# **Projet Web HTML5 – Développement d’une SPA**

Dans ce projet, vous allez réaliser une *Single Page Application* (SPA) en HTML5, CSS3 et JavaScript. L'enjeu de ce projet consiste à exploiter les spécificités avancées du langage JavaScript afin de gérer les interactions avec l’utilisateur et de communiquer avec une API Restful pour l’accès aux données. Pour ce faire, vous intégrerez divers progiciels afin de faciliter la fabrication du produit logiciel et la gestion du projet. Vous devrez également gérer une arborescence plus complexe afin d’organiser les dossiers et fichiers de votre application et de ses dépendances. Enfin, vous utiliserez d’une part un gestionnaire de projet, d’autre part un système de *versioning* afin de gérer les évolutions de votre développement.

Dans le cadre du projet, vous devrez utiliser les outils suivants :

* Bootstrap Twitter (framework HTML, CSS, JS) – <https://getbootstrap.com/>
* jQuery (librairie JS) – <https://jquery.com/>
* jQueryUI (librairie de widgets JS) – <https://jqueryui.com/>
* Trello (gestionnaire de projet) – <https://trello.com/>
* 3 librairies JS au choix – *Voir liste à la fin de ce document*

## Expression des besoins

Vous devez réaliser le site client ***MonCellier***, accessible sur *moncellier.localhost/index.htm*, qui consomme le service web d’une API Restful. et permet à tout utilisateur de consulter le catalogue des vins, et aux utilisateurs authentifiés de modifier les données personnelles via un formulaire optimisé front-end.

## Méthodologie et phases du projet

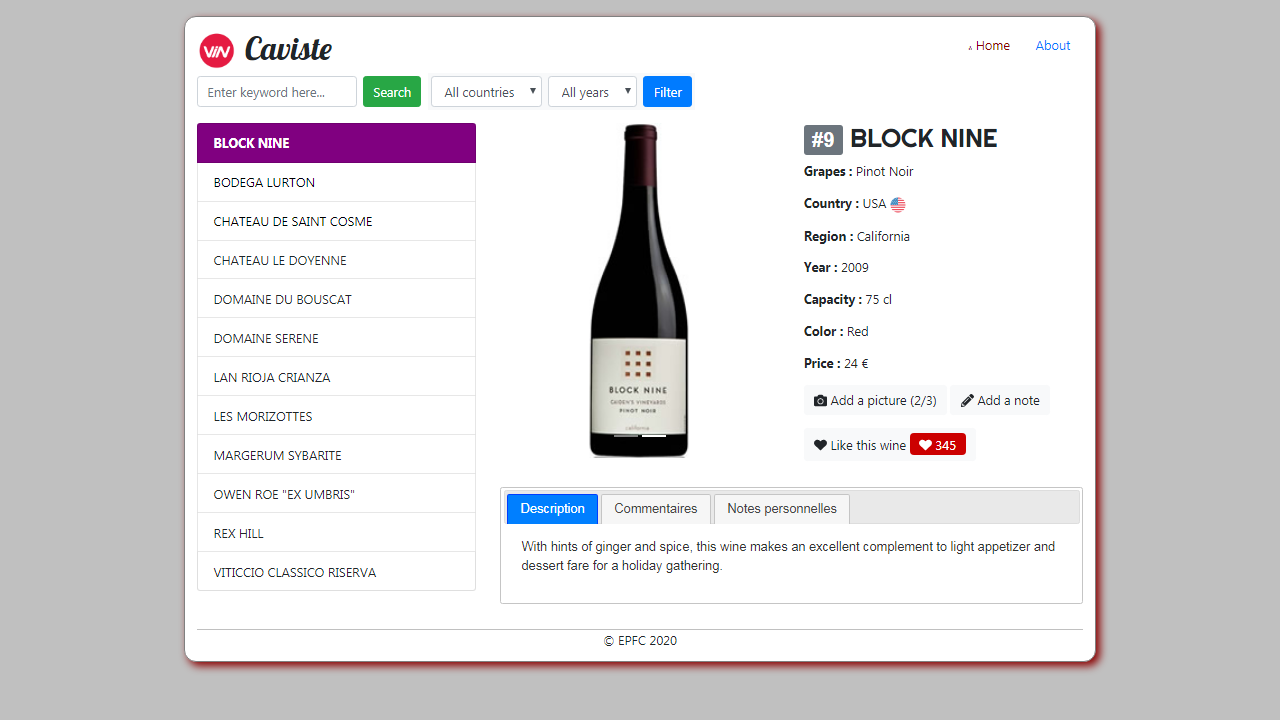
* Identifier les acteurs et leurs fonctionnalités sur base de l’expression des besoins
  + Définition du périmètre du projet
  + => Rédaction du cahier de charges fonctionnel
* Analyser les cas d'utilisation et les séquences système qui devront être implémentés
  + Rédaction du diagramme de cas d’utilisation
  + Rédaction des scénarios
  + Rédaction des diagrammes de séquence système
* Identifier les outils nécessaires (EDI, logiciel graphique,… / framework, librairies,…)
  + Recherche de progiciels (dans le cas présent, choix spécifiques à la demande)
* Installer et configurer les différents outils de développement
* Identifier les contenus gérés par l’application
  + Liste des contenus (type, accessibilité, usage + politique de stockage)
* Identifier les éléments de conception graphique
  + Réaliser le logo, les trames et les maquettes
  + Choisir les polices de caractères (Web font)
* Commencer à développer

## Framework HTML *Twitter Bootstrap* + librairies jQuery/JQueryUI/Ajax

Le développement frontend consiste à réaliser l’application Web one-page, *moncellier.localhost/index.htm*, qui sera cliente du service Web c*aviste/api*.

L'application ***MonCellier*** devra gérer une cave à vin. Il sera possible pour tout utilisateur d’afficher la liste des bouteilles de vin, de sélectionner un vin pour en afficher le détail, de rechercher des vins par nom, de trier la liste par cépage (*grapes*) et de filtrer la liste par pays et par année. Par ailleurs, un utilisateur authentifié pourra ajouter, modifier et supprimer ses propres commentaires pour chaque bouteille de vin, ainsi que modifier et supprimer une note personnelle et des données supplémentaires.

Ci-dessus un exemple incomplet de maquette de l'application cliente *MonCellier*.



Dessinez votre propre maquette fonctionnelle en incluant logo, menu, fonctionnalités de tri, filtre, sélection, sous-pages, formulaire, etc. N’oubliez pas de prévoir un design responsive pour les tablettes et les téléphones mobiles.

## Consommer les données JSON d’un Web Service Restful

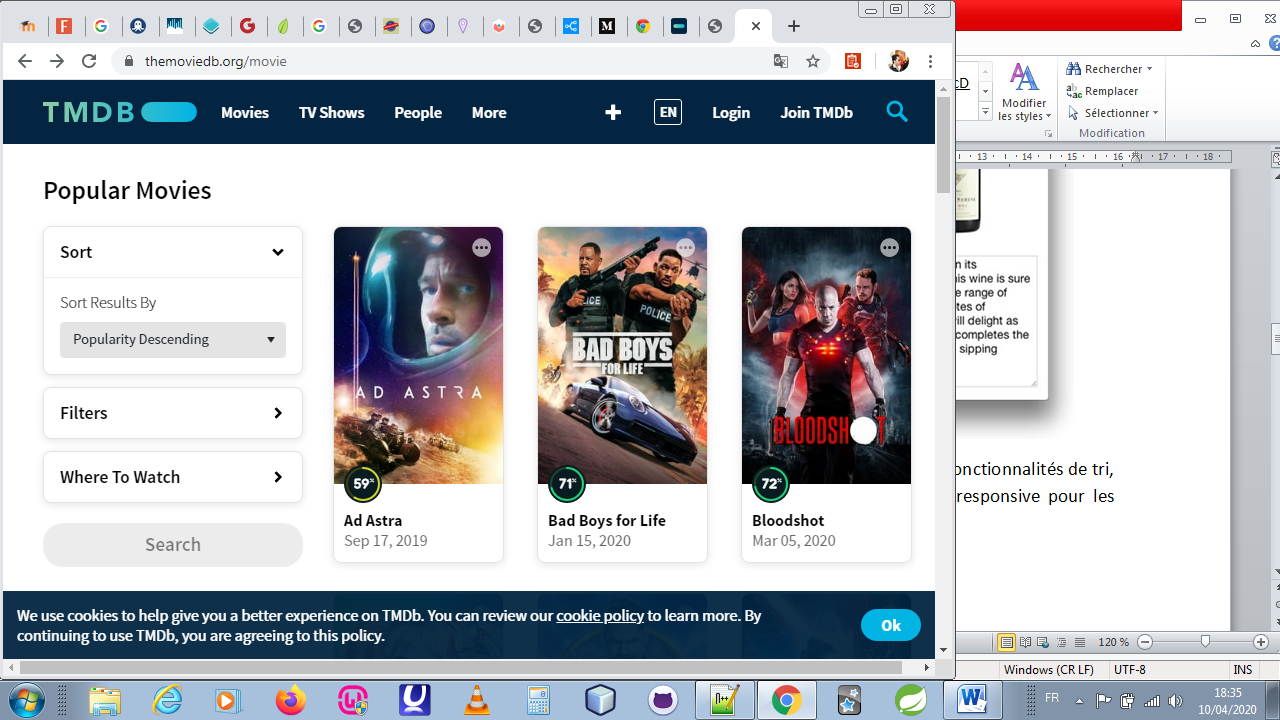
Le backend *caviste/api* distribue un service Web qui fournit les données à l'application cliente au format JSON ou XML. Cette API REST répond aux commandes suivantes :

|  |  |  |
| --- | --- | --- |
| **Méthode** | **URL** | **Action** |
| GET | /api/wines | Retrouve les vins de la base de données |
| GET | /api/wines/10 | Retrouve le vin dont l'id == 10 |
| GET | /api/wines?key=country&val=France&sort=year | Filtre les vins de France triés par année |
| GET | /api/wines/search?keyword=Chateau | Recherche les vins dont le nom contient ‘Chateau’ |
| GET | /api/wines/10/comments | Retrouve les commentaires du vin 10 |
| GET | /api/wines/countries | Retrouve les différents pays |
| GET | /api/wines/10/likes-count | Retrouve le nombre de likes du vin 10 |
| GET | /api/users/5/likes/wines | Retrouve les vins préférés de l’utilisateur 5 |
| PUT | /api/wines/10/like  { "like" : true|false } | Ajoute ou retire le vin 10 parmi ses préférés |
| POST | /api/wines/10/comments  { "content" : "some content" } | Ajoute un commentaire pour le vin 10 |
| PUT | /api/wines/10/comments/3  { "content" : "some new content" } | Modifie le commentaire 3 du vin 10 |
| DELETE | /api/wines/10/comments/3 | Supprime le commentaire 3 du vin 10 |
| POST | /api/wines/10/pictures  FormData | Ajoute une image pour le vin 10 |
| DELETE | /api/wines/10/pictures/2 | Supprime l’image 2 du vin 10 |

### Guidelines pour la réalisation du client Web

* Créez la page Web en définissant l'ossature du document HTML5.
* Incorporez les librairies
  + les librairies *jQuery* et *jQueryUI*,
  + le framework *Bootstrap*,
  + les polices de caractères *Google Fonts*,
  + les icônes *FontAwesome*,…
* Implémentez votre prototype HTML en vous basant sur votre maquette fonctionnelle et en vous conformant au système de grille de *Bootstrap*.
* Ajoutez les composants visuels (widgets) en utilisant ceux proposés par *Bootstrap*, *jQueryUI* ou d’autres librairies (boutons, formulaires et autres composants).
* Incorporez un script JavaScript externe dans lequel vous définirez l’URL de base du service Web.
* Faites une requête Ajax GET vers l’API afin de récupérer la liste des vins.
* Affichez les vins dans la page Web.
* Définissez les traitements liés aux actions de l'utilisateur (clic sur chaque bouton, sélection d'un vin de la liste, etc.).

Exemples de design



### Liste des librairies JavaScript

* Graphique
  + Chart.js – <https://www.chartjs.org/>
* Data visualisation
  + D3.js – <https://d3js.org/>
* Animation
  + Granim.js – <https://sarcadass.github.io/granim.js/>
  + Animate On Scroll Library – <https://michalsnik.github.io/aos/>
  + Slick (Carroussel) – <https://kenwheeler.github.io/slick/>
* Tableau de données
  + Datatable.js – <https://datatables.net/>
* Gestion de formulaire
  + Parsley (Validation) – <http://parsleyjs.org/>
  + Algolia Places (autocomplétion) – <https://community.algolia.com/places/>
* Notification
  + Popper.js – <https://popper.js.org/>
* Data Access Object
  + TaffyDB (JavaScript Database functionality) – <http://taffydb.com/>
* Cartographie
  + Open Layers – <https://openlayers.org/>
  + Leaflet (Cartographie) – <https://leafletjs.com/>