# Oral framework web

Blabla bonjour est ce que vous m’entendez bien toussa toussa

Je vais vous présenter ma version du projet.

Tout d’abord je vais vous parler du thème de mon projet puis de ses fonctionnalités ainsi que les outils que j’ai utilisé.

En ce qui concerne le thème du projet, je me suis inspiré d’un problème que je rencontre et qu’on rencontre sûrement tous. Ce problème c’est : « on mange quoi ce soir ? »

J’ai donc décidé de créer une application pour enregistrer des recettes de cuisine et les proposer aléatoirement.

Pour améliorer l’expérience utilisateur au quotidien j’ai décidé d’ajouter la gestion des ingrédients pour préparer une liste de course à partir des recettes sélectionnées.

Je me suis donc un peu inspiré des applications de commande de courses en drive, en y ajoutant la gestion de recettes de cuisine.

Voici les maquettes que j’ai réalisées pour me guider : [montrer maquette].

Pour m’aider dans la création de cette application j’ai essayé de prédécouper mes maquettes et noter en rouge ce qui pourrait s’apparenter plus tard à des composants VueJs.

Je vous ai parlé principalement de recettes et vous voyez ici ce qui ressemble à une application de liste de courses.

L’explication est que le point commun entre une recette et une liste de courses étant les ingrédients, j’ai choisi de les afficher sur la page d’accueil et de pouvoir les ajouter à la liste de courses.

J’ai aussi décidé de ce mode de fonctionnement car il n’est pas rare de penser à un ingrédient ou un produit qu’il nous manque et de vouloir directement l’ajouter à la liste de courses avant de l’oublier. L’utilisateur n’a pas besoin de naviguer dans l’application il est directement sur la page correspondante, dès l’ouverture, alors que la réflexion du choix des recettes n’est souvent pas aussi instantanée et spontanée.

Le calendrier visible sur cette maquette, destiné à choisir la date à laquelle l’utilisateur souhaite aller chercher ses courses, à été déplacé dans une page dédiée car j’ai estimé après les premiers développements qu’il n'était pas à sa place.

Le but étant de retrouver les ingrédients facilement, ils ont été triés par catégorie comme dans beaucoup de magasins et d’applications dédiées afin de pouvoir les retrouver plus facilement.

La deuxième maquette que j’ai créé représente la page des recettes : 3 recettes sont affichées aléatoirement depuis la base de données.

Chaque recette est composée de 5 éléments :

-          Un titre,

-          une image (optionnelle)

-          une description (optionnelle aussi)

-          une liste d’ingrédients (obligatoire, comment faire une recette sans ingrédients ?)

-          les étapes à réaliser pour la création de la recette (obligatoire également)

Comme précédemment les différents « blocs » sont prédécoupés pour faciliter le développement.

[présentation du site web]

Comme vous pouvez le remarquer, mon site ne correspond pas exactement aux maquettes. Elles étaient là principalement pour me guider lors du développement, pas pour m’imposer des choix de design et de mise en forme qui ne me convenaient plus au fil du développement. J’ai aussi choisi de me concentrer sur la partie fonctionnelle du site car j’ai estimé cette partie plus importante : je préfère avoir un site fonctionnel et ergonomique plutôt que simplement « beau ».

On peut ajouter des ingrédients, en enlever sans que la valeur soit négative.

Les ingrédients sont ajoutés à la liste seulement quand on clique sur le bouton « + » pour éviter de surcharger l’application inutilement. Pour des soucis de performances, j’ai choisi de réaliser une requête qui charge l’ensemble des ingrédients lors du chargement de la page et les trie après pour n’afficher que ceux de la catégorie ciblée.

Pour la gestion du projet j’ai utilisé de nombreux outils qui sont détaillés dans mon compte rendu, mais je peux notamment citer le gitlab universitaire utilisé via gitkraken (version pro maggle). Le gitlab m’a notamment permis d’organiser plus facilement le projet en fonction des différents sprints.

<https://gitlab.univ-lr.fr/aperrin/projet-symfony-vuejs/-/boards>

 En ce qui concerne l’architecture de l’application il y a 3 couches :

-          Une base de donnée MariaDb dans un conteneur docker

-          Une API rest en php symfony alimenté par cette base de donnée

-          L’application VueJs qui interroge l’API php

Dans la première version du projet en symfony j’ai géré l’affichage des dates en français et la connexion des utilisateurs, malheureusement suite à des problèmes techniques je n’ai pas pu le porter en VueJs.

Je vous remercie de m’avoir écoutée.

Questions éventuelles :

-          Pourquoi l’authentification ne fonctionne pas ?

o   A la fin du dernier sprint j’ai géré des problèmes de requêtes CORS, c’était un problème que je n’avais rencontré et que je ne connaissais pas avant, j’ai donc dû comprendre et gérer. Ça m’a malheureusement prit trop de temps puisque je n’ai pas pu finir l’application comme je le voulais.

-          Quelle est l’architecture du projet ?

o   3 couches, avec une API REST qui expose le contenu de la BDD. Elle devrait aussi permettre l’authentification.

-          C’est quoi une API REST ?

o   Une API qui expose ses ressources via http avec des normes comme “une ressource est accessible via une URI portant le nom de l’entité”

-          Est-ce que vous avez aimé ce projet ?

o   Dans l’ensemble oui, c'était une bonne expérience même si ça n’a pas toujours été facile.

-          Ce que vous avez utilisé pour le style ?

o   Pour le design j’ai utilisé bootstrap car j’ai voulu utiliser vuetify au départ mais j’ai eu des soucis d’installation et de configuration donc pour éviter de perdre trop de temps j’ai décidé de basculer sur bootstrap que j’avais déjà un peu utilisé auparavant et beaucoup de CSS natif.

-          Les ingrédients sont ajoutés au panier en cliquant sur le bouton, comment avez-vous fait ?

o   montrer store/index.js => getter/setter de mémoire à vérifier

-          Pour la requête qui charge l’ensemble des ingrédients et le trie, comment avez-vous fait ?

o   montrer store/index.js => mutation de mémoire à vérifier

-          Pourquoi MariaDb ?

o   comme MySQL mais openSource

-          Pourquoi Docker ?

o   Bien plus pratique pour lancer une BDD MariaDB

-          Quelle API avez-vous utiliser ?

o   La mienne

-          Comment avez-vous interrogé votre API ?

o   J’ai interrogé mon API en utilisant Axios. (store/index.js partie mutation à vérif)