

TITRE PROFESSIONNEL
Concepteur Développeur d'Application
Niveau II

Dossier de projet

Lilian LAYRAC
EPSI
École d'ingénierie informatique

TABLE DES MATIERES

Présentation personnelle.....	6
1 - Référentiel de compétences.....	7
2 - Résumé des projets	8
2.1 - Introduction	8
2.2 – Projet IAPM StoreSwap.....	8
2.3 – Projet Mobile	8
3 – Cahier des charges	9
3.1 – Cahier des charges pour IAPM StoreSwap	9
3.1.1 – Description du cahier des charges du projet.....	9
3.1.2 – Utilisateurs des applications.....	9
3.1.3 – Tests et validation	10
3.2 – Cahier des charges pour Mobile IAPM	10
3.2.1 – Description du cahier des charges du projet.....	10
3.2.2 – Utilisateurs des applications.....	11
3.2.3 – Tests et validation	11
4 – Gestion de projet.....	12
4.1 – Gestion de projet IAPM StoreSwap.....	12
4.2 – Gestion de projet Mobile IAPM.....	14
5 – Spécifications fonctionnelles	15
5.1 – Application web StoreSwap.....	15
5.1.1 – Spécificités StoreSwap.....	15
5.1.2 – Diagramme de cas UML	16
5.1.3 – Dictionnaire de données.....	17
5.1.4 – Modèle Conceptuel de Données	18
5.1.5 – Modèle Logique de Données.....	19
5.1.6 – Conventions de Nommages.....	19
5.2 – Application desktop IAPM	20
5.2.1 – Spécificités IAPM.....	20
5.2.2 – Diagramme de cas UML	21
5.2.3 – Diagramme de Classes.....	22
5.2.3 – Base de Données.....	23
5.3 – Application Mobile.....	23
5.1.1 – Spécificités IAPM Mobile.....	23
5.1.2 – Diagramme de cas UML	25

6 – Spécifications techniques	26
6.1 – Application web StoreSwap.....	26
6.1.1 – Elaboration des contraintes techniques	26
6.1.2 – Composants d'accès aux données.....	27
6.1.3 – Principe de développement Modèle Vue Contrôleurs.....	28
6.1.4 – Les différentes balises de Bootstrap	29
6.2 – Application desktop IAPM	30
6.2.1 – Elaboration des contraintes techniques	30
6.2.2 – Composants d'accès aux données.....	30
6.3 – Application Mobile.....	31
6.3.1 – Elaboration des contraintes techniques	31
6.3.2 – Composants d'accès aux données.....	32
7 – Réalisations du candidat	34
7.1 – Application StoreSwap IAPM.....	34
7.1.1 – Choix de style pour StoreSwap	34
7.1.2 – Choix de style pour IAPM	34
7.1.3 – Coté base de données.....	35
7.1.4 – Comment est construite l'application StoreSwap.....	35
7.1.5 – Sécurité de l'application StoreSwap	36
7.1.6 – Tests Unitaires	37
7.1.7 – Intégration et création du CSS avec le framework Bootstrap	38
7.1.8 – Tests de performances de l'application StoreSwap.....	41
7.1.9 – Déploiement de l'application StoreSwap.....	43
7.1.10 – Comment est construite l'application IAPM.....	45
7.1.11 – Sécurité de l'application IAPM	46
7.1.12 – Déploiement de l'application IAPM.....	46
7.2 – Application Mobile.....	46
7.2.1 – Choix de style pour IAPM Mobile	46
7.2.2 – Sécurité de l'application IAPM Mobile	46
7.2.3 – Déploiement de IAPM Mobile	47
8 – Veille Technologique.....	48
8.1 – Les failles de php	48
9 – Situation nécessitant un travail de recherche	49
10 – Remerciements.....	50
11 – Glossaire.....	51
12 – Annexes	52
Annexes pour IAPM	52
Annexe IAPM Maquette.....	52

Annexe IAPM Charte Graphique.....	53
Annexe IAPM Connexion	53
Annexe IAPM BDD.....	54
Annexe IAPM Résultat Trigger	54
Annexe IAPM Tests Unitaires	55
Annexe IAPM App Menu.....	55
Annexe IAPM App Ajout Client.....	55
Annexe IAPM App Liste Client.....	56
Annexe IAPM App Ajout Produit.....	56
Annexe IAPM App Liste Produit.....	57
Annexe IAPM App Liste Commande	58
Annexe IAPM Arborescence	58
Annexes pour StoreSwap	59
Annexe StoreSwap Accueil	59
Annexe StoreSwap Connexion	59
Annexe StoreSwap Inscription	60
Annexe StoreSwap Boutique.....	60
Annexe StoreSwap Panier	61
Annexe StoreSwap Admin Table	61
Annexe StoreSwap Admin Table Client.....	62
Annexe StoreSwap Code Connexion	62
Annexes pour IAPM Mobile	63
Annexe IAPM Mobile Connexion & Déconnexion.....	63
Annexe IAPM Mobile Accueil	63
Annexe IAPM Mobile Client, Recherche, Ajout, Popup Success, Supprimer, Erreur Saisie.....	64
Annexe IAPM Mobile Produit, Info, Ajout, Suppression, Popup Annul Suppression.....	65
Annexe IAPM Mobile Commande, Recherche, Info	66
Annexe IAPM Mobile Statistiques	66

Présentation personnelle

Je me présente, Lilian Layrac, étudiant à l'EPSI pour le passage de mon Bachelor CDA (Concepteur développeur d'applications). Mon parcours professionnel se traduit par le passage d'un Baccalauréat STMG avec option SIG (Système d'Information et de Gestion) dans le lycée Marc Bloch, à Sérignan. Je l'ai obtenu avec mention assez-bien. Je me suis ensuite tourné vers quelque chose qui me passionne, l'informatique, en introduisant un cursus BTS SIO (Services informatiques aux organisations), dans ce même lycée. Durant ce cursus de deux ans, j'ai appris les bases du développement informatique, de par la conception de sites web, mais d'autre part, via la conception de clients lourds comme des applications métiers. Au bout d'un an de formation, j'ai choisi l'option SLAM (Solutions logicielles et applications métiers), car je suis plus attiré par le développement. À la suite de l'obtention de mon BTS, je me suis tourné vers l'EPSI pour effectuer un Bachelor CDA, en suivant un cursus d'alternance.

Mon alternance se déroule dans une petite structure nommée Optim.Solutions (anciennement appelé Algo.Solutions). Nous nous occupons d'intégrer une plateforme open-source nommé VITAM (Programme interministériel archivage numérique). Notre objectif est d'utiliser cette infrastructure, en proposant le déploiement de cette dernière (qui est très fastidieux), mais aussi de créer des portails de connexions pour y accéder. Nous assurerons également la maintenance de la solution si mise à jour il y a.

Les projets que je vais présenter dans ce dossier sont des projets personnels introduits dans une thématique professionnelle, ils ont pour but de valider le référentiel de compétences. Malheureusement, les projets en entreprise ne me permettent pas de valider les compétences, c'est pour cela que je me tourne vers des projets personnel et professionnel réalisé en amont.

En savoir plus sur moi et l'entreprise ou je travaille :

Durant mon année d'alternance chez Optim.Solutions, je me suis formé à Docker et aux technologies qui touchent l'infrastructure système. J'ai acquis des compétences de déploiements et d'optimisation d'architectures qui m'ont beaucoup aidé lors de cette année à l'EPSI. Coté développement informatique, je me suis également auto-formé à des frameworks que je ne connaissais pas, tel que Symfony par exemple (formation sur la plateforme UDEMY). Malgré tout, je suis fier d'avoir réussi à surmonter le manque d'attention de mon tuteur, car cela m'a permis de devenir plus autonome, et surtout d'apprendre davantage en termes de développement informatique.

Notre équipe est composée de 4 alternants et d'un tuteur. Nous travaillons ensemble quand le temps nous le permet, car par soucis de calendrier (nous ne sommes pas tous dans la même école), nous devons nous adapter à chacun, et surtout travailler le soir pour mettre en commun.

Notre cycle de travail était basé sur des réunions journalières, pour faire un point tous les matins, nous étions dans un premier espace de co-working. Le temps est passé, mon ancien tuteur a délégué la gérance de la société à mon tuteur actuel, et nous avons changé d'espace de co-working.

Dans ce nouvel espace, nous y sommes seulement le vendredi, le reste du temps, nous sommes en télétravail. Ce n'est pas un cadre de travail idéal, surtout vue la communication qu'il y a dans l'équipe, et surtout entre nous et nos tuteurs (ex et nouveau). Mais nous parvenons à faire avec, et nous essayons tant bien que mal de sortir la tête de l'eau.

Sur ces notes un peu maussades, je vous laisse découvrir mon dossier pour le titre professionnel Concepteur Développeur d'Application, et vous souhaite une bonne lecture.

1 - Référentiel de compétences

N° Fiche AT	Activités types	N° Fiche CP	Compétences professionnelles
1	Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité	1	Maquetter une application
		2	Développer une interface utilisateur de type desktop
		3	Développer des composants d'accès aux données
		4	Développer la partie front-end d'une interface utilisateur web
		5	Développer la partie back-end d'une interface utilisateur web
2	Concevoir et développer la persistance des données en intégrant les recommandations de sécurité	6	Concevoir une base de données
		7	Mettre en place une base de données
		8	Développer des composants dans le langage d'une base de données
3	Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité	9	Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement
		10	Concevoir une application
		11	Développer des composants métier
		12	Construire une application organisée en couches
		13	Développer une application mobile
		14	Préparer et exécuter les plans de tests d'une application
		15	Préparer et exécuter le déploiement d'une application

2 - Résumé des projets

2.1 - Introduction

Firstly, to validate the title of Designer Application Developer, I need to present you some projects. These projects validate all the competences I showed you above. They are made by different languages of programming, like PHP, HTML/CSS for the front-end, Visual Basic for the back-end, and Dart (with the framework Flutter) for the mobile app.

Our fake company name is Layweb. I'm gonna use this name during all the project to contextualize it as a professional project.

2.2 – Projet IAPM | StoreSwap

The first project is called IAPM | Storeswap, it's a project with a web app (StoreSwap), administered by the desktop app (IAPM). It's a marketplace web app where you can buy candies. Obviously, I will go deeper into the explanations further into the presentation. The desktop app IAPM can administrate the database by handle entities and their content. Both apps have the same database. I created a MySQL database managed by a WAMP server, all my work was made in local. I also created Docker containers on linux operating system, to develop my flutter app (ubuntu).

These applications were made by myself, but otherwise, I managed to make them into a professional scope, by creating workflows like Conceptual Data Model, or some Schedule Management.

2.3 – Projet Mobile

The second project is a mobile app. This mobile app use the same database as IAPM | StoreSwap project. It's a simple mobile app that display the data of the database (like customers, product or orders), but you can also add product, customers and check dashboard of statistic of the marketplace. This app communicate with a MySql server and collect data from here. The MySql container are made by a script caller docker-compose with the docker service. A PhpMyAdmin container are made too.

Also, theses docker containers work for the project IAPM | StoreSwap.

3 – Cahier des charges

3.1 – Cahier des charges pour IAPM | StoreSwap

3.1.1 – Description du cahier des charges du projet

La demande de notre client était de créer une application permettant la vente de bonbons. Cette dernière étant composée d'une interface web, ainsi qu'une interface desktop.

Concernant l'application web StoreSwap, elle est développée en PHP, HTML et CSS, elle gère la vitrine de l'activité Layweb. L'application desktop IAPM, quant-à-elle, gère tous les éléments de la base de données, elle gère donc les stocks et les utilisateurs.

Concernant l'entreprise Layweb, elle aimerait expandre son secteur d'activité, actuellement limité à un simple petit magasin de centre-ville, en créant leur propre marketplace, pour ainsi vendre des bonbons dans toute la France, l'Europe, et pourquoi pas le monde entier.

Mais pour développer une activité en ligne florissante, il faut être rigoureux et bien réfléchir à son architecture ou même sa logistique. C'est pour cela que l'entreprise Layweb a fait appel à nos services pour lui créer une interface web StoreSwap, agrémenté de son progiciel de gestion de base de données IAPM, pour faciliter les insertions de nouveaux produits ou la gestion des clients.

Nous nous occupons également des problématiques de sécurité qui ne sont pas à négliger quand nous traitons des données sensibles comme des objets à caractère personnel des comptes utilisateur.

Les besoins de l'entreprise sont :

- Avoir un site web permettant la navigation des clients
- Avoir une interface progicielle permettant la manipulation des données
- Avoir un contrôle total sur les produits, utilisateurs et commandes

Ce type d'application est totalement adapté à l'activité de l'entreprise Layweb, car elle répond parfaitement à leur attente.

A noter que le client peut ajouter de nouvelles fonctionnalités tout au long du projet, ou même ultérieurement après le livrable de ce dernier (car nous assurons la maintenance de toutes nos applications livrées). Une veille technologique est primordiale en termes de sécurité car il faut constamment corriger des failles.

3.1.2 – Utilisateurs des applications

Les différents utilisateurs des applications sont :

- pour la partie web, les clients, les administrateurs
- pour la partie desktop, les administrateurs

Un client est un utilisateur de l'application, il peut acheter des bonbons, consulter le site, se balader librement dessus mais n'a aucun droit sur l'édition de la base de données, si ce n'est la création, modification ou suppression de son propre compte utilisateur, ou la gestion de son panier. A savoir que le client navigue seulement sur le front-end StoreSwap, et ne touche jamais à l'application IAPM.

Un administrateur, coté web, peut effectuer les mêmes actions que le client, mais il peut également consulter les tables de la base de données sur sa partie administration (intégrée à l'interface web).

Un administrateur, coté IAPM, peut consulter les clients, en ajouter, modifier ou supprimer. Il peut également consulter les produits, en ajouter, modifier ou supprimer. Il peut également consulter les commandes, en ajouter, modifier ou supprimer. Il a également une interface dashboard qui permet de consulter le nombre de clients, de produits et de commandes en temps réel.

3.1.3 – Tests et validation

Les spécifications démontrées ci-dessus ont été validées par l'entreprise Layweb, car elles correspondent parfaitement aux attentes du client.

3.2 – Cahier des charges pour Mobile IAPM

3.2.1 – Description du cahier des charges du projet

Le projet IAPM Mobile consiste à développer une application mobile qui permettra de récupérer les informations des clients, des produits et des commandes, mais aussi d'ajouter ou de supprimer des clients et des produits, à partir de la base de données commune au projet IAPM | StoreSwap.

L'application sera conçue pour consulter, ajouter ou supprimer les données relatives à la vente de bonbons. Elle comprendra une interface conviviale et intuitive, adaptée aux appareils mobiles.

Le projet s'appuiera sur l'infrastructure existante du projet StoreSwap, qui est une application web développée en PHP, HTML et CSS. Cette dernière gère la vitrine de l'activité de l'entreprise Layweb. En complément, l'application mobile IAPM sera dédiée à la consultation de la base de données, comprenant les clients, les produits ainsi que les commandes.

L'objectif principal de l'entreprise Layweb est d'étendre son activité en créant sa propre marketplace, permettant la vente de bonbons à l'échelle nationale, européenne voire mondiale. Ainsi, l'application mobile IAPM sera un outil essentiel pour assurer la consultation rapide et efficace des clients, des produits et des commandes.

Dans le cadre de ce projet, nous accordons une grande importance à la sécurité des données sensibles, telles que les informations personnelles des utilisateurs. Nous mettrons en place des mesures de sécurité robustes pour protéger ces données et garantir la confidentialité des utilisateurs.

Les besoins spécifiques de l'entreprise Layweb sont les suivants :

- Disposer d'une interface mobile conviviale pour les clients
- Avoir un progiciel de gestion permettant la manipulation des données
- Assurer un contrôle total sur les produits, les utilisateurs et les commandes

L'application mobile IAPM répond parfaitement aux attentes de l'entreprise Layweb et lui permettra de développer efficacement son activité en ligne. Nous sommes conscients que de nouvelles fonctionnalités pourraient être ajoutées tout au long du projet ou même après sa livraison, et nous assurons la maintenance continue de nos applications pour répondre aux besoins évolutifs de nos clients.

En conclusion, une veille technologique constante sera maintenue pour garantir la sécurité des données et pour corriger toute vulnérabilité potentielle.

3.2.2 – Utilisateurs des applications

Les utilisateurs de l'application sont :

- des administrateurs

Un administrateur qui utilise l'application IAPM Mobile peut consulter les clients, les produits ainsi que les commandes. Cependant, pour se connecter, il doit tout d'abord fournir ses identifiants MySql pour pouvoir avoir accès à l'application. Ensuite, il pourra consulter les tables client, produit, commande ainsi que des dashboards statistiques. Il pourra également trier les clients, produits et commandes en fonction de leur noms, libelle et identifiant. Mais il pourra aussi ajouter et supprimer des clients et des produits via des interfaces dédiées.

3.2.3 – Tests et validation

Les spécifications démontrées ci-dessus ont été validée par l'entreprise Layweb, car elles correspondent parfaitement aux attentes du client.

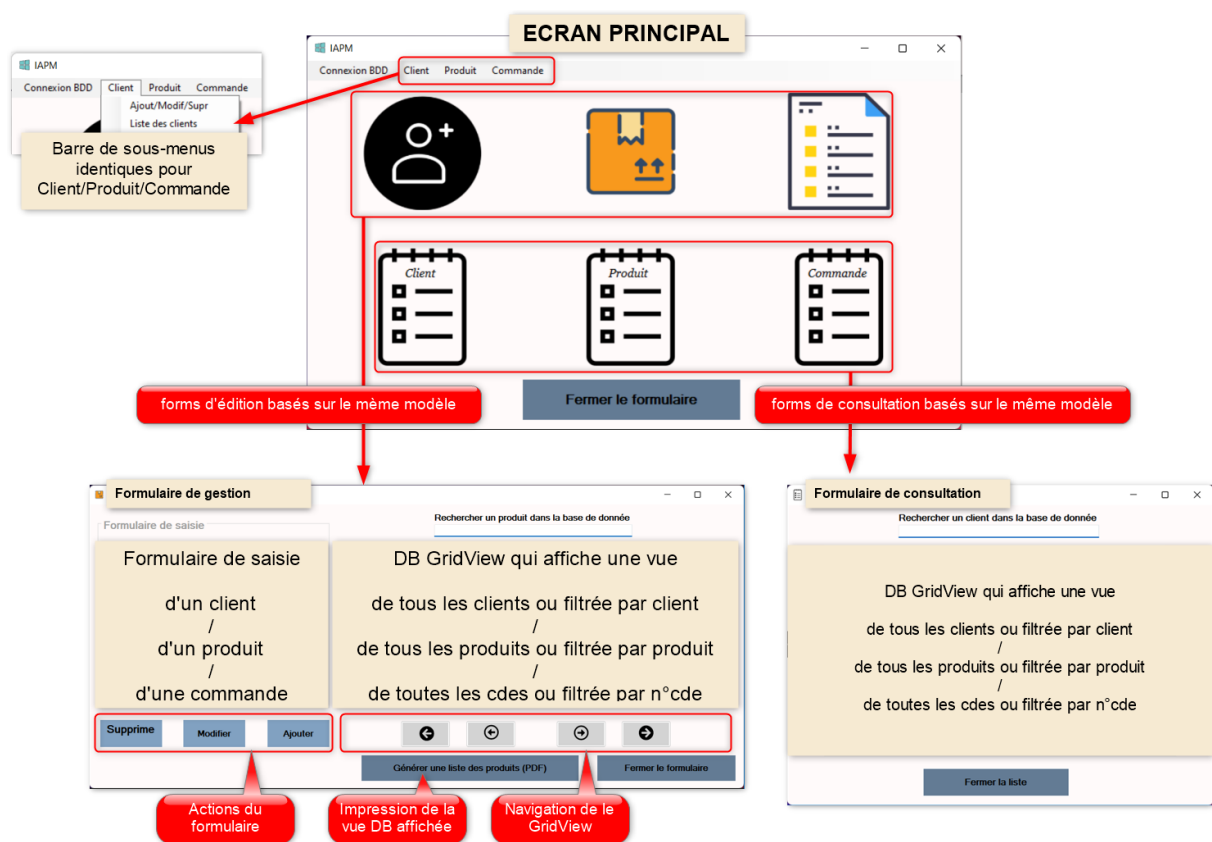
4 – Gestion de projet

4.1 – Gestion de projet IAPM | StoreSwap

Malgré le cas professionnel fictif de ce projet, j'ai tout de même décidé de travailler avec des **outils collaboratifs** telle que GitHub, pour garder un système de versionning et garder une certaine cohérence dans mes commits et ma gestion du code. Grâce à cela, j'ai réussi à rester organisé et surtout à ne pas me perdre dans la masse de travail importante.

J'ai également réalisé une **maquette** de l'application IAPM via le logiciel en ligne **Figma**, pour avoir une idée précise de l'état final du design, avant de me lancer dans la production de cette dernière.

Voici un **Storyboard** qui explique chaque item de l'application :



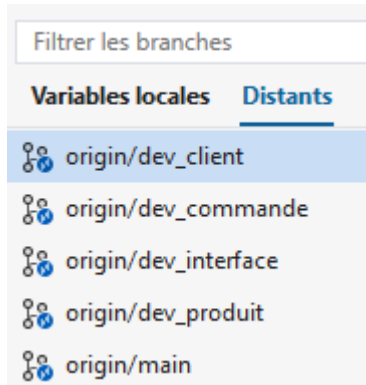
(Pour plus d'annexes, voir Annexes pour IAPM – Annexe IAPM Maquette)

J'ai également intégré la notion de **charte graphique** pour avoir une cohérence des effets visuels (graphique, design, forme des boutons, police) dans l'ensemble de l'application. Elle a également été réalisée sur Figma.

(Voir Annexes pour IAPM – Annexe IAPM Charte Graphiques)

J'ai utilisé **GitHub** pour ma gestion de projet, et j'ai donc créé différentes branches me permettant de m'organiser lors du développement de l'application IAPM Desktop.

Voici les différentes **branches** créées disponibles via l'explorateur de branches sur l'éditeur Microsoft Visual Studio :



Nous avons la branche `dev_client`, qui traite tout le développement en rapport avec la gestion des clients, la branche `dev_commande`, qui traite tout le développement en rapport avec la gestion des commandes, la branche `dev_interface`, qui traite tout le développement en rapport avec la gestion de l'interface graphique, la branche `dev_produit`, qui traite tout le développement en rapport avec la gestion des produits, et enfin la branche `main` où l'on merge le résultat final et où l'on règle les conflits pour que tout soit réuni sur une seule et même branche.

Dans un cadre professionnel, il aurait été préférable de réaliser un diagramme de Gantt. Je n'ai pas jugé nécessaire d'en créer un, car j'étais seul sur le projet et donc seul à effectuer les tâches.

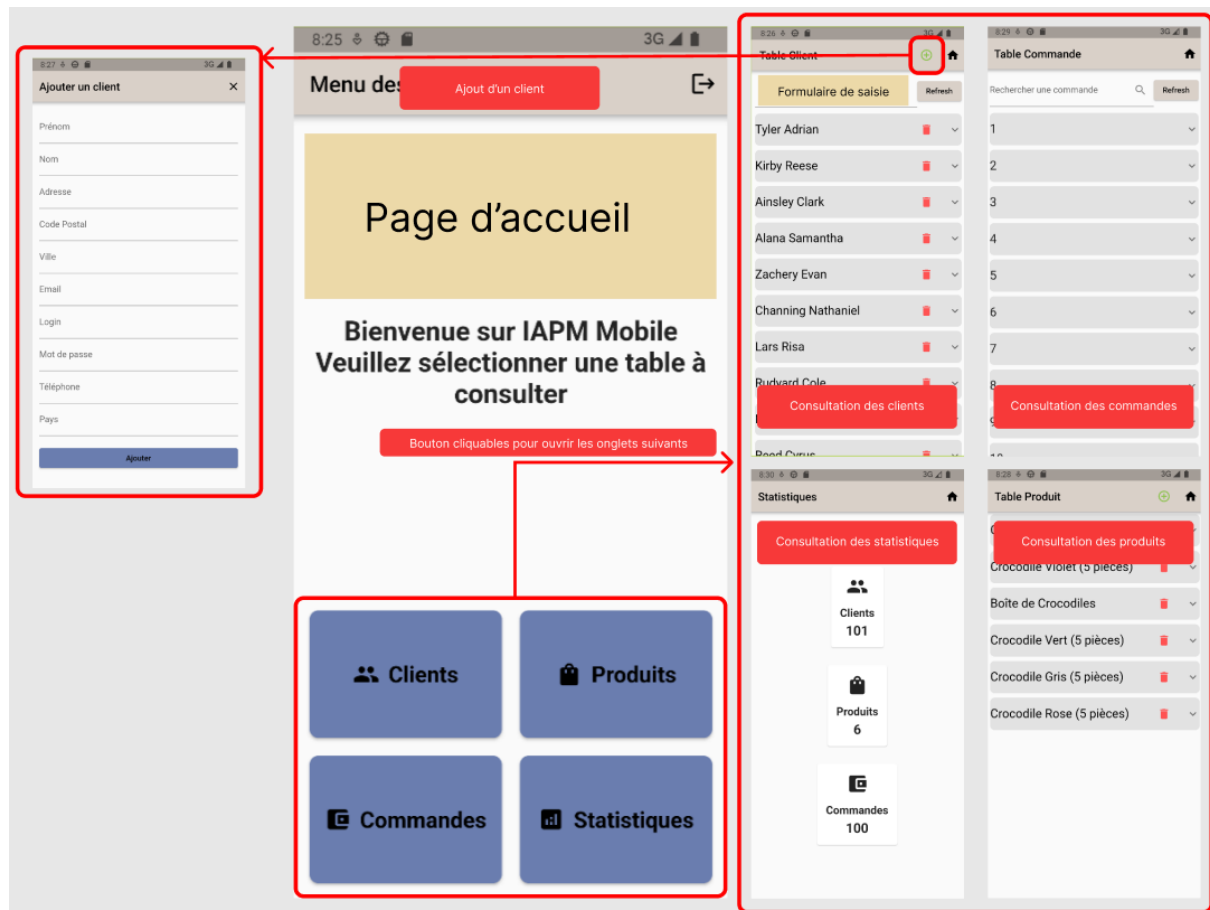
A noter que si le projet avait été en groupe, il aurait été préférable de faire un diagramme de Gantt, car cela nous permet d'avoir un suivi sur l'avancement du projet, et surtout planifier des correctifs si des délais il y a (Il y a d'autres diagrammes de gestion de projet comme le diagramme de PERT, qui permet de représenter graphiquement les tâches à accomplir, mais aussi d'y intégrer la notion temporelle, ce qui sert donc à planifier, organiser et contrôler les différentes étapes d'un projet).

J'ai également effectué des tests unitaires pour l'application desktop IAPM pour tester mes méthodes tels que `AjouterClient` ou `ModifierClient`.

Pour la partie front-end, je n'ai pas réalisé de maquettage, ni de gestion de projet avec GitHub. J'ai simplement posé les idées et coder en fonction de spécificités fonctionnelles et techniques énoncées ci-après dans les rubriques « Spécifications fonctionnelles » et « Spécifications techniques ».

4.2 – Gestion de projet Mobile IAPM

Comme énoncé ci-dessus, malgré le cas fictif, j'ai tout de même utilisé des **outils collaboratifs** comme github, également pour l'application IAPM Mobile. J'ai également réalisé un **Storyboard** expliquant le fonctionnement de l'accueil de mon application IAPM Mobile.



Sur ce **Storyboard**, on peut y apercevoir toutes les informations essentiels permettant la bonne compréhension de l'application IAPM Mobile.

Au milieu nous avons la page d'accueil, où l'on peut voir 5 boutons, 4 pour accéder aux différentes interfaces, et un dans l'AppBar permettant la déconnexion de la base de données. (A noter que pour accéder à cette page, il est obligatoire de s'être connecter avec ses identifiants MySql).

Dans le rectangle rouge à droite, vous pouvez y retrouver les 4 différentes pages, celle qui affiche les clients, celle qui affiche les commandes, celle qui affiche les statistiques et celle qui affiche les produits. Sur la page client, vous pouvez supprimer des clients si vous le désirez (Une popup de confirmation vous sera affiché pour ne pas faire d'erreur lors de la suppression d'un client). Vous avez le même procédé pour les produits. Vous avez également une barre de recherche qui permet de trier en fonction du nom pour les clients, et de l'identifiant de la commande pour les commandes. Les statistiques s'actualisent lors du rechargement de la page. Des contrôles de saisies sont effectué lors de l'ajout d'un client (rectangle rouge de gauche), pour éviter toute erreur d'insertion dans la base de données.

Vous pouvez retrouver différents screenshot de l'application IAPM Mobile.

(Voir Annexes pour IAPM Mobile – Annexe IAPM Mobile (Pages))

5 – Spécifications fonctionnelles

5.1 – Application web StoreSwap

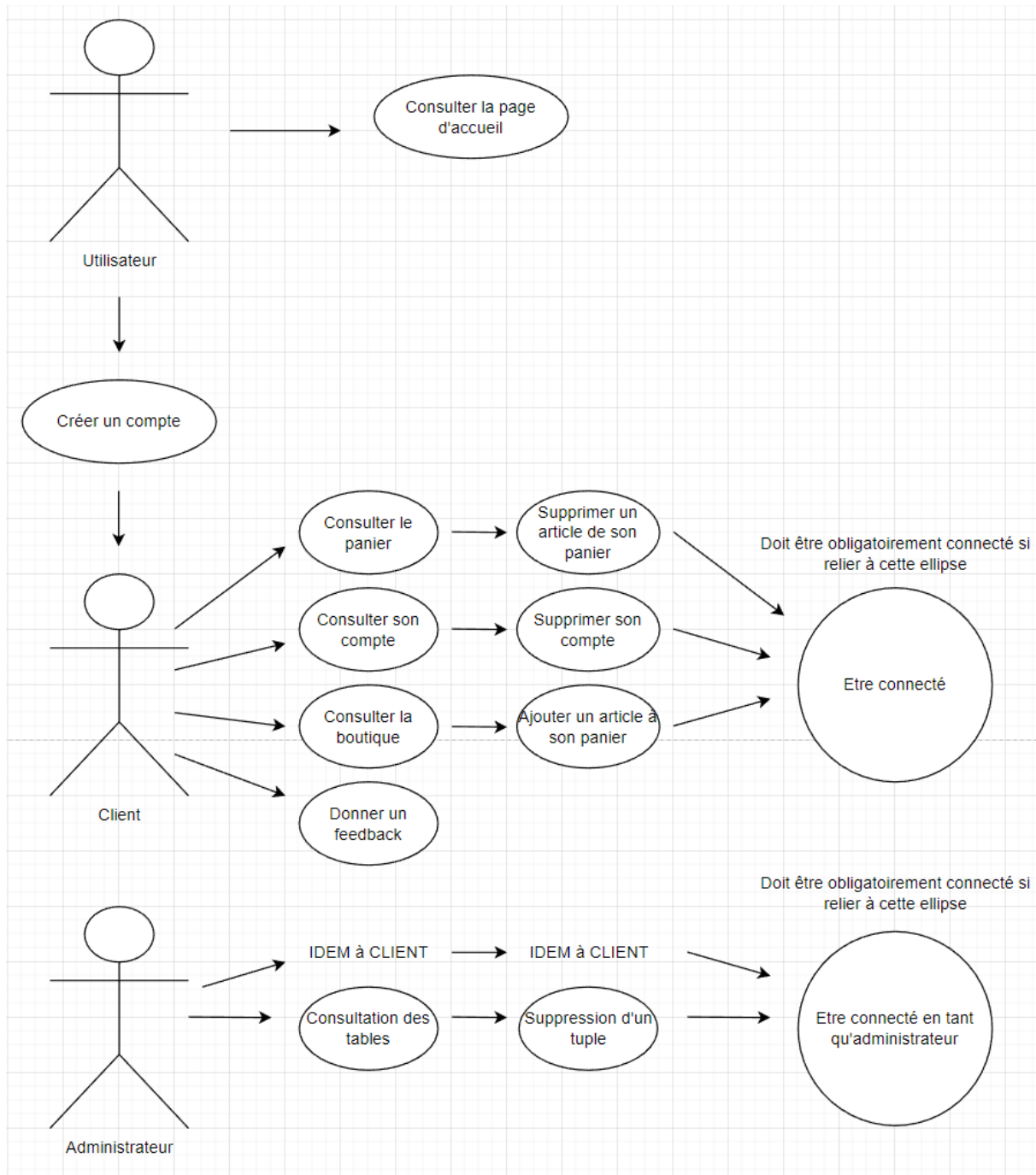
5.1.1 – Spécificités StoreSwap

Voici les spécificités de l'application web StoreSwap :

- Le client peut accéder à la page d'accueil
- Le client peut accéder à la page boutique
- Le client peut accéder à son espace personnel via l'onglet Mon Espace (seulement après s'être connecter)
- Le client peut consulter son panier via l'icône associé
- Le client peut consulter les produits les plus commandés
- Le client peut cliquer sur le bouton Votre Feedback
- Le client peut donner un Feedback en remplissant les champs Prénom, Email, Avis
- L'utilisateur peut créer un compte pour devenir client
- Si l'utilisateur n'a pas de compte, il ne peut pas accéder à la boutique
- Si l'utilisateur n'a pas de compte et qu'il tente de se connecter, il est renvoyé vers le formulaire d'inscription
- Le client peut se connecter via le bouton de connexion
- Si un utilisateur s'inscrit sur le site, il devient client
- Le client peut ajouter un article à son panier
- Le client peut ajouter plusieurs articles à son panier
- Le client peut supprimer un article de son panier
- Le client peut supprimer tout son panier d'un seul coup
- Un administrateur peut effectuer toutes les actions d'un client/utilisateur
- Un administrateur peut cliquer sur l'icône de connexion pour les administrateurs pour s'y connecter
- Un administrateur peut consulter les différentes tables via l'interface d'administration
- Un administrateur peut supprimer un tuple via l'icône corbeille quand il clique sur une table dans l'interface administration
- Un administrateur peut de nouveau afficher toutes les tables quand il a affiché seulement une seule table

5.1.2 – Diagramme de cas UML

Je vais vous présenter ci-dessous le Diagramme de cas UML (Unified Modeling Language). Ce diagramme constitue une représentation visuelle du système, permettant de mieux comprendre son architecture, ses composants et les interactions entre eux. Grâce à ce diagramme, vous pourrez avoir une vision globale de la structure de l'application, des différentes classes qui la composent, ainsi que des relations et des dépendances entre ces classes. En examinant attentivement le Diagramme de cas UML de l'application StoreSwap, vous serez en mesure de saisir plus facilement les fonctionnalités clés de l'application et d'appréhender la manière dont les différents éléments interagissent pour offrir une expérience utilisateur optimale. A savoir que le Diagramme de cas UML reprend les spécifications fonctionnelles énoncées ci-dessus.



5.1.3 – Dictionnaire de données

A présent, il est primordial de présenter le dictionnaire de donnée pour avoir une vue précise sur les différentes données manipulées. Ce précieux outil recense et organise toutes les informations qui sont stockés et manipulées par l'application. Le dictionnaire de données constitue une ressource essentielle pour comprendre la structure, le contenu et les relations entre les différentes entités présentes dans l'application StoreSwap. Il vous permettra d'avoir une vue d'ensemble claire et détaillée des différentes tables, champs, types de données et contraintes utilisé dans la gestion des informations. Grâce à cette présentation, vous pourrez apprécier la rigueur et l'efficacité de la gestion de données de StoreSwap, ainsi que la manière dont elles sont organisées pour assurer un fonctionnement optimal de l'application.

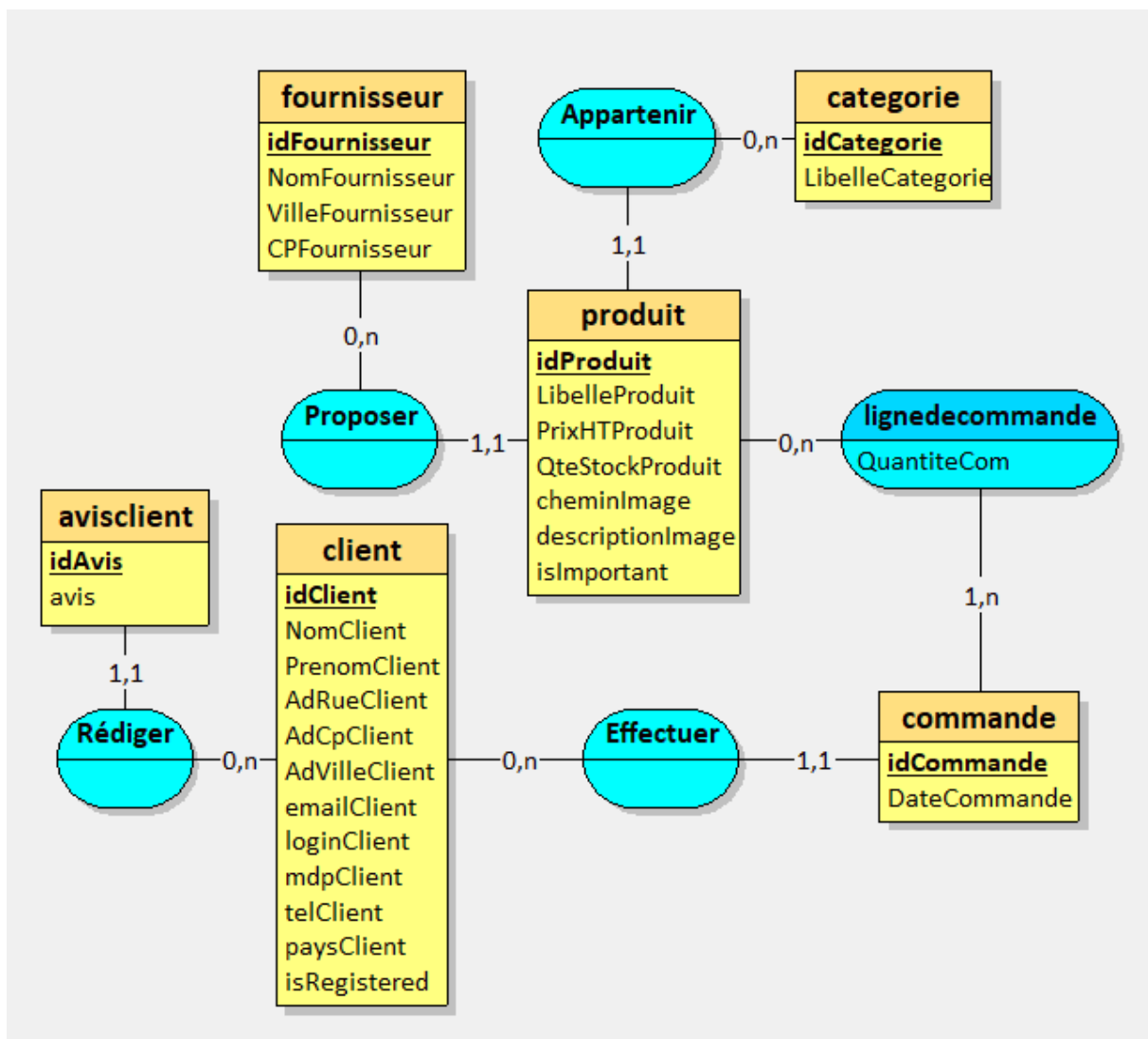
	A	B	C	D	E
1	Donnée	Type	Longueur	Libelle_BDD	Exemple de Donnée
2	Fournisseur Identifiant	INTEGER		idFournisseur	1
3	Fournisseur Nom	VARCHAR	50	NomFournisseur	Abiroh
4	Fournisseur Ville	VARCHAR	50	VilleFournisseur	Montpellier
5	Fournisseur Code Portal	INTEGER		CPFournisseur	34 000
6	Produit Identifiant	INTEGER		idProduit	1
7	Produit Libelle	VARCHAR	100	LibelleProduit	Crocodile Bleu
8	Produit Prix Hors Taxes	INTEGER		PrixHTProduit	1,99
9	Produit Quantité Stock	INTEGER		QteStockProduit	245
10	Produit Chemin Image	VARCHAR	50	cheminImage	croco_bleu.png
11	Produit Description Image	VARCHAR	50	descriptionImage	Excellent crocodile ...
12	Produit Est Important	BOOLEAN		isImportant	0
13	Categorie Identifiant	INTEGER		idCategorie	1
14	Categorie Libelle	VARCHAR	50	LibelleCategorie	Basique
15	Commande Identifiant	INTEGER		idCommande	1
16	Commande Date	DATE		DateCommande	05/05/2020
17	Client Identifiant	INTEGER		idClient	1
18	Client Nom	VARCHAR	150	NomClient	Adrian
19	Client Prenom	VARCHAR	45	PrenomClient	Tyler
20	Client Adresse Rue	VARCHAR	150	AdRueClient	Ap 120 Rue Du Sel
21	Client Adresse Code Portal	INTEGER	11	AdCpClient	34000
22	Client Adresse Ville	VARCHAR	100	AdVilleClient	Montpellier
23	Client Email	VARCHAR	100	emailClient	feugiat.placerat@outlook.com
24	Client Login	VARCHAR	50	loginClient	tempor
25	Client Mot de passe	VARCHAR	50	mdpClient	*****
26	Client Numéro de téléphone	VARCHAR	15	telClient	07 78 32 49 23
27	Client Pays	VARCHAR	15	paysClient	Ukraine
28	Client Est Enregistré	BOOLEAN		isRegistered	1
29	Avis Identifiant	INTEGER		idAvis	1
30	Avis Avis	VARCHAR	500	avis	Très beau site web bla bla bla
31	Ligne De Commande Quantité	INTEGER	11	QuantiteCom	

5.1.4 – Modèle Conceptuel de Données

Maintenant, je vais vous présenter le Modèle Conceptuel de Données. Il a été réalisé sur l'éditeur de MCD Looping MCD, cet éditeur nous permet également de générer le MLD (Modèle Logique de Données), mais aussi le script de base de données pour l'intégrer dans l'interpréteur de script MySQL et donc générer notre base de données.

Ce modèle est une représentation abstraite et visuelle des concepts et des relations qui existent au sein de l'application. Le MCD de StoreSwap met en évidence les entités principales, les attributs et les associations entre ces entités, offrant ainsi une vue claire de la structure et du fonctionnement de la base de données. Les entités représentent les objets distincts, les relations indiquent les liens entre ces objets, et les cardinalités spécifient le nombre d'occurrences d'une entité dans une relation.

En examinant attentivement le MCD, vous pourrez comprendre comment les différentes entités sont liées entre elles. Le MCD constitue donc un outil essentiel pour comprendre la logique sous-jacente de StoreSwap et apprécier la manière dont les données sont modélisées pour répondre aux besoins de l'application.



5.1.5 – Modèle Logique de Données

Avec l'application LoopingMCD, j'ai également généré un Modèle Logique de Données, les clés primaires sont en gras soulignées, les clés étrangères sont précédées d'un # et en bleu, en voici l'image :

```
MLD
client = (idClient INT, NomClient VARCHAR(50), PrenomClient VARCHAR(50), AdRueClient VARCHAR(50), AdCpClient INT, AdVilleClient VARCHAR(50), emailClient VARCHAR(50), loginClient VARCHAR(50), mdpClient VARCHAR(50), telClient VARCHAR(50), paysClient VARCHAR(50), isRegistered INT);
fournisseur = (idFournisseur INT, NomFournisseur VARCHAR(50), VilleFournisseur VARCHAR(50), CPFournisseur INT);
categorie = (idCategorie INT, LibelleCategorie VARCHAR(50));
avisclient = (idAvis INT, avis VARCHAR(50), #idClient);
commande = (idCommande INT, DateCommande DATE, #idClient);
produit = (idProduit INT, LibelleProduit VARCHAR(50), PrixHTProduit INT, QteStockProduit INT, cheminImage VARCHAR(50), descriptionImage VARCHAR(50), isImportant LOGICAL, #idCategorie, #idFournisseur);
lignedecommande = (#idProduit, #idCommande, QuantiteCom INT);
```

5.1.6 – Conventions de Nommages

Voici les conventions de nommages que j'ai utilisé pour la génération de la base de données, ainsi que le développement de l'application StoreSwap.

- Thème globale → Commentaires en anglais si possible, ainsi que le code
- Les classes sont en lowercase → (exemple : produit)
- Les variables sont en lowercase → (exemple : text)
- Les méthodes adoptent le UpperCamelCase et doivent avoir une signification simple et verbeuse → (exemple : DeleteProduit(), GetProduitByCommande())

A savoir que ces conventions son appliquées à la base de données, à l'application web StoreSwap mais aussi à l'application desktop IAPM.

En plus du StoryBoard présenté ci-dessus dans la rubrique Gestion de projet IAPM, j'ai réalisé des maquettes disponibles en annexe.

(Voir Annexes pour IAPM – Annexe IAPM Maquette)

5.2 – Application desktop IAPM

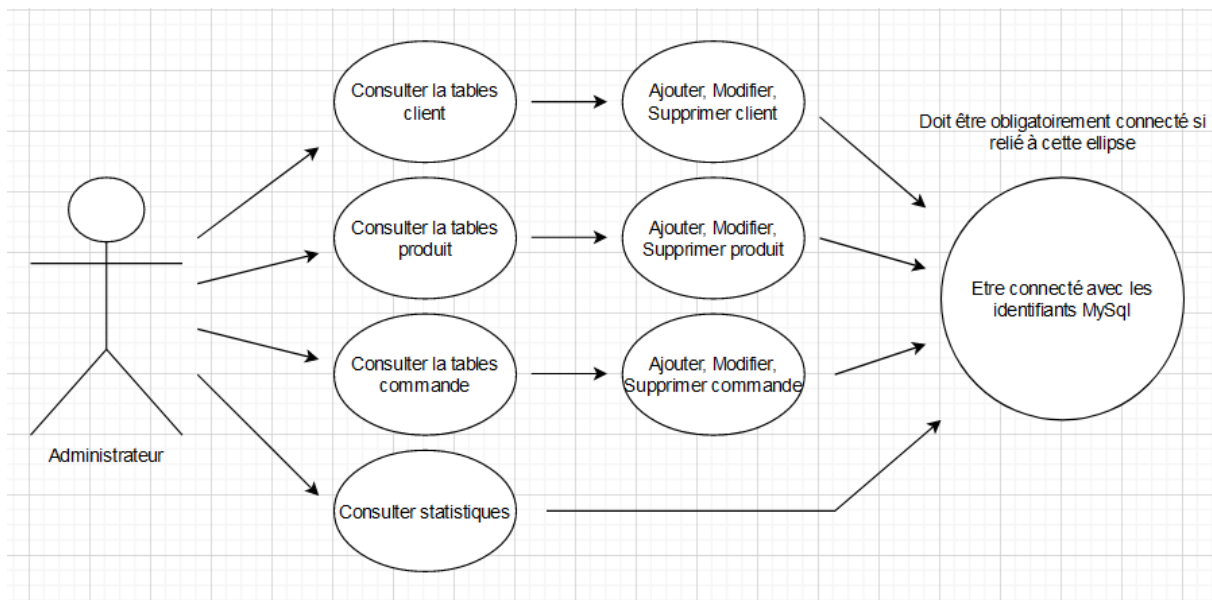
5.2.1 – Spécificités IAPM

Voici les spécificités de l'application desktop IAPM

- Un administrateur peut accéder au menu
- Un administrateur peut connecter le progiciel à une base de données précise via un menu déroulant
- Un administrateur peut ouvrir l'onglet Client via un menu déroulant (Idem pour produits et commandes)
- Un administrateur peut accéder à l'administration des clients via le bouton « Ajout/Modif/Supr » dans le menu déroulant Client (Idem pour produits et commandes)
- Un administrateur peut accéder à la visualisation de la liste des clients via le bouton « Liste des clients » dans le menu déroulant Client (Idem pour produits et commandes)
- Un administrateur peut fermer l'application via le bouton « Fermer le formulaire »
- Un administrateur peut ajouter un client via le bouton « Ajouter »
- Un administrateur peut modifier un client via le bouton « Modifier »
- Un administrateur peut supprimer un client via le bouton « Supprimer »
- Un administrateur peut naviguer entre les différents clients à l'aide des boutons fléchés
- Un administrateur peut rechercher un client via la barre de recherche
- Un administrateur peut ajouter un produit via le bouton « Ajouter »
- Un administrateur peut modifier un produit via le bouton « Modifier »
- Un administrateur peut supprimer un produit via le bouton « Supprimer »
- Un administrateur peut naviguer entre les différents produits à l'aide des boutons fléchés
- Un administrateur peut rechercher un produit via la barre de recherche
- Un administrateur peut ajouter une commande via le bouton « Ajouter »
- Un administrateur peut modifier une commande via le bouton « Modifier »
- Un administrateur peut supprimer une commande via le bouton « Supprimer »
- Un administrateur peut naviguer entre les différentes commandes à l'aide des boutons fléchés

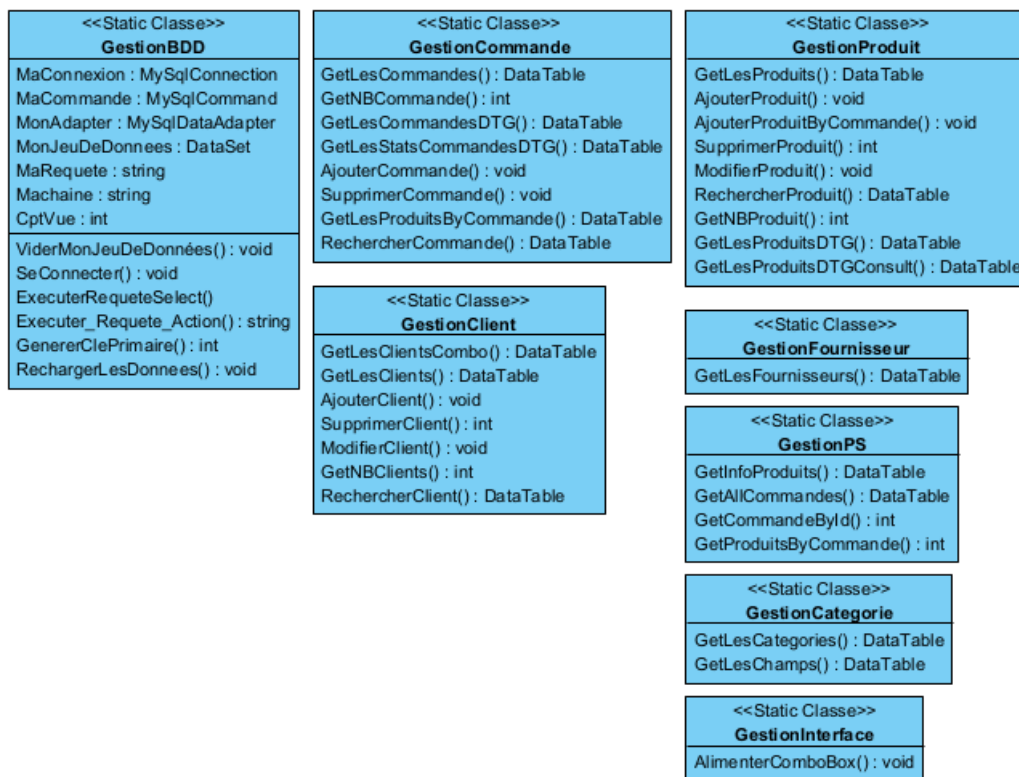
5.2.2 – Diagramme de cas UML

Je vais vous présenter ci-dessous le Diagramme de cas UML (Unified Modeling Language). Ce diagramme constitue une représentation visuelle du système, permettant de mieux comprendre son architecture, ses composants et les interactions entre eux. Grâce à ce diagramme, vous pourrez avoir une vision globale de la structure de l'application, des différentes classes qui la composent, ainsi que des relations et des dépendances entre ces classes. En examinant attentivement le Diagramme de cas UML de l'application IAPM, vous serez en mesure de saisir plus facilement les fonctionnalités clés de l'application et d'appréhender la manière dont les différents éléments interagissent pour offrir une expérience utilisateur optimale. A savoir que le Diagramme de cas UML reprend les spécifications fonctionnelles énoncées ci-dessus.



5.2.3 – Diagramme de Classes

Dans le cadre de la présentation de mon application desktop IAPM, il est essentiel de comprendre la structure et l'organisation du système. Pour cela, je vais vous présenter le Diagramme des classes, un outil visuel qui représente les différentes classes du programme, ainsi que leurs relations et interactions. Ce diagramme sera un guide précieux pour mieux appréhender la conception et le fonctionnement de l'application, en mettant en évidence les entités principales, les attributs et les méthodes de chaque classe. Grâce à cette représentation claire et synthétique, vous pourrez aisément visualiser les différents composants de l'application IAPM et comprendre comment ils interagissent pour offrir des fonctionnalités performantes et intuitives.



5.2.3 – Base de Données

Ci-après, je vais présenter la base de données via des captures d'écrans et en y apportant des explications.

Voici l'arborescence de la base de données, on y retrouve les procédures stockées, les déclencheurs ainsi que les tables.

Ci-dessous, voici la table produit :

Colonnes : Ajouter Supprimer Haut Bas						
#	Nom	Type de données	Taille/Ensemble	Non si...	NULL a...	ZERO...
1	idProduit	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	LibelleProduit	VARCHAR	100	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	PrixHTProduit	FLOAT	6,2	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	CteStockProd...	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
5	idFourm	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	idCat	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
7	cheminImage	CHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	descriptionIm...	VARCHAR	500	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	isImportant	INT	11	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

J'ai également développé des procédures stockées ainsi que des déclencheurs, vous pouvez les retrouver en annexe.

(Voir Annexes pour IAPM – Annexe IAPM BDD)

lilian_ppe_marchand_layrac_adapt		185,2 KiB
!DeleteProduit		
!GetAllCommande		
!GetCAbyProduit(NumProd int)		
!GetCommandeById		
!GetCommandesByClient(NumCli int)		
!GetCommandesByPeriode Entrez jour désiré		
!GetInfoProduit		
!GetProduitsByCommande		
!InsertClient		
!RechercheClient		
avisclient		2,1 KiB
categorie		16,0 KiB
client		16,0 KiB
commande		32,0 KiB
DateComNonValidInsert		
DateComNonValideUpdate		
fournisseur		16,0 KiB
GestionStats		
lignedecommande		48,0 KiB
produit		48,0 KiB
statistiques		4,0 KiB
utilisateur		3,1 KiB
VerifStock		
ViderStats		

Ces procédures stockées et déclencheurs sont appelés dans l'application desktop IAPM pour récupérer les clients, les produits et les commandes.

5.3 – Application Mobile

5.1.1 – Spécificités IAPM Mobile

Voici les spécificités de l'application mobile IAPM Mobile

- l'administrateur peut accéder à la page de connexion
- l'administrateur peut se connecter avec ses identifiants MySQL créer au préalable via PhpMyAdmin

Toutes les informations ci-dessous sont possible seulement si l'administrateur est connecter avec son compte MySQL

- l'administrateur peut accéder à la page d'accueil
- l'administrateur peut accéder à la page Table Client
- l'administrateur peut consulter les clients
- l'administrateur peut créer ou supprimer des clients
- l'administrateur peut effectuer une recherche de clients avec la barre de recherche

- l'administrateur peut consulter la page Table Produit
- l'administrateur peut consulter les produits
- l'administrateur peut créer ou supprimer des produits
- l'administrateur peut consulter la page Table Commande
- l'administrateur peut effectuer une recherche de commandes avec la barre de recherche
- l'administrateur peut consulter la page Statistiques
- l'administrateur peut consulter les statistiques de la boutique (nombre clients, produits et commandes)
- l'administrateur peut se déconnecter et donc retourner à la page de connexion initiale

En résumé, l'application mobile IAPM Mobile présente les spécifications suivantes pour les administrateurs. Ils peuvent se connecter en utilisant leurs identifiants MySQL créés préalablement via PhpMyAdmin. Une fois connectés, les administrateurs ont accès à plusieurs fonctionnalités : la page de connexion, la page d'accueil, la page Table Client pour consulter et gérer les clients, la possibilité de créer ou supprimer des clients, une fonction de recherche de clients, la page Table Produit pour consulter les produits, la possibilité de créer ou supprimer des produits, la page Table Commande pour consulter les commandes, une fonction de recherche de commandes, la page Statistiques pour consulter les données statistiques de la boutique (nombre de clients, produits et commandes), et la possibilité de se déconnecter et revenir à la page de connexion initiale.

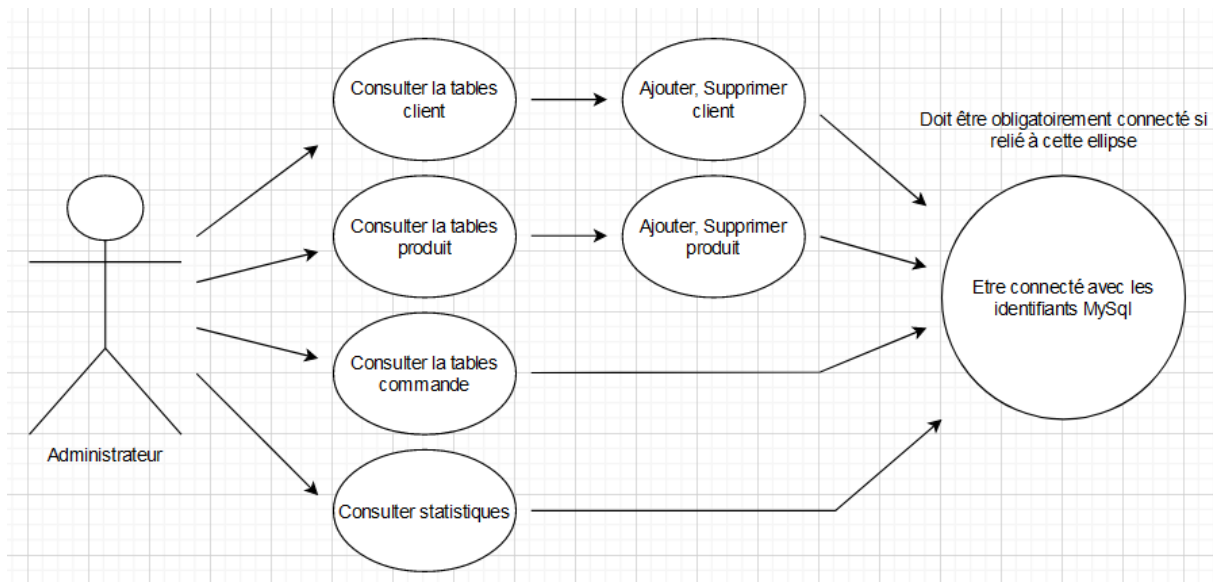
Voici point par point les spécifications fonctionnelles de l'application IAPM Mobile :

1. Connexion et authentification : Les utilisateurs peuvent se connecter à l'application à l'aide de leurs identifiants personnels. L'authentification se fait en utilisant des identifiants MySQL créés au préalable via PhpMyAdmin.
2. Gestion des clients : Les administrateurs ont la possibilité de consulter, créer et supprimer des clients. Une fonctionnalité de recherche permet de retrouver rapidement des clients spécifiques.
3. Gestion des produits : Les administrateurs peuvent accéder à une liste de produits, les consulter, en créer de nouveaux ou supprimer des produits existants.
4. Gestion des commandes : Les administrateurs ont la possibilité de consulter les commandes passées, effectuer des recherches spécifiques, et afficher les détails de chaque commande.
5. Statistiques : Une page dédiée aux statistiques fournit aux administrateurs des informations sur le nombre total de clients, de produits et de commandes enregistrées dans la boutique.
6. Déconnexion : Les administrateurs peuvent se déconnecter de l'application, ce qui les ramènera à la page de connexion initiale.

Ces spécifications fonctionnelles décrivent les principales fonctionnalités de l'application IAPM Mobile et servent de guide pour le développement et la mise en œuvre de l'application.

5.1.2 – Diagramme de cas UML

Je vais vous présenter ci-dessous le Diagramme de cas UML (Unified Modeling Language). Ce diagramme constitue une représentation visuelle du système, permettant de mieux comprendre son architecture, ses composants et les interactions entre eux. Grâce à ce diagramme, vous pourrez avoir une vision globale de la structure de l'application, des différentes classes qui la composent, ainsi que des relations et des dépendances entre ces classes. En examinant attentivement le Diagramme de cas UML de l'application IAPM Mobile, vous serez en mesure de saisir plus facilement les fonctionnalités clés de l'application et d'appréhender la manière dont les différents éléments interagissent pour offrir une expérience utilisateur optimale. A savoir que le Diagramme de cas UML reprend les spécifications fonctionnelles énoncées ci-dessus.



6 – Spécifications techniques

6.1 – Application web StoreSwap

6.1.1 – Elaboration des contraintes techniques

L'éditeur de code utilisé est **Visual Studio Code**, car je le trouve ergonomique et simple d'utilisation. On peut également installer énormément de plugins pour module l'IDE « à sa sauce », et c'est cela qui en fait sa force.

Pour que l'application communique avec la base de données, j'ai utilisé un serveur **WAMP** pour pouvoir travailler en local.

J'ai utilisé la version 7.4.26 de PHP, ainsi que la version 5.7.36 de MySQL.

Pour, la conception des diagrammes UML, j'ai utilisé **diagrams.net**.

Pour la conception du MCD, j'ai utilisé **LoopingMCD**, il m'a également généré le MLD.

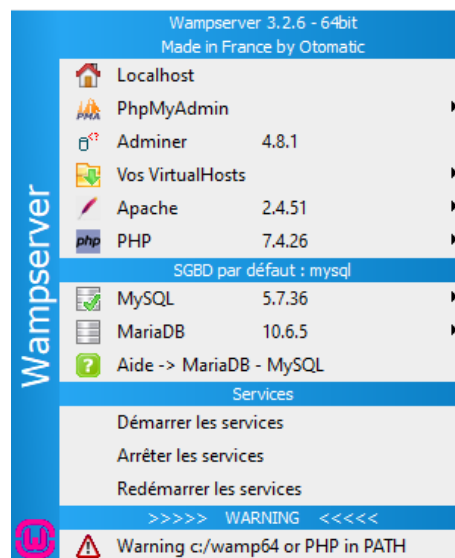
Pour la plateforme, j'ai choisi le langage **PHP** pour le développement ainsi que **HTML 5** et **CSS 3**. Pour le design, j'ai utilisé le **framework CSS Bootstrap 5**. La base de données est en langage **MySQL**.

Pour l'architecture, j'ai utilisé le modèle en **couche MVC**, qui permet d'avoir un contrôle total sur les modèles, les vues, ainsi que les contrôleurs de l'application web. Le serveur tourne sur le protocole de base https, qui permet une certaine sécurité lors de la navigation. Les politiques de gestion de session sont respectées.

Les exigences de **l'interface utilisateur** sont que l'interface soit intuitive et facile de compréhension pour l'utilisateur. Il doit aller d'un point A à un point B simplement et efficacement sans se perdre sur le site, sinon il risque de le quitter et donc arrêter la navigation.

L'aspect sécurité est traité par le biais de hachage du mot de passe lors de l'inscription de l'utilisateur. Un système de cookie de session est également utilisé pour garder la connexion de l'utilisateur, il respecte les spécifications de durée.

Dans le contexte fictif, l'entreprise Layweb assurera la maintenance de l'application si nécessaire et si amélioration est demandé par le client, cependant, elle vous sera facturée.



6.1.2 – Composants d'accès aux données

Pour le projet StoreSwap, pour effectuer la connexion à la base de données, j'ai créé la classe MysqlConfig, qui s'occupe d'affecter les informations de connexion aux variable appelées dans la fonction de connexion seConnecter. Voici le code de la classe MysqlConfig :

configs > mysql_config.class.php > ...

```
1  <?php
2
3  class MysqlConfig {
4      const SERVEUR = 'localhost';
5      const BASE = 'lilian_ppe_marchand_layrac_adapt';
6      const UTILISATEUR = 'root';
7      const MOT_DE_PASSE = '';
8  }
9
10 >
```

La variable **SERVEUR** correspond à l'IP du serveur MySQL cible,

La variable **BASE** correspond au nom de la base de données cible,

La variable **UTILISATEUR** correspond au nom d'utilisateur permettant de se connecter à la base de données,

La variable **MOT_DE_PASSE** correspond au mot de passe permettant de se connecter à la base de données.

Ces variables seront utilisées dans la fonction seConnecter qui se situe dans la classe ModelePDO.

application > modeles > modelePDO.class.php > ModelePDO > seConnecter

```
1  <?php
2
3  class ModelePDO {
4
5      // <editor-fold defaultstate="collapsed" desc="Champs Statiques">
6      //Connexion
7      protected static $serveur = MySqlConfig::SERVEUR;
8      protected static $base = MySqlConfig::BASE;
9      protected static $utilisateur = MySqlConfig::UTILISATEUR;
10     protected static $passe = MySqlConfig::MOT_DE_PASSE;
11     //Manipulate PDO from DB
12     protected static $pdoCnxBase = null;
13     protected static $pdoStResults = null;
14     protected static $requete = "";
15     protected static $resultat = null;
16
17     // </editor-fold>
18     // <editor-fold defaultstate="collapsed" desc="Méthodes Statique">
19
20     public static function seConnecter() {
21         if (!isset(self::$pdoCnxBase)) { //If there is no connexion yet
22             try {
23                 self::$pdoCnxBase = new PDO('mysql:host=' . MySqlConfig::SERVEUR . ';dbname=' .
24                     MySqlConfig::BASE, MySqlConfig::UTILISATEUR, MySqlConfig::MOT_DE_PASSE);
25                 self::$pdoCnxBase->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
26                 self::$pdoCnxBase->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_OBJ);
27                 self::$pdoCnxBase->query("SET CHARACTER SET utf8");
28             } catch (Exception $e) {
29                 //The object pdoCnxBase is generating automatically an object from Exception type
30                 echo 'Erreur : ' . $e->getMessage() . '<br />'; //Method from Exception class
31                 echo 'Code : ' . $e->getCode(); //Method from Exception class
32             }
33         }
34     }
35
36     public static function seDeconnecter() {
37         self::$pdoCnxBase = null;
38     }
39 }
```

Dans cette classe, je redéfinis les variables stockées dans la classe MysqlConfig, ensuite, dans ma fonction seConnecter, je vérifie si la connexion a déjà été établie, si oui, la méthode ne se réexécute pas, cependant, s'il n'y a pas de connexion établie, j'établis ma chaîne de connexion avec les informations stockées dans les variables. La connexion s'effectue à l'aide de l'extension PDO.

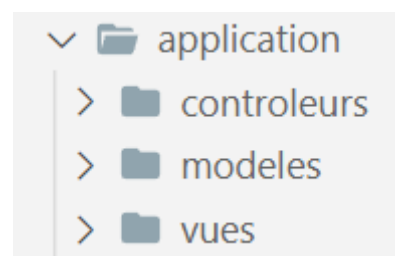
Cette fonction sera appelée quand vues pour avoir constamment accès aux données.

La fonction seDeconnecter met simplement la variable pdoCnxBase à NULL, ce qui annule donc la connexion à la base de données.

L'utilisation de la classe modèle PDO permet de se prémunir d'injections SQL par le biais des requêtes préparées.

6.1.3 – Principe de développement Modèle Vue Contrôleurs

Le modèle vue contrôleur (MVC) est un principe de conception largement utilisé dans le développement d'applications web, y compris dans le cas de StoreSwap. Il divise l'application en trois composants principaux : **le modèle, la vue et le contrôleur**.



Le modèle représente les données et la logique métier de l'application. Il est responsable de la manipulation des données, de leur validation et de leur persistance. Dans le cas de StoreSwap, le modèle peut englober les entités telles que les clients, les produits et les commandes, ainsi que les opérations associées à ces entités.

La vue est responsable de l'affichage des données à l'utilisateur. Elle présente les informations de manière conviviale et attractive, en utilisant des éléments tels que les interfaces graphiques et les templates. Dans StoreSwap, la vue peut représenter les différentes pages et interfaces permettant aux utilisateurs de rechercher, afficher et interagir avec les produits et les fonctionnalités de l'application.

Le contrôleur fait le lien entre le modèle et la vue. Il gère les interactions de l'utilisateur, collecte les données saisies et les transmet au modèle pour traitement. Il récupère également les résultats du modèle et les transmet à la vue pour affichage. Dans le contexte de StoreSwap, le contrôleur peut être responsable de gérer les requêtes de recherche, de valider les informations fournies par les utilisateurs lors de la création de leur compte, et de coordonner les actions entre le modèle et la vue.

En adoptant le modèle vue contrôleur, l'application web StoreSwap peut bénéficier d'une séparation claire des responsabilités, ce qui facilite la maintenance, l'évolutivité et la réutilisabilité du code. Il permet également une meilleure organisation du code, ce qui rend le développement et la collaboration plus efficaces. En comprenant ces principes du modèle vue contrôleur, l'application StoreSwap ne peut qu'être un plaisir à développer.

6.1.4 – Les différentes balises de Bootstrap

Premièrement, qu'est-ce que Bootstrap ?

Bootstrap est un framework de développement front-end largement utilisé pour la création de sites web réactifs et esthétiquement attrayants. Il fournit une collection de styles prédéfinis, de composants et de fonctionnalités JavaScript, ce qui permet aux développeurs de gagner du temps et d'accélérer le processus de conception et de développement. Bootstrap utilise une approche basée sur une grille (grid system) réactive, ce qui signifie que le contenu d'un site est organisé en utilisant une structure en grille flexible qui s'adapte automatiquement à différentes tailles d'écran. Cela garantit que le site est facilement consultable et utilisable sur des appareils de bureau, des tablettes et des smartphones. De plus, Bootstrap offre une variété de composants prêts à l'emploi tels que des menus de navigation, des formulaires, des boutons, des carrousels, etc., facilitant ainsi la création d'une interface utilisateur cohérente et attrayante. Avec sa documentation complète et sa large communauté de développeurs, Bootstrap est un outil puissant pour créer des sites web modernes et réactifs.

Dans le site web StoreSwap, différentes classes du framework Bootstrap sont utilisées, en voici quelques-unes utilisées dans le développement du site :

1. La classe "container" est utilisée pour envelopper le contenu principal et créer une mise en page centrée et réactive.
2. La classe "row" est utilisée pour créer une rangée horizontale qui contient les différentes sections du code.
3. Les classes "col-6" et "col-12-narrower" sont utilisées pour diviser la rangée en colonnes. "col-6" indique que chaque colonne occupe 6 colonnes sur les 12 disponibles, tandis que "col-12-narrower" indique que chaque colonne occupe 12 colonnes sur les 12 disponibles lorsque la vue est étroite.
4. La classe "image-wrapper" est utilisée pour envelopper les images et appliquer des styles spécifiques.
5. La classe "first" est utilisée pour spécifier que l'élément est le premier élément de sa catégorie dans le conteneur parent.
6. La classe "image" est utilisée pour appliquer des styles spécifiques à l'image.
7. La classe "fit" est utilisée pour ajuster l'image de manière à ce qu'elle remplisse complètement son conteneur.
8. La classe "feature" est utilisée pour appliquer des styles spécifiques à chaque section présentant un produit ou une information.
9. La classe "major" est utilisée pour mettre en évidence les en-têtes importants.
10. La classe "actions" est utilisée pour styliser les éléments de liste comme des boutons d'action.
11. La classe "button" est utilisée pour styliser les liens comme des boutons.

6.2 – Application desktop IAPM

6.2.1 – Elaboration des contraintes techniques

L'éditeur de code utilisé est **Microsoft Visual Studio 2022**. Je le trouve ergonomique et facile d'utilisation.

Comme pour l'application StoreSwap, l'application IAPM communique avec une base de données MySQL, hébergée sur un serveur **WAMP** pour le travail en local.

Pour la plateforme, j'ai choisi le langage **Visual Basic** pour le développement. Pour le design, j'ai utilisé **WinForm**, un framework de design proposé dans le package **.NET Core**.

Les exigences de l'**interface administrateur** sont que l'interface soit intuitive et facile de compréhension pour l'administrateur. Il doit aller d'un point A à un point B simplement et efficacement sans se perdre sur le site, sinon il risque de le quitter et donc arrêter la navigation.

Dans le contexte fictif, l'entreprise Layweb assurera la maintenance de l'application si nécessaire et si amélioration est demandé par le client, cependant, elle vous sera facturée.

6.2.2 – Composants d'accès aux données

Pour le projet IAPM, pour effectuer la connexion à la base de données, j'ai créé la classe GestionBDD, qui, comme avec l'application web StoreSwap, s'occupe d'effectuer la connexion à la base de données.

```
1 Imports MySql.Data
2 Imports IAMP_LAYRAC.GestionBDD
3
4 Public Class GestionBDD
5
6     'Filling the variables with the information related to the MySql.data package
7     Public Shared MaConnexion As New MySqlConnection
8     Public Shared MaCommande As New MySqlCommand
9     Public Shared MonDataAdapter As New MySqlDataAdapter
10    Public Shared MonJeuDeDonnees As New DataSet
11    Public Shared MaRequete As String
12    Public Shared Machine As String = "Data Source=localhost;Initial Catalog=lilian_ppe_marchand_layrac_adapt;UserID=root;Password=;Convert Zero Datetime=True"
13    Public Shared CptVue As Integer = 0
14
15    'Dump our dataset
16    Public Shared Sub ViderMonJeuDeDonnees()
17        MonJeuDeDonnees.Clear()
18    End Sub
19
20    'Allows to connect to the database according to the information filled in the variables above
21    Public Shared Sub SeConnecter()
22        MaConnexion.ConnectionString = Machine
23        MaConnexion.Open()
24        MonJeuDeDonnees = New DataSet("IAPM_LAYRAC")
25        MonJeuDeDonnees.Clear()
26        MaCommande.CommandType = CommandType.Text
27        MaCommande.Connection = MaConnexion
28    End Sub
```

Comme pour StoreSwap, je remplis mes variables avec les informations de connexion pour créer une chaîne propre à MySQL. Je mets donc les informations pertinentes.

Ensuite, la fonction SeConnecter me permet d'utiliser ma chaîne de connexion et donc accéder à mes données stockées dans ma base de données.

Bien évidemment, pour effectuer la connexion et remplir les bonnes variables, je dois importer le module **MySql.data**, et utiliser les variables associées comme **MySqlConnection**, qui va s'occuper d'établir la connexion à la base de données.

Cette fonction sera appelée dans chaque formulaires pour continuellement avoir accès aux données.

6.3 – Application Mobile

6.3.1 – Elaboration des contraintes techniques

Pour l'application IAPM Mobile, l'éditeur de code utilisé est **Android Studio**. C'est un éditeur génial pour le développement d'application android, comme son nom l'indique, il est approprié au développement android, car il regroupe tous les outils nécessaires tels que l'éditeur de code, le débogueur, l'émulateur Android et les outils de construction de l'application. Il a une interface conviviale et intuitive, et un support actif et officiel de **Google**.

J'utilise Android Studio sur mon environnement **Linux**, j'utilise donc un serveur **XAMP**, qui regroupe un serveur **MySQL** ainsi qu'un serveur **PhpMyAdmin**, comme WAMP ci-dessus.

J'ai utilisé la version 3.7.1 de **Flutter** qui utilise la version 2.19.1 de **Dart**. J'ai également utilisé la version 5.7.36 de MySQL.

Comme énoncé ci-dessus, j'ai utilisé le langage framework **Flutter** qui utilise **Dart** comme langage. Utiliser Flutter et Dart pour le développement d'applications mobiles présente plusieurs avantages significatifs.

Tout d'abord, Flutter est un framework **open source** développé par Google qui permet de créer des applications mobiles pour **Android et iOS** à partir d'une seule base de code. Cela signifie que les développeurs peuvent **économiser** du temps et des efforts en évitant de créer des applications séparées pour chaque plateforme.

Ensuite, Flutter offre une **expérience utilisateur** fluide et native grâce à son moteur de rendu graphique intégré, qui permet d'obtenir des interfaces réactives et esthétiquement attrayantes. De plus, Dart, le langage de programmation utilisé par Flutter, est facile à apprendre et offre une syntaxe claire et concise, ce qui facilite la création de code propre et maintenable.

Dart possède également des fonctionnalités avancées telles que le ramasse-miettes (**garbage collector**) et le typage statique optionnel, ce qui contribue à améliorer les performances et la fiabilité des applications.

En outre, la vaste bibliothèque de widgets prêts à l'emploi de Flutter permet de créer rapidement des interfaces utilisateur riches et interactives.

Enfin, Flutter bénéficie d'une **communauté active et dynamique**, ce qui signifie qu'il existe de nombreuses ressources, des packages et des exemples de code disponibles pour aider les développeurs à résoudre les problèmes et à accélérer le développement.

En combinant la productivité de Flutter, la simplicité de Dart et la possibilité de créer des applications multiplateformes, il est clair que cette combinaison est un choix solide pour les développeurs d'applications mobiles.

Le modèle utilisé par Flutter est le modèle de **Widget**. C'est un modèle de programmation déclaratif où tout est considéré comme un widget.

Côté utilisateur, Flutter offre une **expérience utilisateur de haute qualité** grâce à sa performance native. Les applications Flutter sont connues pour leur **fluidité** et leur **réactivité**, car elles utilisent un moteur de rendu graphique intégré appelé Skia, qui permet d'obtenir des interfaces utilisateur attrayantes et réactives.

Dans le contexte fictif, l'entreprise Layweb assurera la maintenance de l'application si nécessaire et si amélioration est demandé par le client, cependant, elle vous sera facturée.

6.3.2 – Composants d'accès aux données

Pour le projet IAPM Mobile, pour accéder à la base de données, j'ai utilisé le package `mysql1`. Il suffit d'instancier des variables telles que `host`, `port`, `user`, `password` ainsi que `db`.

Les informations relatives à la connexion sont disponible à la création de votre base de données. Si vous êtes en local, l'ip est 10.0.2.2, et le port MySQL est 3306. Le `username` et le `password` dépendent de l'utilisateur que vous avez créé sur votre interface PhpMyAdmin pour votre serveur MySQL.

Quant à la variable `db`, c'est tout simplement le nom de votre base de donnée.

Voici comment s'effectue la connexion dans mon fichier `connexion_screen.dart`. A savoir que je passe par ce fichier car c'est par celui-ci que je me connecte à mon application et donc à ma base de données.

```
Future<void> _connectToDatabase() async {  
  final host = _hostController.text.trim();  
  final port = int.tryParse(_portController.text.trim());  
  final username = _usernameController.text.trim();  
  final password = _passwordController.text.trim();  
  final dbName = _databaseController.text.trim();  
  
  final settings = ConnectionSettings(  
    host: '10.0.2.2',  
    port: 3306,  
    user: username,  
    password: password,  
    db: 'lilian_ppe_marchand_layrac_adapt',  
  );  
  
  try {  
    await DatabaseConnection.connect(settings);  
  
    // Notify when connexion successful  
    _showSuccessNotification('Connexion réussie');  
  
    // Redirect to home_screen.dart  
    Navigator.pushReplacement(  

```

Une fois que ces variables sont remplies, elles sont utilisées dans mon fichier `database.dart`, qui s'occupe d'effectuer la connexion avec ma base de donnée et mon serveur MySQL.

```
import 'package:mysql1/mysql1.dart';  
  
class DatabaseConnection {  
  static late MySqlConnection _connection;  
  
  static MySqlConnection get connection => _connection;  
  
  static Future<void> connect(ConnectionSettings settings) async {  
    _connection = await MySqlConnection.connect(settings);  
  }  
  
  static Future<void> disconnect() async {  
    await _connection.close();  
  }  
}
```

Voici comment fonctionne le package `mysql1` et de quoi il est composé :

Le package `mysql1` est une bibliothèque de gestion de base de données MySQL pour Flutter. Il facilite la connexion à une base de données MySQL et l'exécution de requêtes SQL.

Voici un aperçu du fonctionnement du package **mysql1** :

1. Établissement de la connexion : Vous devez fournir les informations de connexion telles que l'hôte, le port, le nom d'utilisateur, le mot de passe et le nom de la base de données. Vous utilisez l'objet **ConnectionSettings** pour spécifier ces informations et vous utilisez **MySqlConnection.connect()** pour établir une connexion à la base de données.

2. Exécution des requêtes : Une fois la connexion établie, vous pouvez exécuter des requêtes SQL en utilisant la méthode **query()** de l'objet de connexion. Vous spécifiez la requête SQL en tant que chaîne et les résultats sont renvoyés sous la forme d'un objet **Results** contenant les données retournées par la requête.

3. Traitement des résultats : Vous pouvez parcourir les résultats renvoyés par la requête en utilisant une boucle **for-in** ou d'autres méthodes de parcours. Chaque ligne du résultat est représentée par un objet **Row**, qui vous permet d'accéder aux valeurs individuelles des colonnes.

4. Gestion des erreurs : Il est important de gérer les erreurs qui peuvent survenir lors de la connexion à la base de données ou lors de l'exécution des requêtes. Vous pouvez utiliser des blocs **try-catch** pour capturer les exceptions et prendre les mesures appropriées en cas d'erreur.

5. Fermeture de la connexion : Après avoir terminé les opérations de base de données, il est recommandé de fermer la connexion à l'aide de la méthode **close()** de l'objet de connexion.

Le package **mysql1** fournit également des fonctionnalités avancées telles que la préparation de requêtes paramétrées, la gestion des transactions et le streaming de résultats pour les requêtes volumineuses.

Pour importer le package, il suffit de le mettre dans le fichier **pubspec.yml** :

```
# versions available, run `flutter pub outdated`.
dependencies:
  another_flushbar: ^1.10.28
  flutter:
    sdk: flutter

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  mysql1: ^0.20.0
```

Ensuite, il vous faut réaliser la commande **flutter pub get** pour récupérer les dépendances manquantes, puis il faut lancer la commande **flutter pub upgrade**, pour upgrade les dépendances et avoir les nouvelles versions.

7 – Réalisations du candidat

7.1 – Application StoreSwap | IAPM

7.1.1 – Choix de style pour StoreSwap

Le choix du style pour cette application web a été très simple et efficace. Partons d'une thématique avec un client, s'il veut une boutique de bonbon, il faut plutôt des tons claires, c'est pour cela que j'ai choisis la prédominance de la couleur saumon. Pour ce qui est du design de l'interface, j'ai essayé de la faire la plus intuitive possible avec le moins de clique à faire pour accéder à l'information attendu par l'utilisateur. L'accueil est simple, épuré, regroupe les meilleures ventes, les pages de connexion et d'inscription sont basiques. La page boutique est simple et compréhensible, nous avons le prix du produit, sa quantité, son image et un bouton pour l'ajouter au panier.

7.1.2 – Choix de style pour IAPM

Le choix du style pour l'application desktop IAPM est totalement contraire au site web, de ce fait, j'ai choisis certes, une palette de couleur pâle, mais plus stylisée pour des applications bureau. J'ai donc utilisé du bleu pour les boutons, et des tons gris clair pour les fonds. Partie graphique, il fallait que toutes les informations soient affichées, par exemple, si l'on veut un client, en tant qu'utilisateur, je veux l'information la plus rapidement possible pour travailler efficacement, c'est pour cela que j'ai rendu disponible toutes les informations des clients sur une seule page.

Ici, voici l'interface de gestion des clients :

Formulaire de gestion des clients

Rechercher un client dans la base de donnée

Formulaire de saisie

Code Client : 1

Nom : Adrian

Prenom : Tyler

Rue : Ap #257-4830 Accumsan St.

Code Postal : 55137

Ville : Poitiers

idClient	NomClient	PrenomClient	AdRueClient	AdCpClient	AdVilleClient
1	Adrian	Tyler	Ap #257-48...	55137	Poitiers
2	Reese	Kirby	P.O. Box 13...	10357	Tournefeuille
3	Clark	Ainsley	959-5721 A...	68833	La Roche-s...
4	Samantha	Alana	Ap #113-71...	79986	Douai
5	Evan	Zachery	Ap #633-39...	68378	Angoulême
6	Nathaniel	Channing	Ap #437-53...	89910	Beauvais
7	Risa	Lars	791-277 Se...	83333	Saint-Dizier
8	Cole	Rudyard	345-3960 S...	77815	Saint-Dizier
9	Hanna	Macauley	7597 Lobort...	85829	Alès
10	Cyrus	Reed	Ap #294-32...	1601	Colomiers
11	Willa	Donna	Ap #823-21...	79321	Nîmes
12	Wallace	Tucker	892-2263 Et...	64979	Dijon

Supprime Modifier Ajouter

← → ↻ ↺

Fermer le formulaire

Pour plus de screenshot de l'application (Voir Annexes pour IAPM – Annexe IAPM App « Page »)

7.1.3 – Coté base de données

Maintenant, nous allons évoquer le sujet de la base de donnée, j'ai réalisé la base de donnée en commençant par le MCD sur le logiciel LoopingMCD, j'ai ensuite exporter le script et l'ai introduit pour créer ma base de données dans le logiciel d'édition de base de données HeidiSQL (C'est un logiciel gratuit). Cette base de donnée à été adaptée tout au long du projet, pour y rajouter des procédures stockées et des déclencheurs. (Toujours dans cet optique d'optimisation du code et effectuer les requêtes coté serveur).

J'évoquerais plus bas le cas d'utilisation d'une procédure stockée dans l'application desktop.

7.1.4 – Comment est construite l'application StoreSwap

L'application web StoreSwap adopte une architecture MVC, premièrement, nous avons la couche model, c'est la couche qui regroupe les principales classes de notre application. (En l'occurrence, nos classes munis de nos méthodes pour accéder aux données).



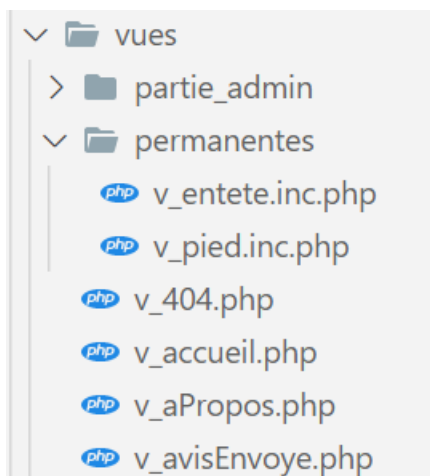
Voici une méthode permettant la récupération des catégories stockées en base de données.

```
public static function getLesCategorie() {
    self::seConnecter();

    self::$requete = "SELECT * FROM categorie";
    self::$pdoStResults = self::$pdoCnxBase->prepare(self::$requete);
    self::$pdoStResults->execute();
    self::$resultat = self::$pdoStResults->fetchAll();

    self::$pdoStResults->closeCursor();

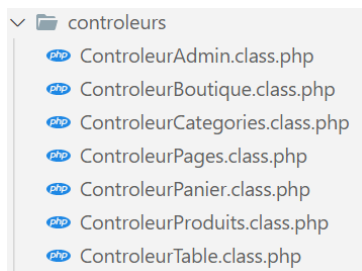
    return self::$resultat;
}
```



Ensuite, nous avons la couche vue, qui regroupe toutes les vues (templates) de notre application, c'est ce qui gère notre affichage.

Ci-dessous, voici un bout de code de la vue permettant l'affichage de la boutique.

```
1 <div class="wrapper">
2     <div class="container" id="main">
3         <div class="row gtr-150">
4             <div class="row features">
5
6                 <nav>
7
8                 </nav>
9
10                <?php
11                if (!isset($_SESSION['login_client'])) {
12                    require_once 'v_seConnecter.php';
13                }
14                else
15                {
16                    foreach (VariablesGlobales::$lesProduits as $unProduit) {
17
18                }
19                <section class="col-4 col-12-narrower feature">
```



Enfin, nous avons la couche contrôleur, il nous permet de gérer les interactions entre le modèle et la vue.

Voici une fonction dans un contrôleur permettant la gestion des erreur (page 404)

```
public function afficherBoutique() {

    VariablesGlobales::$lesProduits = GestionBoutiquePPE::getLesProduits();
    if(VariablesGlobales::$lesProduits == NULL) {
        require Chemins::VUES . 'v_404.php';
    }
    else {
        require Chemins::VUES . 'v_boutique.php';
    }
}
```

A noter que les dossiers Modèle, Vue et Contrôleur sont tous trois stockés dans le dossier Application.

7.1.5 – Sécurité de l'application StoreSwap

La classe modèle PDO (PHP Data Objects) est utilisée pour sécuriser les requêtes dans les sites web PHP. PDO fournit une interface uniforme pour accéder à différentes bases de données, ce qui permet de réduire les vulnérabilités potentielles. En utilisant PDO, vous pouvez utiliser des requêtes préparées, qui sont des requêtes SQL avec des paramètres de substitution. Cela empêche les attaques par injection SQL, car les paramètres sont automatiquement échappés et évite les problèmes de concaténation de chaînes. De plus, PDO prend en charge les transactions, ce qui vous permet d'exécuter des opérations de base de données de manière atomique, garantissant ainsi l'intégrité des données. En utilisant PDO, vous pouvez améliorer la sécurité de votre site web PHP en évitant les erreurs courantes liées à la manipulation directe de requêtes SQL.

```
protected static function getLesTuples($table) {
    self::$seConnecter();
    self::$requete = "SELECT * FROM " . $table;
    self::$pdoStResults = self::$pdoCnxBase->prepare(self::$requete);
    self::$pdoStResults->execute();
    self::$resultat = self::$pdoStResults->fetchAll(PDO::FETCH_OBJ);
    self::$pdoStResults->closeCursor();
    return self::$resultat;
}
```

Nom de la fonction et son paramètre associé (Table).

Affichage du résultat.

Récupération des objets avec l'attribut fetchAll.

Exécution de la requête.

Méthode de connexion à la base de données.

Requête SQL avec la variable \$table passée en paramètre.

Libération des ressources en indiquant à PDO que le travail est finit.

Préparation de la requête avec la fonction « prepare ».

7.1.6 – Tests Unitaires

Les **tests unitaires** sont une pratique essentielle dans le développement logiciel pour garantir la qualité et la fiabilité des applications. Ils permettent de vérifier que chaque composant du code fonctionne correctement de manière isolée, en testant individuellement ses fonctionnalités.

Dans notre projet, nous avons utilisé un framework de tests appelé **PHPUnit**, qui est spécialement conçu pour PHP. PHPUnit fournit des fonctionnalités et des outils pour écrire et exécuter facilement des tests unitaires.

Nous avons créé une classe de test appelée `GestionBoutiquePPETest`, qui contient des méthodes de test pour vérifier le bon fonctionnement de notre application. Chaque méthode de test correspond à une fonctionnalité spécifique que nous souhaitons tester.

```
1  <?php
2  use PHPUnit\Framework\TestCase;
3
4  require_once '/wamp64/www/Storeswap/application/modeles/modelePDO.class.php';
5  require_once '/wamp64/www/Storeswap/configs/mysql_config.class.php';
6  require_once '/wamp64/www/Storeswap/application/modeles/gestion_boutique.class.php';
7
8  class GestionBoutiquePPETest extends TestCase
9  {
```

Dans notre classe, nous avons écrit une méthode de test nommée `testGetLesClients` pour vérifier la fonction `getLesClients` de notre application. Cette fonction est responsable de la récupération des clients.

```
class GestionBoutiquePPETest extends TestCase
{
    public function testGetLesClients()
    {
        ModelePDO::seConnecter(); 1
        // Appel de la méthode à tester 2
        $result = GestionBoutiquePPE::getLesClients();
        // Assertions 3
        $this->assertIsArray($result);
    }
}
```

Je vais vous expliquer chaque composants de la méthode en se référant à l'image ci-dessus.

Pour ce test, nous avons mis en place les préparatifs nécessaires, tels que la connexion à la base de données, afin de s'assurer que le test est exécuté dans un environnement approprié. (1)

Ensuite, nous avons appelé la fonction `getLesClients` (2)` et vérifié que le résultat retourné est un tableau à l'aide de l'assertion `$this->assertIsArray($result)` (3)`. Cela nous permet de nous assurer que la fonction renvoie bien les clients sous forme de tableau, comme attendu.

L'exécution des tests unitaires se fait à l'aide d'une commande spéciale, `vendor/bin/phpunit`, qui lance tous les tests définis dans notre application. Nous obtenons un rapport de test détaillé indiquant le nombre de tests exécutés, le nombre d'assertions réalisées et les éventuelles erreurs rencontrées.

```
lilia@MSI MINGW64 /c/wamp64/www/Storeswap
$ vendor/bin/phpunit application/tests/GestionBoutiquePPETest.php

PHPUnit 10.1.3 by Sebastian Bergmann and contributors.

Runtime:       PHP 8.1.6

.                                                     1 / 1 (100%)

Time: 00:00.020, Memory: 6.00 MB

OK (1 test, 1 assertion)
```

Après avoir exécuté les tests, j'ai obtenu un rapport détaillé de PHPUnit. Il m'a indiqué que tous les tests ont été exécutés avec succès, ce qui signifie que mon application a passé avec succès toutes les vérifications que j'ai définies.

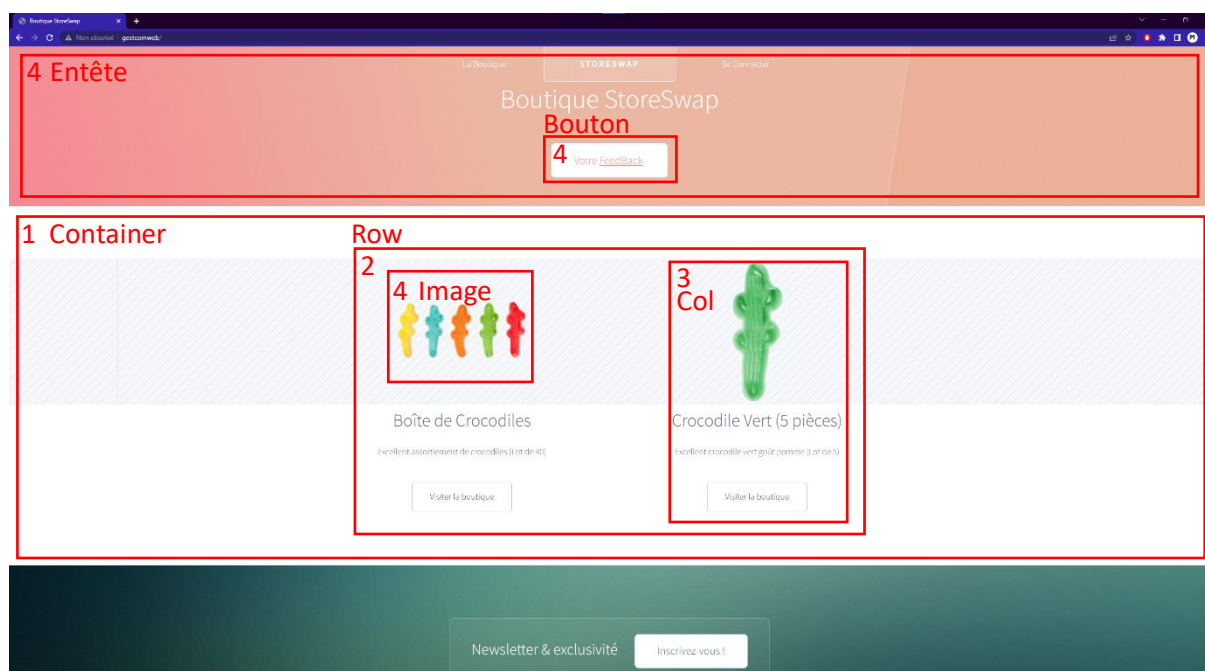
Le rapport affiche également des informations utiles, telles que la version de PHPUnit utilisée (ici la version 10.1.3), la version de PHP (8.1.6), le temps d'exécution total des tests (0,019 secondes) et la consommation mémoire (6,00 MB).

En utilisant cette approche de tests unitaires, j'ai pu identifier rapidement d'éventuels problèmes dans mon application et les corriger avant qu'ils ne se propagent à d'autres parties du code. Cela m'a permis de garantir la qualité et la fiabilité de mon application tout au long du processus de développement.

7.1.7 – Intégration et création du CSS avec le framework Bootstrap

Pour mon site web StoreSwap, j'ai utilisé Bootstrap pour faciliter la découpe de la page en utilisant une grille réactive. Voici comment j'ai découpé les différentes parties de mon site en utilisant Bootstrap :

Voici un schéma explicatif permettant de comprendre quelle balise fait quoi et surtout on est localisé sur la page d'accueil de mon application web StoreSwap :



1. J'ai enveloppé le contenu principal de la page avec la classe "container". Cela m'a permis de créer une zone centrée avec une marge fixe de chaque côté, assurant ainsi un alignement et un centrage horizontal du contenu sur la page.

2. J'ai utilisé la classe "row" pour créer des rangées horizontales à l'intérieur du conteneur. Ces rangées m'ont permis de diviser l'espace horizontal en plusieurs colonnes.

3. J'ai utilisé les classes "col-*" pour définir la largeur des colonnes à l'intérieur d'une rangée. J'ai remplacé l'étoile (*) par un nombre représentant le nombre de colonnes sur lesquelles je voulais étendre ma colonne.

Par exemple, dans le code que j'ai utilisé, j'ai utilisé "col-6" pour diviser une rangée en deux colonnes égales, et "col-4" pour diviser une rangée en trois colonnes égales. Cela signifie que chaque colonne occupe 6 colonnes sur les 12 disponibles dans le premier cas, et 4 colonnes sur les 12 disponibles dans le deuxième cas.



J'ai également combiné les classes "col-*" pour créer des mises en page plus complexes et réactives. Par exemple, j'ai utilisé "col-6 col-12-narrower" pour indiquer qu'une colonne occupe 6 colonnes sur 12, mais passe à 12 colonnes sur 12 lorsque la vue est étroite, comme sur un écran mobile.

4. J'ai placé les différentes parties de mon site, comme les sections, les en-têtes, les images et les boutons, à l'intérieur des colonnes pour créer la mise en page souhaitée. J'ai pu appliquer des classes spécifiques de Bootstrap à ces éléments pour styliser et aligner le contenu.

En utilisant la grille Bootstrap, j'ai pu découper et organiser facilement les différentes parties de mon site de manière réactive. Ainsi, la disposition s'adapte automatiquement en fonction de la taille de l'écran, assurant une expérience utilisateur cohérente sur différents appareils.

Voici le code associé au screenshot de l'accueil du site web StoreSwap :

Encadré en rouge, vous retrouverez les différentes classes Bootstrap pour le CSS.

application > vues >  v_accueil.php >  div.wrapper

```
1 <div class="wrapper">
2   <div class="container">
3     <div class="row">
4       <?php
5         //var_dump(VariablesGlobales::$lesProduits);
6         foreach (VariablesGlobales::$lesProduits as $unProduit) {
7           ?>
8           <section class="col-6 col-12-narrower feature">
9             <div class="image-wrapper first image">
10               <a href="index.php?controlleur=Boutique&action=afficherBoutique">
19               <li><a href="index.php?controlleur=Boutique&action=afficherBoutique" class="button">Visiter la boutique</a></li>
20             </ul>
21           </section>
22         <?php
23       }
24     ?>
25   </div>
26 </div>
27
28 <div id="promo-wrapper">
29   <section id="promo">
30     <h2>Newsletter & exclusivité</h2>
31     <a href="index.php?controlleur=Admin&action=afficherInscription" class="button">Inscrivez-vous !</a>
32   </section>
33 </div>
34
35 <a id="apropos"></a>
36 <div class="wrapper">
37   <section class="container">
38     <header class="major">
39       <h2>Voici quelques informations sur la boutique StoreSwap</h2>
40       <p><i>Les produits présentés sur ces produits peuvent être momentanément indisponibles à la vente,
41       veuillez nous excuser...</i></p>
42     </header>
43     <div class="row features">
44       <section class="col-4 col-12-narrower feature">
45         <div class="image-wrapper first">
46           <a href="#" class="image featured">" alt="" /></a>
47         </div>
48         <p>Nos convictions. Une équipe à l'écoute de ses clients.</p>
49       </section>
50       <section class="col-4 col-12-narrower feature">
51         <div class="image-wrapper">
52           <a href="#" class="image featured">" alt="" /></a>
53         </div>
54         <p>Quelques avis clients, pour forger votre opinion !</p>
55       </section>
56       <section class="col-4 col-12-narrower feature">
57         <div class="image-wrapper">
58           <a href="#" class="image featured">" alt="" /></a>
59         </div>
60         <p>Pourquoi StoreSwap ? Tout savoir sur notre team.</p>
61       </section>
62     </div>
63   </section>
64   <ul class="actions major">
65     <li><a href="#feedback" class="button">Votre Feedback</a></li>
66   </ul>
67 </div>
68
```


7.1.8 – Tests de performances de l’application StoreSwap

7.1.8.1 – Tests de performances avec Phpstan

Pour vérifier les performances de l’application StoreSwap, j’ai utilisé PhpStan. Phpstan est un outil de test de performances largement utilisé dans le développement d'applications web en PHP. Il fonctionne en analysant statiquement le code source de l'application pour détecter les erreurs potentielles, les incohérences et les problèmes de performance. En se basant sur l'inférence de types, Phpstan est capable d'identifier les variables non définies, les appels de méthodes inexistantes et les erreurs de type. Il fournit également des informations détaillées sur les performances du code, en identifiant les parties du programme qui pourraient être optimisées pour améliorer l'efficacité et la vitesse d'exécution. En résumé, phpstan est un outil essentiel pour garantir la qualité et les performances de votre application web PHP.

Voici comment j’ai utilisé Phpstan pour mon projet StoreSwap, j’ai analysé ma classe GestionAdmin, en spécifiant la configuration spécifique pour rediriger l’analyse vers le fichier voulu :

```
phpstan.neon
1 parameters:|
2     level: 7
3     paths:
4         - /application/controleurs/ControleurAdmin.class.php
```

Voici le résultat de l’analyse par phpstan en exécutant la commande `vendor/bin/phpstan analyse` :

```
lilia@MSI MINGW64 /c/wamp64/www/Storeswap
$ vendor/bin/phpstan analyse
Note: Using configuration file C:\wamp64\www\Storeswap\phpstan.neon.
1/1 [=====] 100%
```

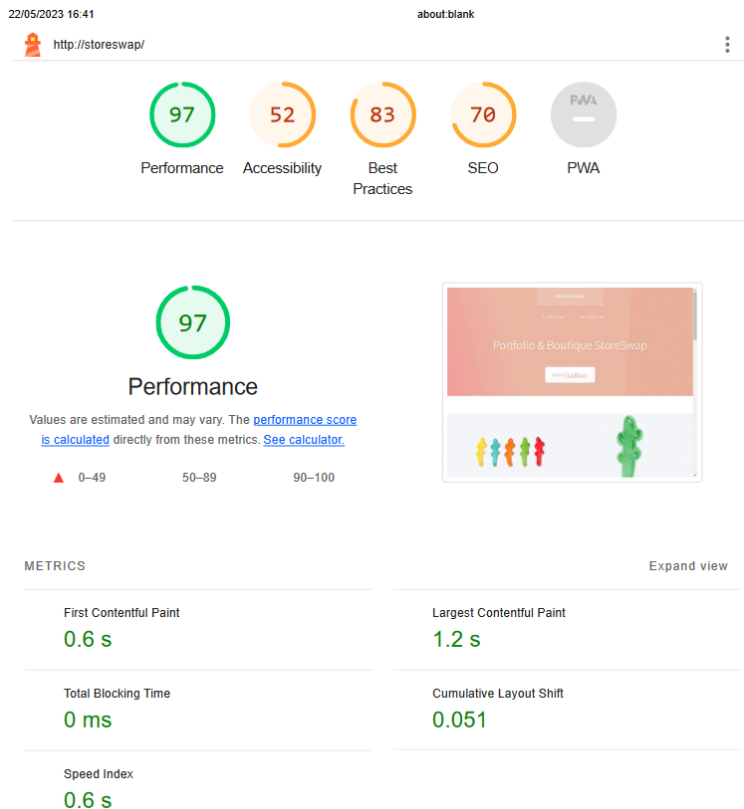
[OK] No errors

7.1.8.2 – Tests de performances avec LighthouseTab

J’ai effectué des tests de performances de mon site en local avec l’outil LighthouseTab.

LighthouseTab est une extension de navigateur Chrome qui permet d'exécuter des audits de performance, d'accessibilité, de bonnes pratiques et de référencement pour des sites web. Il s'appuie sur Lighthouse, une puissante bibliothèque d'audit de performance développée par Google. LighthouseTab facilite l'utilisation de Lighthouse en intégrant ses fonctionnalités directement dans l'interface du navigateur Chrome.

Les résultats de l'audit sont présentés de manière claire et détaillée, avec des scores et des recommandations pour chaque catégorie évaluée. Vous pouvez consulter des informations précises sur les aspects à améliorer, y compris des suggestions de correction et des liens vers des ressources supplémentaires pour approfondir chaque point.



Sur le screenshot à gauche, vous pouvez apercevoir une analyse de performance de la page d'accueil de mon site web StoreSwap.

7.1.8.3 – Respect des normes W3C et WCAG

Le site web StoreSwap respecte les normes W3C, car l'outil de validation LighthouseTab me permet de vérifier cela. Le site a été évalué selon les critères rigoureux du W3C et a réussi à surpasser les attentes. Sa conformité aux normes garantit une accessibilité accrue pour tous les utilisateurs, y compris ceux ayant des besoins spéciaux ou utilisant des technologies d'assistance. De plus, StoreSwap offre une performance exceptionnelle, se charge rapidement et suit les meilleures pratiques de développement web, ce qui permet aux utilisateurs de naviguer en toute fluidité sur le site.

1. Le code HTML ainsi que CSS est conforme aux normes W3C.
2. Le site utilise des couleurs attractives ainsi que du texte et des images lisibles et dissociables des autres éléments du site web.
3. Les balises appropriées sont utilisées, comme les balises sémantiques :
 - a. <header>
 - b. <nav>
 - c. <main>
 - d. <section>
 - e. <article>
 - f. <footer>

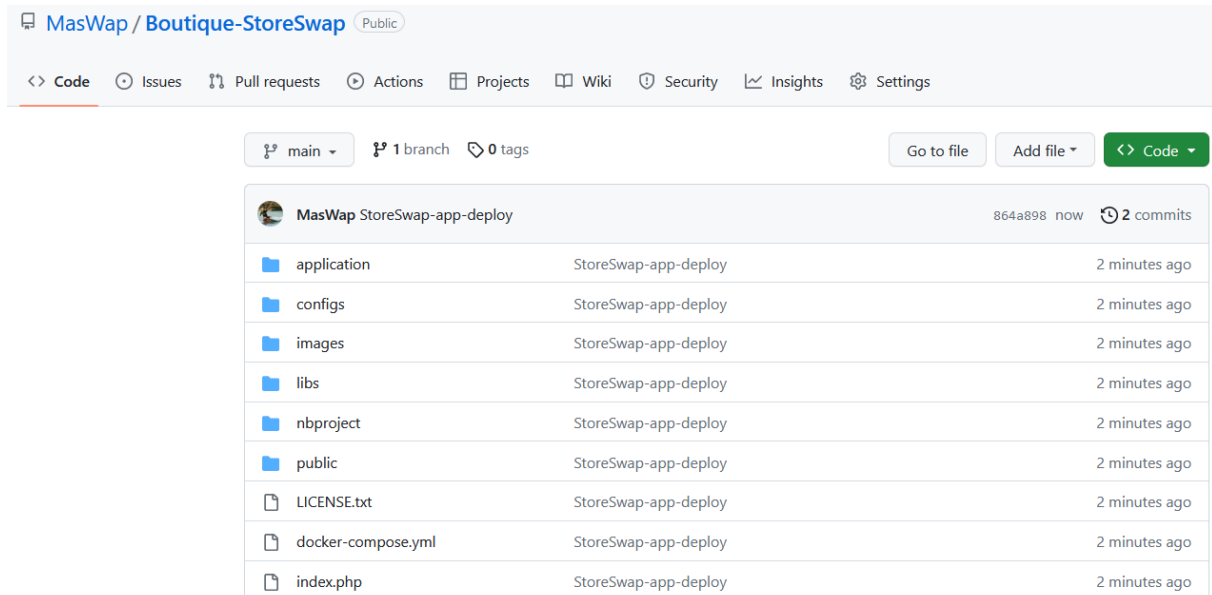
Elles permettent de structurer le code et ainsi une meilleure compréhension du moteur de recherche.

4. Le site web a été testé sur différents navigateurs tel que Chrome, Firefox, Safari et Microsoft Edge. Le responsive est également respecté pour offrir une accessibilité sur n'importe quel appareil.

7.1.9 – Déploiement de l'application StoreSwap

J'ai également réalisé le déploiement de l'application en proposant un dépôt github pour télécharger le site web, ainsi qu'un tutoriel détailler permettant le déploiement de l'application web StoreSwap ainsi que sa base de donnée et son serveur de base de donnée local.

Voici un screenshot du repository github. Il vous suffit simplement de cloner le dépôt sur votre répertoire local :



Voici la commande pour cloner le dépôt github :

```
[18:10:05] lilian@lilian-Modern-15-A11MU  
~ > git clone git@github.com:MasWap/Boutique-StoreSwap.git
```

Voici la commande pour accéder au bon dossier, puis ensuite exécuter le fichier docker-composer.yml pour instancier les conteneurs :

```
[18:11:12] lilian@lilian-Modern-15-A11MU  
~ > cd Boutique-StoreSwap/
```

```
[18:12:22] lilian@lilian-Modern-15-A11MU  
~/Boutique-StoreSwap > docker-compose up -d  
Creating network "boutique-storeswap_default" with the default driver  
Creating mysql_container ... done  
Creating phpmyadmin_container ... done  
[18:12:30] lilian@lilian-Modern-15-A11MU  
~/Boutique-StoreSwap >
```

Une fois les conteneurs instanciés, il faut vérifier qu'ils sont bien lancés, pour cela, il faut donc exécuter la commande suivante :

```
18:13:05 lilian@lilian-Modern-15-A11MU
~/Boutique-StoreSwap > docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
79a141207068   phpmyadmin/phpmyadmin  "/docker-entrypoint.s..." 37 seconds ago Up 36 seconds 0.0.0.0:8080->80/tcp, :::8080->80/tcp  phpmyadmin_container
0f2a3755ad0e   mysql         "docker-entrypoint.s..." 37 seconds ago Up 37 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp  mysql_container
18:13:07 lilian@lilian-Modern-15-A11MU
~/Boutique-StoreSwap > 
```

Sur le screenshot ci-dessus, on peut apercevoir que les deux conteneurs se sont instanciés. Le conteneur phpmyadmin permet d'héberger le serveur web permettant d'accéder à l'interface PhpMyAdmin (et donc administrer la base de données).

Le conteneur mysql permet d'héberger le serveur de base de données MySQL.

Une fois les étapes effectuées ci-dessus, vous pouvez vous connecter sur l'interface PhpMyAdmin à l'adresse suivante **http://localhost:8080** :



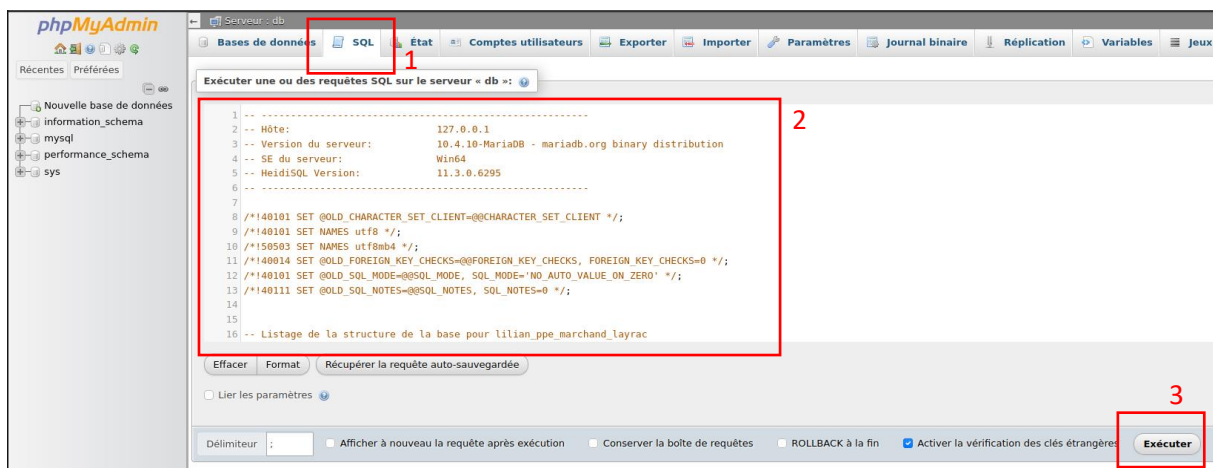
Dans ce formulaire, entrez les informations spécifiées dans le fichier `docker-compose.yml`. (En l'occurrence, l'utilisateur est root, et le mot de passe est un champ vide).

ATTENTION :

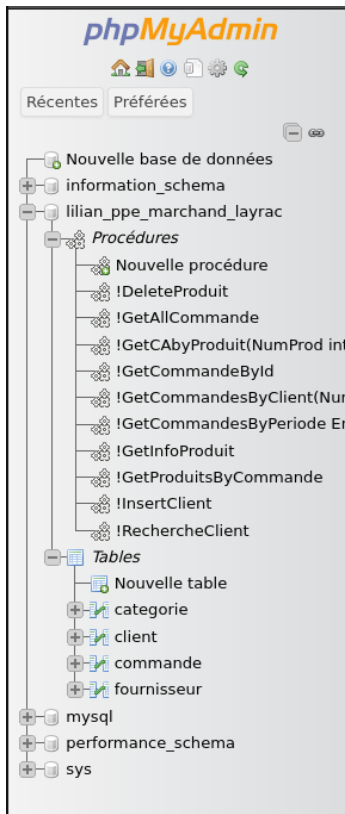
Il est fortement déconseillé de garder les identifiants par défaut dans un environnement de production.

Après la connexion, vous allez être redirigé vers le gestionnaire de base de données. Bien évidemment, ce dernier ne contiendra pas la base de données du site StoreSwap, il faudra donc copier le script de génération fournie dans le dépôt Boutique-StoreSwap, et l'exécuter comme une requête SQL simple.

Voici l'interface de gestionnaire de base de données :



En 1, vous avez l'onglet permettant d'exécuter les scripts SQL (requêtes SQL), en 2 vous avez le script SQL (disponible dans le dépôt github Boutique-StoreSwap), en 3 vous avez le bouton d'exécution de la requête.



Après exécution de la requête, vous devrez rafraîchir la page, et après vous aurez la base de données créée dans l'arborescence comme décrit à gauche :

Ensuite, vous n'avez plus qu'à créer vos virtual hosts, et ensuite vous connectez au site web à l'adresse suivante : **http://localhost/[VotreVirtualhost]**

7.1.10 – Comment est construite l'application IAPM

L'application IAPM est composée de classes et de fichiers WinForm permettant la modélisation graphique avec le framework intégré au package .NET.

Vous pouvez retrouver l'arborescence du projet en annexe.

(Voir Annexes pour IAPM – Annexe IAPM App « Page »)

Lorsque l'application effectue une requête à la base de données, elle appelle une requête stockée en base de données, via une procédure stockée, en voici les images :

Voici la procédure stockée appelée dans le code en fonction de l'identifiant de la commande passée en paramètre :

```
Public Shared Function GetProduitsByCommande(idCommande As Integer)
    Return GestionBDD.ExecuterRequeteSelect("Call '!GetProduitsByCommande'(" & idCommande & ")")
End Function
```

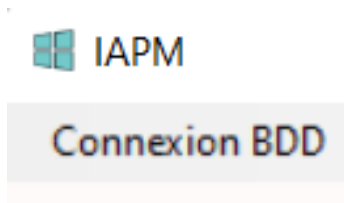
Voici le contenu de la procédure stockée en base de données :

Corps de la routine :

```
1 BEGIN
2
3 SELECT LibelleProduit, QteStockProduit, LibelleCategorie
4      FROM Produit, Fournisseur, Categorie, lignedecommande
5      WHERE Categorie.idCategorie = produit.idCat
6            AND Fournisseur.idFournisseur = Produit.idFourn
7            AND LignedeCommande.idCommande = Produit.idProduit
8            AND LignedeCommande.idCommande = " & idcom & " " " ;
9
10 END
```

7.1.11 – Sécurité de l’application IAPM

Pour la sécurité de l’accès à la base de donnée pour l’application IAPM, j’ai obligé le fait que l’utilisateur doit se connecter à la base de données avec ses identifiants MySql. Sinon, il n’a pas accès à l’application.



7.1.12 – Déploiement de l’application IAPM

Comme pour l’application IAPM, j’ai effectué un guide de déploiement avec un dépôt GitHub, ainsi qu’un déploiement automatisé avec docker-compose pour instancier les serveurs MySql et PhpMyAdmin.

7.2 – Application Mobile

7.2.1 – Choix de style pour IAPM Mobile

Le choix de style pour cette application mobile a été de se baser sur la même charte graphique que l’application desktop IAPM. L’interface est simple et épurée, avec très peu d’actions à effectuer pour arriver au but de l’utilisateur.

L’administrateur doit avoir l’information qu’il recherche instantanément, sans perdre de temps.

7.2.2 – Sécurité de l’application IAPM Mobile

Pour la sécurité de mon application IAPM Mobile, ainsi que la sécurité de la base de donnée, j’ai décidé d’effectuer une connexion systématique lorsque l’administrateur se connecte à l’application, cela veut dire qu’il est obligé de se connecter pour accéder à l’application.

Quand l'application est lancée, l'administrateur doit saisir son nom d'utilisateur ainsi que son mot de passe MySQL. C'est un compte qui a été créé au préalable dans la base de donnée MySQL via l'interface PhpMyAdmin.

Cela assure une sécurité totale des données, car, pas de connexion, est égale à pas d'accès et surtout manipulation des données.

Voici l'interface de connexion :



Aperçu des comptes utilisateurs

	Nom d'utilisateur	Nom d'hôte	Mot de passe	Privilèges globaux	« Grant »	Action
<input type="checkbox"/>	lilian	localhost	Oui	ALL PRIVILEGES	Oui	Éditer les privilèges Exporter
<input type="checkbox"/>	mysql.infoschema	localhost	Oui	SELECT	Non	Éditer les privilèges Exporter
<input type="checkbox"/>	mysql.session	localhost	Oui	SHUTDOWN, SUPER	Non	Éditer les privilèges Exporter
<input type="checkbox"/>	mysql.sys	localhost	Oui	USAGE	Non	Éditer les privilèges Exporter
<input type="checkbox"/>	root	localhost	Non	ALL PRIVILEGES	Oui	Éditer les privilèges Exporter

Ci-dessus, vous retrouvez les utilisateur MySQL, sur l'application Mobile IAPM ainsi que l'application IAPM Desktop, nous utilisons l'utilisateur lilian.

7.2.3 – Déploiement de IAPM Mobile

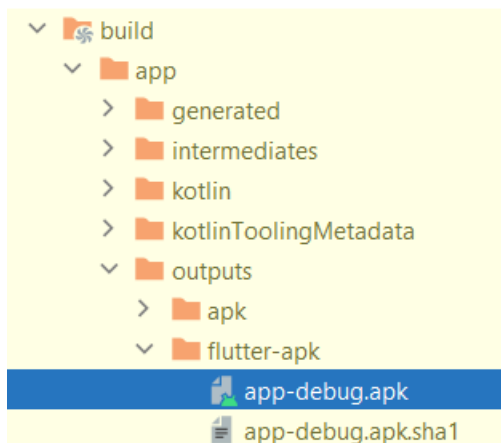
J'ai généré le .apk de l'application pour l'exécuter sur n'importe quel appareil android.

Il m'a suffi d'accéder à l'aide d'un terminal à la racine de mon projet IAPM Mobile, ensuite, j'ai exécuter la commande **flutter build apk**, et j'ai patienter pendant que flutter compile l'application et génère le .apk.

```
PS C:\Users\lilia\Desktop\EPSI\Certif_CDA_Contenue\Applications\Appli Mobile\iapm_mobile> flutter build apk

Running Gradle task 'assembleRelease'... 85,3s
✓ Built build\app\outputs\flutter-apk\app-release.apk (19.0MB).
```

Enfin, pour retrouver son fichier .apk avec son application compilée, il suffit d'aller dans



8 – Veille Technologique

8.1 – Les failles de php

PHP renforce sa sécurité en corrigeant plusieurs failles majeures

PHP, l'un des langages de programmation les plus populaires pour le développement web, a récemment annoncé la correction de plusieurs failles de sécurité critiques. Ces vulnérabilités ont été identifiées et résolues dans les versions récentes de PHP, renforçant ainsi la sécurité et la fiabilité de cette technologie fondamentale pour de nombreux sites web et applications en ligne.

Voici un aperçu des principales failles de sécurité corrigées dans PHP :

Vulnérabilité d'exécution de code à distance (RCE) : Cette faille permettait aux attaquants d'exécuter du code à distance sur un serveur PHP vulnérable, ce qui pouvait entraîner une compromission complète du système. Les développeurs de PHP ont pris des mesures pour renforcer les mécanismes de sécurité afin de prévenir cette menace. **(Source 1)**

Voici quel était le but de la faille :

Le but de la fonctionnalité testée était de permettre à l'utilisateur d'uploader des fichiers sur une plateforme pour qu'ils puissent être réutilisés ailleurs. Lorsque le fichier uploadé est un fichier audio ou vidéo, l'application PHP va lancer une commande sur le serveur afin de récupérer la durée du fichier et ainsi de pouvoir la communiquer à l'utilisateur. **(Source 1)**

Injection SQL : L'injection SQL est une faille courante dans les applications web qui utilisent des bases de données. Elle permet aux attaquants d'injecter du code SQL malveillant dans les requêtes, compromettant ainsi la sécurité des données. Les nouvelles versions de PHP ont introduit des mesures pour prévenir efficacement les attaques par injection SQL. **(Source 2)**

Cross-Site Scripting (XSS) : Le XSS est une vulnérabilité qui permet aux attaquants d'injecter des scripts malveillants dans des pages web visitées par d'autres utilisateurs. Cela peut entraîner le vol de données sensibles, le vol de session utilisateur ou encore des attaques de phishing. PHP a renforcé ses mécanismes de filtrage et d'échappement des données pour prévenir les attaques XSS. **(Source 2)**

5. *Divulgaration d'informations sensibles* : Cette faille permettait aux attaquants d'obtenir des informations sensibles telles que les mots de passe, les clés d'API ou les informations d'identification d'utilisateur à partir du code source PHP. Les correctifs récents ont renforcé la protection des informations sensibles, limitant ainsi les risques de divulgation.

6. *Exploitation de faille include avec les sessions php* : Cette exploitation de la faille "include" avec les sessions PHP se produit lorsque l'attaquant parvient à contrôler l'identifiant de session ou à injecter du code malveillant dans les données de session. Dans ce cas, l'attaquant peut manipuler l'inclusion de fichiers en fournissant un chemin de fichier relatif ou absolu malveillant. Cela peut entraîner l'inclusion de fichiers arbitraires sur le serveur, y compris des fichiers sensibles qui ne devraient pas être accessibles. **(Source 3)**

Il est essentiel pour les développeurs et les administrateurs de sites web d'adopter les dernières versions de PHP afin de bénéficier de ces corrections de sécurité. La mise à jour régulière du langage de programmation et des bibliothèques associées contribue à garantir un environnement en ligne plus sécurisé.

En conclusion, les failles de sécurité corrigées dans PHP témoignent de l'engagement de la communauté des développeurs à maintenir cette technologie de premier plan à jour et sécurisée. La vigilance continue et la mise à jour des systèmes avec les dernières versions de PHP restent essentielles pour assurer la sécurité des applications web et la protection des données des utilisateurs.

Sources :

<https://www.vaadata.com/blog/fr/vulnerabilite-rce-dans-un-nom-de-fichier/>

<https://www.techniques-ingenieur.fr/actualite/articles/lautre-menace-les-attaques-par-injections-sql-11046/>

<https://www.segmentationfault.fr/securite-informatique/exploitation-de-faille-include-avec-les-sessions-php/>

9 – Situation nécessitant un travail de recherche

Tout au long de la réalisation de ces projets, j'ai été confronté à de nombreuses situations qui ont nécessité un travail de recherche approfondi de ma part. Chaque projet, qu'il s'agisse de l'application web, de l'application IAPM Desktop ou de l'application IAPM Mobile, a présenté ses propres défis et exigences spécifiques. Dans chaque cas, j'ai dû me plonger dans des recherches approfondies pour acquérir les connaissances nécessaires et relever ces défis avec succès.

Pour l'application web StoreSwap, une étape cruciale a été de me familiariser avec la documentation de Bootstrap. Ce framework CSS très populaire offre une multitude de composants et d'outils pour le développement d'interfaces utilisateur réactives et esthétiquement plaisantes. En étudiant attentivement la documentation, j'ai pu exploiter les fonctionnalités offertes par Bootstrap de manière optimale, ce qui a considérablement facilité le processus de conception et de mise en page de l'interface utilisateur de mon application web.

En ce qui concerne l'application IAPM Desktop, j'ai également dû effectuer des recherches approfondies pour améliorer le design de l'application et optimiser la qualité du code. J'ai même dû refactoriser le code à plusieurs reprises pour parvenir à une architecture plus claire et modulaire. Au cours de mes recherches, j'ai découvert l'utilisation de procédures stockées, qui m'ont permis d'optimiser certaines parties du code en les rendant plus efficaces et réutilisables. Cette approche a grandement contribué à l'amélioration globale de la performance et de la maintenabilité de l'application.

Quant à l'application IAPM Mobile, j'ai consacré beaucoup de temps à la recherche de bonnes pratiques en Flutter, le framework de développement d'applications mobiles multiplateformes que j'ai utilisé. J'ai étudié les recommandations concernant l'organisation de l'arborescence des fichiers dans le projet, ainsi que la gestion efficace de la disposition des éléments de l'interface utilisateur pour offrir une expérience utilisateur fluide et agréable. Ces recherches m'ont permis d'optimiser la structure de mon projet Flutter, de rendre le code plus lisible et maintenable, et de garantir une expérience utilisateur cohérente sur différentes plateformes.

Dans l'ensemble, toutes ces recherches ont été extrêmement bénéfiques pour mes projets. Elles m'ont permis d'améliorer mes compétences, de perfectionner mes applications et de les personnaliser

davantage. De plus, ces efforts de recherche m'ont aidé à développer une attitude plus assidue et impliquée dans mes projets, car j'ai compris l'importance de rester constamment à jour avec les meilleures pratiques et les technologies émergentes.

10 – Remerciements

Je vous remercie pour votre lecture, votre attention et l'intérêt que vous avez apporté à ce dossier de projet destiné à la validation du diplôme Concepteur et Développeur d'Application.

Sincères remerciements,

Lilian Layrac

Terminé le 24/05/2023

11 – Glossaire

APK : Android Package Kit

BDD : Base de Données

CDA : Concepteur Développeur d'Application

CSS : Cascading Style Sheets

EPSI : Ecole d'Ingénierie Informatique

HTML : HyperText Markup Language

IDE : Integrated Development Environment

IAPM : Interface Applicatif de Produits Marchands

INT : Integer

IP : Internet Protocol

MCD : Modèle Conceptuel de Données

MB : MegaByte

MLD : Modèle Logique de Données

MVC : Model-View-Controller

MVVM : Model-View-ViewModel

PDO : PHP Data Objects

PHP : Hypertext Preprocessor

SIG : Système d'Information et de Gestion

SLAM : Solutions Logicielles et Applications Métiers

SQL : Structured Query Language

STMG : Sciences et Technologies du Management et de la Gestion

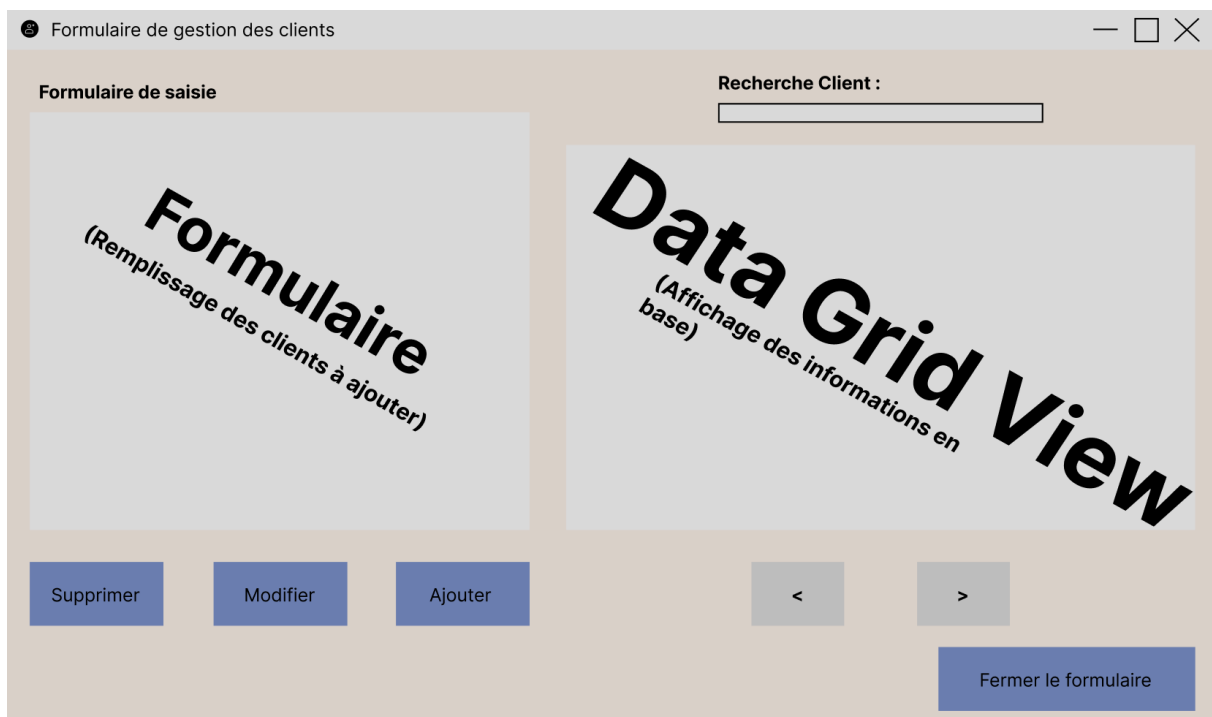
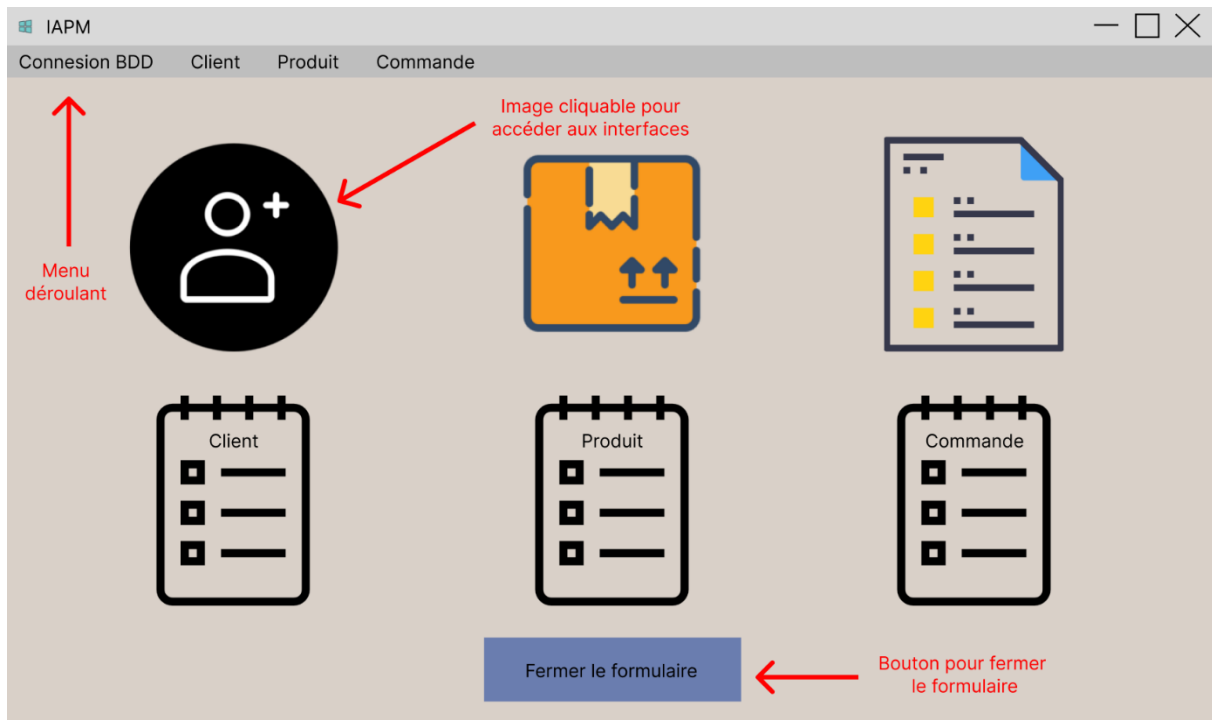
UML : Unified Modeling Language

XSS : Cross-Site Scripting

12 – Annexes

Annexes pour IAPM

Annexe IAPM Maquette



Charte Graphique

(Variants)

Background

D9D0C8

Boutons

6A7DAF

BDBDBD

Text

000000

E34444

Annexe IAPM Connexion

```
1 Imports MySql.Data
2 Imports IAMP_LAYRAC.GestionBDD
3
4 Public Class GestionBDD
5
6     'Filling the variables with the information related to the MySql.data package
7     Public Shared MaConnexion As New MySqlConnection
8     Public Shared MaCommande As New MySqlCommand
9     Public Shared MonAdapter As New MySqlDataAdapter
10    Public Shared MonJeuDeDonnees As New DataSet
11    Public Shared MaRequete As String
12    Public Shared Machine As String = "Data Source=localhost;Initial Catalog=lilian_ppe_marchand_layrac_adapt;UserID=root;Password=;Convert Zero Datetime=True"
13    Public Shared CptVue As Integer = 0
14
15    'Dump our dataset
16    Public Shared Sub ViderMonJeuDeDonnees()
17        MonJeuDeDonnees.Clear()
18    End Sub
19
20    'Allows to connect to the database according to the information filled in the variables above
21    Public Shared Sub SeConnecter()
22        MaConnexion.ConnectionString = Machine
23        MaConnexion.Open()
24        MonJeuDeDonnees = New DataSet("IAPM_LAYRAC")
25        MonJeuDeDonnees.Clear()
26        MaCommande.CommandType = CommandType.Text
27        MaCommande.Connection = MaConnexion
28    End Sub
```

Annexe IAPM BDD

Une procédure stockée qui effectue la recherche d'un client :

Corps de la routine :

```
1 CREATE DEFINER=`root`@`localhost` PROCEDURE `!RechercheClient`(  
2     IN `Critere1` CHAR(20),  
3     IN `Critere2` CHAR(20)  
4 )  
5 LANGUAGE SQL  
6 NOT DETERMINISTIC  
7 CONTAINS SQL  
8 SQL SECURITY DEFINER  
9 COMMENT ''  
10 BEGIN  
11  
12     SELECT idClient, nomClient, prenomClient  
13     FROM Client  
14     WHERE nomClient LIKE CONCAT('%',Critere1,'%')  
15     OR prenomClient LIKE CONCAT('%',critere2,'%');  
16  
17 END
```

Un déclencheur qui s'occupe de la gestion des statistiques :

Corps d'exécution :

```
1 CREATE DEFINER=`root`@`localhost` EVENT `GestionStats`  
2 ON SCHEDULE  
3     EVERY 10 SECOND STARTS '2023-03-17 07:46:15'  
4 ON COMPLETION PRESERVE  
5 ENABLE  
6 COMMENT ''  
7 DO BEGIN  
8  
9 DECLARE commande INT;  
10 DECLARE client INT;  
11 DECLARE produit INT;  
12  
13 SET commande = (SELECT COUNT(idCommande) FROM commande);  
14 SET client = (SELECT COUNT(idClient) FROM client);  
15 SET produit = (SELECT COUNT(idProduit) FROM produit);  
16  
17 INSERT INTO statistiques (NBCommandes, NBClients, NBProduits, DateStatistiques)  
18 VALUES (commande, client, produit, NOW());  
19  
20  
21 END
```

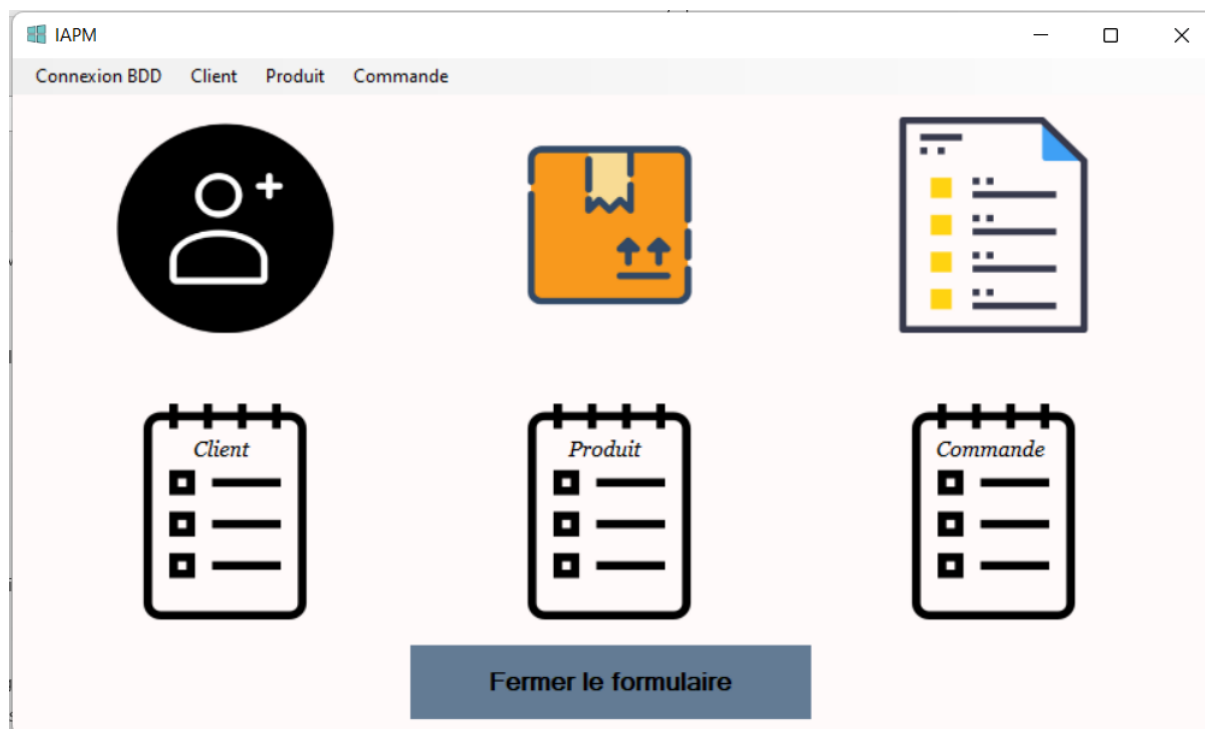
Annexe IAPM Résultat Trigger

	NBCommandes	NBClients	NBProduits	DateStatistiques
►	100	103	6	10/05/2023 12:02
*				

Nombre de commandes en temps réel

Annexe IAPM Tests Unitaires

Annexe IAPM App Menu



Annexe IAPM App Ajout Client

The screenshot shows the 'Formulaire de gestion des clients' interface. On the left, there is a 'Formulaire de saisie' section with input fields for 'Code Client' (value: 1), 'Nom' (value: Adrian), 'Prenom' (value: Tyler), 'Rue' (value: Ap #257-4830 Accumsan St.), 'Code Postal' (value: 55137), and 'Ville' (value: Poitiers). Below these fields are three buttons: 'Supprime', 'Modifier', and 'Ajouter'. On the right, there is a table titled 'Rechercher un client dans la base de donnée'. The table has columns: 'idClient', 'NomClient', 'PrenomClient', 'AdRueClient', 'AdCpClient', and 'AdVilleClient'. The table contains 12 rows of data. Below the table are four navigation buttons: a left arrow, a right arrow, a double left arrow, and a double right arrow. At the bottom right is a large blue button labeled 'Fermer le formulaire'.

idClient	NomClient	PrenomClient	AdRueClient	AdCpClient	AdVilleClient
1	Adrian	Tyler	Ap #257-48...	55137	Poitiers
2	Reese	Kirby	P.O. Box 13...	10357	Tournefeuille
3	Clark	Ainsley	959-5721 A...	68833	La Roche-s...
4	Samantha	Alana	Ap #113-71...	79986	Douai
5	Evan	Zachery	Ap #633-39...	68378	Angoulême
6	Nathaniel	Channing	Ap #437-53...	89910	Beauvais
7	Risa	Lars	791-277 Se...	83333	Saint-Dizier
8	Cole	Rudyard	345-3960 S...	77815	Saint-Dizier
9	Hanna	Macaulay	7597 Lobort...	85829	Alès
10	Cyrus	Reed	Ap #294-32...	1601	Colomiers
11	Willa	Donna	Ap #823-21...	79321	Nîmes
12	Wallace	Tucker	892-2263 Et...	64979	Dijon

Annexe IAPM App Liste Client

Formulaire de la liste des clients

Rechercher un client dans la base de donnée

	idClient	NomClient	PrenomClient	AdRueClient	AdCpClient	AdVilleClient
▶	1	Adrian	Tyler	Ap #257-48...	55137	Poitiers
	2	Reese	Kirby	P.O. Box 13...	10357	Tournefeuille
	3	Clark	Ainsley	959-5721 A...	68833	La Roche-s...
	4	Samantha	Alana	Ap #113-71...	79986	Douai
	5	Evan	Zachery	Ap #633-39...	68378	Angoulême
	6	Nathaniel	Channing	Ap #437-53...	89910	Beauvais
	7	Risa	Lars	791-277 Se...	83333	Saint-Dizier
	8	Cole	Rudyard	345-3960 S...	77815	Saint-Dizier
	9	Hanna	Macaulay	7597 Lobort...	85829	Alès
	10	Cyrus	Reed	Ap #294-32...	1601	Colomiers
	11	Willa	Donna	Ap #823-21...	79321	Nîmes
	12	Wallace	Tucker	892-2263 Et...	64979	Dijon
	13	Shad	Shad	138-6227 N...	42094	Le Havre
	14	Cynthia	Shelley	394-3303 Si...	84110	Roubaix
	15	Cade	Sonya	6413 Enim. ...	24175	Pessac
	16	Movada	Kimberly	Ap #529-84...	19632	Vitrac-sur-Sei

Fermer la liste

Annexe IAPM App Ajout Produit

Formulaire de gestion des produits

Rechercher un produit dans la base de donnée

Formulaire de saisie

Code Produit : 1

Libellé : Crocodile Bleu (5 pièces)

Prix HT : 2

Quantité Prod: 245

Fournisseur : Abiroh

Catégorie : Basique

	idProduit	LibelleProduit	PrixHTProduit	QteStockProduit	NomFournisse	LibelleCategor	isImportant
▶	1	Crocodile Bl...	2.00 €	245	Abiroh	Basique	0
	2	Crocodile Vi...	1.99 €	235	Bonbon&co	Basique	0
	3	Crocodile Gr...	1.99 €	568	LyrcCorp	Piquant	0
	4	Boîte de Cro...	19.99 €	58	Abiroh	Basique	1
	5	Crocodile R...	1.99 €	457	Abiroh	Piquant	0
	6	Crocodile V...	1.99 €	154	Abiroh	Basique	1
*							

Supprimer

Modifier

Ajouter

Générer une liste des produits (PDF)

Fermer le formulaire

Annexe IAPM App Liste Produit

Formulaire de la liste des produits

Rechercher un produit dans la base de donnée

	idProduit	LibelleProduit	PrixHTProduit	QteStockProduit	idFourn	idCat	cheminImage	descriptionIma	isImportant
▶	1	Crocodile Bl...	2	245	1	1	croco_bleu....	Excellent cr...	0
	2	Crocodile Vi...	1,99	235	2	1	croco_violet...	Excellent cr...	0
	3	Crocodile Gr...	1,99	568	3	2	croco_gris.p...	Excellent cr...	0
	4	Boîte de Cro...	19,99	58	1	1	croco_grou...	Excellent as...	1
	5	Crocodile R...	1,99	457	1	2	croco_rose....	Excellent cr...	0
	6	Crocodile V...	1,99	154	1	1	croco_vert.p...	Excellent cr...	1
*									

Fermer la liste

Annexe IAPM App Liste Commande

Formulaire de la liste des commandes

Rechercher une commande dans la base de donnée

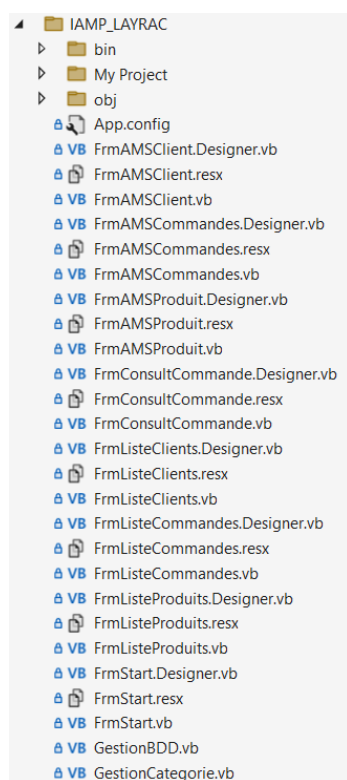
	idCommande	DateCommanc	idCli
▶	1	08/01/2018	74
	2	05/09/2019	98
	3	28/12/2017	95
	4	07/12/2017	68
	5	05/05/2020	25
	6	14/07/2018	18
	7	07/04/2017	66
	8	29/07/2019	27
	9	28/06/2017	39
	10	28/10/2019	19
	11	22/09/2017	40
	12	02/04/2020	00

	NBCommande	NBClients	NBProduits	DateStatistique
▶	100	103	6	18/05/2023 ...
*				

Nombre de commandes en temps réel

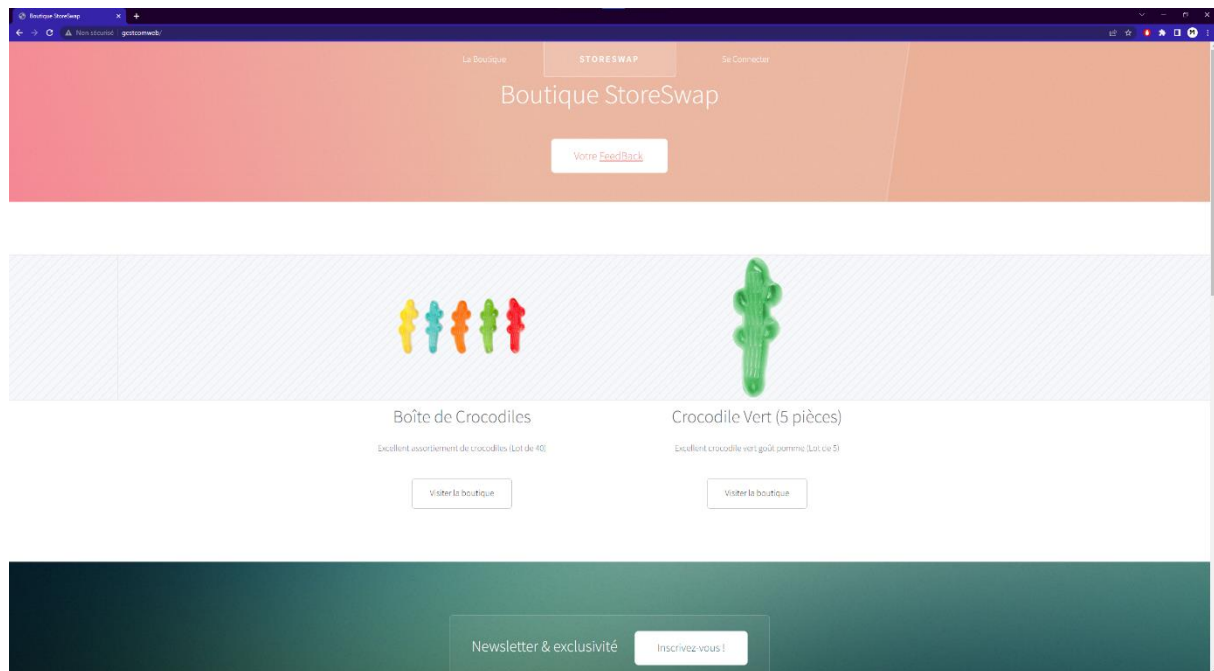
Fermer la liste

Annexe IAPM Arborescence

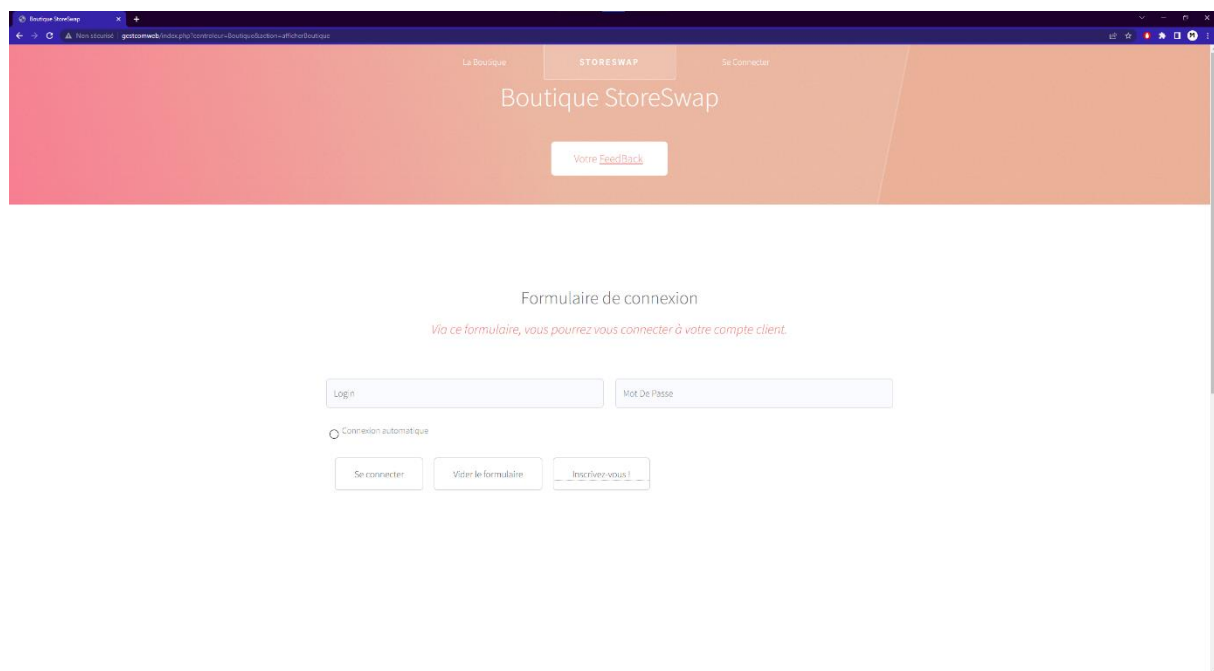


Annexes pour StoreSwap

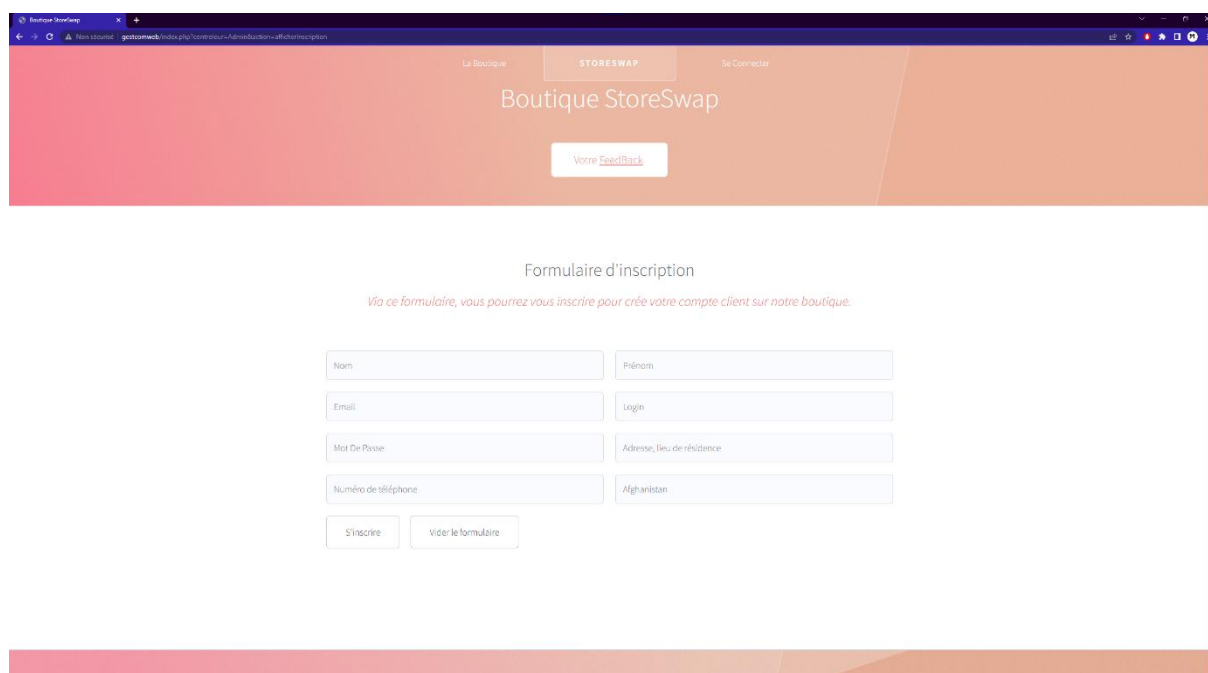
Annexe StoreSwap Accueil



Annexe StoreSwap Connexion



Annexe StoreSwap Inscription



The screenshot shows the 'Boutique StoreSwap' registration page. At the top, there's a navigation bar with 'La Boutique', 'STORESWAP', and 'Se Connecter'. Below this is a header with the store name and a 'Votre Feedback' button. The main section is titled 'Formulaire d'inscription' with a subtext: 'Via ce formulaire, vous pourrez vous inscrire pour créer votre compte client sur notre boutique.' The form consists of two columns of input fields: 'Nom', 'Prénom', 'Email', 'Login', 'Mot De Passe', 'Adresse, lieu de résidence', 'Numéro de téléphone', and 'Afghanistan'. At the bottom of the form are two buttons: 'S'inscrire' and 'Vider le formulaire'.

La Boutique STORESWAP Se Connecter

Boutique StoreSwap

Votre Feedback

Formulaire d'inscription

Via ce formulaire, vous pourrez vous inscrire pour créer votre compte client sur notre boutique.

Nom Prénom

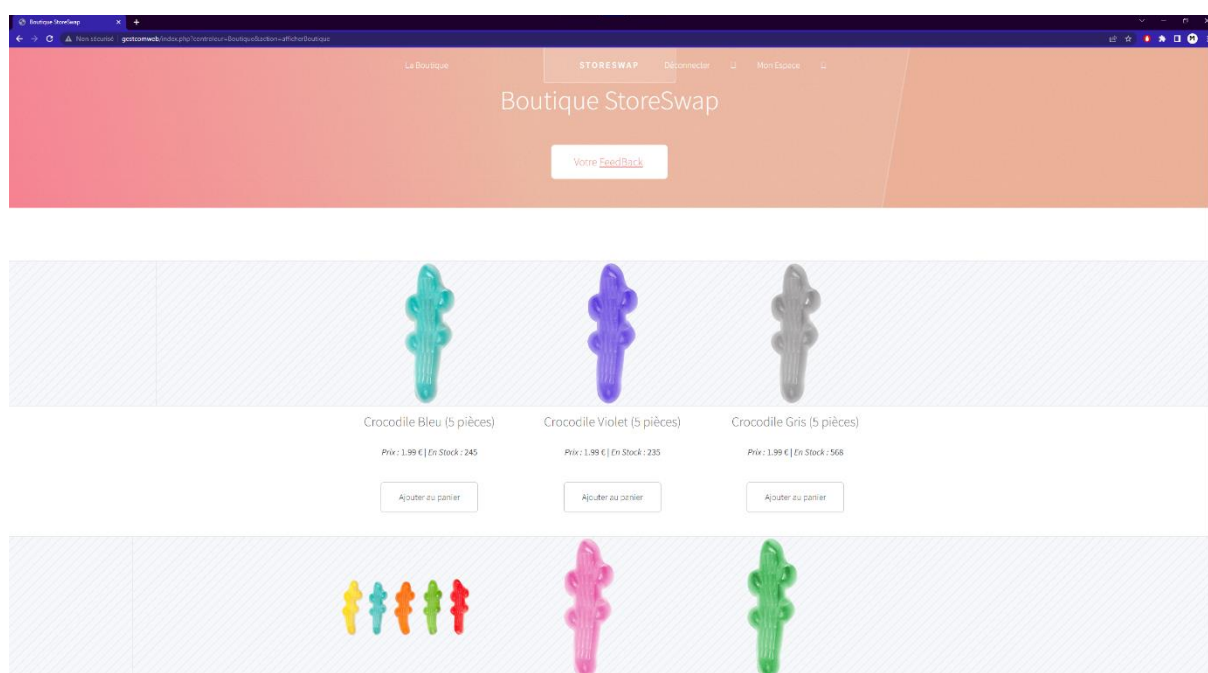
Email Login

Mot De Passe Adresse, lieu de résidence

Numéro de téléphone Afghanistan

S'inscrire Vider le formulaire

Annexe StoreSwap Boutique



Annexe StoreSwap Panier

The screenshot shows the 'Boutique StoreSwap' shopping cart page. The header includes the store name, a 'STORESWAP' logo, and links for 'Déconnecter', 'Mon Espace', and 'La Boutique'. A 'Votre Feedback' button is visible. The main content area displays a table of items in the cart:

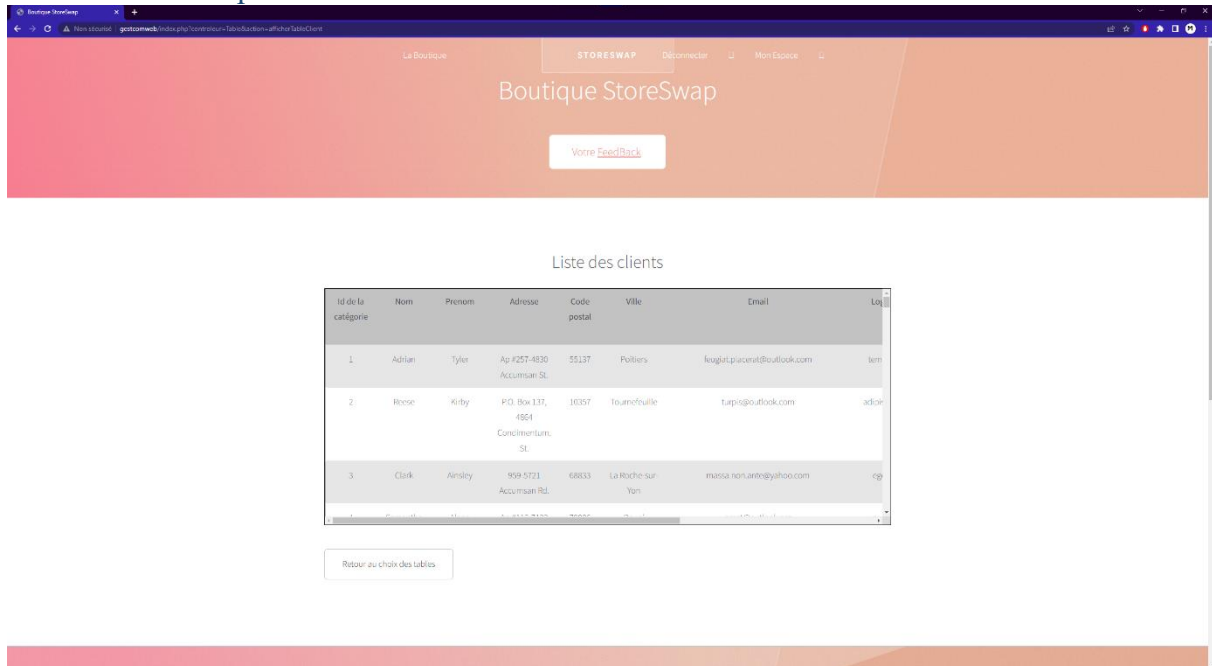
Produit	Disponibilité	Quantité	Prix	
Crocodile Bleu (5 pièces)	En stock	<input type="text" value="1"/>	1.99 €	
Crocodile Violet (5 pièces)	En stock	<input type="text" value="1"/>	1.99 €	
Crocodile Rose (5 pièces)	En stock	<input type="text" value="1"/>	1.99 €	
Boîte de Crocodiles	En stock	<input type="text" value="1"/>	15.99 €	
			Sous-total	20.96 €
			Livraison	5 €
			Total	30.96 €

Below the table are three buttons: 'Vider le panier', 'Acheter le panier', and 'Retour à la boutique'. At the bottom of the page is a 'Formulaire d'avis client'.

Annexe StoreSwap Admin Table

The screenshot shows the 'Administration du site (Accès réservé)' page. The header is identical to the shopping cart page. The main content area includes a greeting '- Bonjour admin -' and links to 'Accès à la boutique' and 'Déconnexion (admin)'. Below this is a section titled 'Naviguer sur les tables de la base de donnée' with several buttons: 'Avis Client', 'Catégories', 'Client', 'Commande', 'Fournisseur', 'Produit', and 'Utilisateur'. At the bottom of the page is a 'Formulaire d'avis client'.

Annexe StoreSwap Admin Table Client



Annexe StoreSwap Code Connexion

application > modeles > `modelePDO.class.php` > `ModelePDO` > `seConnecter`

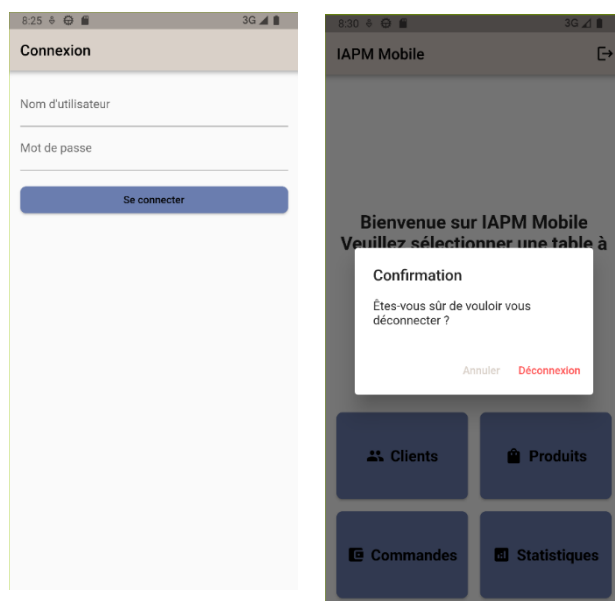
```

1  <?php
2
3  class ModelePDO {
4
5      // <editor-fold defaultstate="collapsed" desc="Champs Statiques">
6      //Connexion
7      protected static $serveur = MySQLConfig::SERVEUR;
8      protected static $base = MySQLConfig::BASE;
9      protected static $utilisateur = MySQLConfig::UTILISATEUR;
10     protected static $passe = MySQLConfig::MOT_DE_PASSE;
11     //Manipulate PDO from DB
12     protected static $pdoCnxBase = null;
13     protected static $pdoStResults = null;
14     protected static $requete = "";
15     protected static $resultat = null;
16
17     // </editor-fold>
18     // <editor-fold defaultstate="collapsed" desc="Méthodes Statique">
19
20     public static function seConnecter() {
21         if (!isset(self::$pdoCnxBase)) { //If there is no connexion yet
22             try {
23                 self::$pdoCnxBase = new PDO('mysql:host=' . MySQLConfig::SERVEUR . ';dbname=' .
24                     MySQLConfig::BASE, MySQLConfig::UTILISATEUR, MySQLConfig::MOT_DE_PASSE);
25                 self::$pdoCnxBase->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
26                 self::$pdoCnxBase->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_OBJ);
27                 self::$pdoCnxBase->query("SET CHARACTER SET utf8");
28             } catch (Exception $e) {
29                 //The object pdoCnxBase is generating automatically an object from Exception type
30                 echo 'Erreur : ' . $e->getMessage() . '<br />'; //Method from Exception class
31                 echo 'Code : ' . $e->getCode(); //Method from Exception class
32             }
33         }
34     }
35
36     public static function seDeconnecter() {
37         self::$pdoCnxBase = null;
38     }

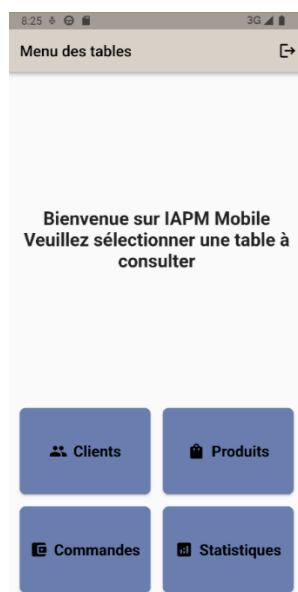
```

Annexes pour IAPM Mobile

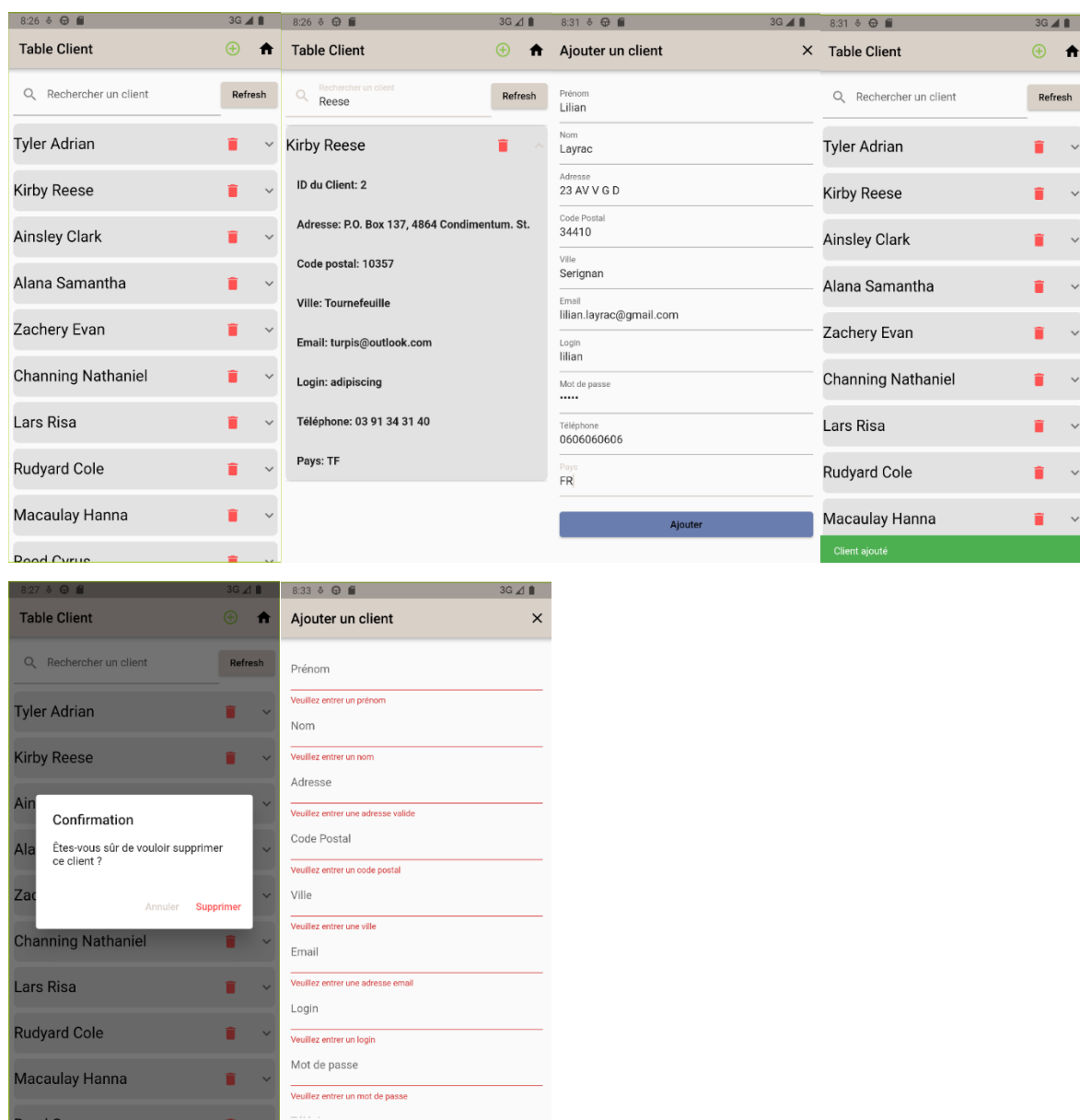
Annexe IAPM Mobile Connexion & Déconnexion



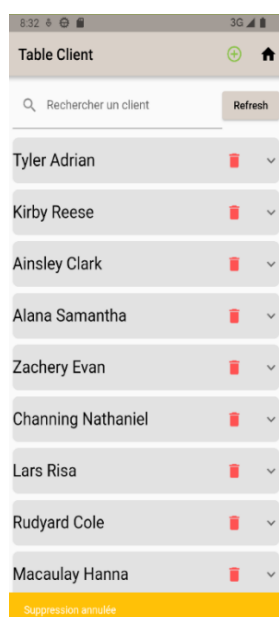
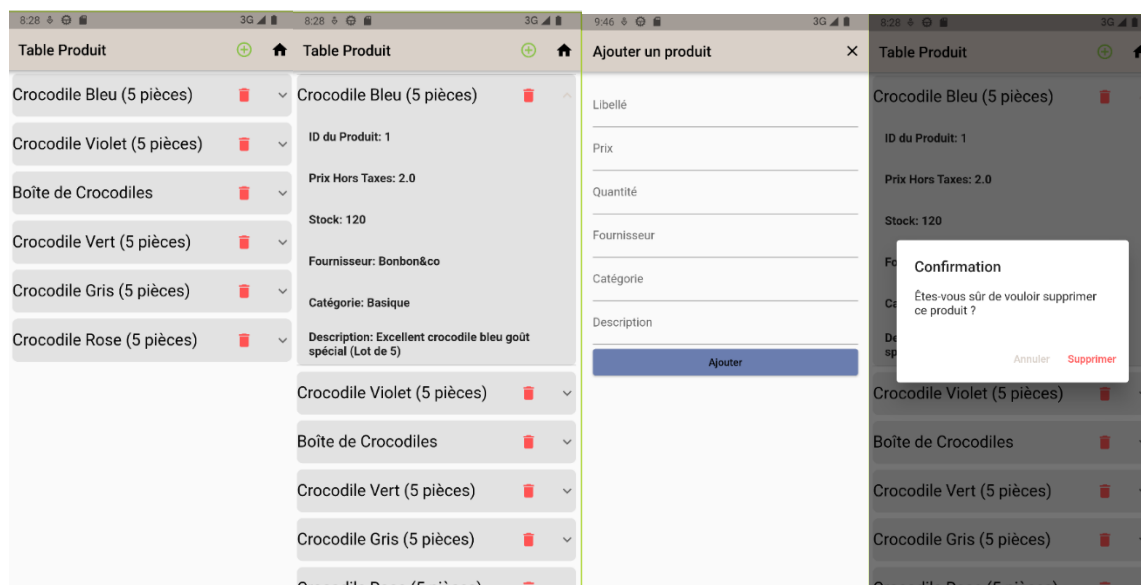
Annexe IAPM Mobile Accueil



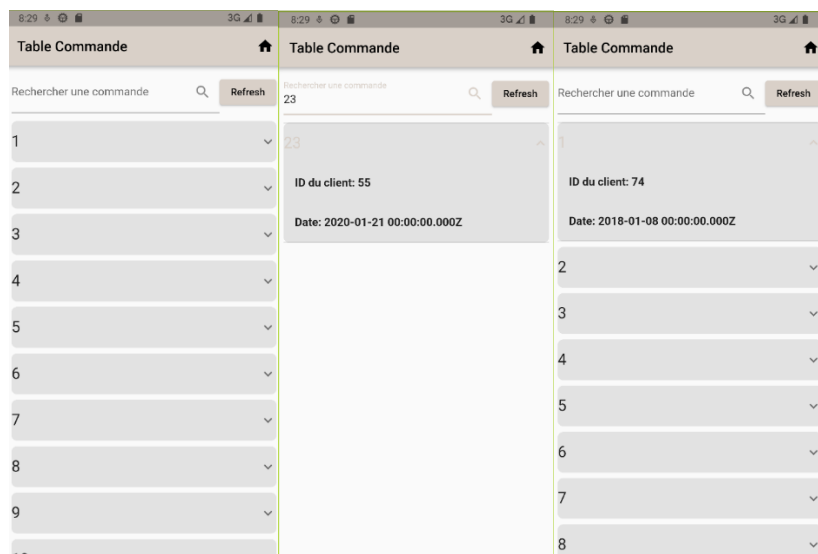
Annexe IAPM Mobile Client, Recherche, Ajout, Popup Success, Supprimer, Erreur Saisie



Annexe IAPM Mobile Produit, Info, Ajout, Suppression, Popup Annul Suppression



Annexe IAPM Mobile Commande, Recherche, Info



Annexe IAPM Mobile Statistiques

