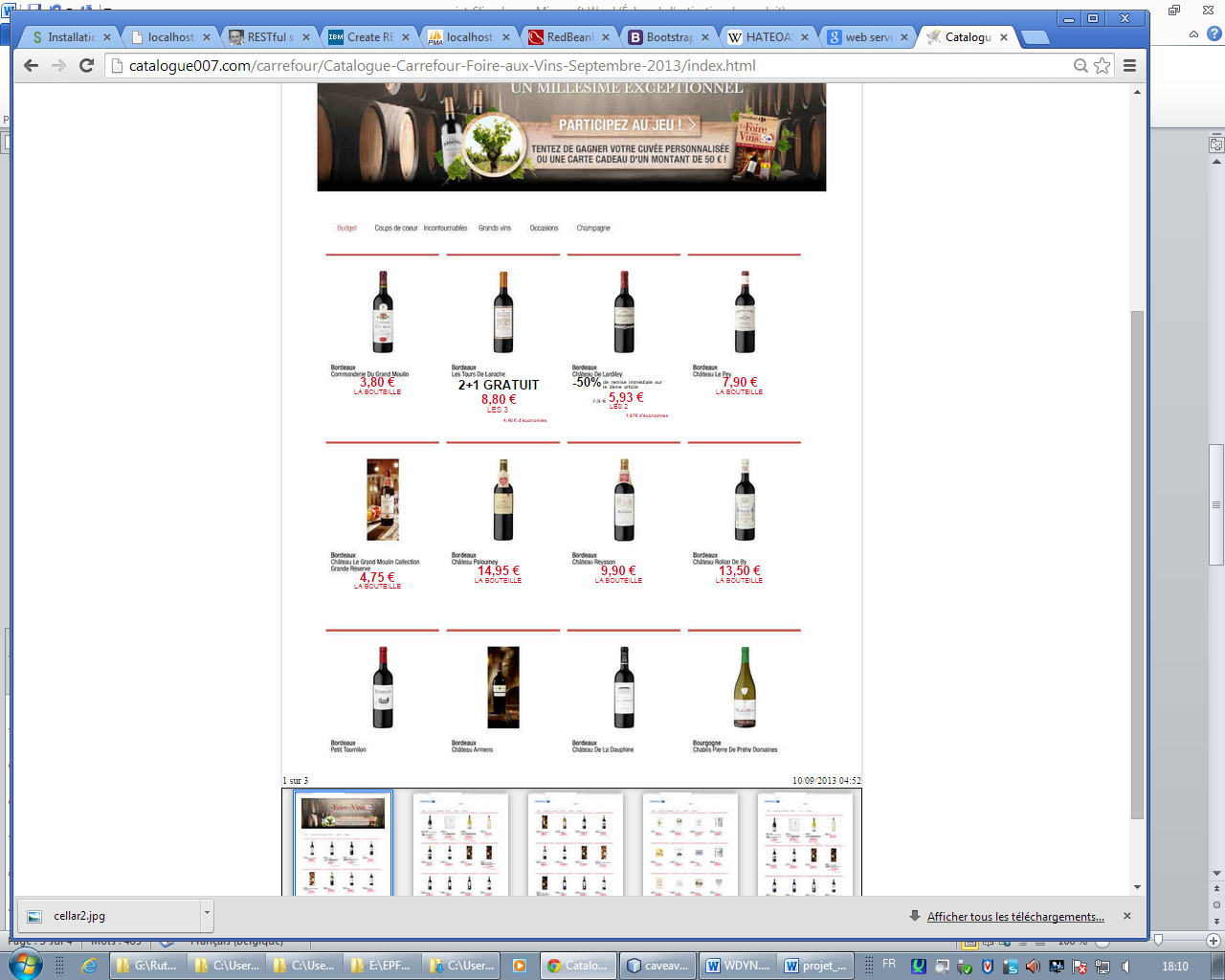
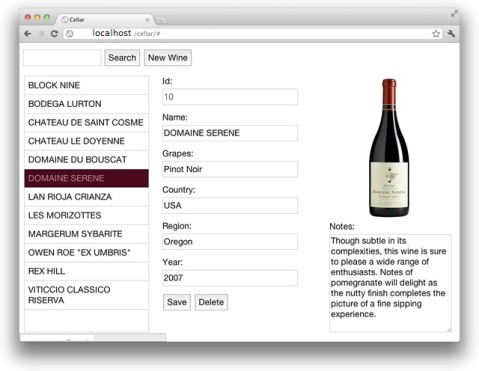
# **Projet Web I – Initiation aux framework**

L'enjeu de ce projet consiste à travailler avec plusieurs solutions open source afin de développer rapidement une application Web qui répond à des publics-cibles différents. Dans le cadre du projet, vous devrez utiliser les outils suivants:

* Slim framework (framework PHP)
* Twig (moteur de template)
* RedBeanPHP (ORM)
* Bootstrap (Framework HTML, CSS, JS)
* jQuery (Librairie JS)

Il vous est demandé de réaliser une application qui d'une part implémente une **API RESTful**, laquelle pourra être interrogée en tant que *web service* par des sites tiers, d'autre part génère les pages d'un *catalogue* de produit.

Il y a donc trois développements en un: le **catalogue**, consultable sur le site de l'application développé, et le **service Web**, qui fournira des données à l'**application cliente**.

JSON

## Étapes de développement

* Analyser les cas d'utilisation et les séquences système qui devront être implémentés
* Identifier les outils nécessaires (framework, librairies,…)
* Télécharger et installer les différents outils
* Intégrer chaque outil dans le framework principal
* Récupérer, classer et/ou charger les ressources de contenu gérées par l'application
* Commencer à développer

## Partie I – Framework PHP / BD MySQL / JSON + jQuery & AJAX

### Frontend – jQuery/JQueryUI/Ajax

Le développement frontend consiste à réaliser une application Web one-page, cliente du service Web précédent.

L'application que vous développerez est une application qui gère une cave à vin. Il sera possible de rechercher des bouteilles de vin, d'ajouter un vin à sa cave, de modifier et de supprimer des bouteilles.

### Backend – Slim Framework

Le backend est un service Web qui doit fournir les données à l'application cliente sous forme d'objet JSON. Votre API REST devra répondre aux commandes suivantes:

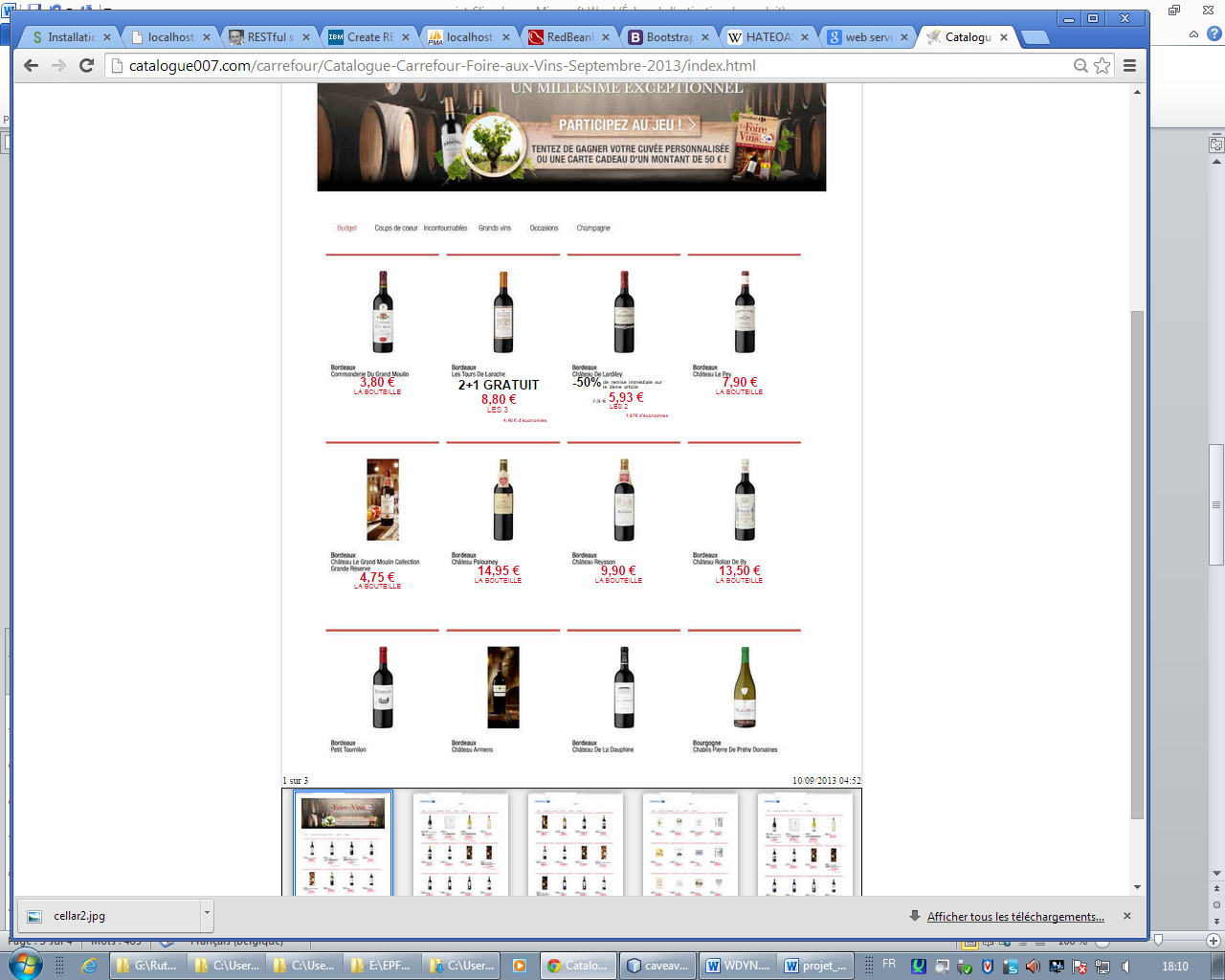
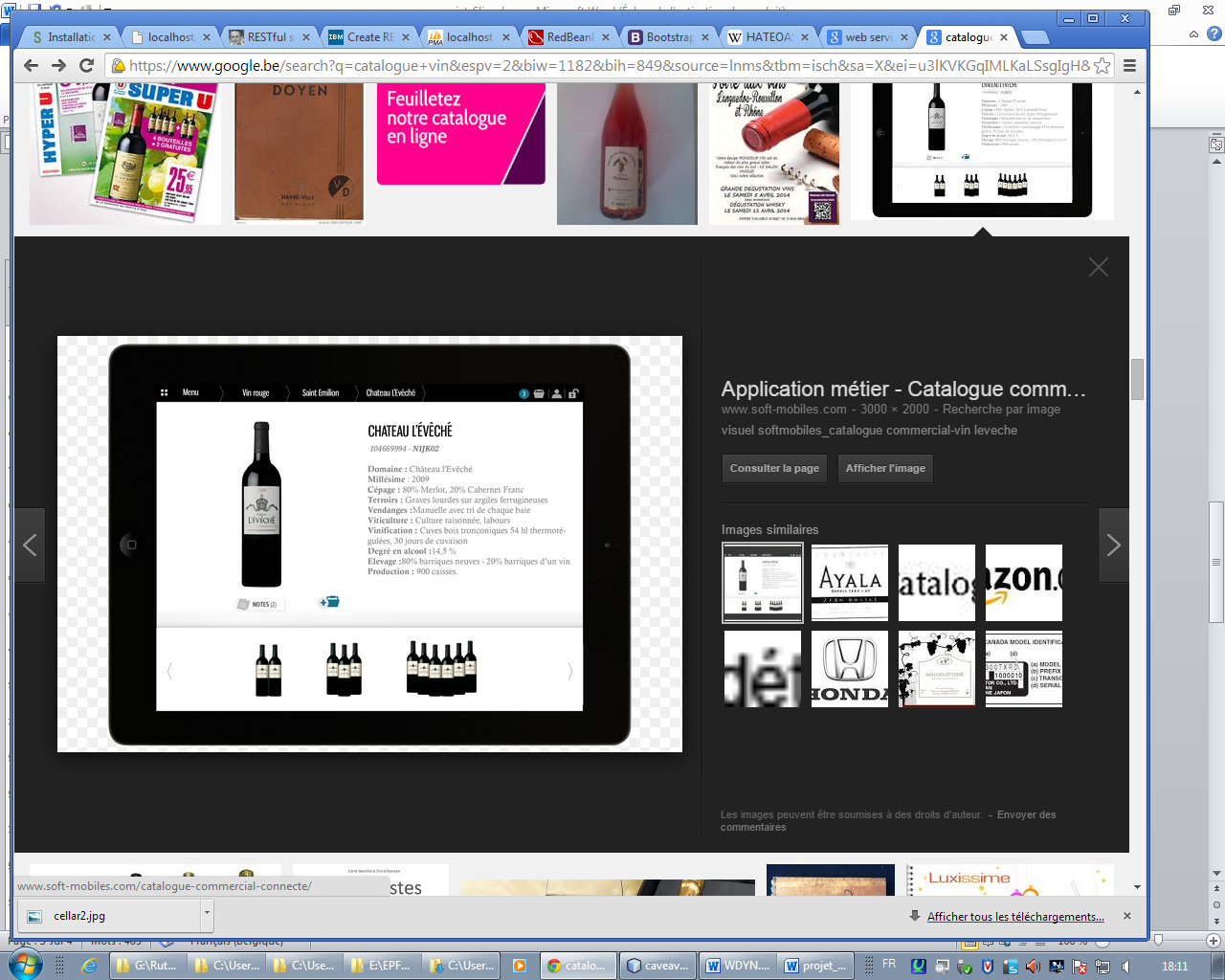
|  |  |  |
| --- | --- | --- |
| **Méthode** | **URL** | **Action** |
| GET | /api/wines | Retrouve tous les vins de la base de données |
| GET | /api/wines/search/Chateau | Recherche les vins dont le nom contient ‘Chateau’ |
| GET | /api/wines/10 | Retrouve le vin dont l'id == 10 |
| POST | /api/wines | Ajoute un vin dans la base de données |
| PUT | /api/wines/10 | Modifie les données du vin dont l'id == 10 |
| DELETE | /api/wines/10 | Supprime le vin dont l'id == 10 |

Il existe de nombreux framework PHP sur le Web. Le choix de Slim est avant tout pédagogique car il est léger et nous permettra de nous concentrer sur l'essentiel avant d'aborder d'autres composantes des frameworks les plus exigeants. Raisons d'adopter Slim dans le cadre de ce projet:

1. Il est léger et se concentre principalement sur l'architecture REST;
2. Il supporte toutes les méthodes HTTP (GET, POST, PUT, DELETE,…) utiles pour la réalisation de ce projet.

Pour la partie catalogue, votre application devra produire des pages Web présentant une liste des bouteilles de vins. Cette liste devra pouvoir être triée, filtrée et paginée. En cliquant sur une des bouteilles de la liste, vous devrez afficher la fiche descriptive complète de la bouteille de vin choisie.

***Exemple de présentation***



Pour cette partie, vous êtes libre d'intégrer comme bon vous semble des outils de présentation et d'animation de la librairie jQueryUI, des feuilles de styles CSS du framework Bootstrap. Veillez néanmoins à implémenter les fonctionnalités minimales du catalogue, à savoir pagination, tri, filtre et sélection.

### Guidelines

#### Réalisation du service Web

* Créez une base de données nommée *cavavin* et y importer les données à partir du fichier SQL fourni (*cellar.sql*).
* Cherchez et testez dans *PhpMyAdmin* les différences requêtes SQL dont votre application aura besoin (SELECT, INSERT, UPDATE, DELETE). Exemples: retrouvez tous les vins, retrouvez tous les vins dont le nom contient…, dont l'id vaut…
* Dans un script nommé index.php placé à la racine de votre site, écrivez la structure de base de Slim.
* Entre l'instanciation de l'objet App de Slim et l'appel à la méthode run(), définissez toutes les routes que le service doit traiter.
* Pour chaque routes, définissez comme traitement l'affichage d'une chaîne de caractères témoin afin de tester vos routes.
  + Exemples:
    - La route GET /api/wines affichera "Affichage de tous les vins"
    - La route GET /api/wines/10 affichera "Affichage du vin d'id 10"
    - La route DELETE /api/wines/10 affichera "Suppression du vin d'id 10"
    - etc.
* Les fonctions devront retourner une chaîne de caractères au format JSON.

#### Réalisation du client Web

* Créez la page Web en définissant l'ossature du document HTML5 et en vous référant à la maquette donnée en illustration.
* Incorporez la librairie jQuery.
* Incorporez le framework Bootstrap.
* Conformez votre prototype HTML au système de grille Bootstrap et ajoutez les classes CSS de mise en forme (boutons, formulaires).
* Incorporez jQueryUI.
* Ajoutez les classes CSS et le script permettant de rendre les éléments de la liste de vins sélectionnables (Selectable).
* Dans un nouveau script JavaScript, définissez les traitements liés aux actions de l'utilisateur: clic sur chaque bouton, sélection d'un vin de la liste.

## Partie II – Framework PHP / BD MySQL via ORM RedBeanPHP

Dans cette partie, vous allez déléguer l'accès et le traitement des requêtes vers la base de données à l'ORM ReadBeanPHP.

## Partie III – Moteur de template / Bootstrap & jQuery