Text

Description automatically generated A picture containing text, clipart

Description automatically generated

*Tutoriel :*

Source :

« br-automation.com »

/ Help

/ BR Tutorial



**« Etude DB »**

**/**

**Système AFAG**

MS 2022/07

**Table des matières :**

[I. Généralité sur le système AFAG 3](#_Toc108105610)

[1. Présentation 3](#_Toc108105611)

[1) Synoptique 3](#_Toc108105612)

[II. BDD 3](#_Toc108105613)

[1. Etude BDD 3](#_Toc108105614)

[1) Expression du besoin 3](#_Toc108105615)

[2) Scénarii 4](#_Toc108105616)

[3) Modèle Entité Association de la base « AFAG » 4](#_Toc108105617)

[4) Diagramme E/A du système AFAG 5](#_Toc108105618)

[5) Établir le modèle relationnel de la base 6](#_Toc108105619)

[6) Optimiser le modèle 7](#_Toc108105620)

[7) Diagramme Relationnel des tables de la base de données AFAG 7](#_Toc108105621)

[8) Création des tables avec l’outil Workbench 7](#_Toc108105622)

[9) Importation de la base vers notre server WAMP 9](#_Toc108105623)

[10) WAMPserver 9](#_Toc108105624)

[11) Paramétrer votre WAMP de la base de données 10](#_Toc108105625)

[2. MappBDD 12](#_Toc108105626)

# Généralité sur le système AFAG

## Présentation

### Synoptique

**Interface WEB :**

**API CLIENT** : WEB

⮱ HTML/php web pages

**Machine HMI**

PP500

Graphical user interface, application

Description automatically generatedA picture containing text, electronics, computer, sitting

Description automatically generated

**FrontEnd**



http

**BackEnd**

A picture containing yellow, orange, bright, honey

Description automatically generated

**B&R controller**

**API SERVER** : Traite les articles de la BDD

**⮱ mappDB** :

* Demande Recette

à la BDD pour les traiter

* liaison avec BDD

(connect DB with python)

**⮱ http** : AShttpService = **not use here** (liaison avec l’API client)

**Lamp Server** (apache + MariDB)

* **Database** to save products

# BDD

## Etude BDD

### Expression du besoin

L’opérateur, par le biais d’une IHM, sélectionne une recette conditionnant un **format de récipient**, entrainant un placement automatique du robot 5 axes associé au lecteur de codes à barres pour une lecture optimisée des informations.

Des données sont traitées par le serveur de la BDD pour **renseigner le responsable** via une interface WEB, **sur le nombre des commandes en attentes, en cours, terminées et des statistiques.**

Des données sont traitées aussi par le serveur de la CPU B&R pour renseigner l’opérateur via une IHM sur l’état d’avancement, le contrôle de l’exactitude des commandes en cours et des alarmes avertissant l’opérateur de la fin d’une recette ou d’une erreur.

L'administration et l'enregistrement des recettes se feront par le biais d'une interface WEB, liée à une base de données, que contrôlera le responsable technique du site.

Les opérateurs auront la possibilité de piloter le système via une IHM tactile: Lancement, et arrêt d’une recette.

Les données du projet seront stockées sur un serveur (base de données).

Le paramétrage et le contrôle du système d’identification sur chaine d’embouteillage automatique se fera par le biais d’une station de travail type PC que contrôleront le responsable technique à l'aide d'une interface Web ou 1 pupitre tactile.

En résumé, la BDD devra **Enregistrer les Entrées/Sorties des commandes clients suivant une recette** établie par unresponsable et exécuter par des opérateurs avec les attentes suivantes :

* Obtenir la liste des commandes passées
* Obtenir le nombre de commandes effectuées, en cours ou en attente
* Obtenir le nombre de récipients passées
* Obtenir une liste Historique des commandes (Date)
* Utilisation de l’interface WEB pour remplir le magasin avec des commandes associé à un code-barres et un format de récipient (hauteur)
* Lecture code-barres des récipients pour des informations produits et leurs comptages.
* Une recette contient seulement un type de produit

### Scénarii

**Personnes et actions à chacun :**

* **Responsable** => **Interface WEB**

⮱ Ajouter ou supprimer **une commande** via l’interface **web** (front end => back end => base de données)

* **L’opérateur machine** => **IHM machine**

⮱ Ne fait que faire ce que lui dit le système : mise en énergie / lancement ou arrêt d’une recette / supervision (**pas d’action sur la DB**)

Donc si on résume :

**Responsable**

Entrée des recettes en fonction des commandes reçues

Sortie des recettes en fonction des commandes reçues

Generation historique

TypeUtilisateur :

Type : 0-responsable (seulement)

**Opérateur** : PAS D’ACTION SUR LA DB

**Actions sur le système :**

* Action :

Type : 0-entrée (lancer) / 1-historique (des recettes)

Statut : 0-non traitée / 1-en cours / 2-finie

Position lecteur : 0-haut / 1-bas (en fonction des récipients)

* historique (date) :

Numéro EAN13 : renseignement produit

Nombre de recettes

Nombre de récipients

### Modèle Entité Association de la base « AFAG »

A partir de l’expression du besoin présentée précédemment, proposez un modèle entité association pour la future base de données.

**Se poser la question : Que voulons-nous garder en mémoires ?**

Icon

Description automatically generated(ex : « user » si nous voulons distinguer les commanditaires par exemple)

**Pas de calcul induit dans la BDD** (ex : calcul du nombre de caisse)

* Repérez les **entités** évidentes : **recipient : définition article(s)** *(référence, Ht (hauteur), description,…)*

*rq : on désigne les entités au singulier*

**user :** pas d’attribut particulier ⇨ entitée pas nécessaire **(responsable seulement et pas d’identification)**

**recette**  **enregistrer les produits d’une commande**

* Indiquez le(s) **association**(s) : **préparer : user/recette** : entrée dans la bdd

**contenir** : recipient/recette

*rq : on désigne les associations avec un verbe (conjugué)*

* Complétez vos entités / associations par des attributs
* Ajoutez un attribut identifiant à vos entités : id (si ce n’est déjà fait)

|  |  |  |  |
| --- | --- | --- | --- |
| **Entités** | **user** | **recette** | **recipient** |
| **Attributs** | Id  username  password  *nom*  *prenom*  *avatar*  *role*  *commentaire* | Id  Ref\_recette  date\_recette  *commentaire* | Id  Ref\_recipient  Ht\_recipient  Description -> codebarres |
| **Associations** | **préparer** | **contenir** |  |
| **Attributs** | date | quantité |  |

### Diagramme E/A du système AFAG

Graphical user interface, application, Teams

Description automatically generated

*(schéma réalisé avec l’outil « drawio »)*

* Puis les rôles : cardinalités :

***Le rôle de l’association est défini par deux nombres (min..max) représentant le nombre de fois minimum et le nombre de fois maximum qu’une entité participe à une association.***

*La question des cardinalités se pose dans les 2 sens (de A vers B et de B vers A) afin d’appliquer les rôles de chaque côté de l’association.*

**PREPARER user / recette :** date

« **user**» vers « recette » : (*cardinalité du coté de user)*

***user peut préparer 1 à n recette***

« **recette**» vers « user » : (*cardinalité du coté de recette)*

***Recette peut être préparée par 1 à 1 user*** (responsable)

**CONTENIR recipient / recette :**quantité

« **recipient** » vers « recette » : (*cardinalité du coté de recipient)*

***recipient peut être contenu dans 0 à n recette***

« **recette** »  vers « recipient » : : (*cardinalité du coté de recette)*

***Recette peut contenir 1 à 1 recipient*** (Une recette contient seulement un type de produit)

### Établir le modèle relationnel de la base

A partir du modèle obtenu, nous allons établir un schéma relationnel qui nous permettra de déterminer les tables de notre base de données.

Modèle E/A -> **Modèle relationnel** : (Relations = Tables)

⮱ **Chaque entité et association deviennent des tables (relations)**: ici, 3 entités et 2 associations soit **5** tables

⮱ **Chaque table est alors définie par des attributs avec une clé primaire et 0 ou n clé(s) étrangère(s).**

Pour les entités : La clé de cette relation est l’identifiant de la classe d’entités correspondante -> clé primaire « id »

Table « **user** » avec les attributs : id(clé primaire), username, mdp, nom, prénom, avatar, comment…

Table « **recette** » avec les attributs : id(clé primaire), ref\_recette, date\_recette

Table « **recipient »** avec les attributs : id(clé primaire), ref\_recipient, ht\_recipient, Codebarres…

Pour les associations : La clé de cette relation est composée d’une clé primaire et de tous les identifiants des entités participantes

⮱clé(s) étrangère(s) : « nomentite\_id »

Table « **preparation**» avec les attributs : id(clé primaire), user\_id(clé étrangère), recette\_id(clé étrangère), date

Table « **contenu** » avec les attributs : id(clé primaire), recipient\_id(clé étrangère), recette\_id(clé étrangère), qte

Rq : la désignation des associations passe du verbe au nom

|  |  |  |  |
| --- | --- | --- | --- |
| **Tables** | **user** | **recette** | **recipient** |
| **Attributs** | Id  username  password  nom  prenom  avatar  role  commentaire | Id  Ref\_recette  date\_recette | Id  Ref\_recipient  ht\_recipient  Codebarres (description) |
| **Tables** | **Préparer**  **⮱ preparation** | **Contenir**  **⮱contenu** |  |
| **Attributs** | Id  user\_id  recette\_id  date | Id  recipient\_id  recette\_id  quantité |  |

### Optimiser le modèle

**Avec les cardinalités 0,1 ou 1,1 toute classe d’associations, reliée à une classe d’entités peut être fusionnées** **avec la classe d’entité** soit ici :

⮱ éliminer la classe d’association**, « preparation » et « contenu\_recette »**

⮱ déplacer le(s) attribut(s) des classes d’associations éliminées vers les classe d’entité adjacentes du côté du lien supprimé avec les cardinalités 0,1 ou 1,1

Résultat **3** tables : **user, recette, recipient** avec les attributs suivants :

|  |  |  |
| --- | --- | --- |
| **user** | **recette** | **recipient** |
| Id  username  password  nom  prenom  avatar  role | Id  Ref\_recette  nb\_recipient (quantité)  date\_recette  user\_id  recipient\_id | Id  Ref\_recipient  ht\_recipient  **Codebarres** (description) |

! les verbes d’actions des associations deviennent des noms pour les entitées : contenir -> contenu

! dans la table recette, mettre les attributs recette\_id, une seule fois car l’attribut recette\_id est inutile dans la table recette avec l’attribut Id (sous-entendu : Id\_recette)

! Vérifier que chaque table à **1** clé primaire et de 0 à n clé(s) étrangère(s)

### Diagramme Relationnel des tables de la base de données AFAG

Graphical user interface, application

Description automatically generated

*(schéma réalisé avec l’outil « workbench »)*

### Création des tables avec l’outil Workbench

Lancer “Workbench” : (tuto TH)

**Créer les tables sous workbench :**

**! Eviter les majuscules, les accents, les caractères spéciaux.**

* File / New model / Double clic sur mydb et rentrer le nom de votre nouvelle base de données ou renommer la base de données existante « mydb »: ici nous l’appellerons **« afagdb**»
* Enregistrer votre modèle :

File / Save Model As / nous l’appellerons « **AfagDB** » enregistrée sous C :Documents / BDD

* / doubles cliquez sur "**Add Diagram**"
* Pour placer une table, maintenez la touche T de votre clavier enfoncée puis faites un clic gauche et relâcher la touche T.

Doubles cliquez sur votre nouvelle table.

En dessous de votre diagramme, vous avez les propriétés de votre table, ajoutez les entités.

Graphical user interface

Description automatically generated

ClicG sur l’onglet Columns :

Dans le champ :

* "Table name" : le nom de votre table
* "Colums Name ": le nom de votre entité
* "Datatype" : le type de champ (INT, VARCHAR, TEXT, DATETIME, etc...)
* "PK" : Primary Key
* "NN" : Not Null (**coché par défaut**)
* "AI" : Auto Increment

**…**

**! Pour les clés primaires cochez 🗹 PK, 🗹 NN et 🗹 AI**

**! Nature des attributs**

Graphical user interface

Description automatically generated

**Pour relier les tables, utiliser l’onglet Foreign Keys :**

⮱ Sélectionner la table avec des clés étrangères en double cliquant dessus ici avec **la table “commande »**

/ Effectuer un clic gauche sur l’onglet **Foreign Keys**

**/** nommer la clé étrangère**: fk\_cde\_client**

/ table cible**: ’tstock21db’ , ‘client’**

**/** table origine**: cocher 🗹 client\_id**

**/** table cible : **id**



Graphical user interface, text, application

Description automatically generated

**! Pour les clés étrangères cochez 🗹 NN seulement !**

Avec **🞏 UN** décoché sinon

(NB : il faut qu’entre les clés nous ayons des champs identiques :

INT / INT ou INT UN / INT UN)

### Importation de la base vers notre server WAMP

Text

Description automatically generated

**! Vérifier que votre serveur** (WampServer ici) **est démarré** (vous devez avoir le **W** vert de *Wamp* dans la zone de notifications Windows**\***).

⮱ Database / **Forward Engineer** / next / next / next…

! si vous êtes sur votre PC, vérifier : localhost / 127.0.0.1 / port 3306 (srv MariaDB)

**! errors**

***MySQL error 1064 :***

*NB la visibilité des index est apparu avec MySQL 8 et non utilisée ??? !!! ...*

* *Je Retire simplement le paramètre VISIBLE dans mon SQL : Ca fonctionne :*

Graphical user interface, text

Description automatically generated

* **Ou** **vérifier la verion mySQL sous phpMyAdmin ici 5.7.3 et non 8.0, effectuer la modif /version->5.7.3**

Graphical user interface, application, Word

Description automatically generated***! successful***

### WAMPserver

Vérifier la présence des tables de votre base AFAG :

* Si http://localhost/phpmyadmin/ NE REPOND PAS ⇨ <http://localhost:90/phpmyadmin/>

(car le port a été changé en 90 au lieu de 80 par défaut)

Graphical user interface, text, application, email

Description automatically generated

Sinon : Paramétrer votre WAMPserver

⮱ Création d’un utilisateur « afaguser »

Il n'est pas très conseillé de travailler en tant que "root" dans MySQL, à moins d'en avoir spécifiquement besoin. En effet, l’utilisateur "root" a tous les droits. Pour éviter cela, il est judicieux de créer un utilisateur pour une base de données avec des droits restreints.

Avant d’aller plus loin, assurez-vous d’avoir démarré WampServer (vous devez avoir le **W** vert de Wamp dans la zone de notifications Windows**\***).

**\*** Si votre icône est rouge ou orange, votre serveur n’est pas correctement démarré. Il faut alors revoir la configuration.

⮱ Lancer « WAMPserveur » : voit tuto « Install Wampserver »

! vérifier la compatibilité MariaDB / MySQL, ici version mariaDB 10.4.10 et MySQL 5.7 sous workbench pour compatibilité)

**! Vérifier les Ports utilisés par : MySQL : 3308**

**MariaDB : 3306**

**! si pb faire une mise à jour de wampserver (important) :**

1. **Supprimer la BDD dans phpmyadmin et effectuer la mise à jour**
2. **ex le 26/06/2020, update 3.2.3.0**

Graphical user interface, text, application

Description automatically generated

1. Lancez PhpMyAdmin

⮱ clicGauche sur le **W** vert présenté ci-dessus > **PhpMyAdmin**

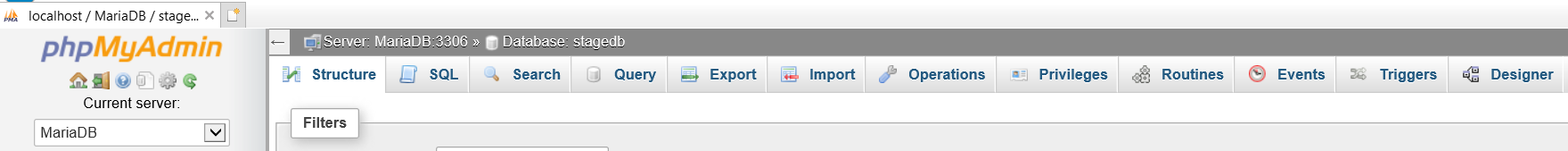
Graphical user interface, text, application, chat or text message

Description automatically generated

Log : Root

Password : (Vide)

1. Sélectionner la base de données : **«** **afagdb »**
2. Sous PhpMyAdmin, à l’accueil de l’interface, cliquez sur le menu « **Privilèges**»



1. Graphical user interface, text

   Description automatically generatedCliquez ensuite sur « **Ajouter un compte d’utilisateur** »
2. Dans la page d’ajout d’un utilisateur, renseignez les paramètres comme suit :

* Graphical user interface, application

  Description automatically generated**Nom d’utilisateur :** *afaguser* (donnons-lui le nom de notre base de données, car il va créer une base au nom de l’utilisateur)
* **Client :** « Local » (nous n’avons pas besoin d’accéder à cette base de données de l’extérieur du PC)
* **Mot de passe :** afag22

! ne rien cocher **si une base à déjà été créée**

Graphical user interface, text, application, email

Description automatically generatedSinon, cochez « Créer une base portant son nom et donner à cet utilisateur tous les privilèges sur cette base. »

Vous constaterez l’apparition de la base de données «afaguser » dans la liste des bases existantes (à gauche) sur votre serveur. (inutile si nous avons déjà une autre base)

! ne rien cocher

**Privilèges globaux :** cliquez sur « **Tout cocher** » afin de donner à cet utilisateur tous les droits sur la base transtocker.

Graphical user interface

Description automatically generated



1. Validez la création de l’utilisateur en cliquant sur « Exécuter » ou « GO »

Quels sont les droits de l’utilisateur « afaguser » ?

Il aura tous les droits sur la base « **afagdb**» (modification de la base, ajout de tables, ajout / récupération d’enregistrements, …), mais il ne verra pas les autres bases du serveur et n’aura aucun droit dessus.

## SQL

### Remplir les tables avec les commandes SQL :

**Help :** tuto D:\VeilleTechno\DocSupport\DocSupportBDD\RapportBDD\_2021\ **BDDlesson\_SQL\_200626**

**Enregistrement de données dans une table :**

Table ‘recipient’ avec

Mustard / ht=120mm / cb=3250546095238

Jam / ht=70mm / cb=3564706683938

Table ‘recette’ avec

AMORA / 20 / 08-07-2022

ANDROS / 10 / 07-07-2022

Table ‘user’ avec :

Responsable /staff1 /pwd1

**Commandes SQL :**

INSERT INTO user (username, password) VALUES ('staff1', 'pwd1')

ok

INSERT INTO recipient (ref\_recipient, ht\_recipient, **codebarres**) VALUES ('mustard', 12.0, **3250546095238**)

* codebarres en **BIGINT** car INT ne retient pas suffisamment de digits)

ok

INSERT INTO recette (ref\_recette, nb\_recipient, date\_recette, user\_id, recipient\_id) VALUES ('amora1', 20,'2022-07-08 13:45:10', ?, ?)

Nok :

User\_id = ?

Recipient\_id =  ?

## MappBDD

Voir Tutoriel :