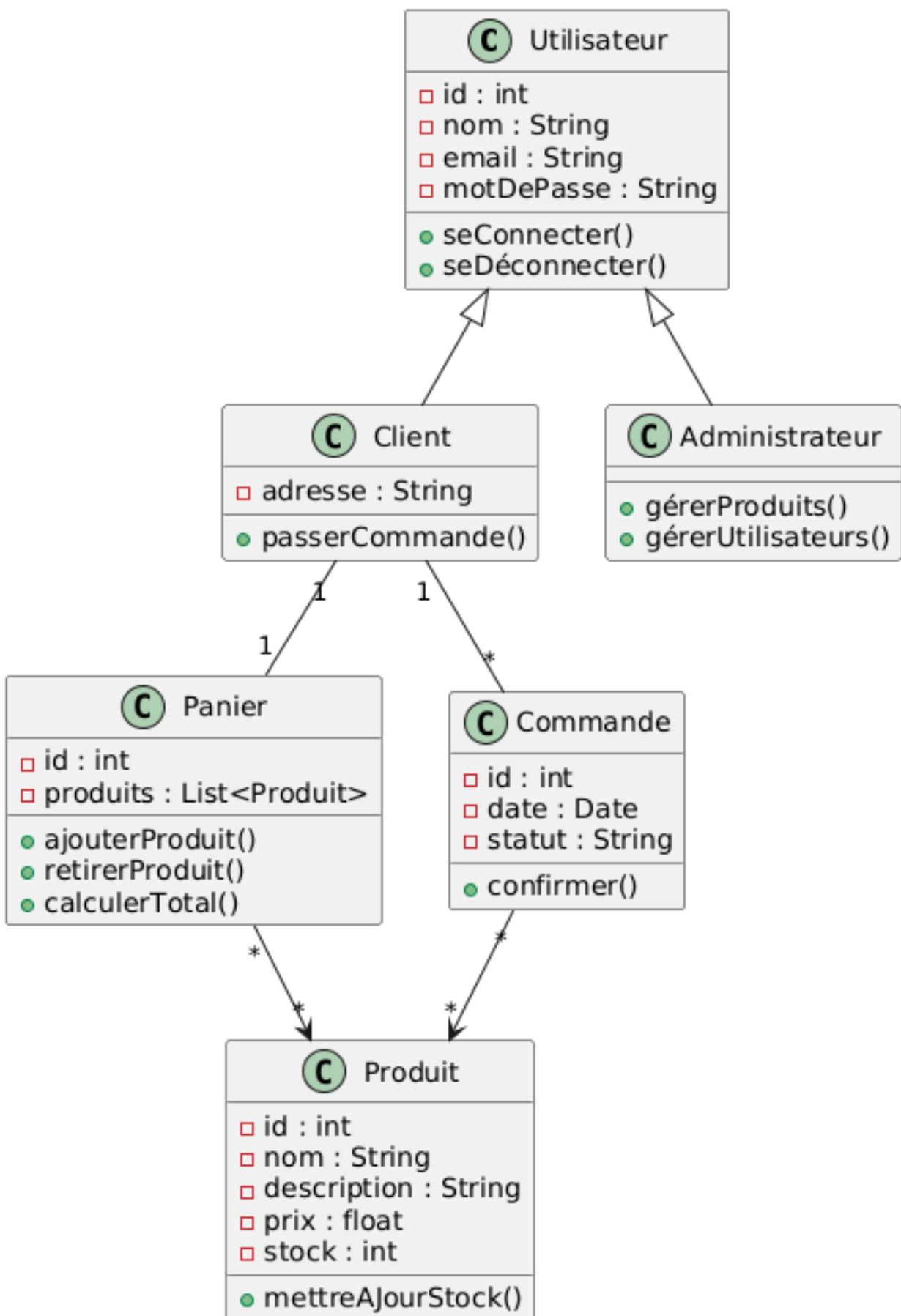


FIGURE 2.4 – Diagramme de classes

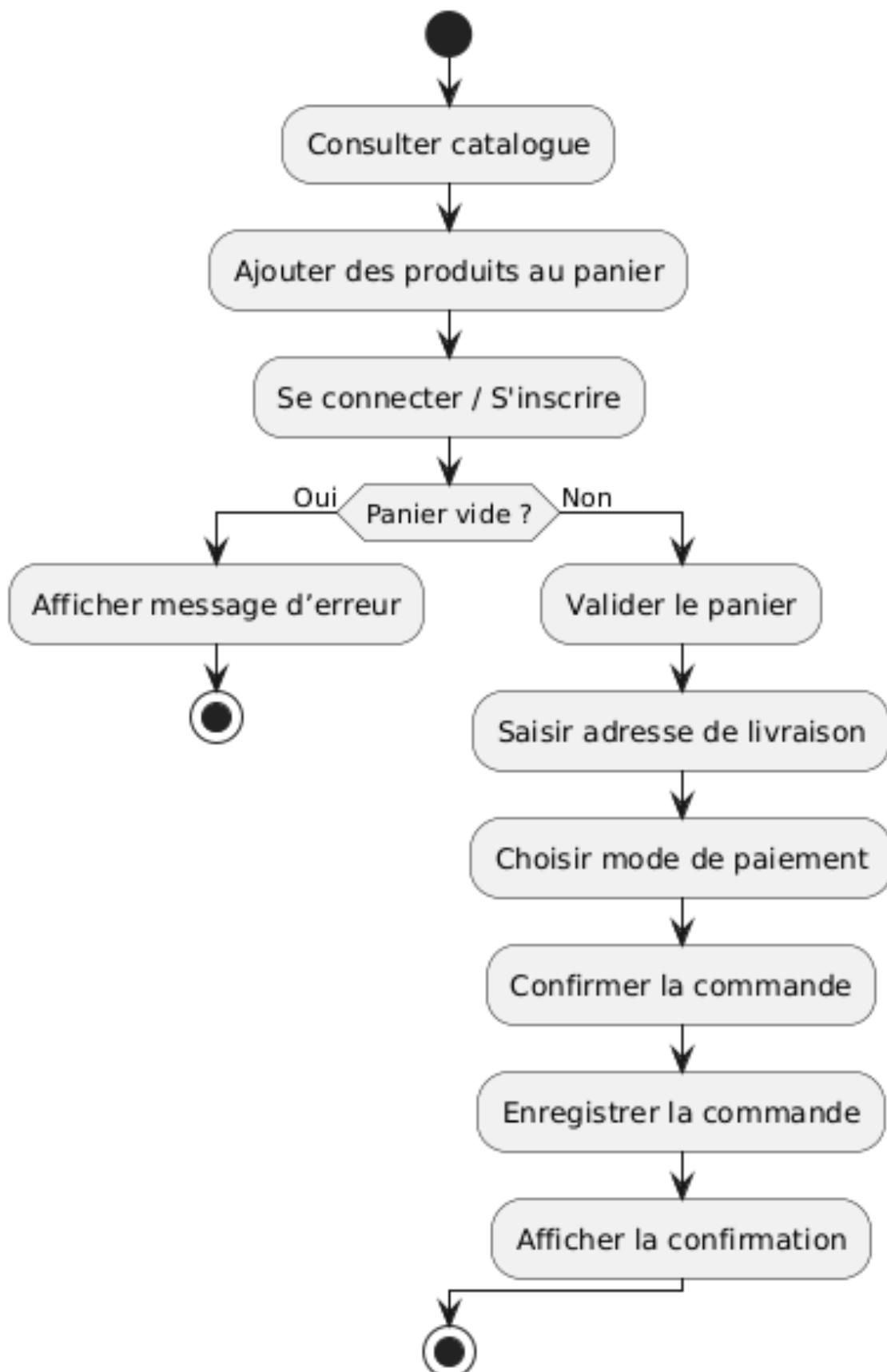


FIGURE 2.5 – Diagramme d’activité - Passer une commande

2.8 Conclusion

La phase de modélisation a permis de représenter de manière claire et structurée le fonctionnement de notre application e-commerce avant son implémentation. À travers les différents diagrammes UML — cas d'utilisation, séquence, classes, états et activités — nous avons pu visualiser les interactions entre les acteurs, les comportements du système, ainsi que son organisation interne. Cette étape essentielle garantit une meilleure compréhension des besoins, une communication efficace entre les parties prenantes, et constitue une base solide pour le développement technique à venir.

Chapitre 3

Réalisation

3.1 Introduction

Après avoir défini les besoins, structuré les fonctionnalités et modélisé le fonctionnement de l'application à travers les différents diagrammes UML, la phase de **réalisation** constitue une étape cruciale du projet. Elle correspond à la **concrétisation technique** des spécifications établies durant la phase d'analyse et de conception. Cette section détaille le **choix des technologies** utilisées (comme Python, Django, HTML/CSS, Bootstrap, SQLite), l'**architecture générale de l'application**, ainsi que le **développement progressif des modules** (authentification, gestion des produits, panier, commandes, etc.). L'objectif est de traduire les modèles théoriques en un système fonctionnel, robuste, et conforme aux attentes des utilisateurs.

3.2 Architecture de la solution

L'architecture de l'application repose sur le **modèle MVC** (Modèle-Vue-Contrôleur), implémenté dans Django sous la forme **MTV** (Model-Template-View), ce qui permet une **séparation claire des responsabilités** entre les différentes couches de l'application.

3.3 Outils et technologies utilisés

Pour mener à bien la réalisation de notre application e-commerce, nous avons choisi un ensemble d'outils et de technologies adaptés à nos besoins en matière de **développement web**, de **gestion des données** et de **conception d'interface utilisateur**. Ces outils nous ont permis de garantir un développement efficace, une bonne organisation du code, et une interface fluide et intuitive.

3.3.1 Langages de programmation



FIGURE 3.1 – Logo HTML

HTML (HyperText Markup Language) est le langage de balisage conçu pour représenter les pages web. Ce langage nous permet d'écrire de l'hypertexte sous forme des balises, d'où son nom, de structurer sémantiquement la page, de mettre en forme son contenu.



FIGURE 3.2 – Logo CSS

CSS (Cascading Style Sheets) qui signifie feuilles de style en cascade. Le mot cascade rappelle ici que les styles peuvent être classés selon différents degrés d'importance. CSS est un langage informatique qui décrit de manière précise comment des documents HTML et XML sont présentés.



FIGURE 3.3 – Logo JavaScript

Javascript Est un langage de programmation de scripts principalement utilisé dans les pages Web interactives. L'interaction de JavaScript avec HTML est gérée par des événements qui se produisent lorsque l'utilisateur ou le navigateur manipule une page.



FIGURE 3.4 – Logo Python

Python Python est un langage de programmation interprété, simple et polyvalent, conçu pour écrire du code clair et lisible. Il est utilisé dans des domaines variés comme le web, l'IA, la data science et l'automatisation.

3.3.2 Framework

Django Django est un framework web Python open source qui permet de développer rapidement des applications web sécurisées, modulaires et maintenables grâce à une architecture MVC (Modèle-Vue-Contrôleur).

3.3.3 Système de gestion de base de données

SQLite base de données légère, intégrée à Django, parfaite pour les projets de petite à moyenne taille et idéale pour la phase de développement.



FIGURE 3.5 – Logo Django



FIGURE 3.6 – Logo SQLite

3.3.4 Outils de développement

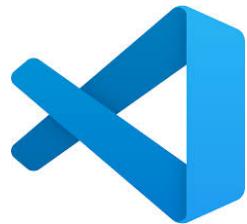


FIGURE 3.7 – Logo Visual Studio Code

Visual studio code Un éditeur de code source développé par Microsoft, très populaire pour le développement web grâce à ses nombreuses extensions et fonctionnalités.

GitHub GitHub est une plateforme en ligne qui permet d'héberger, de gérer et de collaborer sur du code en utilisant le système de contrôle de version Git.



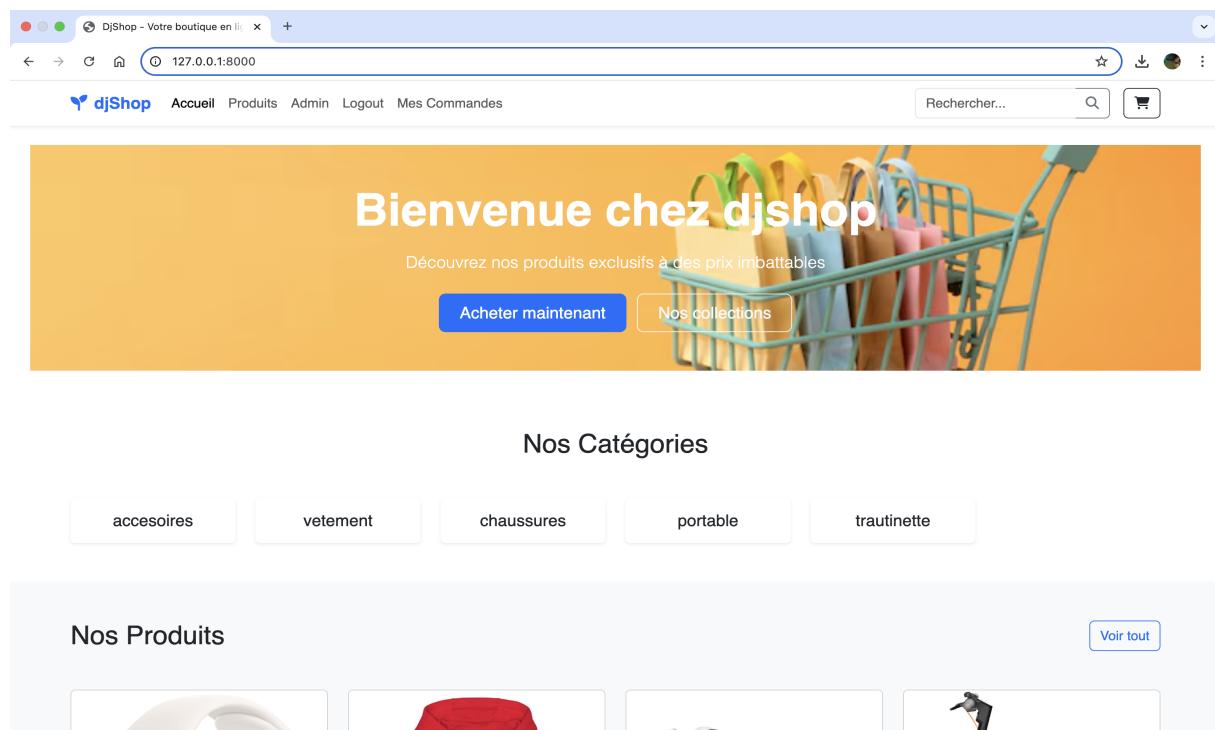
FIGURE 3.8 – Logo GitHub

3.4 Interfaces de l'application

L'interface utilisateur de notre application e-commerce a été conçue pour être **ergonomique, claire et responsive**, afin d'offrir une expérience fluide aussi bien sur ordinateur que sur mobile. Grâce à l'utilisation de **Django Templates** combinés avec **Bootstrap**, nous avons pu créer des pages web dynamiques, cohérentes et faciles à naviguer.

3.4.1 Interface d'accueil

- Présente la boutique avec une sélection de produits mis en avant.
- Barre de navigation : accès rapide à l'accueil, au catalogue, au panier, à l'espace utilisateur.
- Barre de recherche pour filtrer les produits par nom ou catégorie.



The screenshot shows the homepage of the DjShop website. At the top, there is a header bar with the title "djShop" and links for "Accueil", "Produits", "Admin", "Logout", and "Mes Commandes". A search bar with the placeholder "Rechercher..." and a magnifying glass icon is also present. Below the header, a large yellow banner features the text "Bienvenue chez djshop" and "Découvrez nos produits exclusifs à des prix imbattables". It includes two buttons: "Acheter maintenant" and "Nos collections". To the right of the banner is a shopping cart filled with colorful bags. Below the banner, the heading "Nos Catégories" is displayed, followed by five categories: "accesoires", "vetement", "chaussures", "portable", and "trautinette". Underneath these categories, the heading "Nos Produits" is shown, followed by five product thumbnails: a white headphones case, a red cloth, a white electronic device, a black electronic device, and a black jacket. A "Voir tout" button is located in the top right corner of this section.

FIGURE 3.9 – Interface d'accueil

3.4.2 Interface de catalogue/produits

- Affichage en grille ou en liste des produits avec image, titre, prix et bouton "Ajouter au panier".
- Filtres possibles par catégorie, prix, ou mots-clés.
- Affichage détaillé du produit : image agrandie, description, prix, stock disponible.
- Recommandations de produits similaires (optionnel).

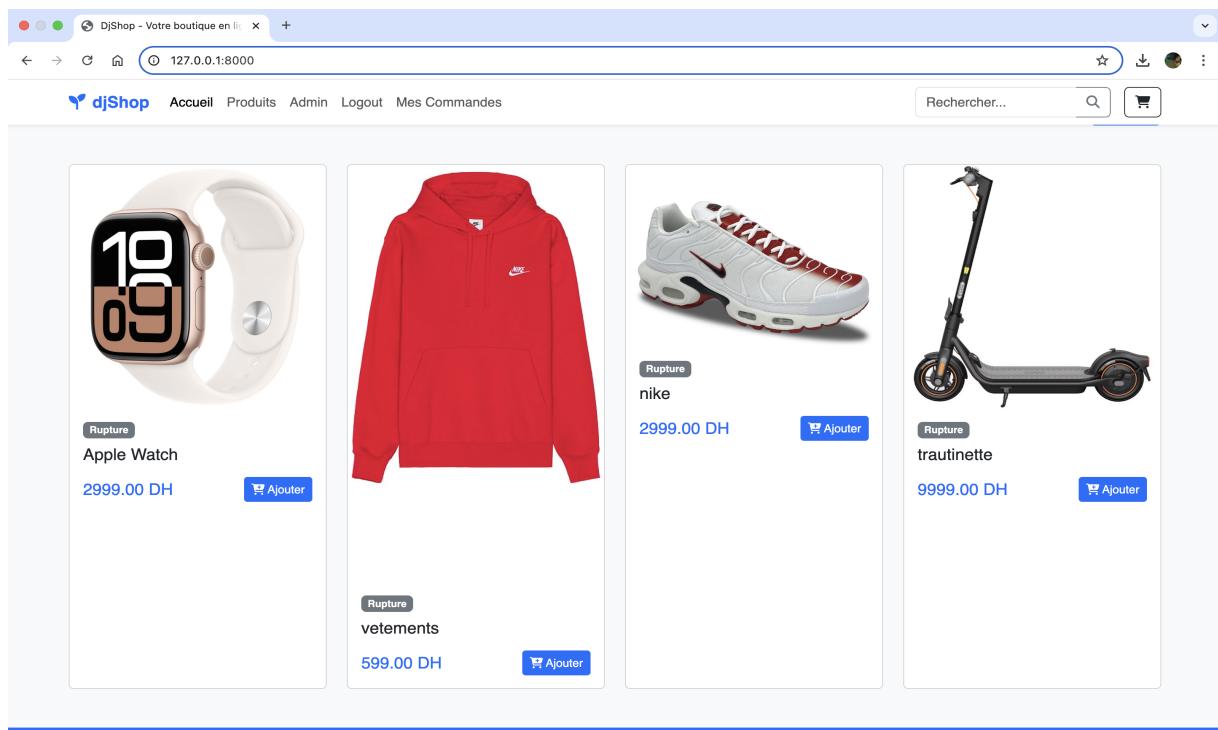


FIGURE 3.10 – Interface catalogue/produits

3.4.3 Interface de panier

- Liste des produits ajoutés avec quantités, sous-total, et bouton de suppression.
- Calcul automatique du total de la commande.
- Bouton "Passer commande" pour finaliser l'achat.

The screenshot shows a web browser window for 'Mon Panier | DjShop' at the URL '127.0.0.1:8000/panier/'. The header includes a logo, navigation links for 'Accueil', 'Produits', 'Admin', 'Logout', and 'Mes Commandes', and a search bar. The main content features a yellow banner with the text 'Bienvenue dans votre panier' and a sub-instruction 'Vérifiez les produits que vous avez ajoutés et procédez à l'achat'. Below the banner is a shopping cart containing three items: 'Apple Watch' (3 units, 2999.00 DH each, total 8997.00 DH), 'vetements' (4 units, 599.00 DH each, total 2396.00 DH), and 'trautinette' (3 units, 9999.00 DH each, total 29997.00 DH). Each item has a 'Supprimer' (Delete) button. A total sum of 'Total : 41390.00 DH' is displayed, followed by a green button labeled 'Procéder à l'achat' (Proceed to purchase).

Image	Produit	Quantité	Prix unitaire	Sous-total	Actions
	Apple Watch	3	2999.00 DH	8997.00 DH	<button>Supprimer</button>
	vetements	4	599.00 DH	2396.00 DH	<button>Supprimer</button>
	trautinette	3	9999.00 DH	29997.00 DH	<button>Supprimer</button>

FIGURE 3.11 – Interface panier

3.4.4 Interface de commande

- Formulaire de validation de commande (informations de livraison, récapitulatif).
- Confirmation de la commande et génération d'un numéro de suivi.

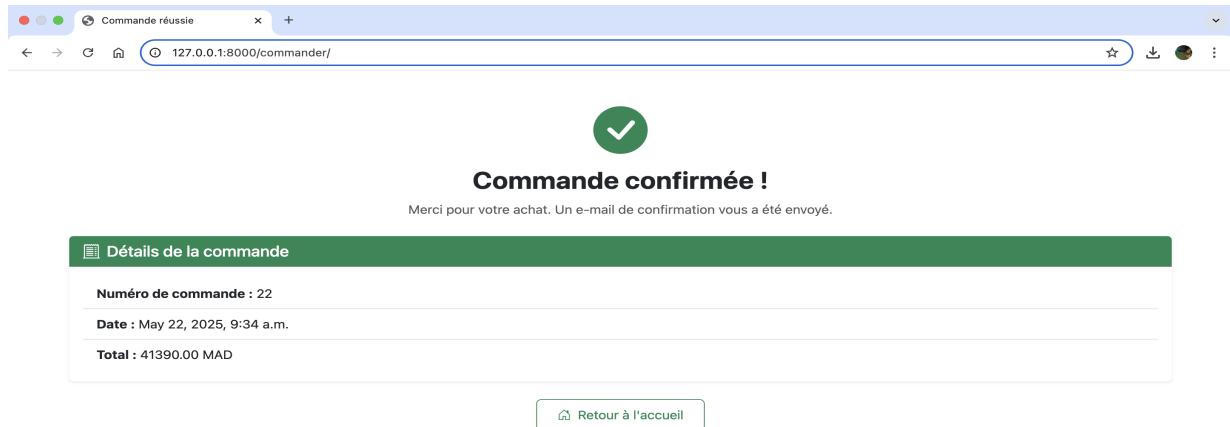


FIGURE 3.12 – Interface commande

3.4.5 Interfaces d'authentification

- **Connexion** : formulaire de login avec vérification des identifiants.
- **Inscription** : formulaire de création de compte (nom, e-mail, mot de passe).

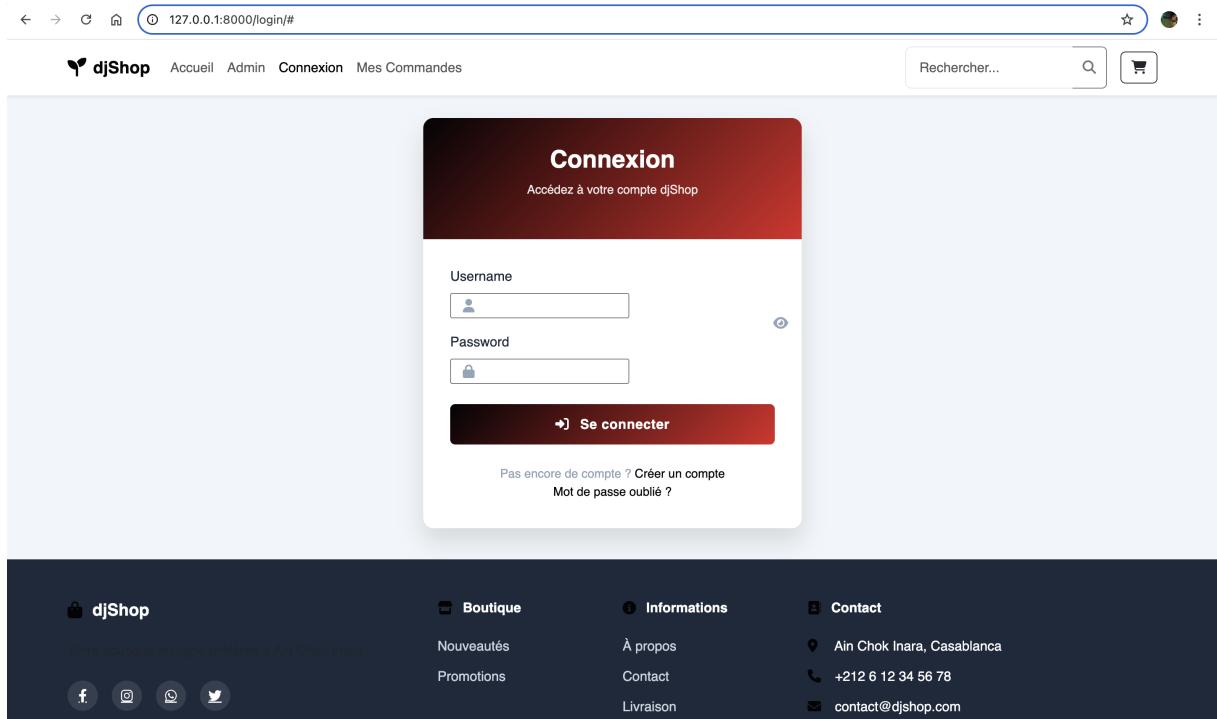


FIGURE 3.13 – Interface d'authentification

3.4.6 Interface des commandes

- Historique des commandes.

Numéro de commande	Date	Status	Total	Actions
22	May 22, 2025, 9:34 a.m.	pending	41390.00 MAD	Voir les détails
19	May 20, 2025, 3:47 p.m.	pending	5998.00 MAD	Voir les détails
18	May 20, 2025, 3:23 p.m.	pending	599.00 MAD	Voir les détails
17	May 20, 2025, 3:01 p.m.	pending	5998.00 MAD	Voir les détails
16	May 20, 2025, 2:55 p.m.	pending	5998.00 MAD	Voir les détails
14	May 20, 2025, 2:36 p.m.	pending	1198.00 MAD	Voir les détails
13	May 20, 2025, 9:09 a.m.	pending	5998.00 MAD	Voir les détails
11	May 19, 2025, 2:55 p.m.	pending	2999.00 MAD	Voir les détails
10	May 19, 2025, 2:12 p.m.	pending	2999.00 MAD	Voir les détails
6	May 17, 2025, 7:37 p.m.	pending	2999.00 MAD	Voir les détails
5	May 17, 2025, 7:19 p.m.	pending	2999.00 MAD	Voir les détails
4	May 13, 2025, 10:52 p.m.	pending	0 MAD	Voir les détails
3	May 9, 2025, 11:07 p.m.	pending	0 MAD	Voir les détails

FIGURE 3.14 – Interface historique des commandes

3.4.7 Interface d'administration (réservée au personnel)

- Gérée via l'interface Django Admin.
- Permet de créer, modifier ou supprimer des produits, gérer les commandes, consulter les utilisateurs inscrits.

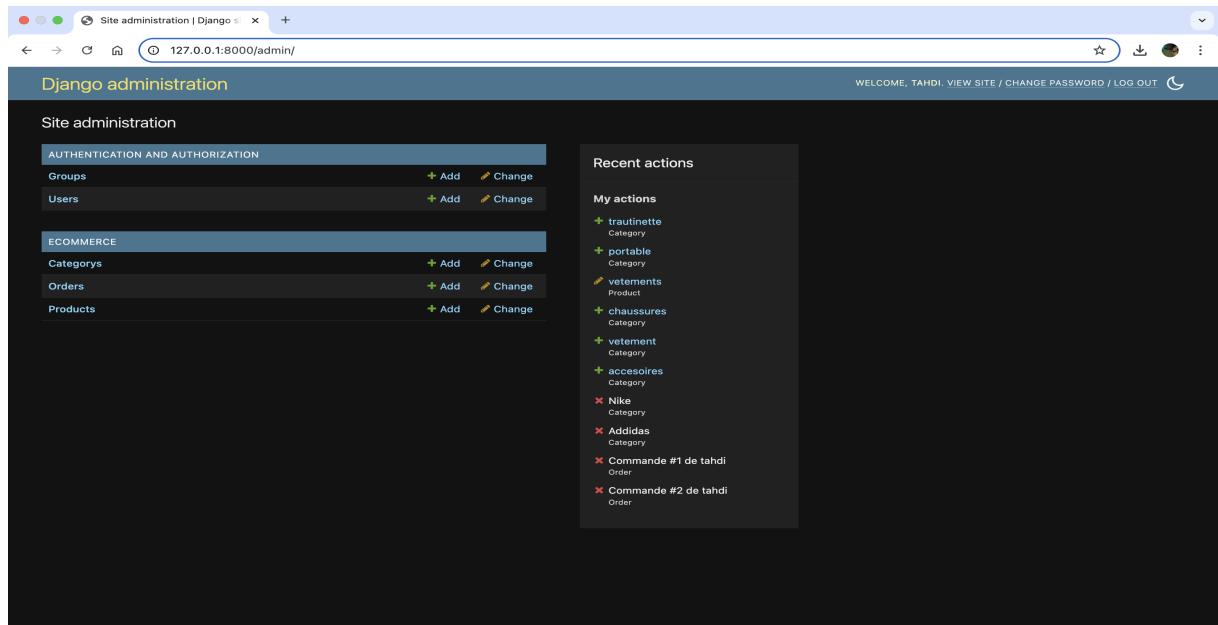


FIGURE 3.15 – Interface d'administration

3.4.8 Dashboard

- Nombre total de produits en ligne.
- Nombre de commandes effectuées.

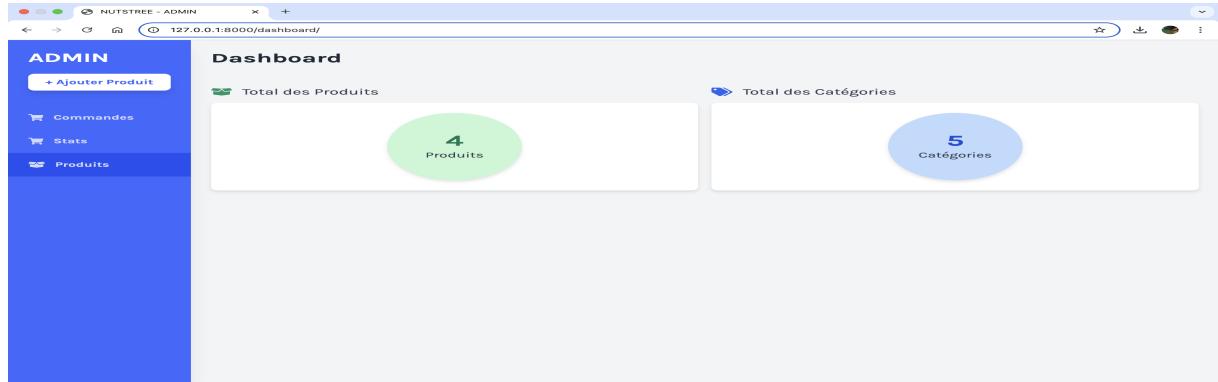


FIGURE 3.16 – Dashboard

3.5 Conclusion

La phase de **réalisation** constitue le cœur technique du projet. Elle a permis de concrétiser la conception théorique de l'application en une plateforme e-commerce fonctionnelle, dynamique et conviviale. En s'appuyant sur des technologies robustes comme **Django**, **Bootstrap**, **SQLite** et des outils modernes de développement, nous avons pu construire une solution répondant efficacement aux besoins identifiés.

Chaque interface a été pensée pour garantir une **expérience utilisateur fluide**, tant pour les clients que pour les administrateurs. L'intégration du dashboard a facilité la **gestion quotidienne de la boutique**, tandis que l'architecture mise en place assure **la maintenabilité et l'évolutivité** du système.

Cette étape marque une avancée significative dans le cycle de vie du projet et ouvre la voie à des phases ultérieures comme **les tests, le déploiement** et éventuellement **l'amélioration continue** de l'application.