**Dossier de projet pour le titre développeur web - web mobile**

SOMMAIRE

I Développer la partie frontend d’une application web ou web mobile en intégrant les recommandations de sécurité

1 Maquetter une application

2 réaliser une interface utilisateur web statique et adaptable

3 Développer une interface utilisateur web dynamique

4 Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce

II Développer la partie backend d’une application web ou web mobile en intégrant les recommandations de sécurité

5 Créer une base de données

6 Développer les composants d’accès aux données

7 développer la partie backend d’une application web ou web mobile

8 Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce

INTRODUCTION

I Développer la partie frontend d’une application web ou web mobile en intégrant les recommandations de sécurité

1 Maquetter une application

2 réaliser une interface utilisateur web statique et adaptable

3 Développer une interface utilisateur web dynamique

4 Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce

1 Maquetter une application

Cahier des charges

Le cahier des charges est le support écrit entre l’équipe de développement et le client, c’est le lien entre eux. C’est un document signé par les deux parties et qui contractualise le projet ses spécificités, son coût, les délais, etc...

Il est organisé en plusieurs parties (le plan détaillé en annexe), il est présenté ci-après de manière succincte.

Le projet est celui d’un blog de cuisine adossé à un site de réservation de repas pour un foodtruck.

1) Projet et objectifs

La société n’est pas encore créée et il n’y a pas d’identité actuellement. Elle n’a ni marque ni identité à l’heure actuelle.

Les fondateurs par goût pour la cuisine et également pour satisfaire la curiosité des tous ont décidé de créer un foodtruck basé sur une alimentation essentiellement végétale et mettant en avant le pois chiche.

Le secteur d’activité est celui de la restauration à emporter.

Le projet s’articule sur la création d’un blog de cuisine associé à un foodtruck qui distribuerai les plats issus des recettes du blog pour de la vente à emporter avec un système de réservation des plats.

2) Etude du besoin

Le besoin du client est la mise en place d’un site web pour la partie blog et d’une possibilité de réservation de produits en ligne. Il n’y a actuellement aucun site existant.

3) Concurrence

a) contexte et enjeux

La concurrence est très importante pour les blogs de cuisine mais pour les foodtrucks cela s’avère être moins sévère. Le marché de l’alimentation végétale est en plein essor. Les propositions des foodtrucks d’alimentation saine et de qualité autour de la métropole toulousaine sont actuellement peu nombreuse. Le secteur géographique visé est celui des zones industrielles de la périphérie toulousaine. Le service traiteur du foodtruck est tout d’abord envisagé pour le service du midi.

b) objectifs marketing

L’objectif est de pouvoir offrir une visibilité au foodtruck par la tenue du blog. Une analyse de la concurrence a été faite tant au niveau marketing que technique. Elles sont abordées ultérieurement dans le document.

c) positionnement du futur produit

Le marché est celui des recettes alimentaires et plus particulièrement la restauration du midi pour le foodtruck localement à Toulouse.

d) analyse SWOT

Force : concept original

Faiblesses : peu de moyens mis à disposition

Opportunités : pouvoir s’installer sur un marché non encore saturé de l’alimentation végétale du midi

Menaces : non décollage de l’activité et cessation de l’activité et être noyé dans la masse des blogs de cuisine

B) Cible

1) public

Pour le blog : les personnes à sensibilité végétarienne entre 18 et 65 ans.

Pour le foodtruck : les personnes travaillant dans les zones de desserte.

2) équipements

Le site devra être géré de façon simple pour que le restaurateur puisse le mettre à jour facilement et rapidement en fonction des menus du jour.

C) Typologie d’application

L’application est une application web s’adaptant aux écrans des tablettes et smartphones. Il n’y a pas d’application mobile de prévue.

D) Périmètre

Le contrat est l’établissement du blog et de la plateforme de réservation de repas. Le contenu est fourni par le client ainsi que les images des plats et recettes.

Graphisme et ergonomie

Cette partie est encore à définir avec le client qui doit apporter l’identité graphique et le logo.

A) Graphisme

L’identité graphique et le logo sont à créer.

B) Maquettage - Arborescence

Le maquettage sera effectué avec https://wireframe.cc/ et le logiciel draw.io.

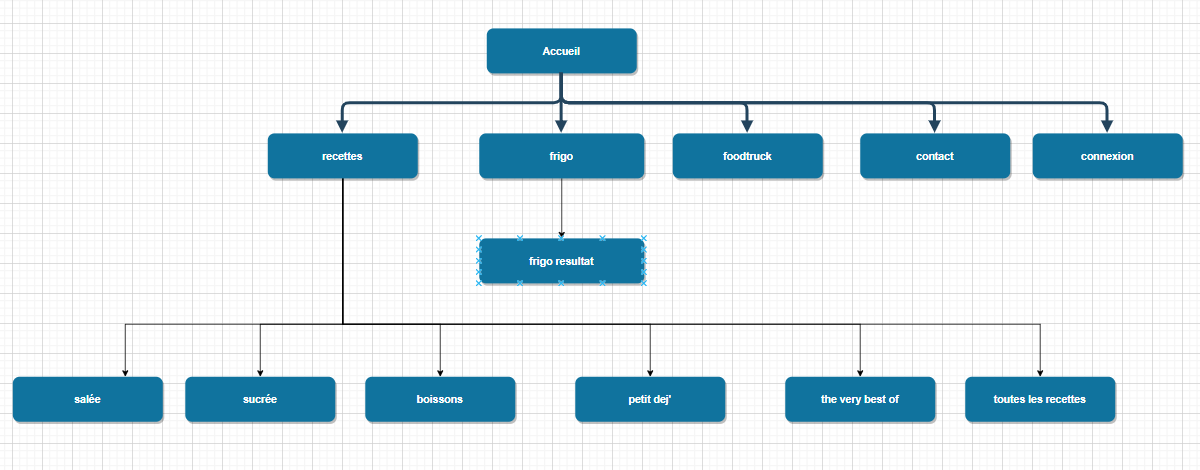
Le maquettage permet en collaboration avec le client d’établir le visuel du site qu’il souhaite. C’est une représentation graphique non définitive du projet. Cela a pour but de cibler les attentes du client en matière de rendu utilisateur.

L'organisation avec le client a été faite dans une volonté de gestion de projet en mode agile.

La gestion de projet agile est une manière de gérer un projet de façon dynamique avec une volonté de cycle de validation continue avec le client. L'objectif est de présenter au client des livrables fonctionnalités par fonctionnalités contrairement à la gestion de projet classique ou la livraison intervient à la fin du cycle. Concernant le maquettage, la méthode a été appliquée à chaque étape ce qui a permis d’adapter à chaque fois la vision du client au projet.

Les étapes du maquettage sont au nombre de quatre. La première est l’arborescence qui permet d’organiser la structure du site. Les trois autres sont des représentations visuelles de plus en plus réaliste de la version du site.

Ci après la structure du site avec son arborescence.



A travers cette arborescence on peut voir que le seuil des trois clics pour accéder à l’information est respecté.

Le site s’ouvre sur une page d’accueil qui permet d’accéder via un menu déroulant et des liens à plusieurs pages.

Le menu déroulant concerne les recettes classées par catégorie et un lien permet d’accéder à l’ensemble des recettes.

Les liens directs permettent d’accéder au formulaire de contact et à la connexion au compte utilisateur et à la page de recherche de recettes par ingrédient appelée ”Frigo”.

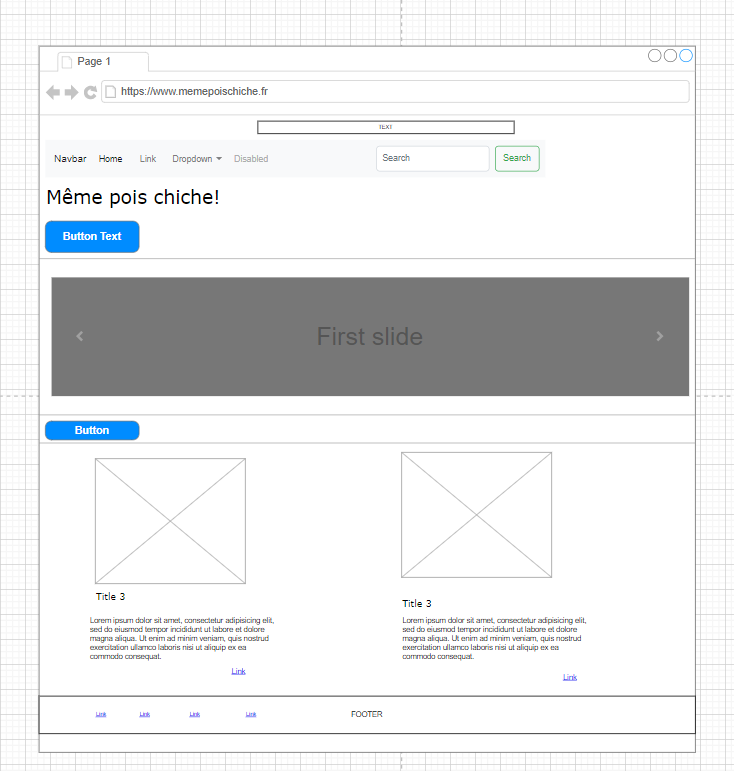
La page frigo comporte deux versions : la première est celle ou l’utilisateur choisi les ingrédients et la seconde montre le résultat de la recherche.

L’étape du zoning qui permet de représenter l’organisation générale de la page en bloc a été pensée comme il suit :



En échangeant avec le client à la suite il en ressort que ce dernier souhaite un visuel très épuré. L’ajout excessif de photos et d’éléments dynamique est donc réduit. Le logiciel utilisé est draw.io pour les visuels qui suivent.

Le wireframe qui en ressort est la suite du zoning et permet de visualiser l’emplacement des différentes zones de textes d’images etc :



Cette représentation est plus réaliste et permet de visualiser de manière plus concrète l’organisation de la page.

Ci-dessus la page d’accueil et les pages concernant le frigo et les pages de connexion et de liste de recettes sont en annexes. (Annexe Maquettage).

La dernière phase du maquettage est le mockup c’est à dire la représentation visuelle du site dont un exemple est ci après. Il a été réalisé via le site WordPress.



**Spécificités et livrables**

Une analyse technique du secteur a été effectuée autour des blogs de cuisine et les foodtrucks.

Il en ressort que les sites des foodtrucks sont très simples et avec peu de contenus.

Les blogs sont eux quasiment tous présenté de la même manière. WordPress est la présentation la plus répandue.

Un échantillon de 3 blogs de cuisine et de 5 sites de foodtrucks ou restaurants végétarien locaux a composé l’analyse.

Il en ressort le choix des caractéristiques suivantes:

A) contenu

1) Contenu du site

Le site est à la fois un blog et un site marchand pour le foodtruck. Le nombre de pages n'est pas encore définitif.

Il se compose à minima de plusieurs pages:

⦁ Une page d’accueil (dont la composition n’est pas encore définie)

⦁ Une page générale de recettes et une fonctionnalité de recherche de recettes à partir d’ingrédients précisés

⦁ Un onglet spécifique pour la vente en ligne avec les plats et menus

⦁ Une page de contact

⦁ Une page d’information

2) Différents rôles (cas d'usages)

L’utilisateur peut consulter le site, utiliser le moteur de recherche “frigo” et créer un compte et se connecter.

L’utilisateur identifié peut télécharger les recettes mettre des commentaires.

Ensuite il y a le client du foodtruck qui peut acheter en ligne et récupérer directement sa commande au foodtruck.

3) Intégration de page référencement

Le référencement est important vu la concurrence expliquée auparavant. Il s’effectuera par google analytic.

4) Droits

Les droits administrateurs seront laissés au client post livraison pour ce qui est de la création modification et effacements des contenus mais le reste est assuré par la maintenance.

5) Formulaires

Les formulaires sont présents au nombre des deux.

Le premier pour s’inscrire sur le site et l’autre pour la prise en charge des commandes au foodtruck.

6) Moteur de recherche interne.

Il y aura un moteur de recherche interne pour la recherche des recettes par mot clés et un moteur de recherche correspondant à la recherche par ingrédient.

7) Blog

Le système de gestion de contenus se fera directement par via les droits administrateur du site.

8) Actualités

Un onglet actualités dans la page d’accueil sera inséré (non fait).

9) Newsletter

L’envoi d’une newsletter sur inscription est prévu. La fréquence sera laissée au choix du client post réception du site. Les spécificités d’envoi de l’email ne sont pas encore connues.

B) contraintes techniques

1) Architecture

L’architecture du site sera approfondie ultérieurement, une arborescence non définitive est présentée dans la partie 1.

2) Back-office:

Le back office est fait en php avec l’utilisation de mysql pour la gestion de la base de données.

3) Front end

Le front end sera géré via la librairie Boostrap et du Javascript pour le popup. Le site est responsive.

L’éditeur de vidéo est à déterminer avec le client en fonction des formats vidéo qu’il fournira.

4) Maintenance

La maintenance est assurée sur un contrat à part. Elle sera effectuée par le Prestataire.

Une astreinte pour le dépannage est mise en place.

5) Sécurité

La sécurité sera gérée avec reCAPTCHA.

6) Hébergement

Le site sera hébergé par OVH (sous confirmation du client) sous serveur Apache. Il est actuellement hébergé en local.

7) Spécifications fonctionnelles

Le langage de programmation sera en PHP et les bases de données gérées avec MySQL.

Le e-commerce sera géré avec Wix e-commerce. Les widgets avec wheelio

C) livrables

Les livrables serotn fait tout au long du processus tel que le préconise la méthode agile. Des points de validation avec le client seront fait régulièrement.

D) Planning

1) Formations et Accompagnement

La formation et l’accompagnement sont prévus et planifié post réception du site. Un accompagnement est prévu pour le mois suivant la réception du site.

2) Délais

Les délais sont les suivants (seront modifiables d’un commun accord avec le client:

31 octobre 2021 premier visuel du site, établissement de charte graphique.

1er mars 2022 point d’étape sur le visuel et le fonctionnement du site

1er août 2022 livraison du site, mise en place du planning de formation.

3) Devis

Un devis sera proposé au client en fonction des choix qu’il établira pour les partenaires comme l’hébergeur et le système de gestion de contenus.

Ce cahier des charges ne sera effectif que lors de la signature d’un contrat régissant les relations avec le client durant la prestation et après la livraison.

**Conception**

La conception est l’étape dans laquelle est pensé la structure du site. Elle intervient après le cahier des charges qui décrit les souhaits du client.

La conception se fait en plusieurs étapes. La première est la description des cas d’utilisation. C’est la suite de l’arborescence décrite antérieurement.

Les cas d’utilisation (ou use case)

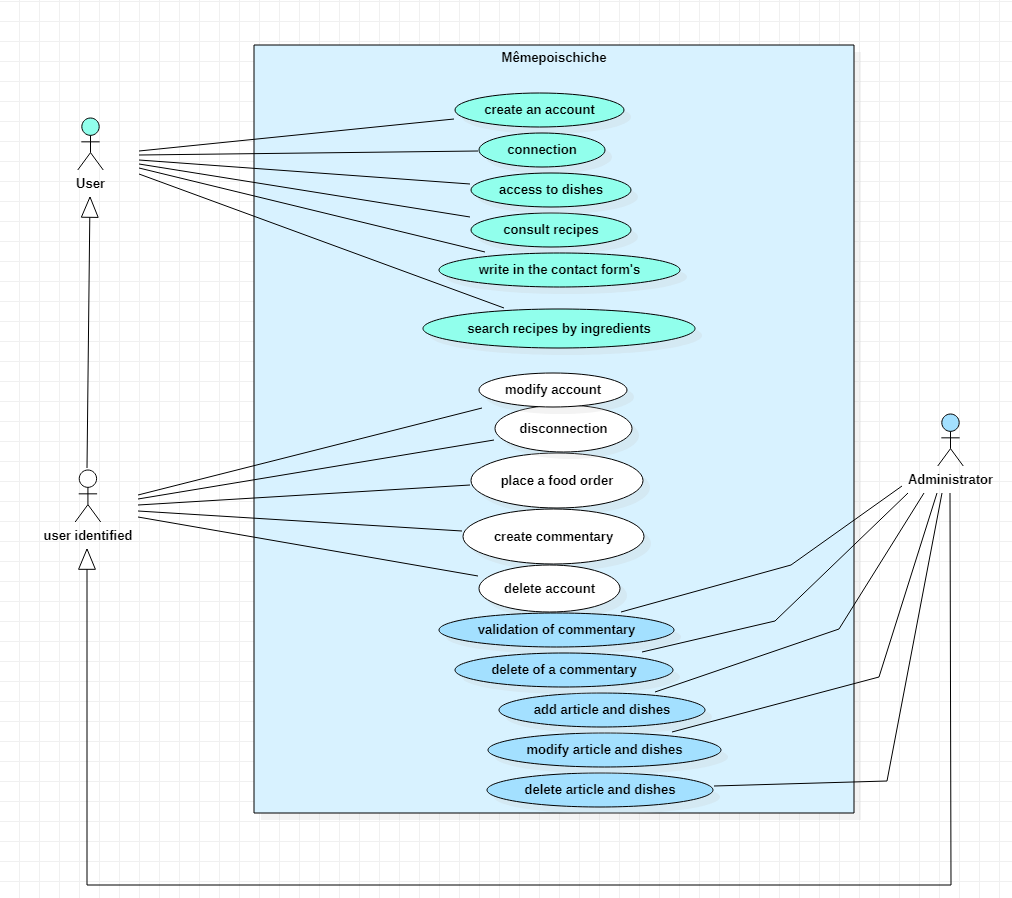
C’est la description sous forme de schéma l’ensemble des utilisations du site par l’utilisateur. Il décrit les actions possibles et le rôle des personnes. Chaque cas d'utilisation correspond à une fonctionnalité du site.

Les acteurs représentent un type d'utilisateur. Dans ce projet, ils sont de trois types: l’utilisateur, l’utilisateur identifié et l’administrateur.

L’utilisateur (user) peut créer un compte, se connecter, avoir accès aux recettes, écrire dans le formulaire de contact et accéder au moteur de recherche des recettes.

L’utilisateur identifié (user identified) peut notamment se déconnecter, modifier son compte etc... Il hérite des cas d’utilisation de l’utilisateur, c’est à dire qui peut faire toutes les actions auquel ce dernier a droit.

Il en est de même pout l’administrateur (administrator) qui hérite des cas d’utilisation de l’utilisateur identifié. Il peut quant à lui exercer des actions comme valider ou supprimer un commentaire ou une recette.



Les diagrammes d’activité

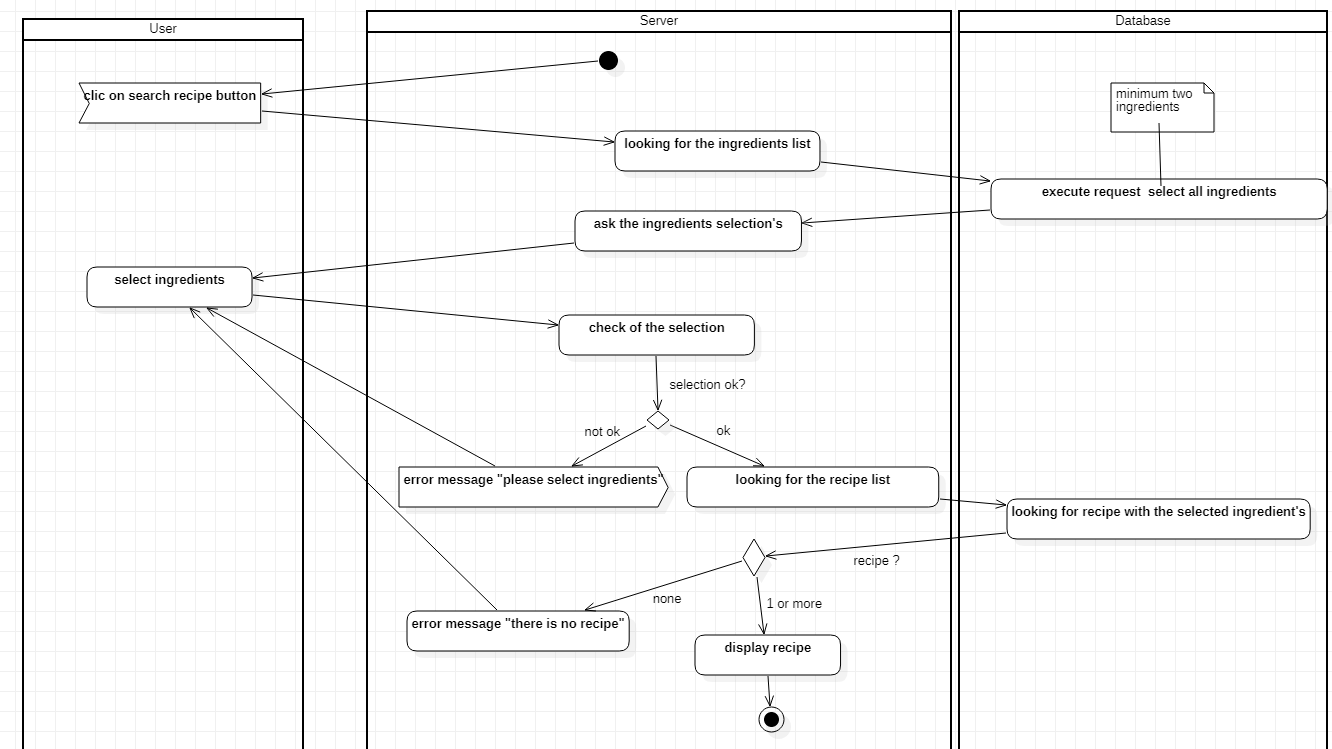
Le diagramme d'activité est la représentation chronologique d'un cas d'utilisation.

Le cas d’utilisation représenté ici est la fonctionnalité de recherche de recette par ingrédient. Il s’établit entre l’utilisateur le serveur et la base données

Le flow intial commence au serveur qui est en attente d'une requête de l'utilisateur. Il s’en suit le déroulement chronologique de la fonctionnalité, qui demande à l’utilisateur de choisir les ingrédients pour rechercher dans la base de données de recettes du site et afficher celles qui correspond aux ingrédients sélectionnés.

Une vérification est faite si les ingrédients sélectionnés sont corrects. Si les ingrédients ne sont pas correctement sélectionnés, la sélection est de nouveau demandée. Une fis cette dernière correcte, il est procédé à la recherche en base de données. Celle-ci renvoi le résultat. Si il est positif (une ou plusieurs recettes) elles sont affichées; si il n’y a pas de recette, la selection d’ingrédient est de nouveau demandée par l’affichage d’un message d’absence de recettes.

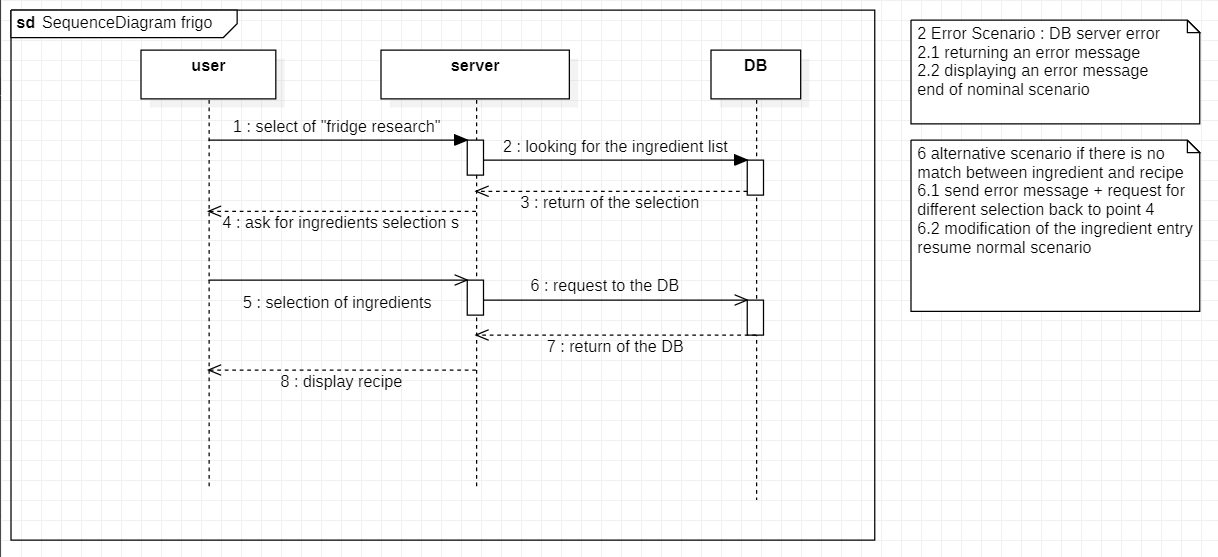
L'état final est l'état de fin de processus attendu, ici au niveau du serveur.



Le diagramme de séquence:

Le diagramme de séquence est la représentation de l'activité ici frigo de manière chronologique. C'est le scénario nomimal qui est présenté ci après. La fonctionnalité est décrite action après action et suit le flux de l’information entre l’utilisateur le serveur et la base de données.

Il est également fait mention des scenario alternatifs, qui décrivent les actions en cas de non applications du scénario nominal.



Le diagramme de classe:

C’est la description du système d’un point de vue statique. Cette phase est plus orientée vers le code. En effet, ce diagramme représente la manière dont les données vont être stockées et leurs accès. Des associations relient les classes entre elles. Les classes sont des modèles pour une donnée et représente un objet.

Une association est une connexion entre les classes. Elle se décrit par un verbe par exemple entre recipe page et ingredient il y a “contain”. La page de recettes contient des ingrédients.

Les agrégations sont la représentation de la question d’appartenance. L’appartenance suis le sens de la flèche d’agrégation. Par exemple l’ingrédient est agrégé à une recette il est contenu dans la recette.

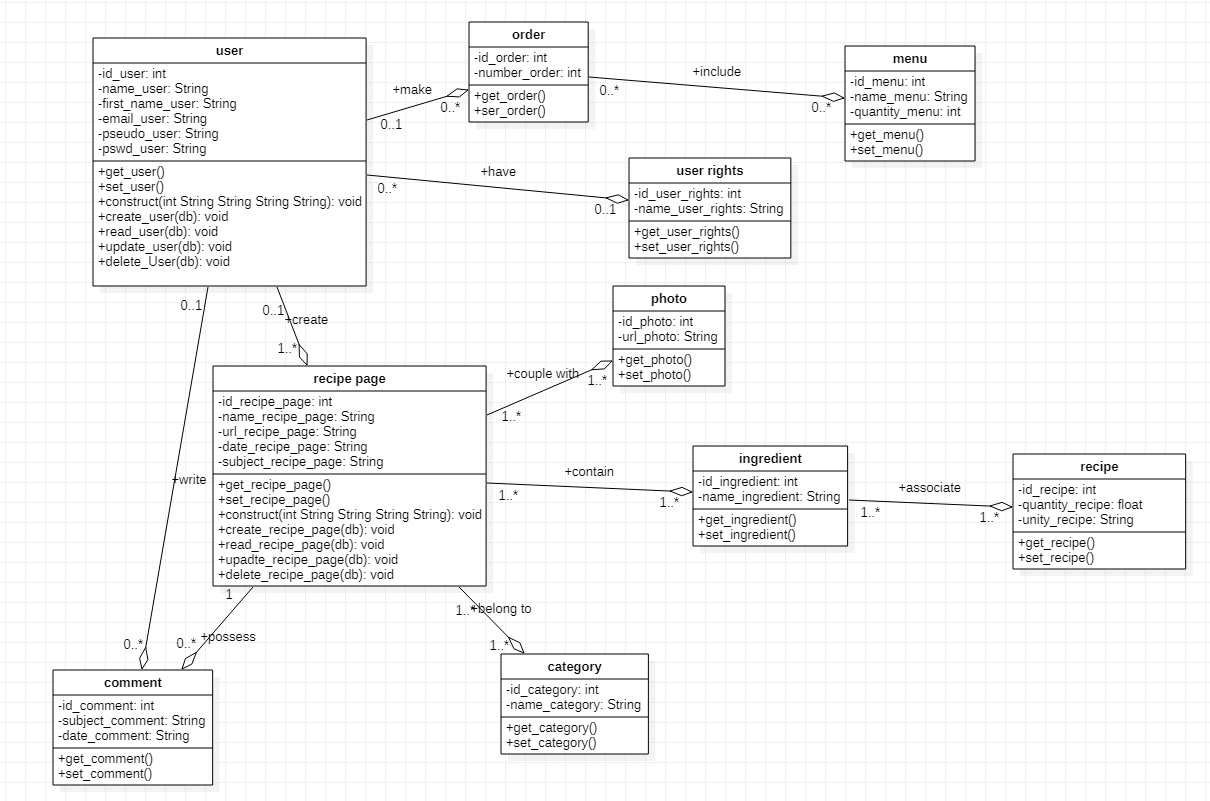
Les cardianalités sont les représentations du nombre de fois ou participe une association. Elles peuvent être de 0 1 ou N. la cardinalité zéro est la minimale et N la maximale. Le N est annoté \*.

En continuant l’exemple de l’ingrédient et des recettes, il faut à minima un ingrédient pour un recette et un ingrédient fait parti à minima d’une recette. Au maximum il y a N ingrédient pour N recettes. D’ou qu’il figure 1 \* car c’est un minimum de un et il n’y a pas de maximum théorique.

Ici nous voyons l’objet user, recipe page ingredient etc.... Chaque objet aura les mêmes caractéristiques appelé attributs. Par exemple un ingredient est un objet qui a pour attribut un name. Pour chaque création d’objet une instance est créée, ingrédient 1, ingrédient 2 etc...

Chaque instance est identifiée par une clé primaire, clé qui est unique à chaque objet.

Les méthodes décrivent les procédures de chaque classe. Ici les méthodes du CRUD (Create Read Update Delete) sont présentent dans plusieurs classes. Elles permettent d’effectuer l’action de créer par exemple un objet.



**Réaliser une interface utilisateur web statique et adaptable**

L’interface utilisateur

**Développer la partie backend d’une application web ou web mobile en intégrant les recommandations de sécurité**

*5 Créer une base de données*

*6 Développer les composants d’accès aux données*

*7 développer la partie backend d’une application web ou web mobile*

*8 Elaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce*

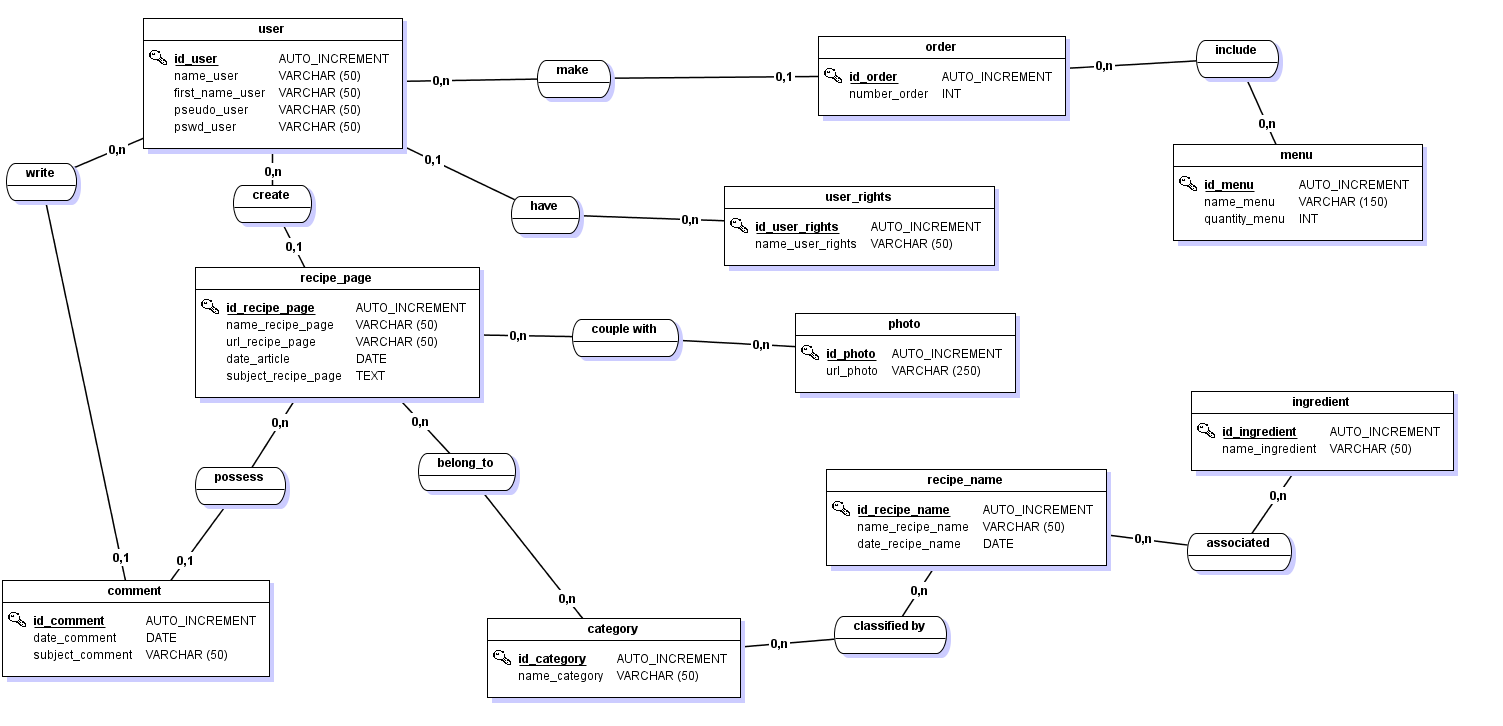
*MCD MLD interaction avec la bdd*

Modèle conceptuel de données:

Le modèle conceptuel de données est la représentation des entités du site.

Ici on voit qu’il y a notamment un user, une recipe\_page une category etc...

Les cardinalités décrivent comme pour le diagramme de classe le nombre de fois que qu’une entité peut interagir avec l’autre. Par exemple, user et user rights ont une cardinalité have exprimant qu’un user peut avoir zéro ou un droit (O,1) alors que les droits sont possédés par zéro ou plusieurs users (O,N).



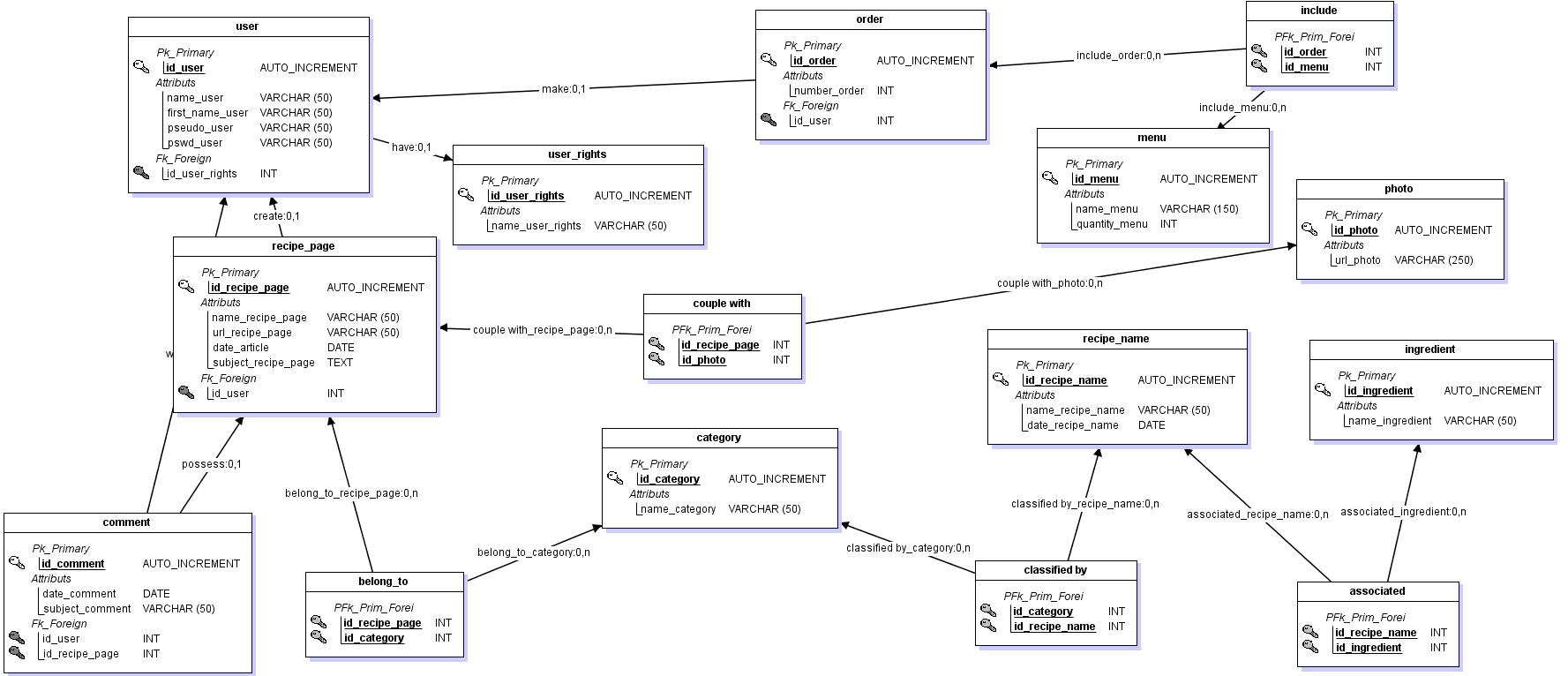
Le modèle logique de données:

C’est la description des tables qui organisent les données de l’application.

La logique de cardinalité est la même que dans le diagramme de classe. Ci après les données sont organisées en tables. Elles ont des attributs. Elles possèdent toutes un id qui est la clé primaire.

On voit ici que la table recipe\_page est centrale ainsi que la table user.

La table recipe\_page et la table comment sont reliées par une cardinalité “possess” 0 N et 0 1. Cela signifie qu’une page de recette peut posséder zéro ou N commentaires. Cependant, un commentaire n’appartient qu’à maximum une page de recette (cardinalité 0 1).



La relation entre la table ingredient et la table recipe\_name est faite par la table d’association “associated “. Elle devient une table dans le modèle logique de données dont la clé primaire est composée des deux clés étrangères (FK Foreign Key) qui viennent des tables qui la composent.

SQL

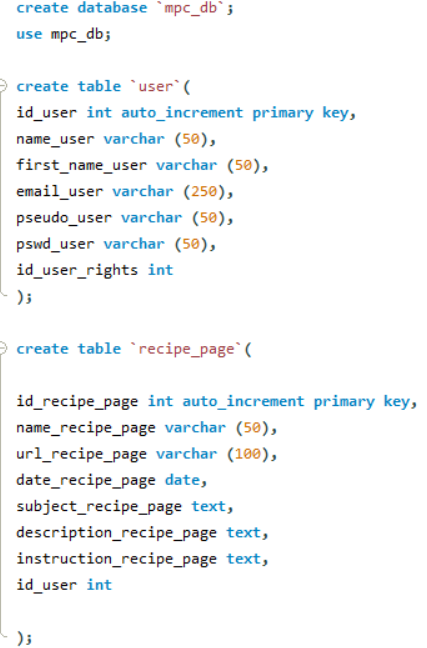
Le SQL ou Structured Query Language est un langage de requête structurée.

C’est le langage de la base de données, base qui permet le stockage des données de l’application. Elles sont écrites de manière organisée à travers des champs et des tables et des index . Si les tables ont des relations entre elles, il s’agit alors de base de données relationnelles.

C’est via le système de gestion qui utilise le SQL que sera géré la base de données. Ici est utilisé My SQL.

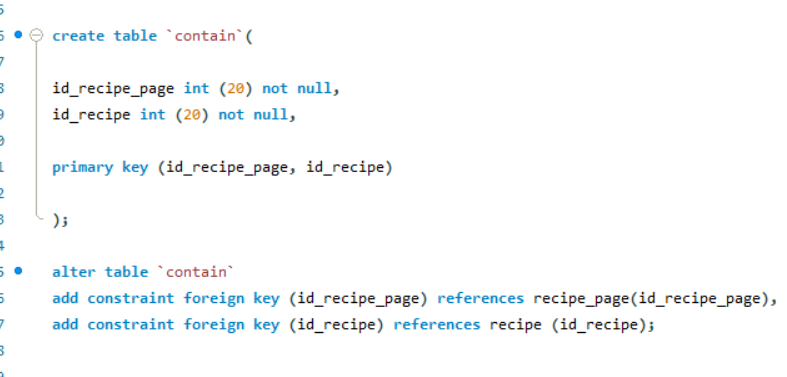
Le SQL permet de créer lire modifier et supprimer des données et des tables. Il gère également les utilisateurs et leurs droits.

Ci après des extrait de la création de la base de données, des tables, des tables d’association et des données.



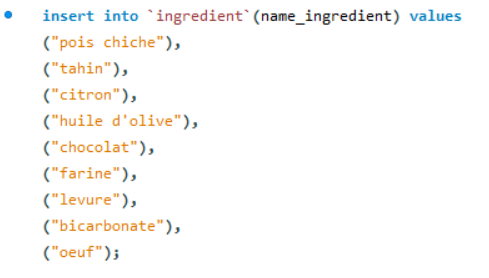
Ci-dessus la création de la base de données “mpc\_db”. Et son utilisation par “use”.

Ensuite il est créé les tables ici user et recipe\_page. L’attribut id\_user est la clé primaire qui s’auto incrémente puis les autres attributs correspondent à chaque catégorie que l’on veut avoir dans la table. Par exemple le nom de l’utilisateur son email etc...



Ici est crée la table d’association “contain” qui est formée par les deux clés primaires de recipe page et recipe.

La formule “alter table ‘contain’” permet de modifier la table “contain” et “add constraint foreign key” d’y ajouter une clé étrangère en référence au champ “id recipe\_page” de la table “contain” et fera reference au champ “id\_recipe\_page” de la table “recipe\_page”.

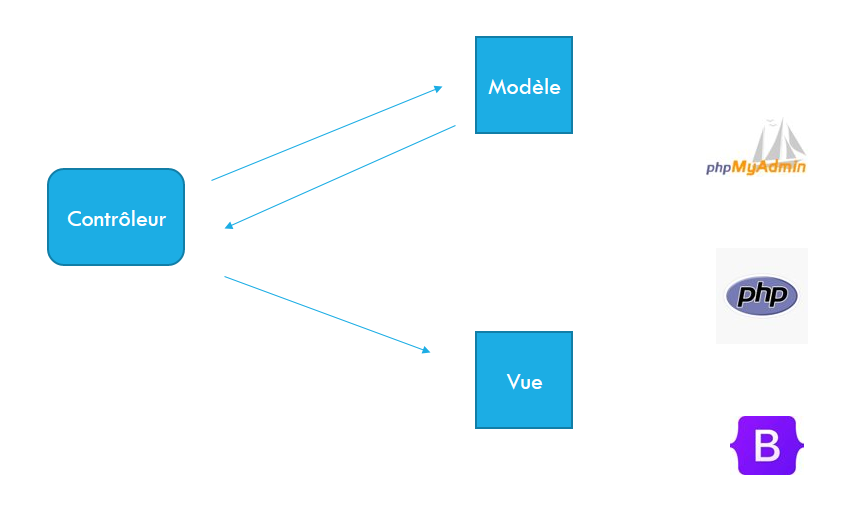


Ici il est ajouté des données par la formule “insert into” dans la table “ingredient” qui ont pour valeurs (“values”) la liste décrite dans le champ “name\_ingredient”.

Interaction avec la base de données :

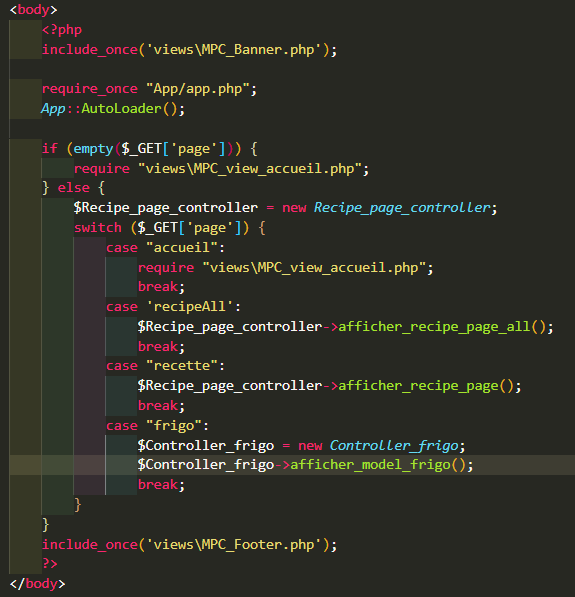
Une fois la base de données créée, l’utilisation des fonctionnalités di site requièrent des interactions avec celle-ci. En effet les fonctionnalités comme la recherche par ingrédients ou la présentation de l’ensemble des recettes requièrent une manipulation des données. Cette manipulation se fait via des requêtes SQL directement dans le code de l’application. Ci-après sont présentées deux requêtes. La première est une requête de sélection, ma seconde une requête imbriquée.

A titre liminaire, il faut présenter l’organisation du code. Ce dernier est structuré en programmation orientée objet selon le modèle d’organisation Modèle vue Contrôleur (Model View Controller). C’est à dire que les fonctionnalités vont être divisées entre un contrôleur (le chef d’orchestre) le modèle (ce qui va gérer l’interaction avec les données) et la vue (l’interface utilisateur). Il n’y a aucun lien entre le modèle et la vue comme le montre le schéma ci après.



Les technologies utilisées sont phpMyAdmin pour la gestion de la base de données, PHP pour la rédaction du code back end et Bootstrap pour le front end.

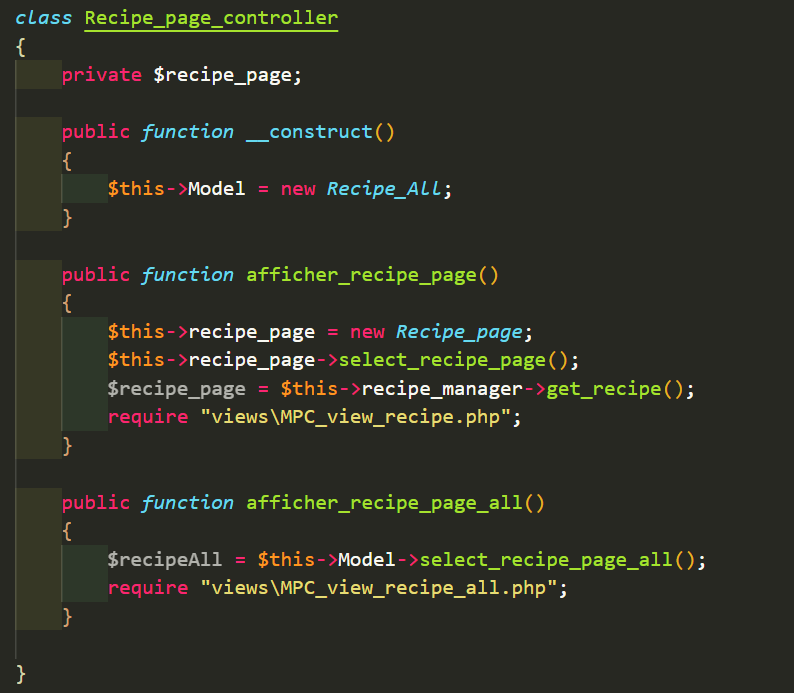
Tout d’abord, nous allons voir l’index.



C’est l’organisation de l’application. Il comporte:

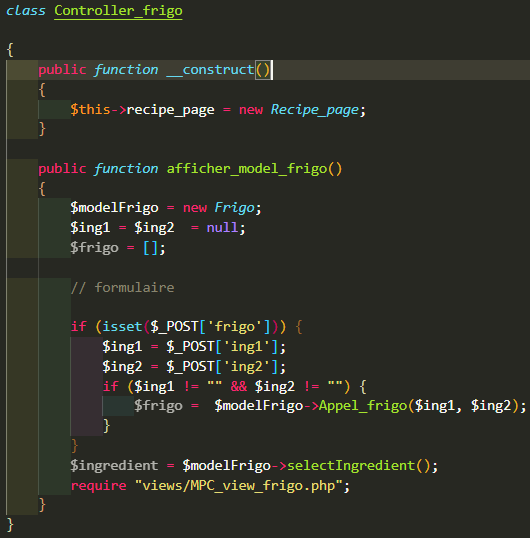
* Des include des vues de la bannière et du footer;
* Un require\_once de la fonction app autoloader qui charge automatiquement les pages et les vues;
* Une structure “If else” qui met en place le routage des pages de l’application. Ce routage se fait via des cas référençant les pages de l’application qui actionne un contrôleur et une vue.

Le contrôleur



Ici c’est le contrôleur de la fonctionnalité d'affichage des recettes. Il est isolé dans une classe pour y faire appel. Il se compose d’un attribut privé qui est la variable recipe\_page puis d’attributs publics dont un constructeur et des fonctions d’affichage de pages (soit une soit toutes). La fonction d’affichage “afficher recipe\_page” instancie un nouvel objet qui récupère auprès du manager “recipe\_manager” la recette. Le require affiche la vue correspondante.

Un autre contrôleur, celui de la fonctionnalité “frigo”:



Le controller permet d'appeler l'objet frigo qui récupère les informations du formulaire avec un tableau

Si les champs sont remplis, la fonctionnalité lance “Appel frigo”

Cette fonction contient deux variables ing1 et 2 instanciées à 'null'

Un formulaire parcourt le tableau $frigo avec un if si les entrées de $ing ne sont pas vides

Puis un appel frigo est fait dans le modèle frigo.

Une sélection des ingrédients est faite dans le modèle frigo par la fonction de sélection des ingrédients.

Enfin, un appel de la vue avec le “requiere” termine la fontionnalité.

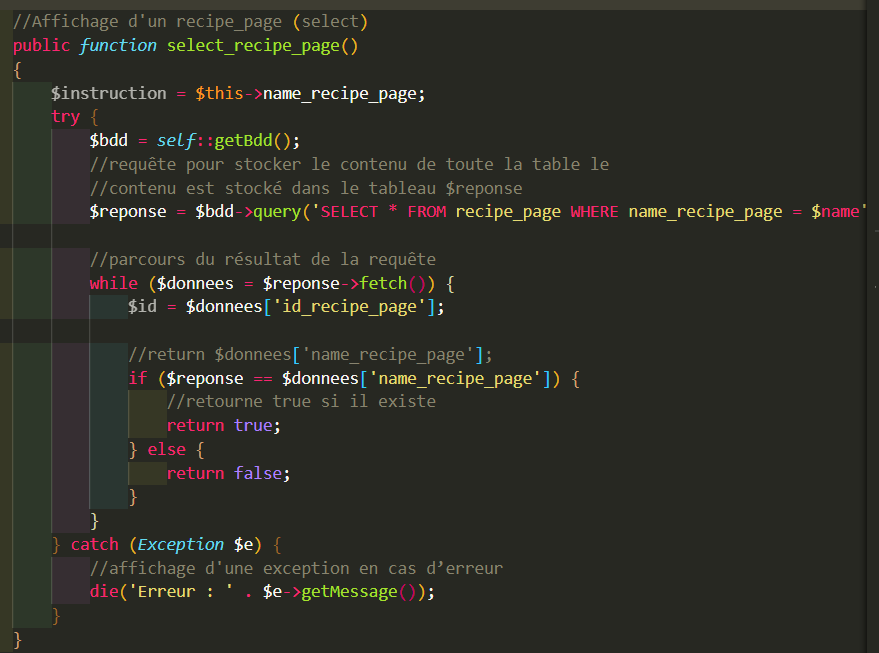
Le modèle

Le modèle de cette fonctionnalité va interagir avec les données.

Il se décompose en fonctions. Elles comportent des attributs privés et des fonctions getter et setter (annexe getter and setter) Ces dernières sont des fonctions publiques (accessibles en dehors de la classe elle-même) qui vont encapsuler les données, c’est à dire donner un accès aux données de l’objet qui comporte un attribut privé ou protégé. Pour accéder à l’information en dehors de la classe il faut passer par ces méthodes, cela permet de formater ou traduire les informations pour les afficher.

C’est également dans le modèle qu’est mis en place le CRUD c’est à dire les fonctionnalités de create read update and delete, soit toutes les modifications que l’on peut effectuer sur les données en base. C’est le modèle qui interagit avec la base de données.

Les requêtes sont faites directement dans une fonction, par le système de “Try and Catch” qui permet une gestion des erreurs et ainsi, éviter l’arrêt du programme si une erreur survient.



Cette méthode permet d’exécuter le code de la manière suivante:

Tout d’abord, dans le “try” on récupère la donnée voulue ici le nom de la recette. La requête est isolée dans une variable ici “réponse” qui encapsule la requête en base de données avec un “select all from la table recipe page ou le nom correspond au nom recherché”. Le résultat revient sous forme de tableau qui est parcouru et retourné. Le “cqtch” est alors ignoré.

Ensuite si une erreur survient le ‘try” est arrêté et le “catch” s’enclenche, le code attrape ainsi l’erreur est gérée ici en envoyant un message.

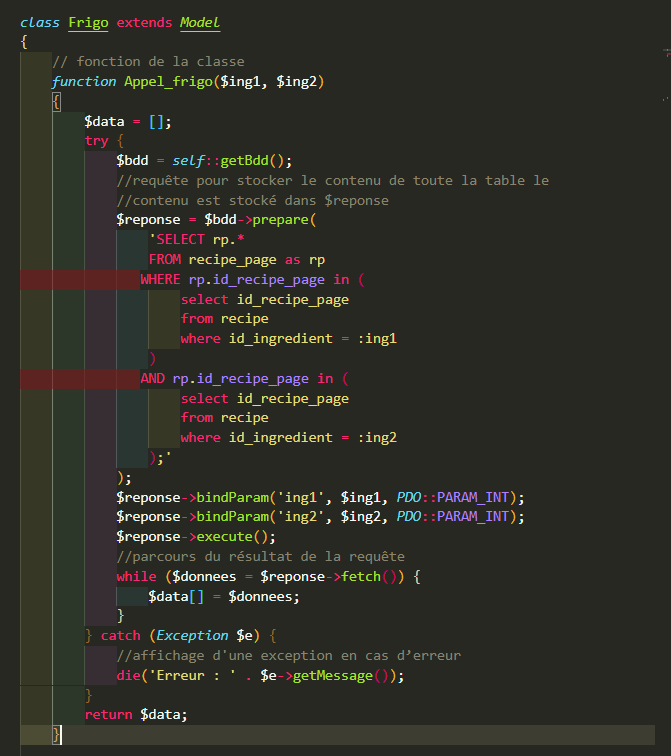
Interaction avec la base de données: requête préparée et imbriquée

Ci après est présenté la fonctionnalité “frigo”. Elle permet de récupérer en base de données les recettes contenant les ingrédients choisis.

C'est une fonction qui retourne le résultat de la requête select c’est à dire la liste des recettes.

Dans la fonction Appel frigo avec une requête préparée ($reponse) et imbriquée les ingrédients sont sélectionnés par leurs identifiants (id) dans la table recipe avec la condition d'égalité entre l'ingrédient sélectionné et celui stockés en base de données.

La $reponse est liée par le bind param et ensuite exécutée.



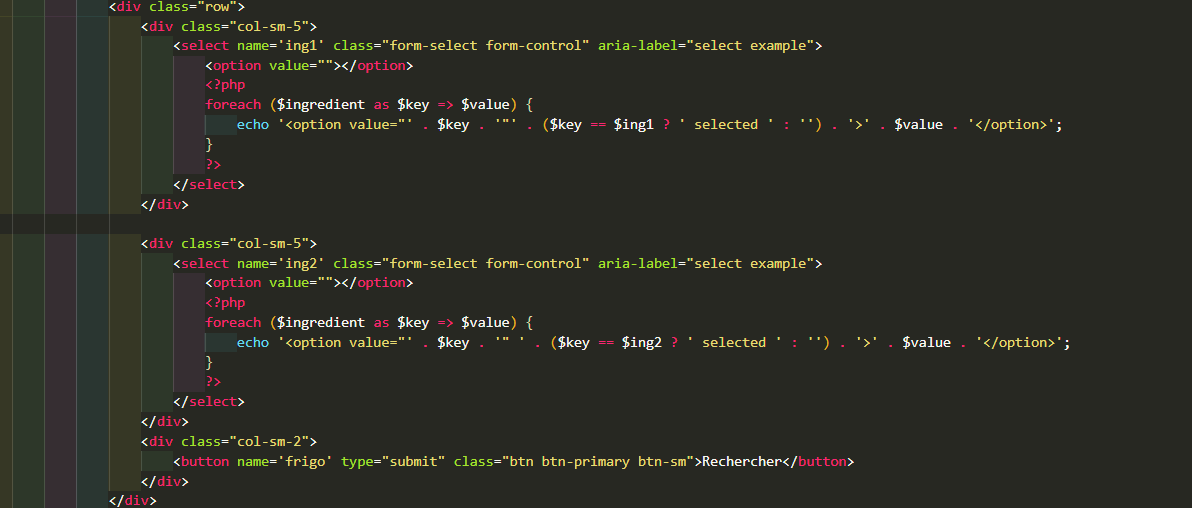
La vue

Elle est rédigée avec Boostrap et html et css.

Elle est appelée par le contrôleur via un require. Ici est présentée la vue de la page frigo qui est en deux temps.

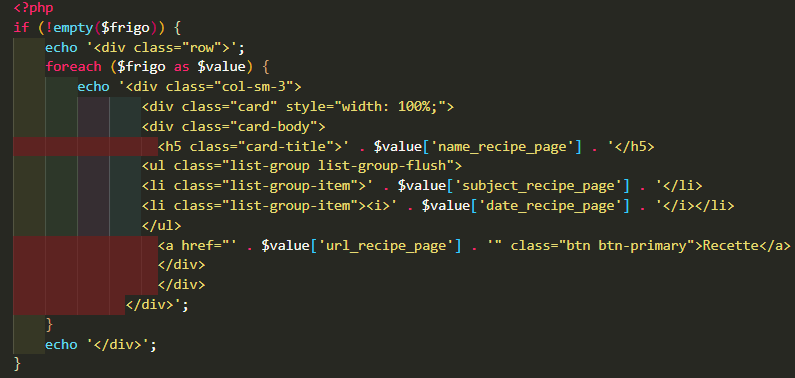
Le premier est le choix des ingrédients via la création d'un formulaire des ingrédients sélectionnés dans la base de données en rappelant les variables ing1 et 2 dans deux sélecteurs différents (option value). Le selected permet de prendre la valeur par défaut.

Un bouton permet de lancer la recherche.

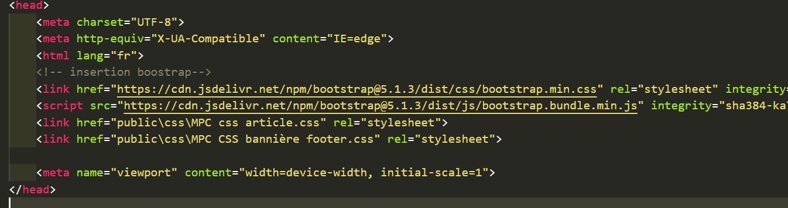


Le second temps est le résultat de la sélection.

En parcourant la variable frigo : mise en forme des informations relatives aux recettes à savoir le nom le sujet et la date de la recette sélectionnée et le lien de la recette.



La responsivité est gérée par Boostrap. C’est la dernière ligne qui permet l’adaptation de la taille de l’écran en fonction du support utilisé.



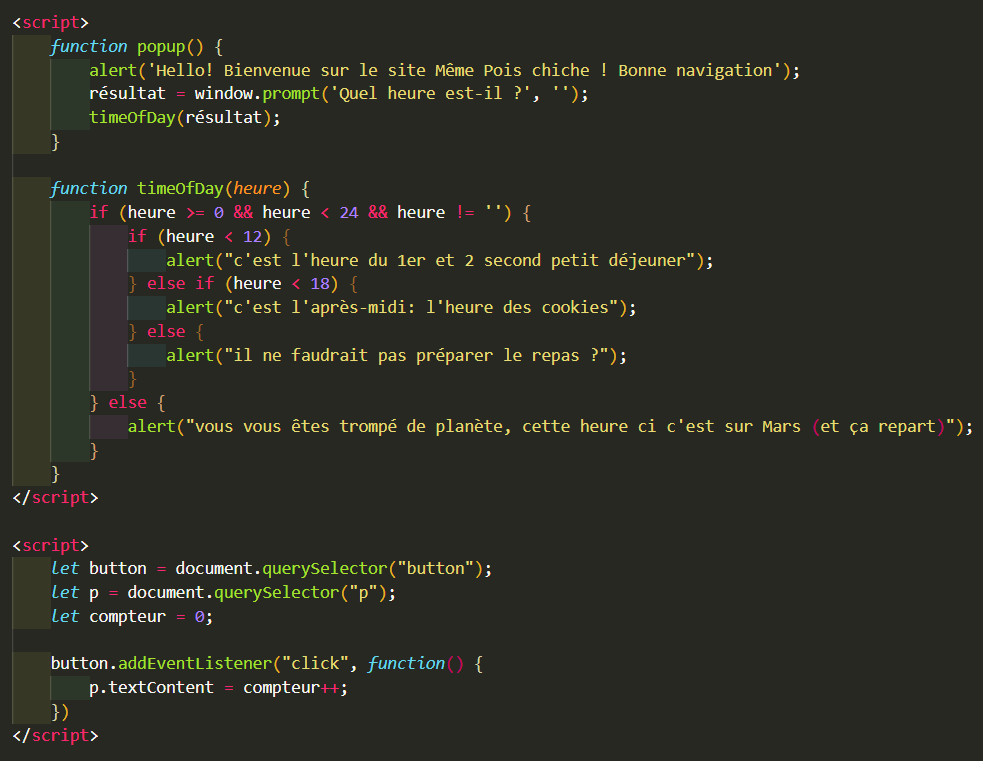
Le rendu visuel se trouve en annexe “affichage”.

Inclusion de Javascript

Ici un popup en langage JavaScript. Il demande l’heure et vous répond en fonction de ce qui est renseigné.

Il est construit avec une structure “if, else”. En effet, cela permet d’établir un circuit fermé de réponse, c’est à dire que si la réponse n’entre pas dans les cas prévus du if, le else prend le relais. Cela permet de sécuriser l’entrée de données dans le popup.

Ici il n’y a pas d’accès à la base de données c’est une interaction uniquement sur la page d’affichage ce qui minime le risque d’accès malveillant.



Interaction avec la base de données: la connexion à la base de données

La connexion à la base de données se fait par modèle class.

L’abstract oblige l'appel de la classe via l'extend par les autres classes. Le static permet d'avoir une variable unique partagée. Le protected permet la fonction soit utilisée uniquement par les enfants.



Sécurisation

Faire une slide sur le pdo avec une connexion root et “ “ et une avec qqch d’autre

Ici se présente la classe encapsulant la connexion.

C’est un modèle de classe abstraite permettant de l’appeler via l’extend par les autres classes. L’attribut static permet d’avoir une variable unique partagée et le protected permet aux enfants d’utiliser la fonction.

Cette fonction permet la connexion à la base de données.



Droits sql et règles de protection

Lors des développements, le travail s’est effectué sur un serveur local. Dès que l’application a vocation à être mise en production pour ensuite être mise en ligne, il faut sécuriser l’accès à la base de données pour éviter les risques d’altérations ou de vol de la base données. En effet, ses données sont la valeur du projet. C’est pour cela que l’identification et les droits des utilisateurs sont définis et restreints aux seules fonctionnalités qui lui sont accessibles.

Ci-après, il sera vu la création d’une base de données avec des utilisateur et des droits appliqués en fonction des rôles de chacun d’eux, ceci étant fait en langage SQL.

Ici on attribue à l’user laure en local le mot de passe projetdevcyber sur labase de données MPC\_DB.



Ici, on voit que sur la base de données tous les privilèges sont donnés à l’utilisateur laure

Puis, à l’administrateur les fonctionnalités de select insert ….

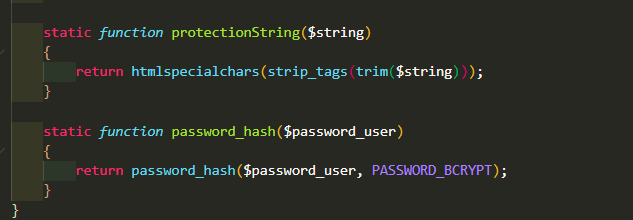
Enfin, l’utilisateur identifié a dans ce cas la possibilité de faire des requêtes SELECT. En particulier cela lui permet d’utiliser le moteur de recherches de recettes.

Le flush privilège permet de mettre à jour les droits lors de la création.



Sécuriser l’application :

Afin d’éviter les failles, il a été mis en place dans le fichier app, les fonctions suivantes:



La fonction protectionString permet d’appliquer les recommandations pour nettoyer les données transmises par l’utilisateur et ainsi se prémunir d’attaque sous forme d'injection de code xss.

Pour s’en prémunir :

⦁ enlever les balises html avec la fonction strip\_tags()

⦁ neutraliser les caractères spéciaux : htmlspecialchars()

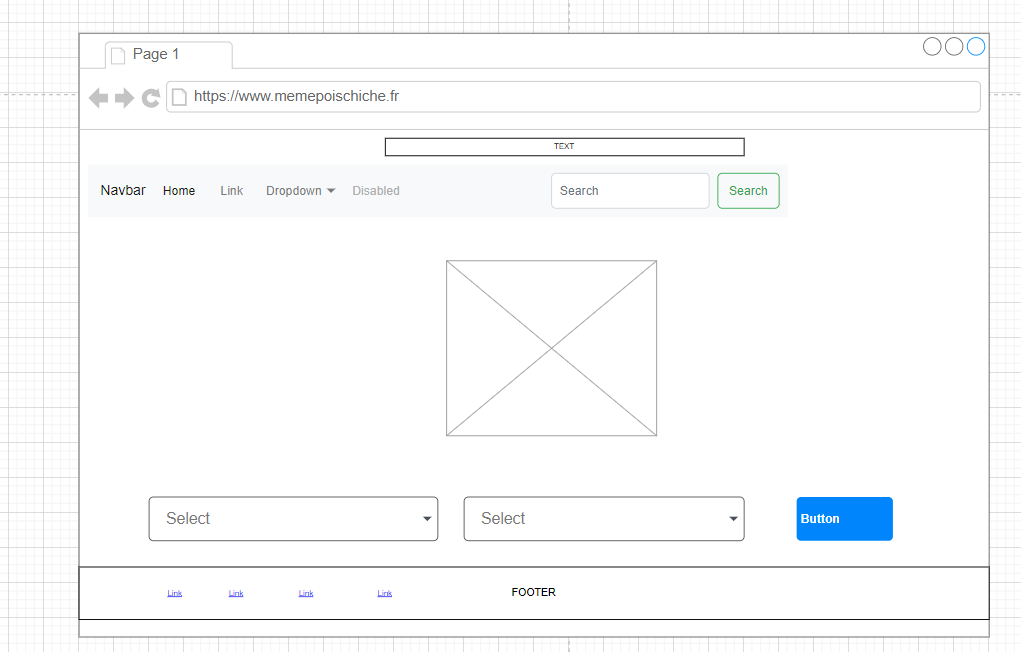
⦁ supprimer espace en début et fin de chaîne : fonction trim()

La fonction password\_hash quant à elle chiffre le mot de passe de l’utilisateur en base de données.

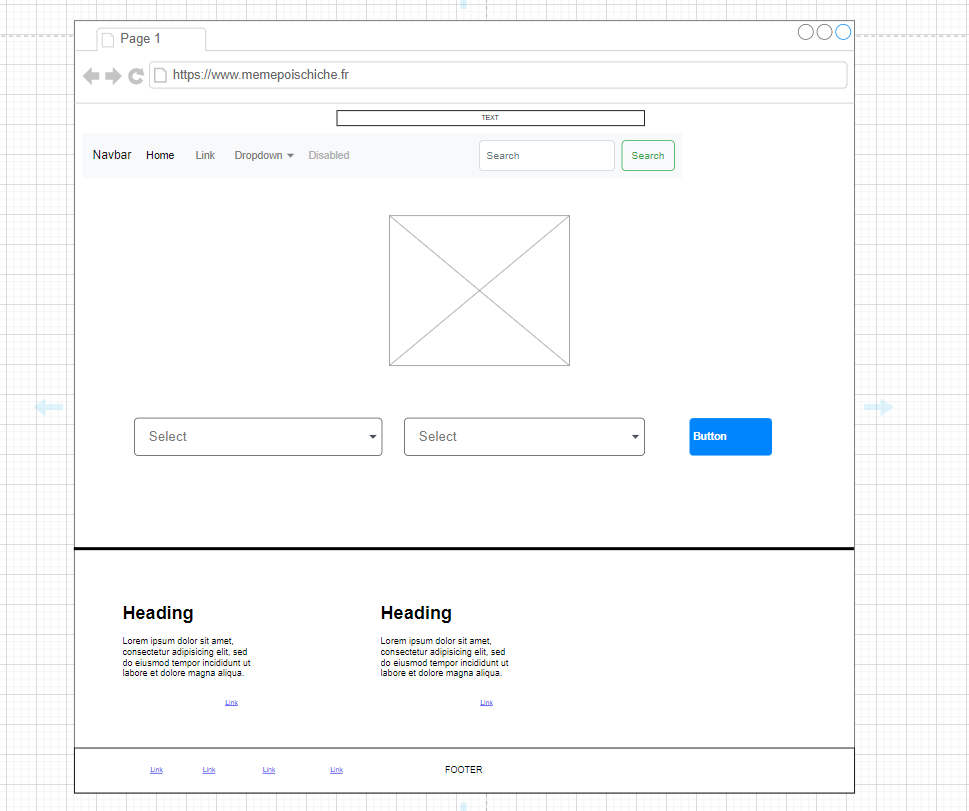
**ANNEXES**

Maquettage

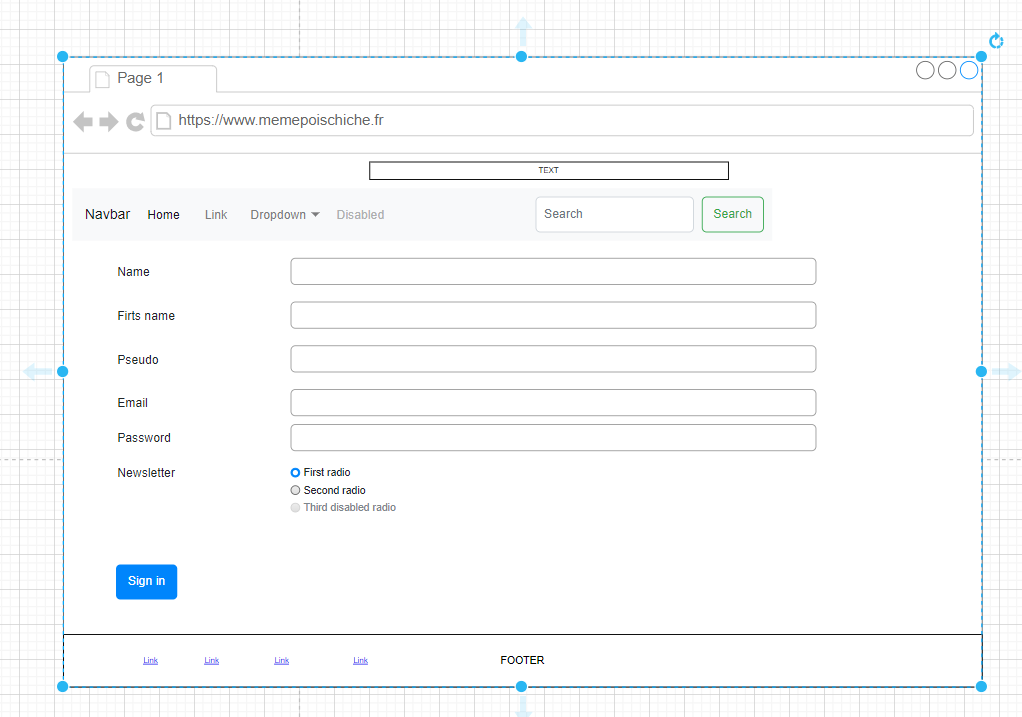
Wireframe Frigo page 1



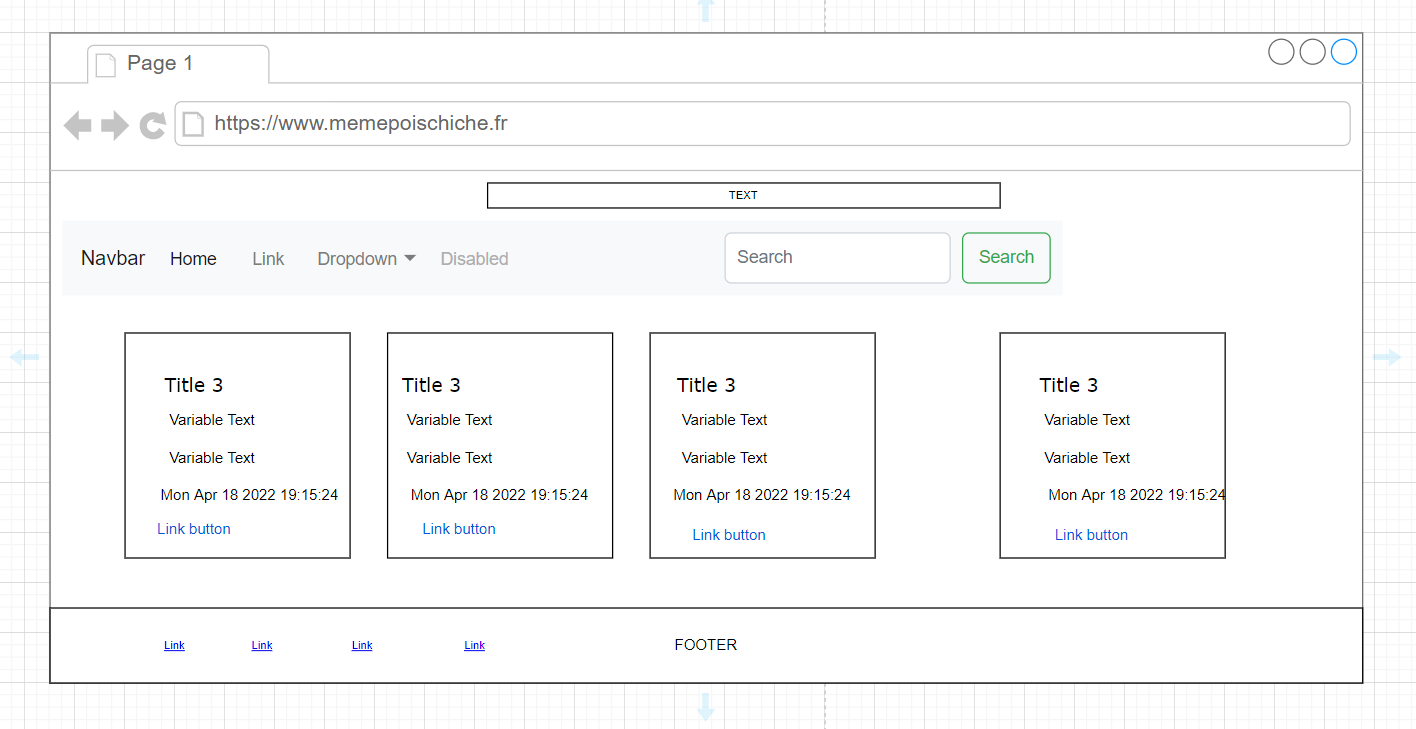
Wireframe Frigo page 2



Wireframe page de connexion



Wireframe page de “toutes les recettes”



Cahier des charges

SOMMAIRE

I Présentation du projet

A) Présentation du projet

1) Projet et objectifs

a) statut actuel marque et identité

b) histoire

c) secteur d’activité

d) le projet

2) étude du besoin

3) concurrence

a) contexte et enjeux

b) objectifs marketing

c) positionnement du futur produit

d) analyse SWOT

B) Cible

1) public

2) équipements

C) Typologie d’application

D) Périmètre

E) Intervenants

II Graphisme et ergonomie

A) Graphisme

B) Maquettage - Arborescence

III Spécificités et livrables

A) contenu

1) Contenu du site

2) Différents rôles (cas d'usages)

3) Intégration de page référencement

4) Droits

5) Formulaires

6) Moteur de recherche interne

7) Blog

8) Actualités

9) Newsletter

B) contraintes techniques

1) Architecture

2) Back-office:

3) Front end

4) Maintenance

5) Sécurité

6) Hébergement

7) Spécifications fonctionnelles

C) Livrables

D) Planning

1) Formations et Accompagnement

2) Délais

3) Devis

IV Spécifications et IHM

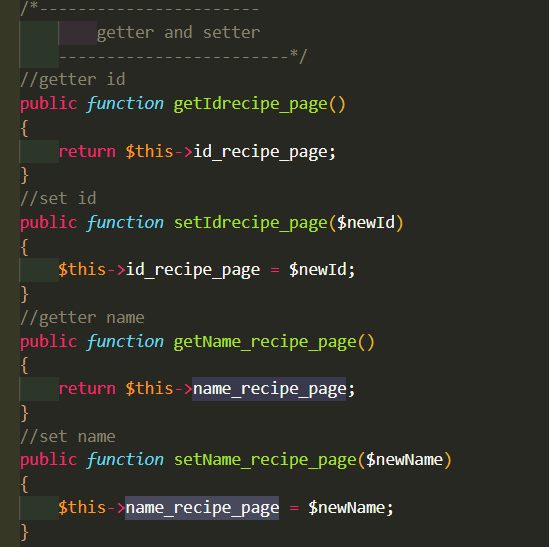
A) utilisateur x

B) utilisateur inscrit

C) Client

Code

Getter and setter



Html css

?

Affichage



