Le site de vente de la sorcière Giselle

Rapport de projet

**GiselleMagicArts.com**



Table des matières

[I. Présentation du projet 3](#_Toc478646897)

[1. Contexte 3](#_Toc478646898)

[2. Besoins 4](#_Toc478646899)

[3. L’équipe projet 4](#_Toc478646900)

[II. Planification du projet 5](#_Toc478646901)

[1. Découpage du projet : le WBS 5](#_Toc478646902)

[2. Planning prévisionnel 5](#_Toc478646903)

[3. Planning effectif 9](#_Toc478646904)

[III. Mise en place de la solution proposée 10](#_Toc478646905)

[1. Modélisation : la méthode MERISE 10](#_Toc478646906)

[a. Modèle conceptuel de données (MCD) 10](#_Toc478646907)

[b. Modèle logique de données (MLD) 12](#_Toc478646908)

[c. Choix du SGBDR 13](#_Toc478646909)

[d. Modèle physique de données (MPD) 13](#_Toc478646910)

[2. Les requêtes SQL 13](#_Toc478646911)

[a. Remplissage de la base de données 13](#_Toc478646912)

[b. Procédures stockées 13](#_Toc478646913)

[c. Soumission de recettes 13](#_Toc478646914)

[d. Rechercher et consulter 13](#_Toc478646915)

[e. Sauvegarder 13](#_Toc478646916)

[3. Tests 13](#_Toc478646917)

[ANNEXE 1 : Créer un serveur mysql sur un Raspberry Pi 14](#_Toc478646918)

[ANNEXE 2 : Script de création de la base de données 15](#_Toc478646919)

# Présentation du projet

## Contexte

Giselle, jeune sorcière ayant obtenue récemment son permis de magie souhaite se lancer dans la vente de potions et d’ingrédient. Elle souhaite donc réaliser un site e-commerce de magie : gisellemagicarts.com.

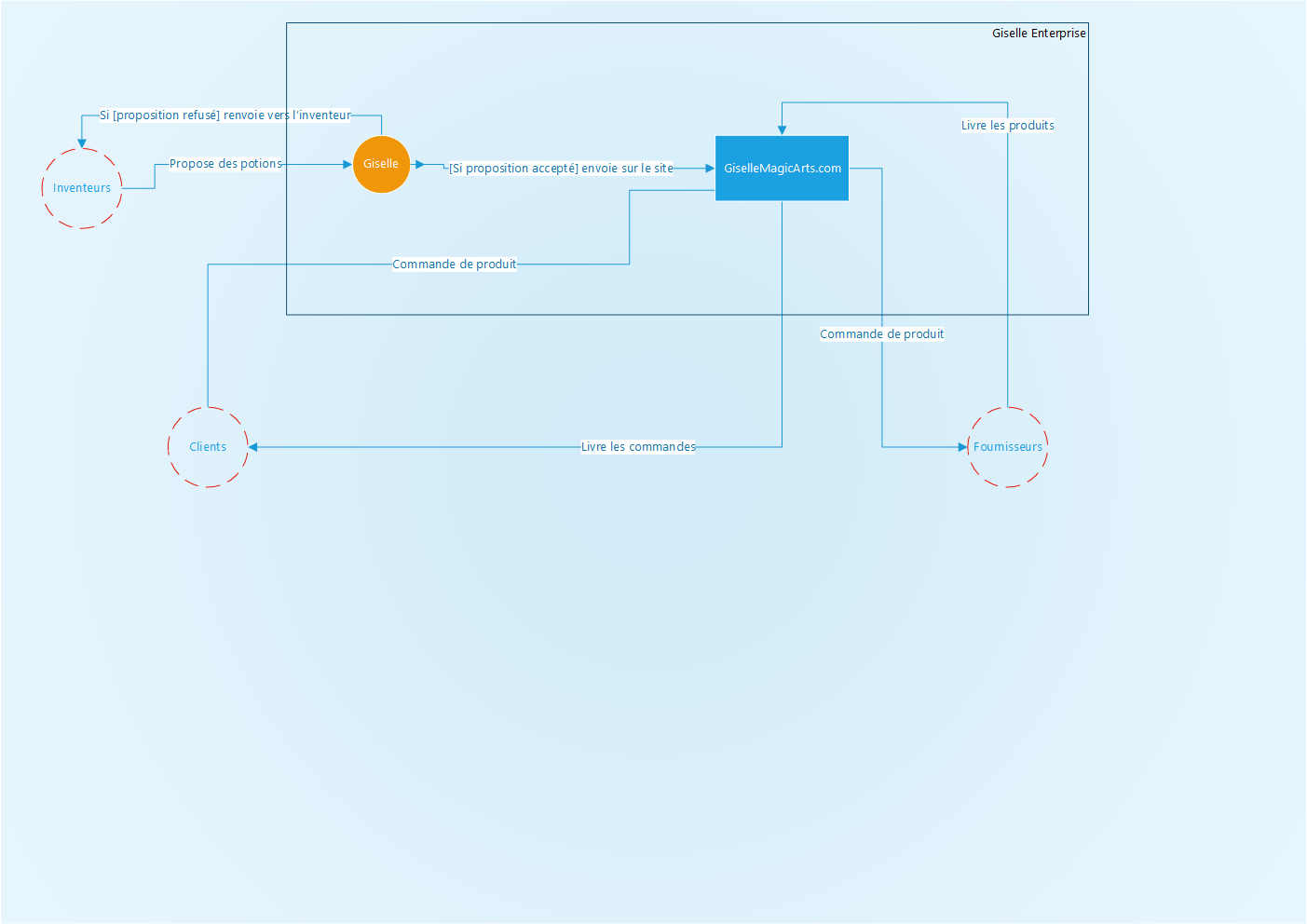
La sorcière a donc besoin d’une interface web ainsi que d’une base de données pour stocker toutes les informations nécessaires au bon fonctionnement de son site. Sur ce dernier sera vendu :

* Des potions
* Des onguents
* Des ingrédients (pour préparer des potions et des onguents)3

Pour expliquer le fonctionnement du site GiselleMagicArts nous avons décidé d’établir un diagramme du flux. Le but est de bien comprendre le fonctionnement du futur site pour développer un modèle contextuel de données adapté.

Tout d’abord listons quelques informations complémentaires concernant le fonctionnement du site :

* Giselle ne se procure pas elle-même les ingrédients mais fait appel à des fournisseurs.
* Les clients commandent puis sont livrés
* De nouveaux inventeurs peuvent proposer des recettes de nouvelles potions. Giselle choisit ensuite si elle souhaite ajouter les propositions de potions au site ou non.

Les acteurs externes sont représentés par des cercles à traits pointillés, les acteurs internes par des cercles à traits pleins et les flux par des flèches de l’émetteur vers le récepteur.

## Besoins

Un ami de Giselle, Miguel se charge de réaliser le site web. Ce projet consistera donc en la création d’une base de données et des requêtes nécessaires au bon fonctionnement de ce site.

Giselle souhaite surveiller de manière consciencieuse la fraîcheur de chaque ingrédient. En effet, la fraîcheur d’un ingrédient peut avoir un effet sur la qualité d’une potion. Elle souhaite également que les ingrédients qui ne sont plus frais soient automatiquement supprimés de la base de données. De plus les clients peuvent préciser la fraîcheur souhaitée des ingrédients qu’ils commandent (un même ingrédient peut être présent deux fois dans une commande mais avec des fraîcheurs différentes).

Les clients, lorsqu’ils commandent des potions, doivent choisir dans quel récipient celle-ci sera contenu (fioles, pots…). Le prix variera en fonction du type de récipient.

Il faudra préparer un certain nombre de requêtes en langage SQL pour extraire de la base de données des informations clés pour la sorcière.

Giselle veut mettre en place un système de proposition de recettes. Les clients pourront proposer de nouvelles recettes et Giselle choisira si elle décide de les intégrer ou non dans son catalogue.

Le dernier point important pour la base de données de Giselle est que celle-ci doit être sauvegardée. Ainsi, elle pourra être facilement restaurer en cas d’éventuel problèmes ou de migration.

## L’équipe projet

L’équipe chargée de réaliser le projet de Giselle est composée de quatre membres :

* Florian FRITSH
* Aurélien KLEIN
* Pierre GEERAERT
* Alexandre LECOMTE (chef de projet)

Dans la suite de ce rapport, « nous » désignera l’équipe du projet présentée ci-dessus.

Nous disposons de 7 jours ouvrés pour une équipe de quatre personnes soit 28 jours homme pour réaliser une base de données fonctionnelle et répondant aux besoins du futur site web de la sorcière Giselle.

# Planification du projet

## Découpage du projet : le WBS

Afin de se répartir correctement les tâches et avoir une vision globale du travail pour réaliser le projet nous avons décidé de définir un Work Breakdown Structure (WBS).

Le WBS est définit par un « découpage hiérarchique en livrables spécifiques des travaux à exécuter ». On trouve plusieurs niveaux dans un WBS, chaque niveau détaille un peu plus les tâches à réaliser jusqu’à arriver au niveau final. L’idée est d’obtenir un diagramme facilement compréhensible reprenant tout ce que l’équipe du projet doit réaliser.

Le dernier niveau doit correspondre à une tâche que l’on peut attribuer à une ou plusieurs personnes de l’équipe.

Après avoir clarifié le contexte et défini le périmètre du projet dans la partie précédente, nous avons donc découpé le travail de la manière représenté par la figure 1 de la page suivante.

## Planning prévisionnel

A partir de ce WBS nous avons établi une répartition des tâches (figure 2) ainsi qu’une estimation de la durée de celles-ci.

A partir du tableau de répartition des tâches nous avons construit un diagramme de Gantt correspondant (figure 3).

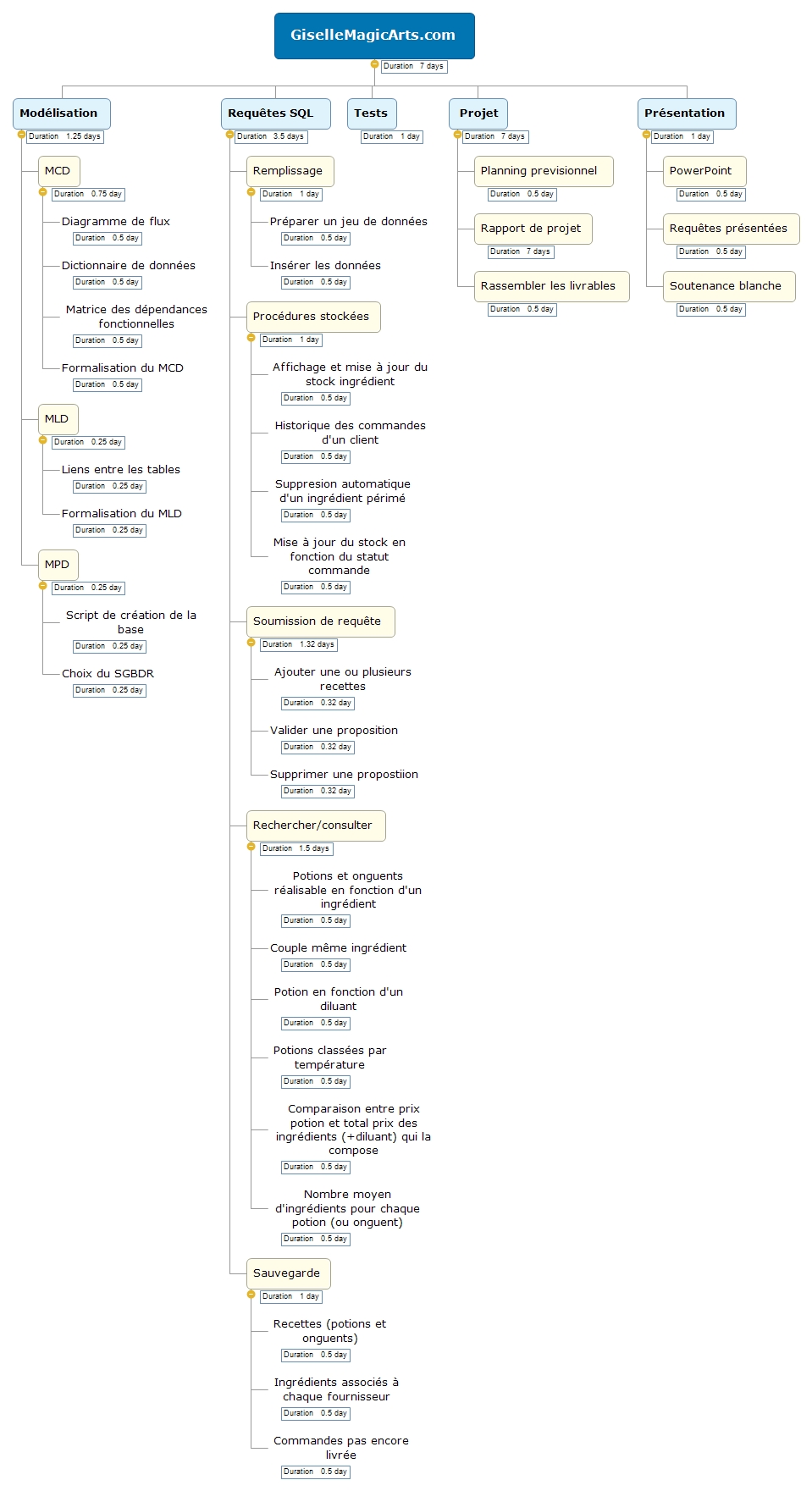


Figure 1 - WBS du projet



Figure 2 - Répartition des tâches

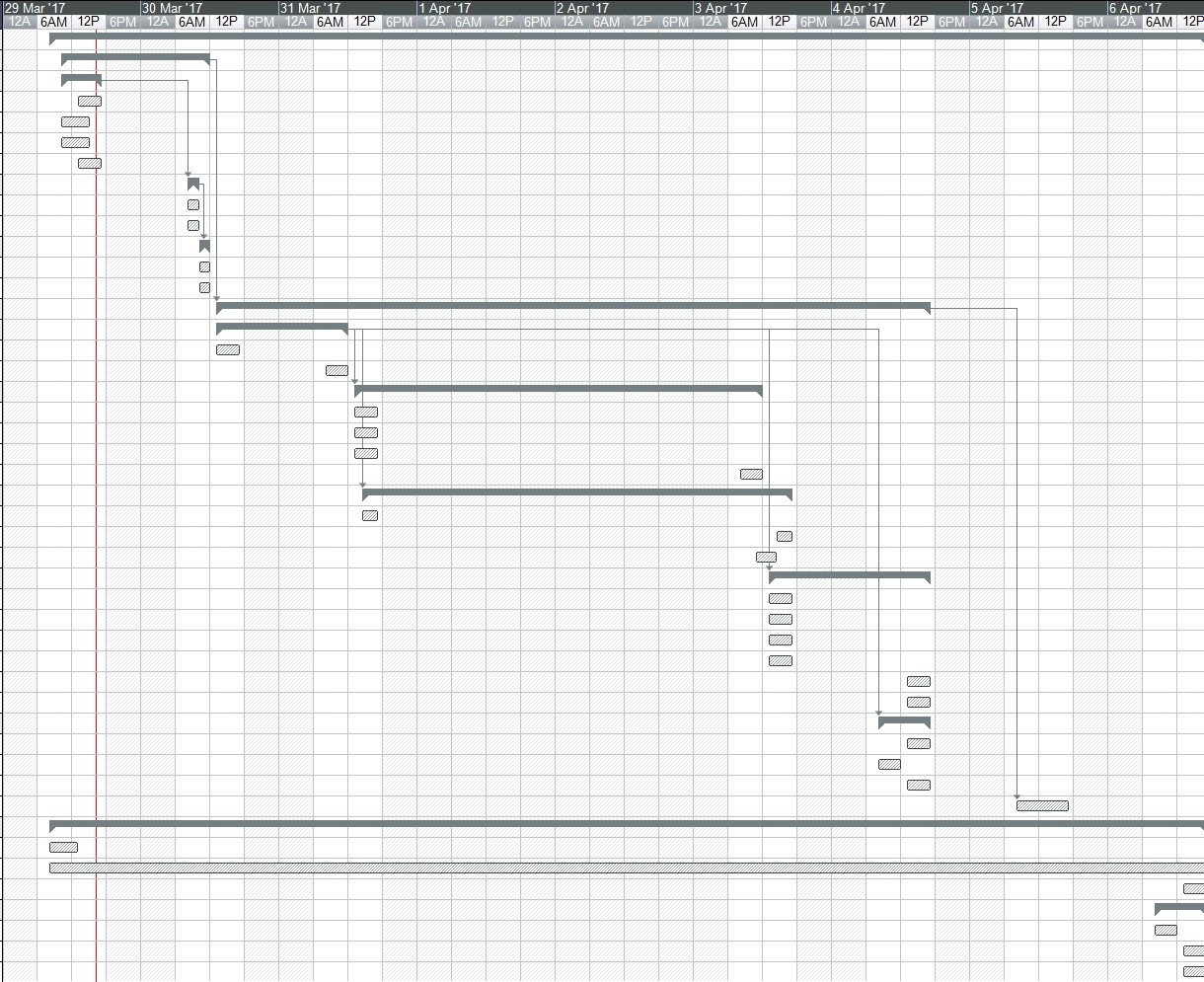


Figure 3 - Diagramme de Gantt

## Planning effectif

# Mise en place de la solution proposée

## Modélisation : la méthode MERISE

### Modèle conceptuel de données (MCD)

Pour pouvoir construire le modèle conceptuel de données on procède en deux étapes :

* La création d’un dictionnaire de données (DD)
* La création d’une matrice des dépendances fonctionnelles (Matrice des DF)

Le dictionnaire de données ou recueil d'informations consiste à collecter les données. Nous avons donc répertorié l’ensemble des données que nous jugions nécessaire en analysant le sujet du projet.

Nous aboutissons au dictionnaire de données suivant :

Figure 4 - Dictionnaire de données (DD)

La deuxième phase de conception du MCD est l’analyse des dépendances. Les dépendances sont les liens qui lient les différentes données. Le document résultant de cette étape est un diagramme ou une matrice des dépendances.

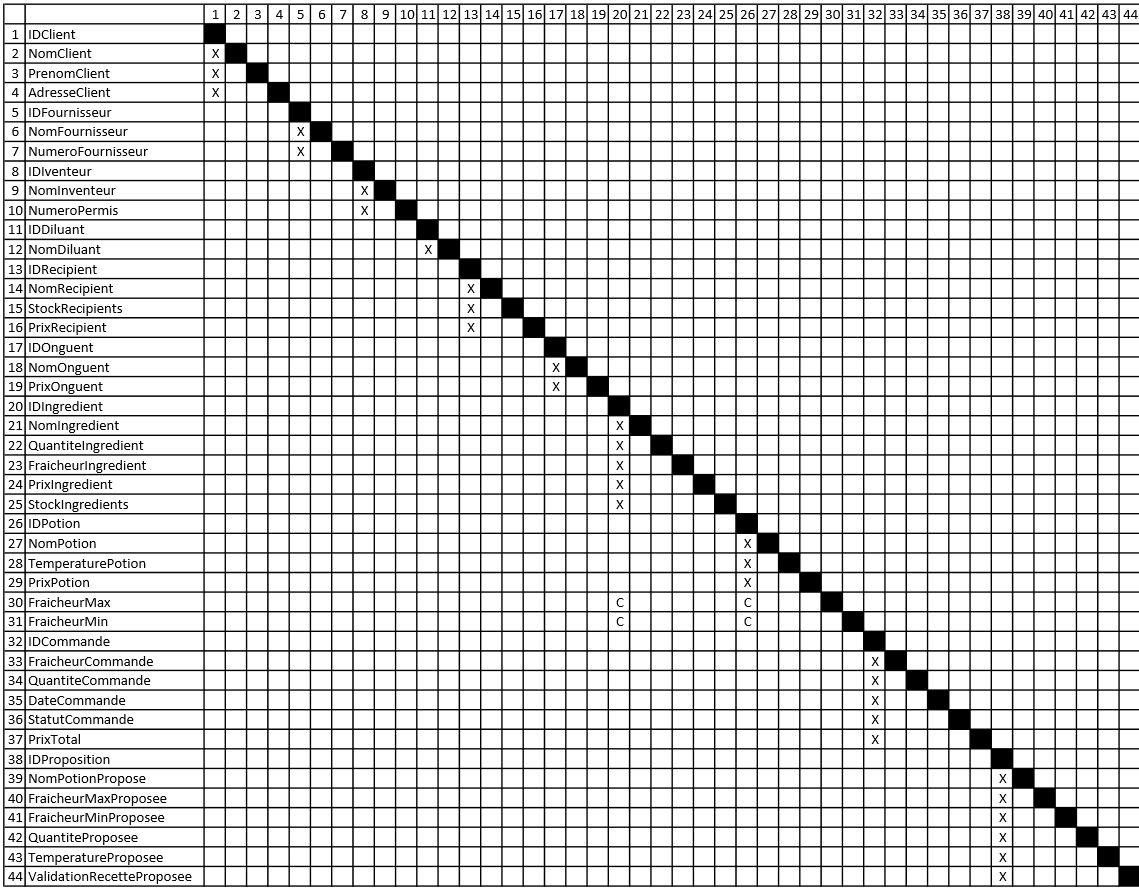
A partir du dictionnaire précédant on construit la matrice des dépendances fonctionnelles suivantes :

Figure 5 - Matrice des dépendances fonctionnelles

A l’aide de ces deux documents (DD et matrice des DF) on établit le modèle conceptuel de données sur le logiciel JMERISE.

On obtient le MCD suivant :

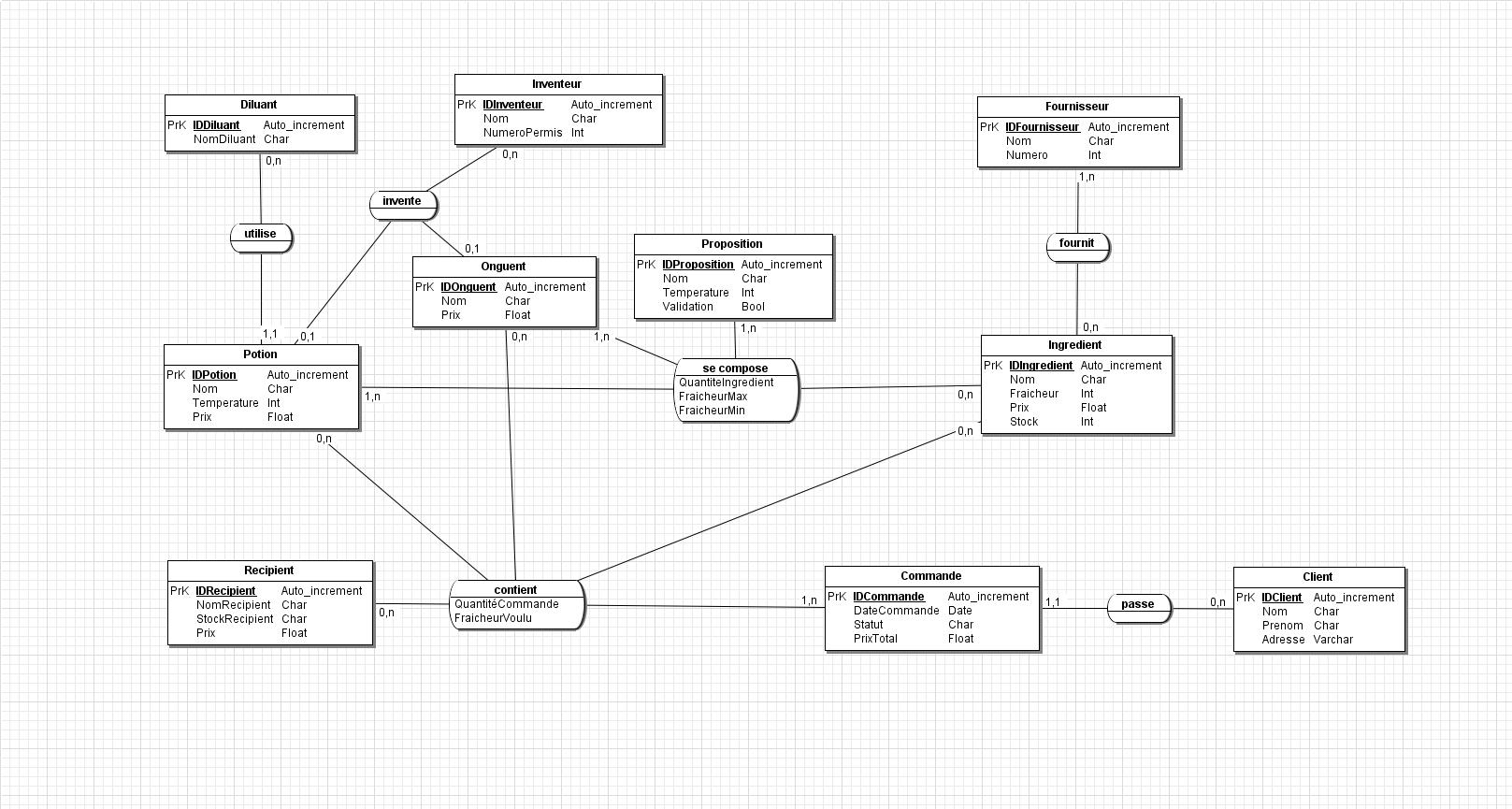


Figure 6 - Modèle Conceptuel de données

### Modèle logique de données (MLD)

Maintenant nous devons transformer le MCD en modèle logique de données (MLD). La quatrième étape de l’analyse des données est donc la traduction du MCD en MLD. Cette traduction s’effectue en suivant des règles précises.

Le MLD est toujours indépendant du SGBD utilisé. Il est composé uniquement de relations. Une relation est composée d’attributs et possède systématiquement une et une seule clé primaire qui permet d’identifier sans ambiguïté chaque occurrence de cette relation.

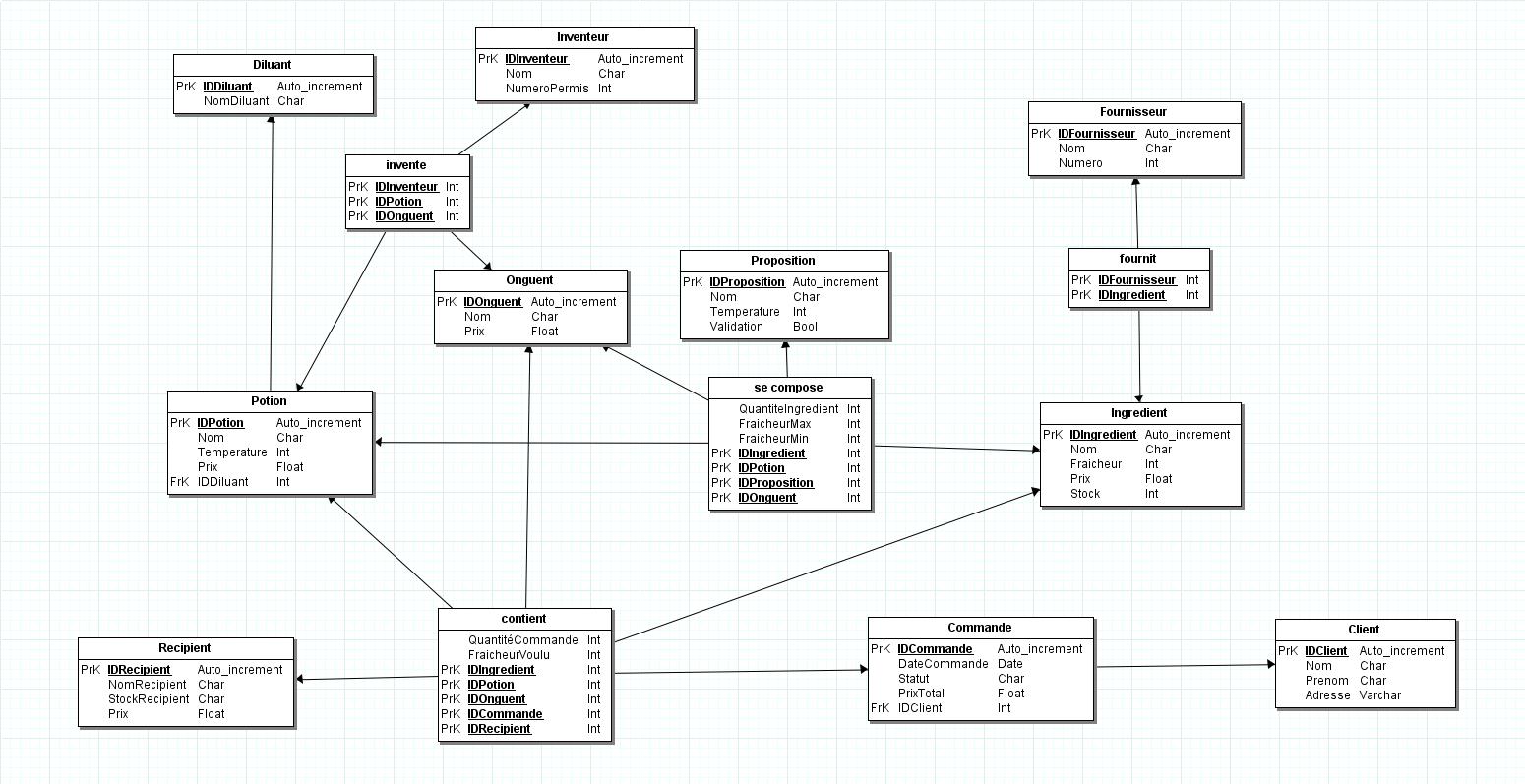


Figure - Modèle logique de données

### Choix du SGBDR

Nous avons choisi le Système de Gestion de Base de Données Relationnel (SGBDR) : MySQL. Comme la majeure partie des SGBDR, il utilise le langage SQL.

Dans un premier temps, nous avons été formés à l’école sur mysql. Cela nous a permis d’avoir plus d’aisance dans la création de la base de données sur ce SGBDR.

De plus mysql est gratuit et disponible sur plusieurs systèmes d’exploitation. Il est multi-utilisateur, cela nous permet donc de travailler tous ensemble sur la même base de données. Pour cela nous avons donc hébergé notre serveur MySQL sur un Raspberry Pi 3[[1]](#footnote-1).

MySQL dispose également de fonctionnalité de sécurité. Ce SGBDR contient un système de privilèges qui nous permet de gérer les différents accès sur ce serveur et il dispose d’un système de sauvegardes.

### Modèle physique de données (MPD)

Le modèle physique de données est l’implémentation du MLD par un logiciel. On va devoir traduire le MLD pour obtenir un MPD. On va notamment préciser le type et la taille (octets, bits) des données pour savoir comment les stocker.

La traduction d’un MLD relationnel en MPD passe par la création d’une base de données hébergée par un SGBDR (ici mysql).

Le MPD est donc le script de création de l’ensemble des tables et des relations entre elles en langage SQL (Structured Query Language).

Vous trouverez le fichier script\_creation\_bdd dans l’annexe 2.

## Les requêtes SQL

### Remplissage de la base de données

### Procédures stockées

### Soumission de recettes

### Rechercher et consulter

### Sauvegarder

## Tests

# ANNEXE 1 : Créer un serveur mysql sur un Raspberry Pi

Nous voulions que l’ensemble des membres de l’équipe du projet puisse se connecter et travailler sur la base de données de manière simultanée. Nous avons donc décidé d’installer un serveur mysql (version 5.7) sur un raspberry pi 3. Ce dernier est connecté au domicile d’un membre de l’équipe et accessible depuis l’extérieur.

A l’aide de l’adresse IP publique du routeur domestique et à la création de comptes utilisateurs sur la base de données mysql, les membres de l’équipe peuvent travailler sur la base de données en même temps.

Détaillons les étapes de déploiement de ce serveur sur un raspberry pi 3.

Tout d’abord nous conseillons de mettre à jour l’ensemble du système d’exploitation de l’ordinateur.

Capture d’écranEnsuite on utilise la commande suivante pour récupérer le paquet d’installation du serveur mysql version 5.7 (dernière version stable actuellement) :

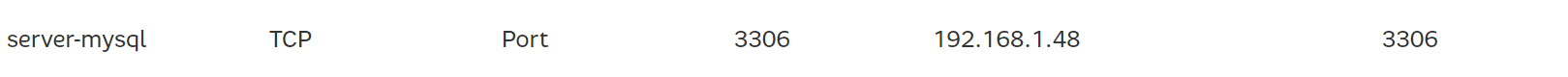
Capture d’écranAprès quelques minutes, le serveur mysql est en place, pour le démarrer on utilise la commande :

Capture d’écranOn peut maintenant se connecter en local au serveur mysql en utilisant la commande :

Capture d’écranUne fois connecté, on crée un compte utilisateur pour chaque membre de l’équipe du projet à l’aide de la clause GRANT.

Ici on attribue tous les privilèges à l’utilisateur florian qui peut se connecter depuis n’importe quelle adresse IP avec le mot de passe password.

Ensuite sur le routeur, on créer une règle de redirection du port 3306 (port mysql par défaut) arrivant de l’extérieur vers l’adresse IP local du Raspberry comme le montre la capture d’écran suivante.



Grâce à cette règle de redirection peut maintenant se connecter à distance au serveur mysql et donc à la base de données du projet. Pour l’administrer on pourra utiliser un terminal ou alors phpmyadmin.

L’installation de phpmyadmin requiert la mise en place d’un serveur Apache et la redirection du port 80.

# ANNEXE 2 : Script de création de la base de données

#------------------------------------------------------------

# Script MySQL.

#------------------------------------------------------------

#------------------------------------------------------------

# Table: Client

#------------------------------------------------------------

CREATE TABLE Client(

IDClient int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

Prenom Char (25) NOT NULL ,

Adresse Varchar (25) NOT NULL ,

PRIMARY KEY (IDClient )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Potion

#------------------------------------------------------------

CREATE TABLE Potion(

IDPotion int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

Temperature Int NOT NULL ,

Prix Float NOT NULL ,

IDDiluant Int NOT NULL ,

PRIMARY KEY (IDPotion )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Ingredient

#------------------------------------------------------------

CREATE TABLE Ingredient(

IDIngredient int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

Fraicheur Int ,

Prix Float NOT NULL ,

Stock Int NOT NULL ,

PRIMARY KEY (IDIngredient )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Onguent

#------------------------------------------------------------

CREATE TABLE Onguent(

IDOnguent int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

Prix Float NOT NULL ,

PRIMARY KEY (IDOnguent )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Diluant

#------------------------------------------------------------

CREATE TABLE Diluant(

IDDiluant int (11) Auto\_increment NOT NULL ,

NomDiluant Char (25) NOT NULL ,

PrixDiluant int(4) NOT NULL ,

PRIMARY KEY (IDDiluant )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Recipient

#------------------------------------------------------------

CREATE TABLE Recipient(

IDRecipient int (11) Auto\_increment NOT NULL ,

NomRecipient Char (25) NOT NULL ,

StockRecipient Char (25) NOT NULL ,

Prix Float NOT NULL ,

PRIMARY KEY (IDRecipient )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Commande

#------------------------------------------------------------

CREATE TABLE Commande(

IDCommande int (11) Auto\_increment NOT NULL ,

DateCommande Date NOT NULL ,

Statut Char (25) ,

PrixTotal Float NOT NULL ,

IDClient Int ,

PRIMARY KEY (IDCommande )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Fournisseur

#------------------------------------------------------------

CREATE TABLE Fournisseur(

IDFournisseur int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

Numero Int NOT NULL ,

PRIMARY KEY (IDFournisseur )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Inventeur

#------------------------------------------------------------

CREATE TABLE Inventeur(

IDInventeur int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

NumeroPermis Int NOT NULL ,

PRIMARY KEY (IDInventeur )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: Proposition

#------------------------------------------------------------

CREATE TABLE Proposition(

IDProposition int (11) Auto\_increment NOT NULL ,

Nom Char (25) NOT NULL ,

Temperature Int ,

Validation Bool ,

Diluant varchar(25) ,

PRIMARY KEY (IDProposition )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: se compose

#------------------------------------------------------------

CREATE TABLE se\_compose(

QuantiteIngredient Int NOT NULL ,

FraicheurMax Int ,

FraicheurMin Int ,

IDIngredient Int NOT NULL ,

IDPotion Int ,

IDProposition Int ,

IDOnguent Int ,

PRIMARY KEY (IDIngredient ,IDPotion ,IDProposition ,IDOnguent )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: contient

#------------------------------------------------------------

CREATE TABLE contient(

QuantiteCommande Int ,

FraicheurVoulu Int ,

IDIngredient Int NOT NULL ,

IDPotion Int NOT NULL ,

IDOnguent Int NOT NULL ,

IDCommande Int NOT NULL ,

IDRecipient Int NOT NULL ,

PRIMARY KEY (IDIngredient ,IDPotion ,IDOnguent ,IDCommande ,IDRecipient )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: fournit

#------------------------------------------------------------

CREATE TABLE fournit(

IDFournisseur Int NOT NULL ,

IDIngredient Int NOT NULL ,

PRIMARY KEY (IDFournisseur ,IDIngredient )

)ENGINE=InnoDB;

#------------------------------------------------------------

# Table: invente

#------------------------------------------------------------

CREATE TABLE invente(

IDInventeur Int NOT NULL ,

IDPotion Int NOT NULL ,

IDOnguent Int NOT NULL ,

PRIMARY KEY (IDInventeur ,IDPotion ,IDOnguent )

)ENGINE=InnoDB;

ALTER TABLE Potion ADD CONSTRAINT FK\_Potion\_IDDiluant FOREIGN KEY (IDDiluant) REFERENCES Diluant(IDDiluant);

ALTER TABLE Commande ADD CONSTRAINT FK\_Commande\_IDClient FOREIGN KEY (IDClient) REFERENCES Client(IDClient);

ALTER TABLE se\_compose ADD CONSTRAINT FK\_se\_compose\_IDIngredient FOREIGN KEY (IDIngredient) REFERENCES Ingredient(IDIngredient);

ALTER TABLE se\_compose ADD CONSTRAINT FK\_se\_compose\_IDPotion FOREIGN KEY (IDPotion) REFERENCES Potion(IDPotion);

ALTER TABLE se\_compose ADD CONSTRAINT FK\_se\_compose\_IDProposition FOREIGN KEY (IDProposition) REFERENCES Proposition(IDProposition);

ALTER TABLE se\_compose ADD CONSTRAINT FK\_se\_compose\_IDOnguent FOREIGN KEY (IDOnguent) REFERENCES Onguent(IDOnguent);

ALTER TABLE contient ADD CONSTRAINT FK\_contient\_IDIngredient FOREIGN KEY (IDIngredient) REFERENCES Ingredient(IDIngredient);

ALTER TABLE contient ADD CONSTRAINT FK\_contient\_IDPotion FOREIGN KEY (IDPotion) REFERENCES Potion(IDPotion);

ALTER TABLE contient ADD CONSTRAINT FK\_contient\_IDOnguent FOREIGN KEY (IDOnguent) REFERENCES Onguent(IDOnguent);

ALTER TABLE contient ADD CONSTRAINT FK\_contient\_IDCommande FOREIGN KEY (IDCommande) REFERENCES Commande(IDCommande);

ALTER TABLE contient ADD CONSTRAINT FK\_contient\_IDRecipient FOREIGN KEY (IDRecipient) REFERENCES Recipient(IDRecipient);

ALTER TABLE fournit ADD CONSTRAINT FK\_fournit\_IDFournisseur FOREIGN KEY (IDFournisseur) REFERENCES Fournisseur(IDFournisseur);

ALTER TABLE fournit ADD CONSTRAINT FK\_fournit\_IDIngredient FOREIGN KEY (IDIngredient) REFERENCES Ingredient(IDIngredient);

ALTER TABLE invente ADD CONSTRAINT FK\_invente\_IDInventeur FOREIGN KEY (IDInventeur) REFERENCES Inventeur(IDInventeur);

ALTER TABLE invente ADD CONSTRAINT FK\_invente\_IDPotion FOREIGN KEY (IDPotion) REFERENCES Potion(IDPotion);

ALTER TABLE invente ADD CONSTRAINT FK\_invente\_IDOnguent FOREIGN KEY (IDOnguent) REFERENCES Onguent(IDOnguent);

1. Pour plus sur la création du serveur Mysql sur un Raspberry Pi 3 se repporter à l’annexe 1 [↑](#footnote-ref-1)