KOULADE GRACE ORNELA

RAPPORT DE PROJET ETUDIANT

DEDICACES

À mes chers parents, Vous êtes les piliers de ma vie, ceux qui m'ont guidé, soutenu et aimé depuis le tout début. Votre amour inconditionnel et votre soutien ont été une source constante d'inspiration pour moi. Merci d'avoir été toujours là, dans les bons moments comme dans les mauvais.

À mes merveilleux frères et sœurs, Vous êtes bien plus que de simples membres de ma famille, vous êtes mes complices, mes amis les plus proches. Nos souvenirs partagés, nos rires et nos défis surmontés ensemble ont forgé des liens indestructibles que je chérirai pour toujours.

À mes camarades de classe, Vous avez été une partie essentielle de mon parcours éducatif. Nos échanges, nos débats et nos collaborations ont enrichi mon expérience scolaire et m'ont permis de grandir en tant que personne. Merci pour ces moments de partage et d'apprentissage.

Enfin, à moi-même, Je veux me rappeler que je suis digne d'amour, de respect et de succès. Je suis fier(e) du chemin parcouru, des défis surmontés et des moments de bonheur que j'ai savourés. Je me promets de continuer à croire en moi-même et à poursuivre mes rêves avec détermination.

À tous ceux qui ont fait partie de mon voyage jusqu'ici, je vous adresse ma plus sincère gratitude et mon amour éternel.

Avec tout mon amour,

GRACE ORNELA

REMERCIEMENTS

À Dieu, Je veux commencer par exprimer ma gratitude envers Dieu, source de toute grâce et de toute bénédiction. C'est grâce à Sa guidance et Sa miséricorde que je suis arrivée là où je suis aujourd'hui. Merci pour les défis surmontés, les leçons apprises et les bénédictions inattendues sur mon chemin.

À moi-même, Je veux prendre un moment pour reconnaître le travail acharné, la persévérance et la détermination que j'ai investis dans mon parcours jusqu'à présent. Je suis fière de mes accomplissements et je me promets de continuer à croire en moi-même et à suivre mes aspirations avec passion et courage.

À mes professeurs, Vous avez été bien plus que des éducateurs pour moi. Vous avez été des mentors, des guides et des sources d'inspiration. Votre dévouement, votre expertise et votre soutien ont été essentiels pour mon développement académique et personnel. Merci pour tout ce que vous avez fait pour moi.

Aux membres du jury de mon examen, Même si vous ne m'avez pas encore examinée, je tiens à exprimer ma reconnaissance pour le rôle crucial que vous jouez dans mon parcours éducatif. Votre expertise, votre objectivité et votre engagement envers l'excellence académique sont essentiels pour assurer l'équité et la qualité de notre système éducatif. Merci pour votre contribution précieuse.

Ensemble, vous avez tous façonné mon parcours et m'avez aidée à devenir la personne que je suis aujourd'hui. Je vous adresse mes plus sincères remerciements et ma gratitude éternelle.

Avec toute ma reconnaissance,

GRACE ORNELA

TABLE DES MATIERES

INTRODUCTION GENERALE

Le développement d’applications mobiles correspondant au processus de conception, de développent, de test, de maintenance et de déploiement d’applications mobiles fiables, sécuriser, évolutives, et accessible permettant aux entreprises d’interagir avec leurs clients.

Le e-commerce ou « commerce électronique » véritable pilier est à l’édifice de la révolution du commerce moderne, il englobe essentiellement les transactions commerciales s’effectuant sur internet à partir des différents types de terminaux sur des sites e-commerces ou applications mobiles marchandes.

Il a largement remplacé la vente par correspondance et est devenu le principal moyen de vente. Le e-commerce inclut principalement les transactions en ligne effectuées sur des sites web ou des applications mobiles dédiées. Ces plateformes offrent aux clients un accès mondial à une variété d'informations, de produits et de services. Les sites de vente en ligne permettent aux entreprises d’interagir indirectement avec les clients tout en leurs fournissant une service impécables. En outre, ces plateformes offrent une visibilité accrue aux entreprises en les exposant à un public large et diversifier.

Le développement d'applications mobiles jouant un rôle important dans l'expansion et l'optimisation du commerce électronique, c’est dans cette et dans le cadre de mon projet de fin d’études que j’ai procéder a la mise en place d’un site e-commerce de ventes qu’on dénotera "AdAsH" et qui permettra de gérer et faciliter les ventes de l’entreprises mon choix s’est donc porter sur la conception d’une plateforme e-commerce de ventes.

Le présent rapport permet de détailler les différentes étapes que nous avons suivies lors du développement de notre projet. Il se compose de quatre chapitres.

Dans le premier chapitre, nous commençons par présenter le cadre général du projet, la mission qui nous a été confiée, saisir les besoins fonctionnels et non fonctionnels, ainsi que le cycle de développement adopté pour la mise en œuvre de notre projet. Le deuxième chapitre est consacré à la description de l’étude technique tout en déterminant les technologies et outils utilisés.

CHAIPTRE I : INTRODUCTION

PRESENTATION GENERALE DU PROJET

Dans le premier chapitre, nous nous intéresserons d'abord à l'introduction du cadre global du projet et à la description des exigences. Il s'agit en fait d'une introduction à l'organisation d'accueil, suivie de la définition de la tâche, de la présentation du projet et des objectifs à atteindre. Ensuite, nous étudions et discutons quelques applications similaires existantes, puis nous présentons la solution proposée. En outre, nous présentons également la méthode de développement adoptée et le choix du modèle conceptuel. Enfin, nous terminerons ce chapitre par la description de la spécification des exigences et finaliserons le plan de version pour produire un backlog de produit initial et le premier plan de sprint

ENTREPRISE D’ACCUEIL

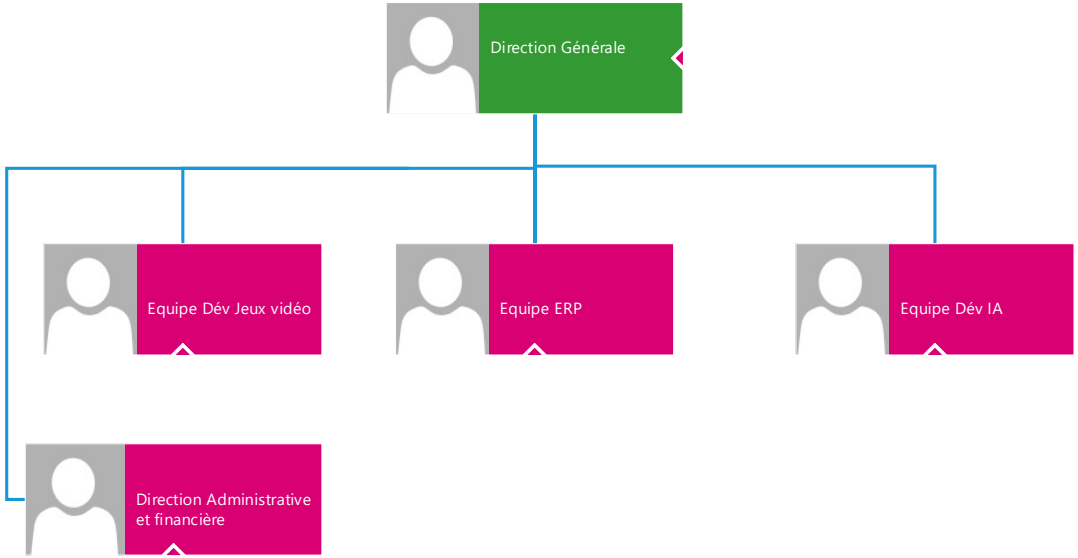
1-Présentation d’Innovative Labs

Innovative Labs est une société de recherche et développement créée en 2023.

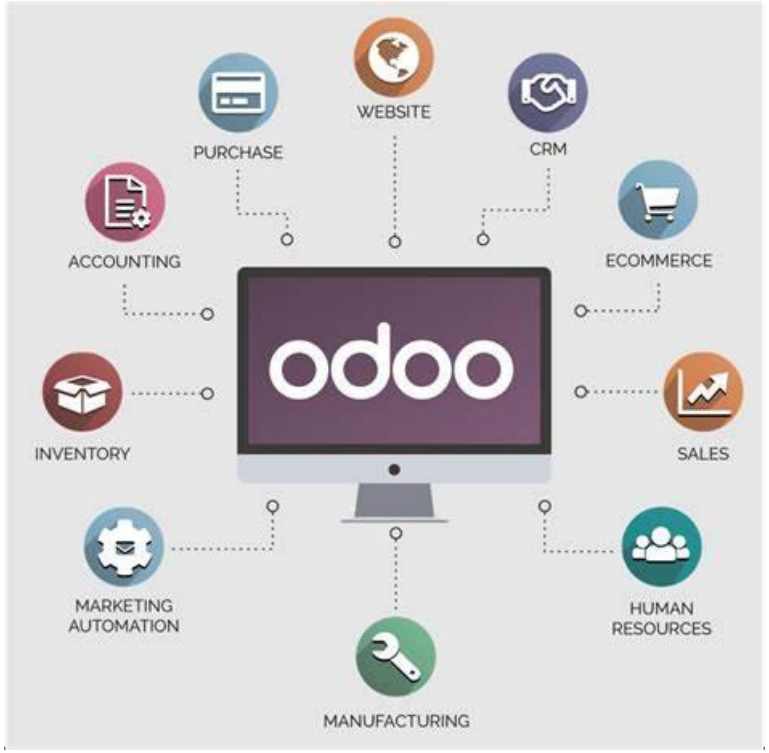
C’est une SAS qui réalise des projets de recherche et de développement en informatique. Actuellement, Innovative Labs sur 3 volets :

- Le développement d’extensions pour les ERPs, essentiellement des applications mobiles, des plates-formes de commerce électronique et des solutions transversales pour les entreprises appartenant à certains secteurs d’activité
- Le développement de jeux 2D et 3D, notamment des jeux du style Serious Games et des jeux en réseau.

Le développement d’applications intelligentes qui utilisent les agents conversationnels (chatbots) intelligents comme ChatGPT et Google Gemini.



2. CADRE DE TRAVAIL



Dans le cadre de cette alternance, l'objectif dans un premier temps était de mettre en pratique les compétences en téléchargeant Odoo, une plateforme open source de gestion intégrée, et en créant une base de données initiale nommée "baseTest1" avec les modules de Ventes, Inventaire et Facturation. L'utilisation de pgAdmin, une application open source de développement et d'administration pour PostgreSQL, est également impliquée pour gérer les données PostgreSQL utilisées par Odoo. De plus,

la communication avec Odoo est établie à travers des pages PHP en utilisant l'API d'Odoo.

Ensuite dans la société Innovative Labs, j’ai intégré l’équipe responsable du développement d’extensions pour les ERPs. J’ai participé ainsi au développement d’une application mobile de type Tableau de bord temps réel. Cette application Android est connectée à l’ERP Odoo à travers sa base de données (PostgreSQL) afin d’en extraire des données permettant d’effectuer des calculs, notamment liés aux indicateurs de performances.

Dans ce cadre, j’avais les tâches suivantes :

- Créer une application Android qui se connecte à la base de données Odoo via des requêtes http
- Comprendre la structure de la base de données Odoo à interroger (Base de données PostgreSQL) afin de savoir quelles seront les tables concernées par les formules de calcul des indicateurs de performance

- Etablir les requêtes de sélection permettant d‘extraire les données de la base de données Odoo.
- Présentation des résultats de ces requêtes dans l’application mobile.

Il reste aussi d’autres éléments pour avoir des tableaux de bord de qualité professionnelle qui ont été pris en charge par les autres membres de l’équipe de développement, notamment :

- La sécurité d’accès à l’application : les tableaux de bord contiennent des informations parfois confidentielles essentielles à la prise de décision. Ils sont destinés aux responsables et pour cette raison, il est essentiel de s’assurer de l’identité de l’utilisateur. C’est pourquoi l’authentification est établie selon les techniques de Google puisque tous les employés de l’entreprise ont des comptes professionnels Google.
- La présentation graphique des indicateurs : Les valeurs des indicateurs doivent être présentées d’une manière conviviale pouvant être comprise par l’utilisateur d’un seul

regard, notamment dans la première page du tableau de bord. Pour cette raison, il y a des règles ergonomiques à respecter dans la conception de la page initiale du tableau de bord (page de signalisation)

- Structure des pages de tableau de bord : La première page du tableau de bord s'appelle page de signalisation. Elle contient des informations générales sur le domaine de gestion ciblé. Cependant, il y a des pages de niveaux inférieurs qui contiennent des informations plus détaillées, voire des informations pertinentes pour prendre la meilleure décision.
- La compatibilité avec d'autres systèmes d'exploitation mobiles comme IOS.

CONTEXTE ET JUSTIFICATION DU CHOIX DE SUJET

DEFINITION DES TABLEAUX DE BORD

Un tableau de bord, également appelé tableau de bord de gestion ou tableau de bord de performance, est un outil utilisé dans la gestion d'une entreprise ou d'une organisation

pour suivre et évaluer ses performances et ses activités. Il s'agit généralement d'un ensemble de données clés présentées de manière graphique ou sous forme de tableaux pour permettre une visualisation rapide et claire de la situation.

Les tableaux de bord peuvent inclure divers indicateurs de performance clés (KPI) pertinents pour l'entreprise ou l'organisation, tels que les ventes, les revenus, les coûts, la satisfaction client, les délais de livraison, etc. Ces indicateurs peuvent être mis à jour régulièrement, parfois en temps réel, afin de fournir une image précise et à jour de la performance.

L'objectif principal d'un tableau de bord est d'aider les décideurs à prendre des décisions éclairées en leur fournissant des informations cruciales sur la santé et la direction de l'entreprise. Ils peuvent être conçus de manière personnalisée pour répondre aux besoins spécifiques de chaque entreprise ou domaine d'activité.
Le tableau de bord joue un rôle essentiel dans la gestion d'une entreprise en fournissant aux dirigeants et aux gestionnaires une vue d'ensemble des performances de l'entreprise et de ses différentes activités. Voici quelques-uns de ses rôles principaux :

1. Suivi des performances : Le tableau de bord permet de suivre les performances clés de l'entreprise en temps réel ou à intervalles réguliers. Cela permet aux

décideurs de repérer rapidement les tendances positives ou négatives et d'identifier les domaines nécessitant une attention particulière.

2. Prise de décision : En fournissant des données et des indicateurs pertinents, le tableau de bord aide les dirigeants à prendre des décisions éclairées. Que ce soit pour allouer des ressources, ajuster des stratégies ou identifier des opportunités, le tableau de bord fournit une base factuelle pour la prise de décision.
3. Communication : Le tableau de bord peut servir de moyen de communication efficace au sein de l'entreprise. En présentant des informations de manière claire et visuelle, il permet de partager les objectifs, les performances et les progrès avec les différents niveaux de l'organisation.
4. Alignement des objectifs : En mettant en évidence les objectifs et les KPIs clés, le tableau de bord aide à aligner les différentes équipes et départements sur les objectifs stratégiques de l'entreprise. Cela favorise la cohérence et la coordination dans l'ensemble de l'organisation.
5. Évaluation de la stratégie : Le tableau de bord permet d'évaluer l'efficacité des stratégies mises en œuvre par l'entreprise. En surveillant les résultats par rapport aux objectifs fixés, il permet d'identifier ce qui fonctionne bien et ce qui nécessite des ajustements stratégiques. Pour conclure, le tableau de bord est un outil précieux pour la gestion d'une entreprise car il fournit des informations essentielles pour suivre les performances, prendre des décisions informées, communiquer efficacement, aligner les objectifs et évaluer la stratégie. Il contribue ainsi à

améliorer la performance globale de l'entreprise et à atteindre ses objectifs commerciaux.

IMPORTANCE DES TABLEAUX DE BORD DANS LA DECISION

Les tableaux de bord jouent un rôle crucial dans la prise de décision en fournissant des informations pertinentes et actualisées sur les activités et les performances de l'entreprise.

1. Centralisation des données : Les tableaux de bord agrègent des données provenant de différentes sources au sein de l'entreprise, telles que les ventes, les finances, les opérations, etc. Cette centralisation permet d'avoir une vue d'ensemble complète de l'entreprise, facilitant ainsi l'analyse et la prise de décision.
2. Visualisation claire et concise : Les tableaux de bord présentent les données de manière visuelle, souvent sous forme de graphiques, de diagrammes ou de tableaux. Cette visualisation claire et concise permet aux décideurs de comprendre rapidement les tendances, les anomalies et les relations entre les différents indicateurs de performance.
3. Identification des tendances : En surveillant les données au fil du temps, les tableaux de bord permettent d'identifier les tendances et les schémas de

comportement. Par exemple, une augmentation régulière des ventes ou une diminution des coûts peuvent indiquer l'efficacité des stratégies mises en œuvre.

4. Détection des problèmes : Les tableaux de bord peuvent mettre en évidence les domaines où les performances ne répondent pas aux attentes ou aux objectifs fixés. Par exemple, une baisse soudaine des ventes dans une région spécifique peut signaler un problème de marché ou de concurrence.
5. Suivi des objectifs : Les tableaux de bord permettent de suivre la progression par rapport aux objectifs fixés. En comparant les performances réelles aux objectifs prédéfinis, les décideurs peuvent évaluer si l'entreprise est sur la bonne voie pour atteindre ses objectifs à long terme.
6. Analyse approfondie : Les tableaux de bord peuvent fournir des outils d'analyse avancée pour explorer les données en profondeur. Par exemple, en utilisant des techniques telles que le forage de données (data mining) ou l'analyse prédictive, les décideurs peuvent identifier des tendances cachées ou prévoir les résultats futurs.
7. Adaptation des stratégies : En comprenant mieux les performances de l'entreprise à travers les tableaux de bord, les décideurs peuvent ajuster et adapter leurs stratégies en conséquence. Par exemple, si les données montrent une demande croissante pour un produit spécifique, l'entreprise peut décider d'investir davantage dans sa production ou sa promotion.

III. OBJECTIFS DU TABLEAU DE BORD ET SON LIEN AVEC L'ERP ODOO

CONTEXTE DU TABLEAUX DE BORD

Le tableau de bord vise à fournir une vue consolidée des performances du Projet , en mettant en évidence les principaux indicateurs de performance et en facilitant la prise de décision. Créer un tableau de bord pour communiquer avec l'ERP Odoo permet de visualiser les données de manière claire et intuitive, de personnaliser les indicateurs de performance, de centraliser les informations, de faciliter la prise de décision, de favoriser la communication et d'intégrer d'autres outils pour une meilleure efficacité opérationnelle.

III.II MODULES ODOO A PRENDRE EN CONSIDERATION

Pour ce faire, j'ai choisi les modules suivants pour mon site :

1. Ventes (Sales) : Ce module permet de gérer tout le processus de vente, y compris la création de devis, les commandes clients, la facturation et le suivi des paiements.

Intégrer ce module dans votre application vous permettra de traiter efficacement les commandes des clients et de suivre les ventes en temps réel.

2. Stock (Inventory) : Le module de gestion des stocks d'Odoo vous permet de suivre les niveaux de stock de vos livres en temps réel. Vous pouvez gérer les entrées et les sorties de stock, effectuer des réapprovisionnements automatiques et suivre les mouvements de stock. Cela garantit que vous disposez toujours des livres disponibles pour répondre à la demande des clients.
3. Le CRM, ou Customer Relationship Management : désigne un ensemble de pratiques, de stratégies et de technologies utilisées par une entreprise pour gérer ses interactions avec ses clients actuels et potentiels. L'objectif principal du CRM est de développer et de maintenir de bonnes relations avec les clients, ce qui peut conduire à une fidélisation accrue, à une augmentation des ventes et à une meilleure satisfaction client.
4. Gestion des produits (Product Management) : Ce module vous permettra de gérer tous les aspects liés à vos produits, y compris leur création, leur modification, leur classification par catégorie, ainsi que la gestion des prix et des descriptions.
5. Gestion de l'Inventaire : désigne la gestion des stocks et des mouvements de produits au sein de votre entreprise. Le module d'inventaire dans Odoo permet de suivre et de contrôler les quantités de produits disponibles, ainsi que de gérer les mouvements de stocks tels que les réceptions, les livraisons, les transferts entre entrepôts, etc.

6. Gestion de la Facturation : fait référence au processus de création et de gestion des factures pour les transactions commerciales effectuées par votre entreprise. Le module de facturation dans Odoo offre une gamme de fonctionnalités pour simplifier ce processus.

CHAPITRE II : CAHIER DE CHARGES

DEFINITION DES BESOINS ET DES FONCTIONNALITES DU TABLEAU DE BORD

I.I. EXIGENCES D’UN BON TABLEAU DE BORD

Un bon tableau de bord doit répondre à plusieurs exigences pour être efficace et utile pour les décideurs. Voici une liste des fonctionnalités attendues et des informations à afficher dans un tableau de bord :

Fonctionnalités attendues :

Personnalisation : Le tableau de bord doit être personnalisable pour répondre aux besoins spécifiques de l'entreprise et des utilisateurs.

Actualisation automatique : Les données doivent être mises à jour automatiquement, de préférence en temps réel ou à intervalles réguliers.

Visualisation graphique : Utilisation de graphiques, de diagrammes et d'autres éléments visuels pour faciliter la compréhension des données.

Interactivité : Capacité à interagir avec les données, par exemple en permettant de filtrer, de zoomer ou de sélectionner des éléments spécifiques.

Accessibilité multi-appareils : Le tableau de bord doit être accessible sur différents appareils, y compris les ordinateurs de bureau, les tablettes et les smartphones.

Sécurité : Assurer la sécurité

Alertes et notifications : Capacité à définir des alertes pour signaler les anomalies ou les événements importants dans les données.

Exportation des données : Possibilité d'exporter les données affichées dans le tableau de bord sous forme de fichiers CSV, Excel ou PDF.

Analyse de données avancée : Intégration d'outils d'analyse avancée tels que le forage de données (data mining), l'analyse prédictive ou la modélisation statistique.

Historique des données : Archivage des données précédentes pour permettre une analyse comparative sur plusieurs périodes.

Collaboration : Fonctionnalités de partage et de collaboration pour permettre aux utilisateurs de commenter, de discuter et de collaborer sur les données.
Compatibilité avec d'autres systèmes : Intégration avec d'autres systèmes et applications utilisés par l'entreprise pour une vue d'ensemble complète.

I.II. INDICATEURS DE PERFOMANCES PERTINENTS A IMPLEMENTER

Les indicateurs de performances identifiés incluent le temps de cycle, le temps passé, les jours de production vs jours calendaires, la capacité des ressources, le taux de réussite des projets, le coût du projet, le taux de satisfaction du client, le nombre de bugs, le taux de rétention des employés et le taux de rotation des employés.

Ainsi voici quelques indicateurs de performance clés (KPI) pertinents à suivre via un tableau de bord :

1. Taux de conversion : Pourcentage de visiteurs du site web qui effectuent un achat. Cela permet de mesurer l'efficacité de votre site en transformant les visiteurs en clients.

2. Chiffre d'affaires : Montant total des ventes réalisées sur votre site web sur une période donnée.
3. Panier moyen : Montant moyen dépensé par chaque client lors d'une transaction. Cela peut aider à identifier les opportunités de ventes incitatives ou de promotions.
4. Taux de rebond : Pourcentage de visiteurs qui quittent votre site web après avoir consulté une seule page. Un taux de rebond élevé peut indiquer des problèmes d'engagement ou de convivialité sur votre site.
5. Taux d'abandon de panier : Pourcentage de clients qui ajoutent des articles à leur panier mais n'achètent pas. Cela peut révéler des obstacles dans le processus de paiement ou des problèmes de prix.
6. Taux de satisfaction client : Mesure de la satisfaction globale des clients, généralement basée sur des enquêtes ou des évaluations post-achat.
7. Taux de retour : Pourcentage des ventes qui sont retournées par les clients. Un taux de retour élevé peut indiquer des problèmes de qualité ou de description des produits.
8. Coût d'acquisition client (CAC) : Coût moyen pour acquérir un nouveau client. Cela peut inclure les dépenses publicitaires, les coûts de marketing et les frais de développement du site web.
9. Valeur de durée de vie client (LTV) : Valeur moyenne des ventes qu'un client génère sur toute sa durée de vie en tant que client. Cela aide à évaluer la rentabilité à long terme des clients.

10. Taux de rétention client : Pourcentage de clients qui reviennent et achètent à nouveau sur votre site web. Une bonne rétention client est essentielle pour la croissance à long terme de votre entreprise.
11. Taux de clics (CTR) : Pourcentage de visiteurs du site web qui cliquent sur des liens spécifiques, tels que des annonces, des produits recommandés, etc. Cela peut aider à évaluer l'efficacité de vos stratégies de marketing en ligne.
12. Taux de conversion par source de trafic : Pourcentage de visiteurs convertis en clients en fonction de la source de leur trafic (organique, payant, direct, etc.). Cela permet d'identifier les canaux de marketing les plus performants.

I. SPECIFICATIONS TECHNIQUES ET CONTRAINTES DU PROJET

II.I. STRUCTURE LOGICIELLE DE L'ERP ODOO

L'ERP Odoo est basé sur une architecture logicielle flexible et modulaire, ce qui en fait un système puissant et adaptable pour les entreprises de toutes tailles et de tous secteurs. Voici les principaux composants de l'architecture logicielle d'Odoo :

1. Framework Odoo : Au cœur de l'architecture d'Odoo se trouve son framework open-source, basé sur le langage de programmation Python. Ce framework fournit une base solide pour le développement d'applications modulaires et évolutives.

2. Modularité : Odoo est organisé en différents modules fonctionnels, chacun couvrant un aspect spécifique de la gestion d'entreprise, tels que la comptabilité, la gestion des ventes, des achats, des stocks, des ressources humaines, etc. Les modules peuvent être activés ou désactivés en fonction des besoins spécifiques de l'entreprise.
3. Base de données PostgreSQL : Odoo utilise la base de données PostgreSQL comme backend pour stocker toutes les données de l'entreprise. PostgreSQL est un système de gestion de base de données robuste et performant, offrant une sécurité, une fiabilité et des performances élevées.
4. Interface utilisateur : Odoo propose une interface utilisateur moderne et conviviale, accessible via un navigateur web. L'interface utilisateur est hautement personnalisable et permet aux utilisateurs de naviguer facilement entre les différents modules et fonctionnalités.
5. Web Services : Odoo expose ses fonctionnalités via des services web RESTful, ce qui permet une intégration facile avec d'autres systèmes et applications externes.
6. Rapports et Business Intelligence : Odoo intègre des fonctionnalités de reporting et d'analyse des données, permettant aux utilisateurs de générer des rapports personnalisés, des tableaux de bord et des analyses pour prendre des décisions éclairées.
7. Extensions et Personnalisation : Odoo offre une grande flexibilité pour étendre et personnaliser ses fonctionnalités. Les développeurs peuvent créer de nouveaux

modules, modifier des modules existants, créer des workflows personnalisés et intégrer des applications tierces selon les besoins spécifiques de l'entreprise.

8. Système de sécurité : Odoo intègre un système de sécurité robuste, permettant de définir des règles d'accès, des permissions et des rôles utilisateur pour contrôler l'accès aux données et aux fonctionnalités sensibles.

En résumé, l'architecture logicielle d'Odoo repose sur un framework modulaire et extensible, une base de données PostgreSQL, une interface utilisateur conviviale, des web services, des fonctionnalités de reporting et d'analyse, des capacités d'extension et de personnalisation, ainsi qu'un système de sécurité avancé. Cette architecture permet à Odoo de s'adapter aux besoins variés des entreprises et de fournir une solution complète et intégrée pour la gestion d'entreprise.

II.II. CHOIX DES BONS OUTILS POUR COMMUNIQUER AVEC LA BASE DE DONNEES D’ODOO

Pour interagir efficacement avec la base de données d'Odoo, il est primordial de sélectionner des outils qui sont compatibles avec PostgreSQL, le système de gestion de base de données utilisé par Odoo. Parmi ces outils, on peut citer pgAdmin, qui est une interface utilisateur graphique permettant de gérer les bases de données PostgreSQL de manière intuitive.

En parallèle, pour communiquer avec Odoo depuis notre application il était nécessaire de créer des pages PHP qui utilisent l'API d'Odoo. Cette API offre des fonctionnalités qui permettent de récupérer, modifier et ajouter des données dans Odoo de manière sécurisée et efficace.

Le choix de pgAdmin pour interagir avec Odoo est souvent recommandé en raison de plusieurs facteurs :

Compatibilité avec PostgreSQL : Odoo utilise PostgreSQL comme système de gestion de base de données. pgAdmin est spécifiquement conçu pour fonctionner avec PostgreSQL, offrant ainsi une compatibilité optimale pour la gestion des bases de données utilisées par Odoo.

Interface utilisateur conviviale : pgAdmin propose une interface utilisateur graphique conviviale qui facilite la gestion et l'administration des bases de données PostgreSQL. Cela rend la navigation et la manipulation des données plus intuitives, ce qui est particulièrement utile pour les utilisateurs qui ne sont pas familiers avec les commandes SQL.

1. Fonctionnalités avancées : pgAdmin offre une gamme de fonctionnalités avancées pour gérer les bases de données PostgreSQL, telles que la création et la

modification de tables, l'exécution de requêtes SQL, la sauvegarde et la restauration de données, etc. Ces fonctionnalités sont essentielles pour maintenir et optimiser la base de données d'Odoo.

2. Support communautaire : pgAdmin bénéficie d'une communauté active d'utilisateurs et de développeurs, ce qui signifie qu'il est relativement facile d'obtenir de l'aide en cas de problème ou de poser des questions sur son utilisation. Cela peut être un avantage précieux lorsque vous travaillez avec un outil critique comme la base de données d'Odoo.

II.III.OUTILS DE SECURITE DE L'APPLICATION A DEVELOPPER

Lors du développement d'une application, en particulier dans le contexte de la gestion d'entreprise où des données sensibles sont manipulées, il est crucial de mettre en place des mesures de sécurité robustes pour protéger ces données contre tout accès non autorisé ou toute violation de la confidentialité. Voici quelques-unes des mesures de sécurité clés à prendre en compte lors du développement de votre application :

1. Authentification et Gestion des Accès : Mise en place un système d'authentification solide pour vérifier l'identité des utilisateurs avant de leur permettre d'accéder à l'application.

2. Chiffrement des Données : Utilisation du chiffrement pour protéger les données sensibles lorsqu'elles sont stockées dans la base de données ou transitent sur le réseau.
3. Protection contre les Injections SQL : Protection de l’application contre les attaques par injection SQL en utilisant des requêtes paramétrées.
4. Protection contre les Attaques CSRF et XSS : Protection de l’application contre les attaques Cross-Site Request Forgery (CSRF) en utilisant des jetons CSRF pour valider les requêtes provenant de formulaires soumis par les utilisateurs

En mettant en œuvre ces mesures de sécurité dans le processus de développement de mon application, j’ai renforcer la protection des données sensibles et assurer la confiance des utilisateurs dans la sécurité de mon application.

III. ETUDE DE FAISABILITE

III.I. EXPLICATION DE CERTAINS DETAILS TECHNIQUES IMPORTANTS A PRENDRE EN CONSIDERATION LORS DU DEVELOPPEMENT DE LA SOLUTION

L'étude de faisabilité implique l'évaluation de la capacité d'intégrer les différents modules d'Odoo avec le tableau de bord, la compatibilité des outils sélectionnés avec l'architecture logicielle existante, ainsi que la

faisabilité technique et financière du projet dans son ensemble. Des tests pilotes peuvent également être réalisés pour valider la performance et la fiabilité de la solution proposée.

Quelques point rencontrés :

1. Évolutivité : Un défi majeur est de concevoir votre application de manière à ce qu'elle puisse évoluer avec l'augmentation du nombre d'utilisateurs et de données. Pour surmonter ce défi, vous pouvez adopter une architecture extensible et évolutive, utiliser des technologies de mise en cache pour améliorer les performances, et concevoir votre application pour qu'elle puisse être déployée sur une infrastructure cloud extensible.
2. Sécurité : Protéger les données sensibles et garantir la sécurité de votre application est essentiel. Pour y parvenir, vous pouvez mettre en œuvre des mesures de sécurité telles que l'authentification forte, le chiffrement des données, la validation des entrées utilisateur, la gestion des sessions sécurisées, les tests de sécurité réguliers et la conformité aux normes de sécurité telles que GDPR ou PCI DSS.
3. Intégrations : Intégrer votre application avec d'autres systèmes et applications peut être complexe en raison de la diversité des technologies, des protocoles et des formats de données. Pour faciliter les intégrations, vous pouvez utiliser des API

bien documentées et standardisées, implémenter des webhooks pour les notifications en temps réel, et envisager l'utilisation de plates-formes d'intégration tierces pour simplifier le processus d'intégration.

4. Performance : Assurer des performances optimales de votre application, en particulier lorsqu'elle gère de grandes quantités de données ou un grand nombre d'utilisateurs simultanés, peut être un défi. Vous pouvez optimiser les requêtes de base de données, mettre en cache les données fréquemment utilisées, utiliser des serveurs d'applications performants, implémenter des mécanismes de mise en cache côté client, et surveiller et optimiser les performances de manière continue à l'aide d'outils de surveillance des performances.
5. Expérience utilisateur : Offrir une expérience utilisateur fluide et intuitive est crucial pour le succès de votre application. Pour ce faire, vous pouvez effectuer des tests d'usabilité et de convivialité avec des utilisateurs réels, concevoir une interface utilisateur attrayante et facile à utiliser, optimiser les temps de chargement des pages, et recueillir les commentaires des utilisateurs pour améliorer constamment l'expérience utilisateur.
6. Maintenance et évolutivité du code : Maintenir et faire évoluer le code de votre application au fil du temps peut devenir difficile, en particulier avec une base de code importante et complexe. Pour faciliter la maintenance et l'évolutivité du code, vous pouvez adopter des pratiques de développement telles que la modularité, la

réutilisation du code, la documentation approfondie, l'automatisation des tests et des déploiements, et l'utilisation de pratiques de développement agile.

CHAPITRE III

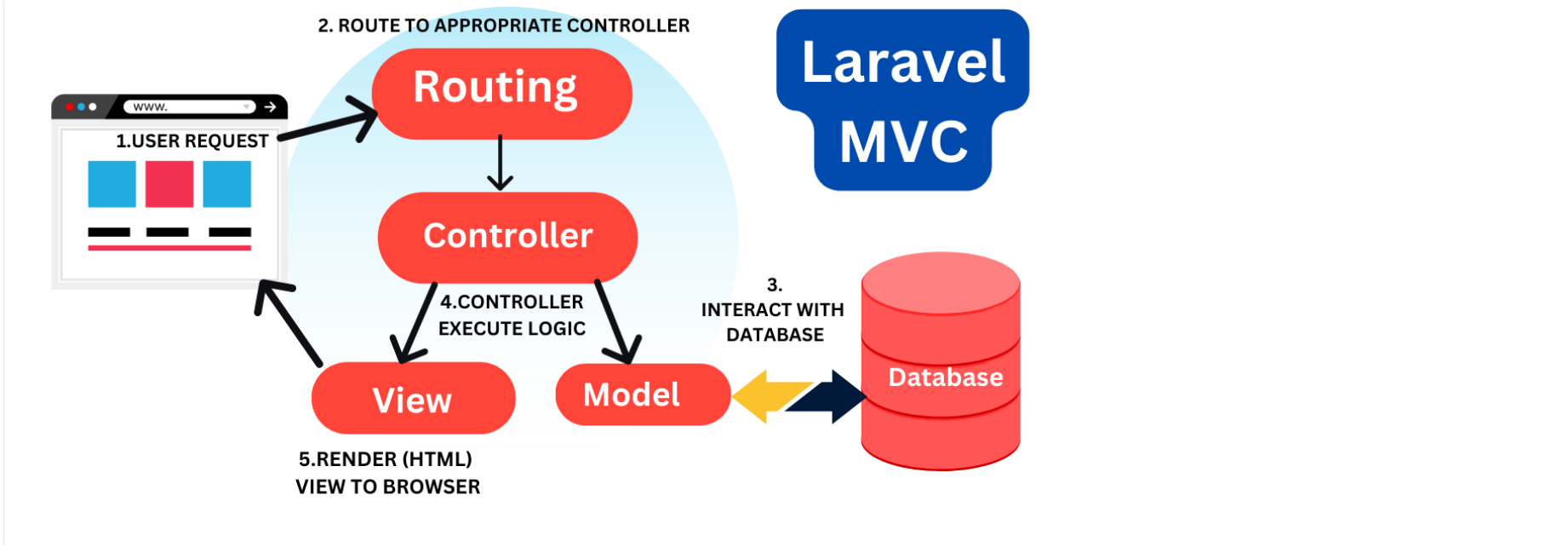
COMCEPTION

I. ARCHITECTURE GLOBALE DE L’APPLICATION

modèle MVC :

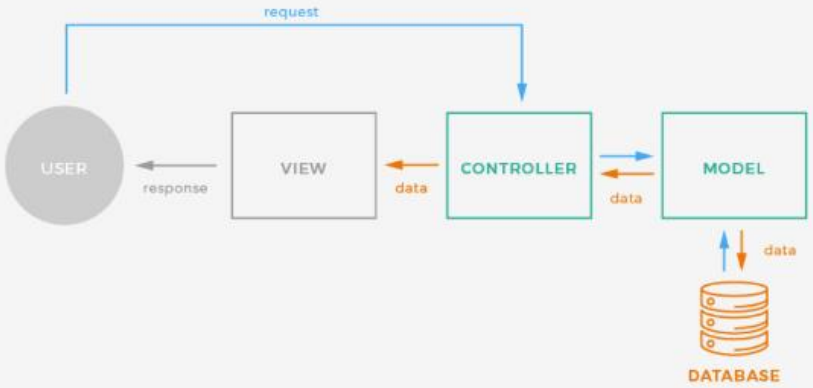
1. Modèle (Model) : Cette partie de l'application est responsable de la gestion des données et de la logique métier. Le modèle représente la structure des données de l'application et gère les opérations de lecture, d'écriture et de manipulation des données. Dans le contexte d'un site e-commerce, le modèle peut comprendre des classes qui représentent les produits, les utilisateurs, les commandes, etc. Cette couche est généralement implémentée en PHP à l'aide de Laravel, qui offre des fonctionnalités puissantes pour la gestion des bases de données et des opérations CRUD (Create, Read, Update, Delete).

2. Vue (View) : La vue est chargée de l'affichage des données à l'utilisateur et de la présentation de l'interface utilisateur. Elle est généralement constituée de fichiers HTML, CSS et JavaScript, qui définissent la structure et le style de l'interface utilisateur de l'application mobile. Dans le cas d'une application mobile de site e-commerce, la vue pourrait inclure des fichiers de modèles de conception (templates) pour afficher les produits, les pages de paiement, etc.
3. Contrôleur (Controller) : Le contrôleur agit comme un intermédiaire entre le modèle et la vue. Il gère les requêtes utilisateur, traite les données entrantes, interagit avec le modèle pour effectuer des opérations sur les données et sélectionne la vue appropriée pour afficher les résultats à l'utilisateur. En utilisant Laravel, les contrôleurs sont généralement implémentés en PHP et peuvent être organisés en fonction des différentes fonctionnalités de l'application, tels que les contrôleurs pour la gestion des produits, des utilisateurs, des commandes, etc.


2. ROUTE TO APPROPRIATE CONTROLLER
Routing
Laravel
MVC
www.
1.USER REQUEST
Controller
4.CONTROLLER
EXECUTE LOGIC
3.
INTERACT WITH
DATABASE
Database
View
Model
5.RENDER (HTML)
VIEW TO BROWSER

Architecture MVC

Laravel utilise une architecture dite **MVC** *(Model – View – Controller)*



II. CHOIX DE TECHNOLOGIE UTILISEES

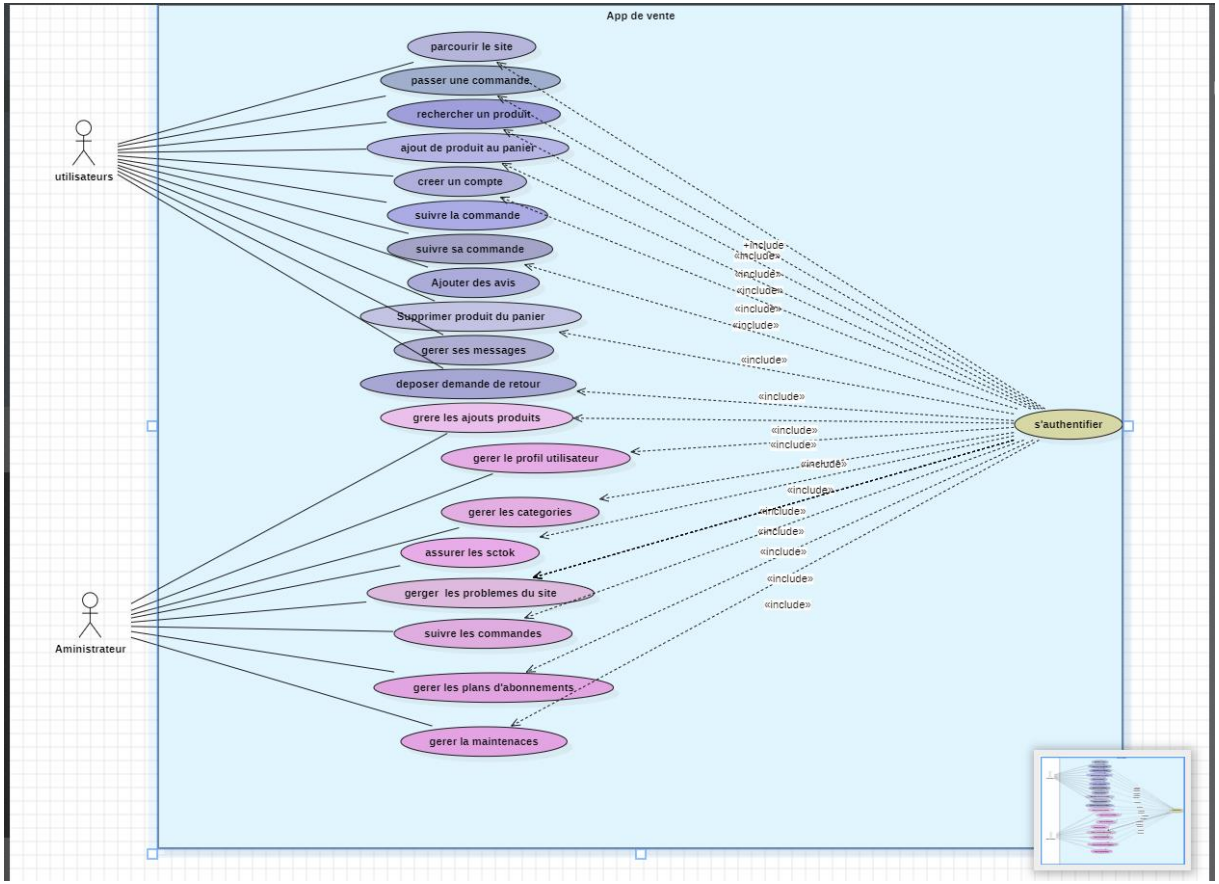
1. Laravel : Laravel est un framework PHP moderne et puissant qui offre une structure et des fonctionnalités avancées pour le développement d'applications web. Vous l'avez utilisé comme base pour développer le backend de votre application mobile de site e-commerce. Laravel facilite la gestion des routes, des contrôleurs, des modèles et des vues, ainsi que l'interaction avec la base de données.
2. PHP : PHP est un langage de programmation côté serveur largement utilisé pour le développement web. Vous avez utilisé PHP en combinaison avec Laravel pour

implémenter la logique métier de votre application, manipuler les données, générer des pages web dynamiques, etc.

3. Visual Studio Code : Visual Studio Code est un éditeur de code source léger et puissant développé par Microsoft. Vous l'avez utilisé comme votre environnement de développement intégré (IDE) pour écrire, éditer et déboguer votre code PHP, HTML, CSS, JavaScript, etc. Visual Studio Code offre des fonctionnalités avancées telles que l'autocomplétion, la coloration syntaxique, le débogage intégré et l'intégration avec des outils de contrôle de version comme Git.
4. Laragon : Laragon est un environnement de développement local tout-en-un pour PHP qui facilite l'installation et la configuration de votre stack de développement (PHP, Apache, MySQL/MariaDB). Vous l'avez utilisé pour mettre en place un environnement de développement local sur votre machine, ce qui vous permet de développer et de tester votre application localement avant de la déployer sur un serveur.
5. MySQL/PHPMyAdmin : MySQL est un système de gestion de base de données relationnelles largement utilisé, et PHPMyAdmin est une interface web pour gérer les bases de données MySQL. Vous avez utilisé MySQL comme votre système de gestion de base de données pour stocker les données de votre application, et PHPMyAdmin pour administrer et interagir avec ces bases de données via une interface web conviviale.

6. pgAdmin : pgAdmin est une interface graphique pour gérer les bases de données PostgreSQL. Bien que vous n'utilisiez pas directement PostgreSQL dans votre application, vous avez mentionné l'utilisation de pgAdmin pour interagir avec la base de données d'Odoo, qui utilise PostgreSQL comme système de gestion de base de données.

II. DIAGRAMMES DE CAS D’UTILISATIONS



CHAPITRE IV

DEVELOPPEMENT

I.PRESENTATION DES ETAPES DE DEVELOPPEMENT

Présentation & installation

tout d'abord nous allons procéder à l'installation de Laravel

Avant de commencer l'installation, vérifier que votre composer est bien installé : ouvrez votre terminal et tapez la commande :

```
composer -v
```

Commençons donc par créer un **nouveau projet Laravel**. Tout en restant sur votre terminal dirigez-vous vers votre dossier à la racine de votre serveur.

D:\laragon\www

Et tapez :

```
composer create-project laravel/laravel laravel1
```

Cette commande vous installera automatiquement la dernière version stable de Laravel

Test d'accès à mon projet

comment je teste l'accès à mon projet ?

utiliser la commande suivante :

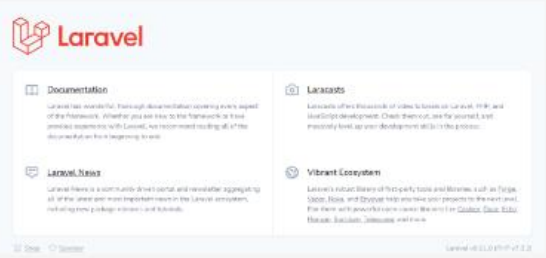
```
cd laravel1
php artisan serve
```

copier/coller le lien, que la console vous renvoie, dans votre navigateur (http://127.0.0.1:8000)

Vous devez voir cette page =>

Cela indique que votre projet

Laravel est bien installé!



Les etapes de developpement et d'execution d'une ROUTE das LARAVEL :
ex : afficher une page contenant la date et le num passés en parametres :

1- declarer une route
Route::get("/{date}/{num}",[OrderController::class,'show'])->name("order.show");

2- créer le controller OrderController.php
php artisan make:controller OrderController

3- developper la methode show($d,$n){} dans le controller OrderController.php
4- developper la vue "show.blade.php" dans le dossier /ressources/views/

nb : n'oublier pas d'importer le controller dans le fichier web.php

use App\Http\Controllers\OrderController;

5- tester votre route :
php artisan serve (pour lancer votre projet)

Voici le planning de développement de 4 pages de site web dans un projet LARAVEL :
1- créer les 4 routes
- accueil : /
- presentation : /presentation
- produits : /produits
- contact : /contact

2- developper le controller WebsiteController.php

3- developper les 4 methodes dans le controller
accueil()
presntation()
produits()
contact()

4- developper les views
accueil.blade.php
presentation.blade.php
produits..blade.php
contact.blade.php

5- developper la barre de navigation

6- je teste mon travail

Les etapes necessaires pou developper des pages avec LARAVEL :
develpper un site web de deux pages

1- la page d'accueil qui affiche une chaine : page d'accueil
2- la page contact qui affiche un formulaire de contact

---
2 routes
2 methodes
1 controlleur

1- créer un controller : SiteController | php artisan make:controller SiteController
elle va générer un fchier SiteController.php dans app/http/controllers/ OK

2- declarer deux methodes :
  2.1 - public function accueil() : retourner la view accueil.blade.php OK
  2.2 - public function contact() : retourner la view contact.blade.php OK

3- developper les views : dans le dossier resources/views/
  3.1 - la page accueil.blade.php qui contient titre : <h1>page d'accueil</h1> OK
  3.2 - la page contact.blade.php qui contient : un formulaire html OK
4- developper 2 routes : OK

  4.1 - importer la classe du controller SiteController OK
5- TEST OK
6- Ajouter un menu principal (deux lien : Accueil, Contactez-nous) OK

Routing

Qu'est-ce que le routage sous Laravel?

Laravel fournit un système de routes simple.

Déclarer une route permet de lier une URI à un code à exécuter.

Le routage est un moyen de créer une URL de requête pour votre application. La meilleure chose à propos du routage Laravel est que vous êtes libre de définir vos routes comme vous le souhaitez.

Le routage est l'un des composants clés du framework Laravel, c'est simplement un mécanisme qui effectue le mappage de vos requêtes **vers une action de contrôleur** spécifique.

Toutes les routes Laravel sont définies dans le fichier situé sous forme de fichier **/routes/web.php** , qui est automatiquement chargé par le framework,

4- Les relations LARAVEL Eloquent

Many to Many:

- Exemple : etudiant et cours => coursetudiant

```
//Création des modèles
php artisan make:model Etudiant -m
php artisan make:model Cour --migration
```

```
//Création de la table de pivot
php artisan make:migration create_cour_etudiant_table
```

- Definir deux clés étrangères(etudiant_id, cour_id) dans la migration cour_etudiant

```
class Etudiant extends Model
{
    public function cour()
    {
// Un etudiant consulte un ou plusieurs cours.

return $this->belongsToMany(Cour::class);
    }
}
```

```
class Cours extends Model
{
    public function etudiant()
    {
// Un cours est consulté par un ou plusieurs etudnts.
return $this->belongsToMany(Etudiant::class);
    }
}
```

5- Génération des CRUDs

- Après la création des models et des tables dans la base de données, nous sommes prêt maintenant pour générer les CRUD (controllers et vues)

```
php artisan make:controller CategorieController  --resource --model=Category
```

Ou créer le tout à la fois (model, migration, controller)

```
php artisan make:model Categorie  -mcr
```

- Dans le fichieir **routes/web.php** ajouter une route ressource

```
Route::resource('categories','CategorieController');
```

- Pour afficher toutes les routes

```
php artisan route:list
```

- Nous remarquons que les routes du controller categorie seront générées

Controllers

- Nous allons voir comment relier une route à un Controller plutôt que d'utiliser une fonction anonyme en deuxième paramètre de nos méthodes de routing.
- Pour commencer créons un nouveau controller

```
php artisan make:controller HomeController
```

- vos controllers se situent dans le dossier **app/Http/Controllers**

Nous allons ensuite créer une nouvelle méthode à notre class WelcomeController que l'on nommera index()

```
public function index() { return view('welcome'); }
```

Blade

- Blade est le moteur de template utilisé par Laravel.
- permettre d'utiliser du php sur notre vue mais d'une manière assez particulière.
- Pour créer un fichier qui utilise le moteur de template Blade il vous faut ajouter l'extension ".blade.php".
- Comme nous l'avons vu dans la présentation de l'architecture de Laravel, les fichiers de vos vues se situent dans le dossier resources/views.

Bonjour.

Blade

Comment ecrire un commentaire?

```
{{-- ceci est un commentaire --}}
```

Comment déclarer une variable?

```
@php
  $nom_var = "bonjour"
@endphp
```

Taches
choisir un template html
selection des pages html
integration du template dans votre projet laravel : creation de **template.bl**
decomposer le fichier template sur plusieurs fichier et en la directive **@inc**
detecter les zone dynamique de votre template en utilisant **@yield()**
importer les fichiers css et js dans votre template en utlisant **{{ asset()}}**
page 1 : accueil
page2 : produits
page 3 : page detail produit
page 4 : page panier
page 5 : page checkout
page 6 : page mon compte
page 7 : inscription
page 8 : connexion
fonctionnalité : icone panier
fonctionnalité : bouton ajouter au panier
fonctionnalité : modifier panier

Taches
choisir un template html
selection des pages html
integration du template dans votre projet laravel : creation de **template.bl**
decomposer le fichier template sur plusieurs fichier et en la directive **@inc**
detecter les zone dynamique de votre template en utilisant **@yield()**
importer les fichiers css et js dans votre template en utlisant **{{ asset()}}**
page 1 : accueil
page2 : produits
page 3 : page detail produit
page 4 : page panier
page 5 : page checkout
page 6 : page mon compte
page 7 : inscription
page 8 : connexion
fonctionnalité : icone panier
fonctionnalité : bouton ajouter au panier
fonctionnalité : modifier panier

Code source du projet laravel fait en classe : integration de template professionnel

https://we.tl/t-LkjqXUlM1v

Etapes à suivre :

pour integrer un template dans laravel

1- choix du template : ok

2- deplacer les fichier selectionner dans votre projet : objectif d'accessibilité

3- deplacer le dossier contenant les fichier css, js, img, fonts

dans le dossier public/ de larvel

4- commencer à integrer le template de la page accueil

5- décomposer le code source du template en plusieurs pages

6- tester la page d'accueil avec l'integration de son contenu

7- developper le menu principal du site web

8- developper la page presentation

9- developper la page contact

10- la page produits

Code source du projet laravel fait en classe : integration de template professionnel

https://we.tl/t-LkjqXUlM1v

Etapes à suivre :

pour integrer un template dans laravel

1- choix du template : ok

2- deplacer les fichier selectionner dans votre projet : objectif d'accessibilité

3- deplacer le dossier contenant les fichier css, js, img, fonts

dans le dossier public/ de larvel

4- commencer à integrer le template de la page accueil

5- décomposer le code source du template en plusieurs pages

6- tester la page d'accueil avec l'integration de son contenu

7- developper le menu principal du site web

8- developper la page presentation

9- developper la page contact

10- la page produits

1- Création et configuration de base de données avec LARAVEL

LARAVEL permet une intégration facile avec les BD en utilisant soit le langage **SQL brut** soit **l'ORM Eloquent.**

LARAVEL utilise **PDO** (PHP Data Objects) pour gérer les requêtes SQL.

1- créer une base de données : ouvrir **phpmyadmin** de votre serveur et céer la base de données de votre projet LARAVEL

2- configurer une base de données dans LARAVEL :

ouvrir le fichier **.env** sur la racine de votre projet et

modifier le bloc suivant :

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laraveldb
DB_USERNAME=root
DB_PASSWORD=
```

1- Création et configuration de base de données avec LARAVEL

LARAVEL permet une intégration facile avec les BD en utilisant soit le langage **SQL brut** soit **l'ORM Eloquent.**

LARAVEL utilise **PDO** (PHP Data Objects) pour gérer les requêtes SQL.

1- créer une base de données : ouvrir **phpmyadmin** de votre serveur et céer la base de données de votre projet LARAVEL

2- configurer une base de données dans LARAVEL :

ouvrir le fichier **.env** sur la racine de votre projet et

modifier le bloc suivant :

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laraveldb
DB_USERNAME=root
DB_PASSWORD=
```

Nous allons commencer par la création d'un model éloquent qui sera généré dans le dossier **App/Models**

en utilisant la commande suivante :

Php artisan make:model Produit -m

-m ou --migration : cela génère aussi la migration

- Allez dans le nouveau fichier de migration du model Produit et ajouter la déclaration des champs de votre table produits

Nous allons commencer par la création d'un model éloquent qui sera généré dans le dossier **App/Models**

en utilisant la commande suivante :

Php artisan make:model Produit -m

-m ou --migration : cela génère aussi la migration

- Allez dans le nouveau fichier de migration du model Produit et ajouter la déclaration des champs de votre table produits

3- Les migrations de base de données dans LARAVEL

Les migrations dans laravel présente un processus de gestion de votre base de données

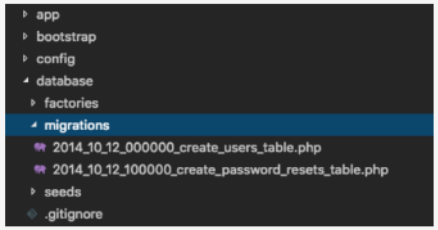
Pour créer une migration vous pouvez utiliser la commande suivante :

php artisan make:migration create_produits_table

Les migrations seront générées dans le dossier databases/migrations et contienent un horodatage qui permet au framework de déterminer l'ordre d'execution des migrations

Exécuter les migration vous pouvez utiliser la commande suivante :

php artisan migrate **=> table créée dans la base de données**



4- Les relations LARAVEL Eloquent

One to Many:

- Exemple : produit et categorie

Créer une méthode **produit()** dans le modèle **Category** qui retourne la methode **hasMany()**

Et une méthode **categorie()** dans le modèle **Produit** qui retourne la methode **belongsTo()**

```
class Categorie extends Model
{
public function produit()
   {
      //return $this->hasMany(Produit::class);
      return $this->hasMany('App\Models\Produit');
   }
}
```

```
class Produit extends Model
{
public function category()
   {
      //return $this->belongsTo(Categorie::class);
      return $this-> belongsTo('App\Models\Category');
   }
}
```

4- Les relations LARAVEL Eloquent

One to Many:

- Exemple : produit et categorie

Créer une méthode **produit()** dans le modèle **Category** qui retourne la methode **hasMany()**

Et une méthode **categorie()** dans le modèle **Produit** qui retourne la methode **belongsTo()**

```
class Categorie extends Model
{
public function produit()
   {
      //return $this->hasMany(Produit::class);
      return $this->hasMany('App\Models\Produit');
   }
}
```

```
class Produit extends Model
{
public function category()
   {
      //return $this->belongsTo(Categorie::class);
      return $this-> belongsTo('App\Models\Category');
   }
}
```

4- Les relations LARAVEL Eloquent

Many to Many:

- Exemple : etudiant et cours => coursetudiant

```
//Création des modèles
php artisan make:model Etudiant -m
php artisan make:model Cour --migration
```

```
//Création de la table de pivot
php artisan make:migration create_cour_etudiant_table
```

- Definir deux clés étrangères(etudiant_id, cour_id) dans la migration cour_etudiant

```
class Etudiant extends Model
{
    public function cour()
    {
// Un etudiant consulte un ou plusieurs cours.

return $this->belongsToMany(Cour::class);
    }
}
```

```
class Cours extends Model
{
    public function etudiant()
    {
// Un cours est consulté par un ou plusieurs etudnts.
return $this->belongsToMany(Etudiant::class);
    }
}
```

4- Les relations LARAVEL Eloquent

Many to Many:

- Exemple : etudiant et cours => coursetudiant

```
//Création des modèles
php artisan make:model Etudiant -m
php artisan make:model Cour --migration
```

```
//Création de la table de pivot
php artisan make:migration create_cour_etudiant_table
```

- Definir deux clés étrangères(etudiant_id, cour_id) dans la migration cour_etudiant

```
class Etudiant extends Model
{
    public function cour()
    {
// Un etudiant consulte un ou plusieurs cours.

return $this->belongsToMany(Cour::class);
    }
}
```

```
class Cours extends Model
{
    public function etudiant()
    {
// Un cours est consulté par un ou plusieurs etudnts.
return $this->belongsToMany(Etudiant::class);
    }
}
```

5- Génération des CRUDs

- Après la création des models et des tables dans la base de données, nous sommes prêt maintenant pour générer les CRUD (controllers et vues)

```
php artisan make:controller CategorieController  --resource --model=Category
```

Ou créer le tout à la fois (model, migration, controller)

```
php artisan make:model Categorie  -mcr
```

- Dans le fichieir **routes/web.php** ajouter une route ressource

```
Route::resource('categories','CategorieController');
```

- Pour afficher toutes les routes

```
php artisan route:list
```

- Nous remarquons que les routes du controller categorie seront générées

5- Génération des CRUDs

- Après la création des models et des tables dans la base de données, nous sommes prêt maintenant pour générer les CRUD (controllers et vues)

```
php artisan make:controller CategorieController  --resource --model=Category
```

Ou créer le tout à la fois (model, migration, controller)

```
php artisan make:model Categorie  -mcr
```

- Dans le fichieir **routes/web.php** ajouter une route ressource

```
Route::resource('categories','CategorieController');
```

- Pour afficher toutes les routes

```
php artisan route:list
```

- Nous remarquons que les routes du controller categorie seront générées

Blade

- Blade est le moteur de template utilisé par Laravel.
- permettre d'utiliser du php sur notre vue mais d'une manière assez particulière.
- Pour créer un fichier qui utilise le moteur de template Blade il vous faut ajouter l'extension ".blade.php".
- Comme nous l'avons vu dans la présentation de l'architecture de Laravel, les fichiers de vos vues se situent dans le dossier resources/views.

Bonjour.

Blade

Comment ecrire un commentaire?

```
{{-- ceci est un commentaire --}}
```

Comment déclarer une variable?

```
@php
  $nom_var = "bonjour"
@endphp
```

Les etapes de developpement et d'execution d'une ROUTE das LARAVEL :
ex : afficher une page contenant la date et le num passés en parametres :

1- declarer une route
Route::get("/{date}/{num}",[OrderController::class,'show'])->name("order.show");

2- créer le controller OrderController.php
php artisan make:controller OrderController

3- developper la methode show($d,$n){} dans le controller OrderController.php
4- developper la vue "show.blade.php" dans le dossier /ressources/views/

nb : n'oublier pas d'importer le controller dans le fichier web.php

use App\Http\Controllers\OrderController;

5- tester votre route :
php artisan serve (pour lancer votre projet)

Voici le planning de développement de 4 pages de site web dans un projet LARAVEL :
1- créer les 4 routes
- accueil : /
- presentation : /presentation
- produits : /produits
- contact : /contact

2- developper le controller WebsiteController.php

3- developper les 4 methodes dans le controller
accueil()
presntation()
produits()
contact()

4- developper les views
accueil.blade.php
presentation.blade.php
produits..blade.php
contact.blade.php

5- developper la barre de navigation

6- je teste mon travail

Nous allons commencer par la création d'un model éloquent qui sera généré dans le dossier **App/Models**

en utilisant la commande suivante :

Php artisan make:model Produit -m

-m ou --migration : cela génère aussi la migration

- Allez dans le nouveau fichier de migration du model Produit et ajouter la déclaration des champs de votre table produits

1- Création et configuration de base de données avec LARAVEL

LARAVEL permet une intégration facile avec les BD en utilisant soit le langage **SQL brut** soit **l'ORM Eloquent.**

LARAVEL utilise **PDO** (PHP Data Objects) pour gérer les requêtes SQL.

1- créer une base de données : ouvrir **phpmyadmin** de votre serveur et céer la base de données de votre projet LARAVEL

2- configurer une base de données dans LARAVEL :

ouvrir le fichier **.env** sur la racine de votre projet et

modifier le bloc suivant :

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laraveldb
DB_USERNAME=root
DB_PASSWORD=
```

3- Les migrations de base de données dans LARAVEL

Les migrations dans laravel présente un processus de gestion de votre base de données

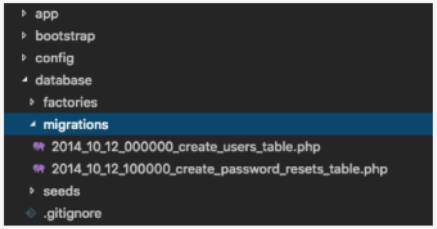
Pour créer une migration vous pouvez utiliser la commande suivante :

php artisan make:migration create_produits_table

Les migrations seront générées dans le dossier databases/migrations et contienent un horodatage qui permet au framework de déterminer l'ordre d'execution des migrations

Exécuter les migration vous pouvez utiliser la commande suivante :

php artisan migrate **=> table créée dans la base de données**



Controllers

- Nous allons voir comment relier une route à un Controller plutôt que d'utiliser une fonction anonyme en deuxième paramètre de nos méthodes de routing.
- Pour commencer créons un nouveau controller

```
php artisan make:controller HomeController
```

- vos controllers se situent dans le dossier **app/Http/Controllers**

Nous allons ensuite créer une nouvelle méthode à notre class WelcomeController que l'on nommera index()

```
public function index() { return view('welcome'); }
```

Routing

Qu'est-ce que le routage sous Laravel?

Laravel fournit un système de routes simple.

Déclarer une route permet de lier une URI à un code à exécuter.

Le routage est un moyen de créer une URL de requête pour votre application. La meilleure chose à propos du routage Laravel est que vous êtes libre de définir vos routes comme vous le souhaitez.

Le routage est l'un des composants clés du framework Laravel, c'est simplement un mécanisme qui effectue le mappage de vos requêtes **vers une action de contrôleur** spécifique.

Toutes les routes Laravel sont définies dans le fichier situé sous forme de fichier **/routes/web.php** , qui est automatiquement chargé par le framework,

Les etapes necessaires pou developper des pages avec LARAVEL :
develpper un site web de deux pages

1- la page d'accueil qui affiche une chaine : page d'accueil
2- la page contact qui affiche un formulaire de contact

---
2 routes
2 methodes
1 controlleur

1- créer un controller : SiteController | php artisan make:controller SiteController
elle va générer un fchier SiteController.php dans app/http/controllers/ OK

2- declarer deux methodes :
  2.1 - public function accueil() : retourner la view accueil.blade.php OK
  2.2 - public function contact() : retourner la view contact.blade.php OK

3- developper les views : dans le dossier resources/views/
  3.1 - la page accueil.blade.php qui contient titre : <h1>page d'accueil</h1> OK
  3.2 - la page contact.blade.php qui contient : un formulaire html OK
4- developper 2 routes : OK

   4.1 - importer la classe du controller SiteController OK
5- TEST OK
6- Ajouter un menu principal (deux lien : Accueil, Contactez-nous) OK

Taches
choisir un template html
selection des pages html
integration du template dans votre projet laravel : creation de **template.bl**
decomposer le fichier template sur plusieurs fichier et en la directive **@inc**
detecter les zone dynamique de votre template en utilisant **@yield()**
importer les fichiers css et js dans votre template en utlisant **{{ asset()}}**
page 1 : accueil
page2 : produits
page 3 : page detail produit
page 4 : page panier
page 5 : page checkout
page 6 : page mon compte
page 7 : inscription
page 8 : connexion
fonctionnalité : icone panier
fonctionnalité : bouton ajouter au panier
fonctionnalité : modifier panier

Architecture MVC

Nous allons maintenant installer **npm** qui est le gestionnaire de paquets officiel de Node.js. (**Node** est une plateforme de logiciel libre et événementielle en Javascript). Il nous permettra de compiler nos assets et d'envoyer la version compilée dans notre dossier « public ».

Pour l'installer, ouvrez votre console et lancez :

```
npm install   Puis   npm run dev
```

Pour ajouter le lien de votre style dans la vue :

```
<link href="{{ asset('css/app.css') }}" rel="stylesheet">
```

4- Les relations LARAVEL Eloquent

One to Many:

- Exemple : produit et categorie

Créer une méthode **produit()** dans le modèle **Category** qui retourne la methode **hasMany()**

Et une méthode **categorie()** dans le modèle **Produit** qui retourne la methode **belongsTo()**

```
class Categorie extends Model
{
public function produit()
   {
      //return $this->hasMany(Produit::class);
      return $this->hasMany('App\Models\Produit');
   }
}
```

```
class Produit extends Model
{
public function category()
   {
      //return $this->belongsTo(Categorie::class);
      return $this-> belongsTo('App\Models\Category');
   }
}
```

3- Les migrations de base de données dans LARAVEL

Les migrations dans laravel présente un processus de gestion de votre base de données

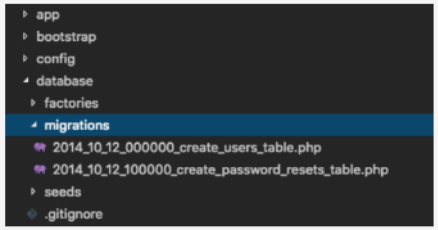
Pour créer une migration vous pouvez utiliser la commande suivante :

php artisan make:migration create_produits_table

Les migrations seront générées dans le dossier databases/migrations et contienent un horodatage qui permet au framework de déterminer l'ordre d'execution des migrations

Exécuter les migration vous pouvez utiliser la commande suivante :

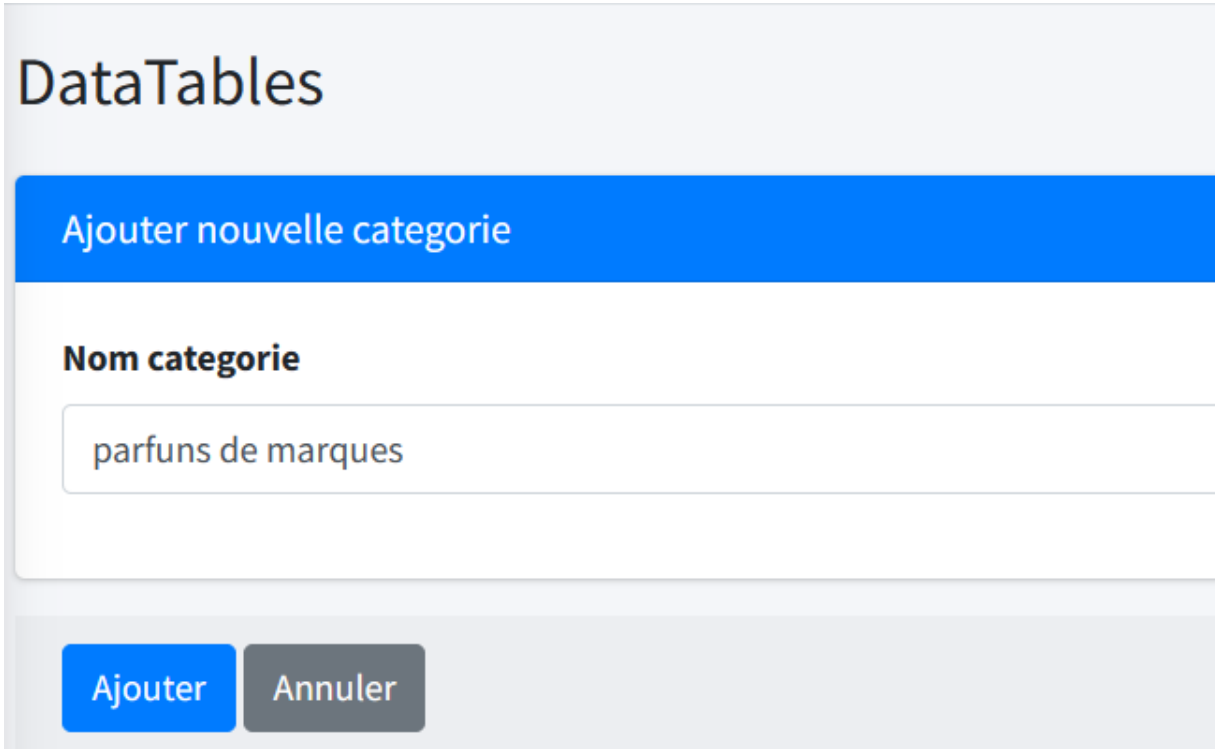
php artisan migrate **=> table créée dans la base de données**



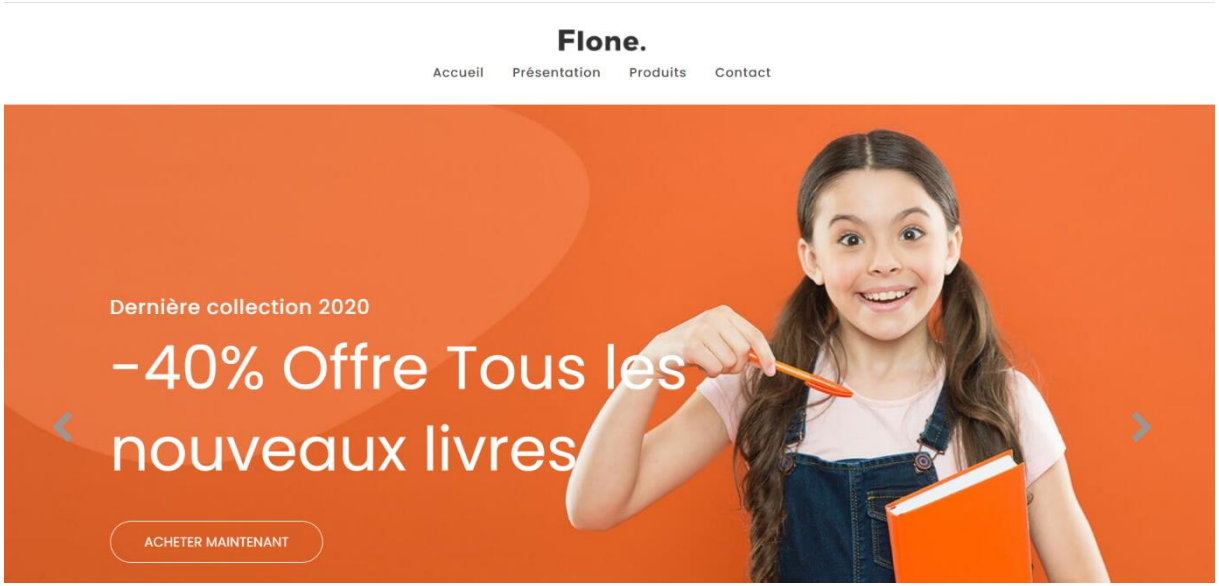
Code source du projet laravel fait en classe : integration de template professionnel
https://we.tl/t-LkjqXUlM1v
Etapes à suivre :
pour integrer un template dans laravel
1- choix du template : ok
2- deplacer les fichier selectionner dans votre projet : objectif d'accessibilité
3- deplacer le dossier contenant les fichier css, js, img, fonts
dans le dossier public/ de larvel
4- commencer à integrer le template de la page accueil
5- décomposer le code source du template en plusieurs pages
6- tester la page d'accueil avec l'integration de son contenu
7- developper le menu principal du site web
8- developper la page presentation
9- developper la page contact
10- la page produits

IIII.BILAN FINAL PERMERTTANT D'EVALUER LE RESULTAT PAR RAPPORT AUX OBJECTIFS FIXES AU DEBUT DU PROJET

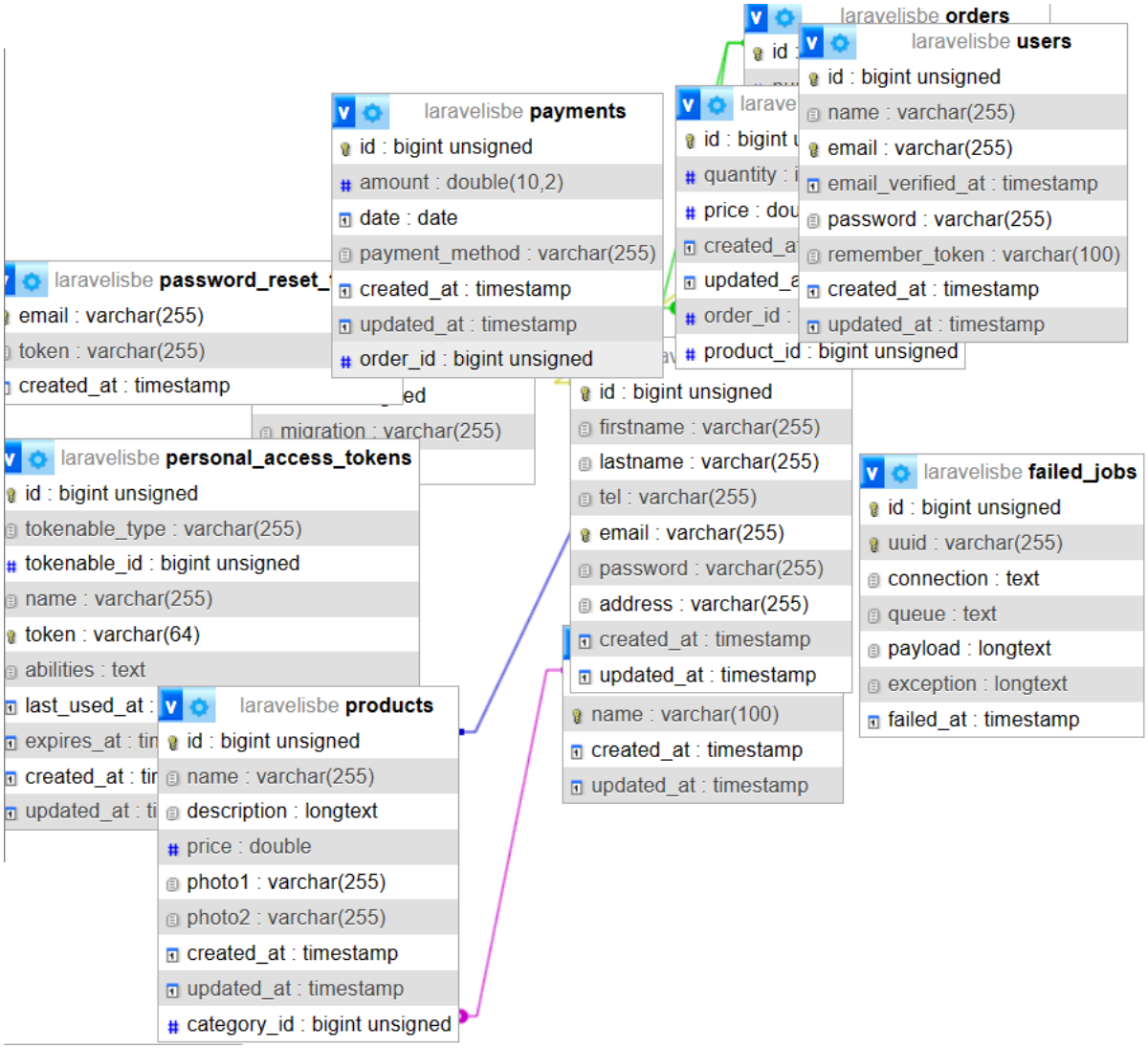
A la fin de la conception , une premier approche en ai sortie et voici quelque images du site :


DataTables
Ajouter nouvelle categorie
Nom categorie
parfuns de marques
Ajouter
Annuler

Accueil Présentation Produits Contact



IIII. MODELE DE DONNEES : STRUCTURE DES DONNEES STOCKEES ET ECHANGER AVEC ODOO


laravelisbe orders
id : bigint unsigned
laravelisbe users
id : bigint unsigned
name : varchar(255)
email : varchar(255)
email_verified_at : timestamp
password : varchar(255)
remember_token : varchar(100)
created_at : timestamp
updated_at : timestamp
laravelisbe payments
id : bigint unsigned
amount : double(10,2)
date : date
payment_method : varchar(255)
created_at : timestamp
updated_at : timestamp
order_id : bigint unsigned
laravelisbe password_reset_
email : varchar(255)
token : varchar(255)
created_at : timestamp
migration : varchar(255)
laravelisbe personal_access_tokens
id : bigint unsigned
tokenable_type : varchar(255)
tokenable_id : bigint unsigned
name : varchar(255)
token : varchar(64)
abilities : text
laravelisbe products
id : bigint unsigned
name : varchar(255)
description : longtext
price : double
photo1 : varchar(255)
photo2 : varchar(255)
created_at : timestamp
updated_at : timestamp
category_id : bigint unsigned
quantity :
price :
created_at
updated_at
order_id :
product_id : bigint unsigned
id : bigint unsigned
firstname : varchar(255)
lastname : varchar(255)
tel : varchar(255)
email : varchar(255)
password : varchar(255)
address : varchar(255)
created_at : timestamp
updated_at : timestamp
name : varchar(100)
created_at : timestamp
updated_at : timestamp
laravelisbe failed_jobs
id : bigint unsigned
uuid : varchar(255)
connection : text
queue : text
payload : longtext
exception : longtext
failed_at : timestamp

CONCLUSION

Le développement de notre application mobile de site e-commerce a été une entreprise passionnante et enrichissante, marquée par des défis stimulants, des succès gratifiants et une croissance personnelle et professionnelle significative. En rétrospective, nous pouvons tirer plusieurs enseignements importants de cette expérience.

Nous avons réussi à atteindre la plupart des objectifs que nous nous étions fixés au début du projet. L'application offre une interface conviviale et réactive, permettant aux utilisateurs de naviguer facilement à travers les produits, de passer des commandes et de gérer leurs comptes. La communication avec l'ERP Odoo a été intégrée avec succès, ouvrant la voie à une synchronisation efficace des données entre notre application et le système ERP.

Cependant, nous reconnaissons également qu'il y a toujours place à l'amélioration. Certaines fonctionnalités pourraient être optimisées pour une meilleure performance et une expérience utilisateur plus fluide. De plus, des efforts supplémentaires pourraient être déployés pour renforcer la sécurité de l'application et améliorer sa robustesse face aux éventuelles vulnérabilités.

II. Perspectives d'Amélioration

À l'avenir, nous envisageons plusieurs pistes d'amélioration pour enrichir davantage notre application mobile de site e-commerce. Parmi les perspectives d'amélioration, nous envisageons :

- L'optimisation des performances pour garantir une expérience utilisateur plus rapide et plus fluide.
- L'ajout de fonctionnalités supplémentaires pour enrichir l'expérience utilisateur, telles que des options de personnalisation des produits, des recommandations personnalisées, etc.
- La mise en œuvre de mesures de sécurité renforcées pour protéger les données des utilisateurs et prévenir les cyberattaques.
- L'exploration de nouvelles technologies et outils pour rester à jour avec les tendances du marché et offrir des fonctionnalités innovantes.

En résumé, bien que nous soyons fiers des réalisations de notre projet actuel, nous sommes également conscients des opportunités d'amélioration qui existent. En continuant à rester engagés dans l'innovation et l'excellence technique, nous sommes confiants que notre application mobile de site e-commerce continuera à évoluer et à prospérer dans un paysage numérique en constante évolution.

```
}
```