|  |  |
| --- | --- |
|  | **2018** |
|  | AFTEC Rennes |

|  |
| --- |
| **[SPECIFICATION tECHNIQUE – algobreizh]** |
| Dossier d’analyse logicielle pour le projet fictif Algobreizh réalisé dans le cadre du BTS SIO (Option SLAM). Auteurs : PILORGE Dorian, BESRET Paul et MARTINEZ Quentin |

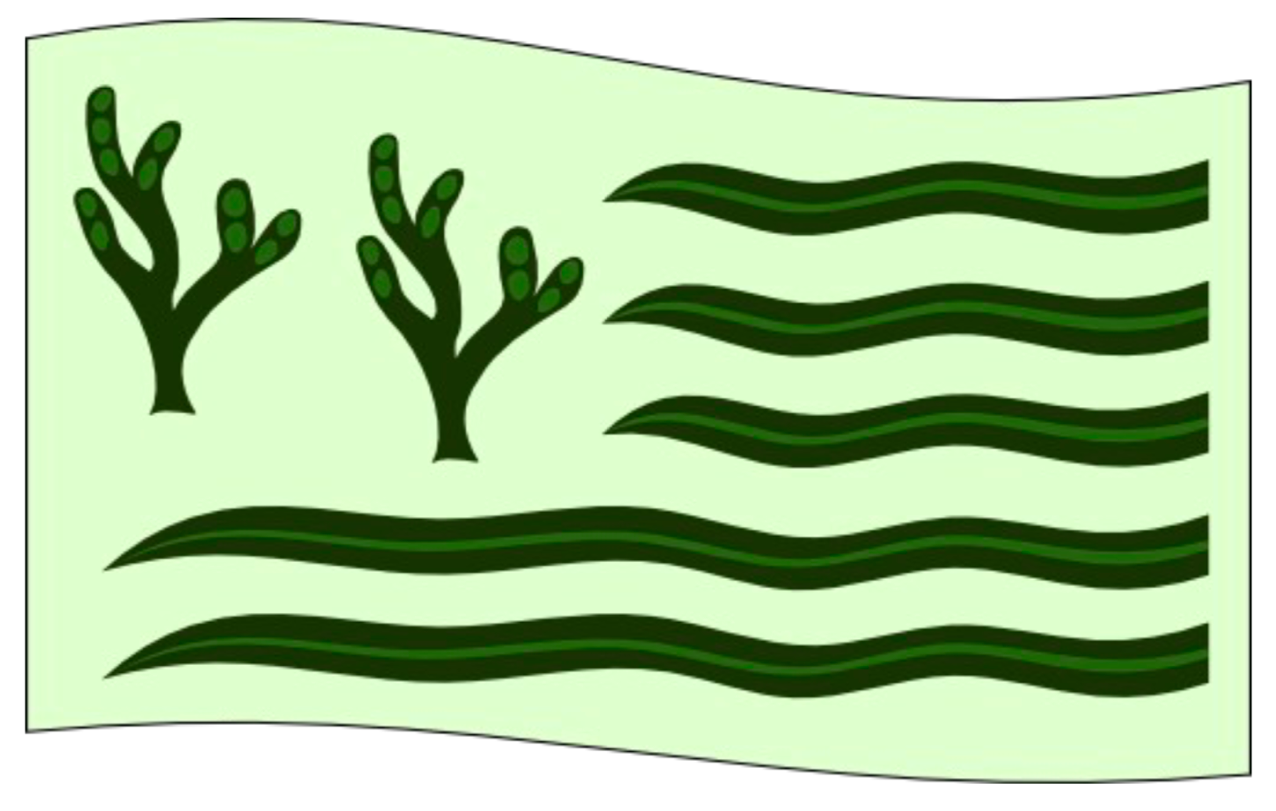


Table des matières

[Commandes en ligne 2](#_Toc508199958)

[Diagramme MCD 3](#_Toc508199959)

[Diagramme LMD 3](#_Toc508199960)

[Diagramme de classe 4](#_Toc508199961)

[Processus 5](#_Toc508199962)

Spécifications techniques

# Commandes en ligne

Nous avons développé un site e-commerce permettant à la société Algobreizh de mettre en vente leurs produits et de passer commande. Pour ce faire nous avons développé l’application en PHP pour l’exécution serveur (back-end) et en HTML/CSS/Javascript pour les interfaces utilisateurs (front end).

Les librairies utilisées pour l’interface utilisateur sont :

* **JQuery** (3.5.1) : afin d’apporter plus de confort à l’utilisateur pour les transitions et les fenêtres modales.
* **Bootstrap** 3 : afin de mettre au point un style moderne et accueillant pour les clients.

Les librairies utilisées en back end :

* **Fpdf** est utilisée pour générer les factures clients.

L’architecture back end de notre application est basé sur un modèle **MVC** (Modèle Vue Contrôleur).

Les différentes parties de notre application sont détaillés ci-dessous.

* Les **modèles** sont hydratés et sauvegardés par nos managers. Cependant ils sont utilisés par les vues pour la sortie utilisateur et par les contrôleurs pour la logique dite « métier ».

Modèle : Customer, Product, Order, AttachedProduct.

* Les **vues** utilisent les objets (modèles) fournit en paramètre, pour compléter le contenu et le gabarit commun à toutes les vues.

*Vues : View, ErrorView, LoginView, RegisterView, CartView, OrdersAdminView, OrdersView, ProductsView, WelcomeView*

* Les **contrôleurs** permettent après un traitement de la requête HTTP produite par l’utilisateur d’effectuer la logique métier et la sortie attendue.

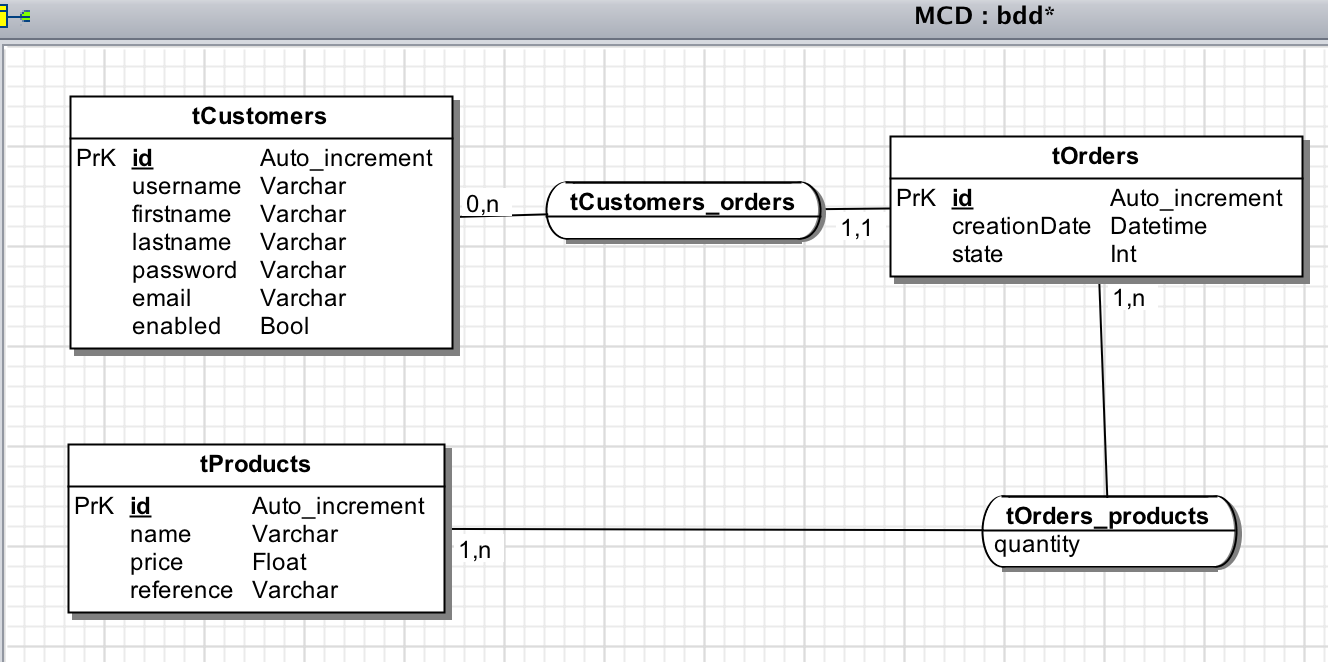
*Contrôleurs : Routeur, OrdersControler, ProductControler, LoginControler, WelcomeControler*

Nous avons utilisé le concept de « manager », ceux-ci forment nos CRUD (Create, Read, Update, Delete). En effet ces objets permettent la lecture, la modification, l’ajout et la suppression en base de données. Ils sont majoritairement utilisés par les contrôleurs.

*Managers : OrdersManager, CustomersManager, CartManager, ProductsManager.*

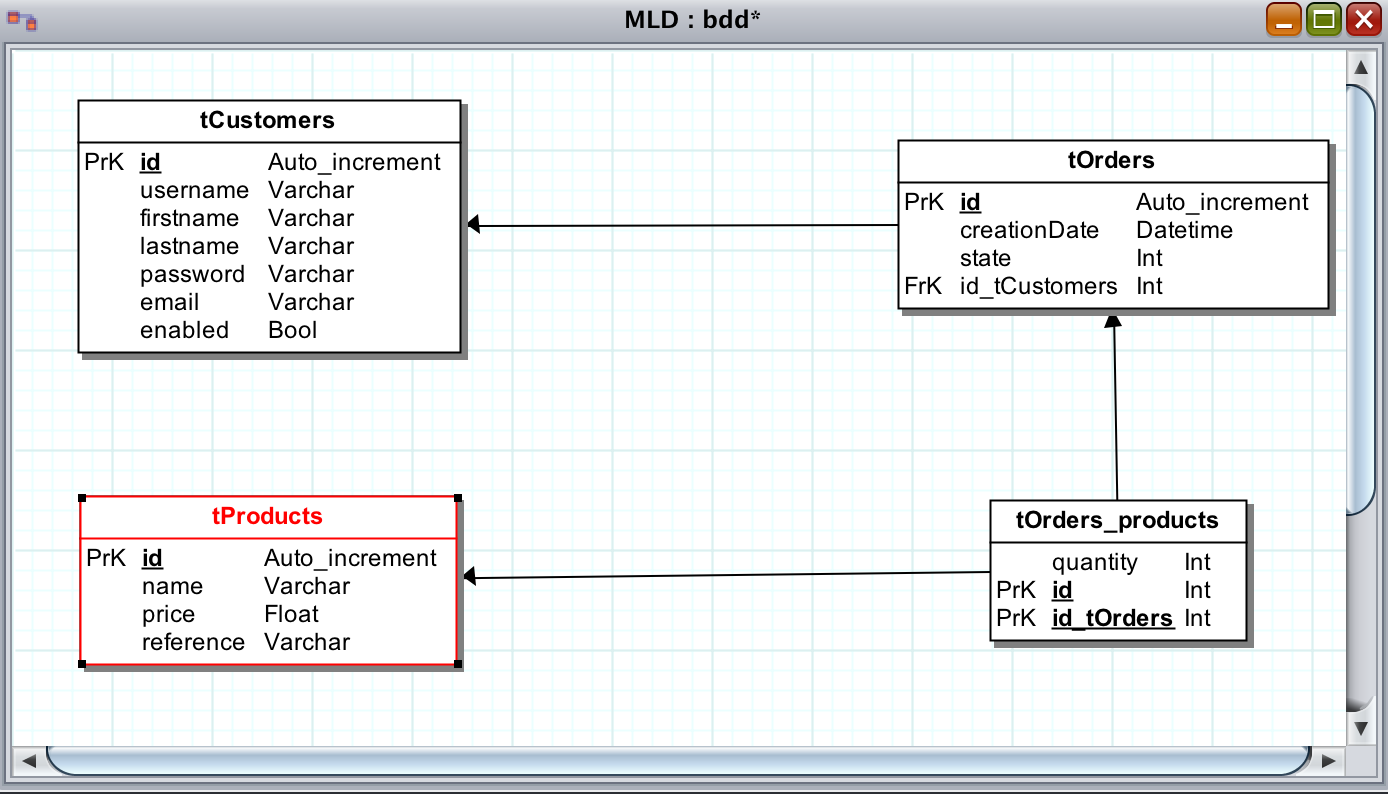
# Diagramme MCD

Pour modéliser notre base de données nous avons utilisé JMerise. Ainsi nous avons pu créer un diagramme MCD (ci-dessous) basé sur les spécifications fonctionnelles.



# Diagramme LMD

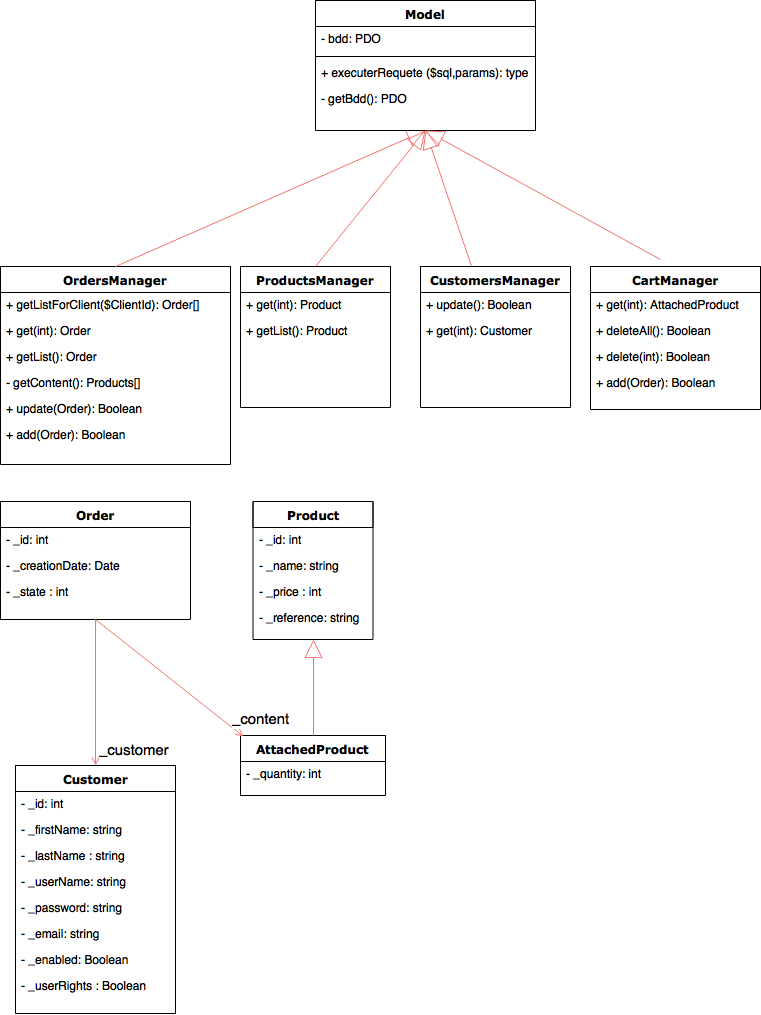
Pour connaître les tables intermédiaires (ci-dessous) nous avons générer un diagramme MLD à partir de la modélisation MCD avec JMerise.



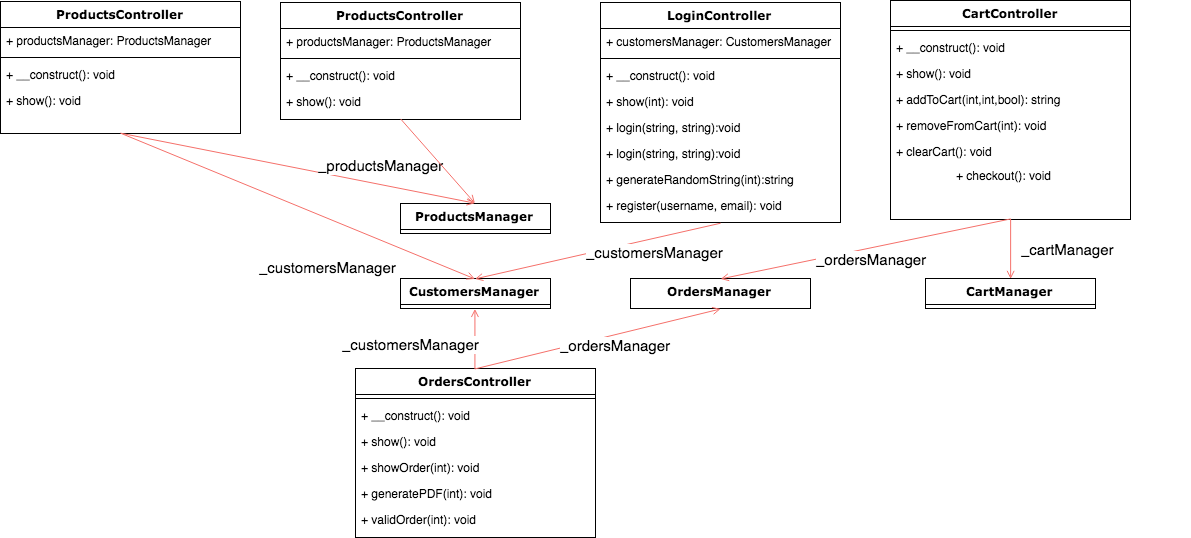
# Diagramme de classe

Tous les « Managers » implémente la classe « Model » et héritent ainsi d’une connexion à la base de données pour exécuter les requêtes SQL (SELECT, UPDATE, INSERT, DELETE) sur les tables de l’application. 4 entités sont déclarées (Order, Product, Customer, AttachedProduct).

L’entité AttachedProduct apporte la notion de quantité et hérite des propriétés de l’entité Product pour le bon fonctionnement du panier utilisateur.



Les contrôleurs utilisent les managers d’objet pour exposés des méthodes publiques utilisable dan



# Processus

Acheminement procédurale d’une action utilisateur.

