

---

# CPE Lyon - 3ETI - 2021-2022

## Techniques et Langages du Web

### Sujet du projet

---



Le sujet ci-dessous présente le travail attendu lors des six séances de projet. Certains termes pourront vous paraître obscurs lors des premières séances ; ils deviendront clairs au fur et à mesure que nous avancerons dans le cours. **Il est cependant fortement conseillé de lire le sujet dans son intégralité.** Les seuls langages autorisés sont ceux vu en cours. En particulier, l'utilisation de frameworks (Bootstrap, Angular...) n'est pas permise.

## I Cahier des charges

Le but de ce projet est de réaliser un site web qui vous a été commandé par une entreprise de **cadeaux personnalisés** (macarons personnalisés, pizzas sur mesure, t-shirts, mugs, vinyles, caleçons, masques chirurgicaux...). Ce produit doit comporter plusieurs gammes (en coffrets, par lots, versions premium ou non...), chacune étant personnalisable par le client directement sur le site web. La MOA (maîtrise d'ouvrage) a exprimé un certain nombre d'exigences fonctionnelles *minimales* (libre à vous de faire preuve d'imagination et d'ajouter des fonctionnalités!). Portez-y la plus grande attention, car **c'est le respect de ces exigences qui déterminera en grande partie votre note** (le client est roi!).


1. Le site doit comporter **au minimum** les pages suivantes :
  - une **page principale** qui présente les différentes gammes du produit proposé, disposées selon une grille
  - une page de **Personnalisation**
  - une page de **Contenu du panier / Commande**
  - une page **A propos & Contact**
2. Le choix d'un article sur la page principale (**en cliquant sur sa photo**) doit amener à la page de personnalisation (**votre site ne doit bien comporter qu'une seule page de personnalisation, et pas une page par article proposé!**). Sur cette page, on retrouve différentes informations sur le produit sélectionné (nom, photo...), ainsi qu'une **zone de personnalisation** (cf. section suivante) (ajout de logo, de textes... soyez imaginatifs!) avec un aperçu dynamique du produit final. On dispose également d'un formulaire permettant de choisir une quantité, d'ajouter le produit personnalisé dans le panier, ou encore de réinitialiser la personnalisation. L'ajout d'un produit au panier conduit à la page **Contenu du panier**
3. La **zone de personnalisation** repose sur **HTML Canvas** (cours 5), afin de superposer l'aperçu du produit de base avec le texte ou le motif sélectionné par l'utilisateur. L'aperçu doit se mettre à jour automatiquement au changement du champ de personnalisation, sans clic sur un bouton.
4. La page **Contenu du panier**, comme son nom l'indique, reprend les informations des différents produits ajoutés, tels qu'ils ont été personnalisés par l'utilisateur. On doit avoir la possibilité de supprimer un élément du panier. Cette page permet également de **valider la commande**, après avoir complété un formulaire. Ce formulaire de commande doit comporter les éléments suivants :
  - nom\*
  - prénom\*
  - adresse mail\*

- téléphone\*
- les champs permettant de spécifier l'adresse de livraison\*
- date de livraison souhaitée\*
- une zone de texte pour laisser une demande de renseignement
- un récapitulatif de la commande (**voir section 6**)
- un bouton de soumission et un bouton de remise à zéro
- tout autre élément que vous jugeriez utile

 Les éléments marqués d'une \* doivent obligatoirement être remplis par l'utilisateur avant soumission du formulaire.

5. **Calcul du prix** : plusieurs éléments entrent en compte dans le calcul du prix de la commande :

- évidemment, le prix des articles personnalisés eux-mêmes ;
- au-delà de 10 articles similaires commandés, une remise de 10% est appliquée sur cet article ;
- **l'entreprise est située à Lyon** (par exemple, dans la Tour de la Part-Dieu) ; les frais de livraison sont gratuits pour les clients situés à moins de 20 km ; au-delà, on applique la formule  $5 + 0.07 \times d$  € (où  $d$  est la distance en km entre l'entreprise et la ville de livraison) ;
- un surplus de 8€ est appliqué pour les livraisons express si le client choisit une date de livraison dans les 72h suivant la commande.

 Pour obtenir la distance entre les deux adresses, vous utiliserez l'API *Directions* de Mapbox : <https://docs.mapbox.com/api/navigation/directions/>

6. **Récapitulatif de la commande** : il s'agit d'une zone d'information détaillant les différents éléments pris en compte dans le prix final (voir section 5). Ces éléments doivent être mis à jour **automatiquement** (par exemple lorsqu'on modifie la quantité d'un article).


7. De base, le modèle de données des produits est très simple : chaque produit est caractérisé par un titre et un tarif de base (sans options). Vous pourrez cependant enrichir librement ce modèle (en ajoutant le prix des personnalisations optionnelles, comme des lignes de texte supplémentaire, des goûts en édition limitée, etc.).

8. La **Page de contact** doit comporter les éléments suivants :

- adresse (postale) de l'agence de voyage
- numéro de téléphone ; un clic sur ce numéro doit ouvrir automatiquement l'application associée à la composition d'appels téléphoniques si l'utilisateur en dispose (par exemple Skype sur PC, ou le téléphone sur smartphone)
- bouton d'envoi d'un mail ; l'objet du mail doit être prérempli avec « Demande de renseignements » et le corps du message doit commencer par « Bonjour, je souhaiterais obtenir des renseignements sur ».

9. Sur toutes les pages on doit retrouver :

- un menu de navigation,
- un pied de page, contenant au minimum les informations légales de l'entreprise (nom et adresse postale),
- un bouton de retour au sommet de la page, qui n'apparaît que lorsque l'utilisateur fait défiler une page trop haute pour tenir sur l'écran.

 Naturellement, vous veillerez à **ne pas dupliquer** de code entre toutes les pages !

10. **Filtrage** : sur la page principale proposant les différents produits, l'utilisateur doit disposer de filtres (prix min / max, coloris, etc.). Les produits ne correspondant pas aux filtres doivent-être **automatiquement** masqués, sans clic sur un bouton de validation des filtres.

💡 On n'exige pas de fonction de **tri** (par exemple par prix croissant).

### Eléments optionnels :

11. En dehors des éléments précédents, qui sont imposés, l'entreprise vous laisse le champ libre pour le style. Il est évident qu'une page HTML nue, sans style, serait largement insuffisante. Laissez-donc libre cours à votre imagination, habillez votre site. Vous pouvez ajouter des effets (par exemple au survol des produits), mais conservez un site agréable à visiter. Nous n'attendons pas un site de qualité professionnelle, même si l'aspect esthétique sera pris en compte, **nous serons surtout attentifs sur le respect du cahier des charges, ainsi que sur la qualité et la clarté de votre code** !
12. **Déploiement** : votre site est terminé, alors pourquoi ne pas le mettre en ligne ?! Vous pouvez le faire très facilement et gratuitement en utilisant des solutions d'hébergement comme [Firebase](#) ou [Netlify](#).
13. **Authentification** : l'utilisateur peut s'authentifier sur le site, à l'aide d'un formulaire simple (nom d'utilisateur et mot de passe).  
Comme vous ne disposez pas de base de données ni de serveur, vous simulerez l'acceptation ou le rejet de connexion (3 ou 4 comptes suffiront). L'utilisateur doit pouvoir se déconnecter.
14. **Historique de commandes** : le site peut être doté d'une fonctionnalité "Historique des commandes" permettant à un utilisateur de consulter ses commandes passées.
15. Le site ne doit pas être nécessairement *responsive*, c'est-à-dire s'adapter à la taille de l'écran sur lequel il est consulté (en particulier sur les écrans de petite taille type smartphone). Cependant, une attention toute particulière sera portée au respect des normes d'*accessibilité*.

## II Organisation du projet

Vous travaillerez en **binôme**. Le projet est à rendre pour le **mardi 9 novembre 2021 à 18h** (heure de fermeture automatique du dépôt). **Aucun rendu ne sera possible après cette date**, et le code qui sera évalué sera celui présent sur le dépôt à ce moment-là.

Chaque binôme dispose d'un dépôt sur Github Classroom. Vous devez commencer par créer un compte GitHub si vous n'en avez pas déjà un, puis une équipe à l'adresse associée à votre groupe (voir sur e-campus) ; **pour des raisons pratiques, cette équipe doit impérativement être désignée par les deux noms de famille des membres du binôme, sous la forme NOM1-NOM2**. L'idée est donc de mettre l'accent sur le travail d'équipe, la collaboration, et le partage de code à l'aide de **Git** : chaque membre du binôme travaille de son côté sur une fonctionnalité précise **en accord avec l'autre membre**, puis **pousse** son code sur le dépôt partagé pour la mise en commun (*fusion*).

Attention à bien gérer votre temps et votre collaboration, ce sont deux composantes fondamentales de la gestion et de la réussite d'un projet !

### Préparation de l'environnement de développement

Sur vos ordinateurs personnels, la première chose à faire est donc de télécharger et installer **Git** (<https://git-scm.com/>) si vous ne l'avez pas déjà ; Git est déjà installé sur les postes de l'école.

Pour le développement de votre site, nous vous conseillons d'utiliser l'éditeur de texte **Visual Studio Code** de Microsoft (ne pas confondre avec *Visual Studio*, qui est un environnement de développement beaucoup plus complet), disponible gratuitement, multi-plateformes et qui fournit une interface graphique pour Git. VS Code dispose d'un *Marketplace* dans lequel vous pourrez trouver des centaines d'extensions pour différents langages de programmation. Nous vous conseillons d'installer les extensions suivantes :

- **French Language Pack for VS Code**, si vous souhaitez disposer de l'interface en français ;
- **W3C Web Validator**, qui vous indique les différentes erreurs contenues dans votre code HTML ;
- **Live Server** ; cette extension vous sera utile lorsque vous utiliserez des fonctionnalités qui nécessitent un serveur. Pour l'utiliser, ouvrez avec VS Code un **dossier** contenant un fichier HTML, et un bouton **Go Live** apparaît dans la barre d'état (de VS Code). Un clic sur ce bouton ouvre une fenêtre de navigateur, qui s'actualisera automatiquement dès que vous modifierez **et enregistrerez** un fichier du dossier ;
- **Git Graph**, **Git History** et **Git Lens** qui facilitent grandement l'utilisation de Git avec VS Code.

Il est également vivement conseillé d'installer, sous Chrome ou Firefox, l'extension **Web Developer**.

### III Etapes du projet

Cette section vous donne le travail attendu à la fin de chaque séance, ce qui doit vous permettre d'évaluer votre avancement et éventuellement votre retard.

#### Séance 1 : HTML - Structure et contenu du site

Lors de la première séance, mettez en application la méthodologie de conception présentée en cours. En particulier, identifiez les différentes tâches et répartissez-vous le travail équitablement dans le binôme. Une fois que vous aurez une bonne compréhension du sujet et du résultat attendu, passez au prototypage (soit papier, soit avec des outils en ligne comme <https://mockflow.com/>, [draw.io](https://draw.io), ou encore avec LibreOffice Draw, Powerpoint ou LibreOffice Impress).

Une fois ce travail réalisé, rassemblez les ressources (photos, icônes...); vous pourrez trouver des photos gratuites sur [pixabay.com](https://pixabay.com) ou [unsplash.com](https://unsplash.com), et des icônes sur [thenounproject.com](https://thenounproject.com) ou [fontawesome.com](https://fontawesome.com).

⚠ N'utilisez que des illustrations libres de droit !

💡 Inutile de partir sur un site proposant 50 produits à personnaliser ! 4 ou 6 suffiront largement pour couvrir les besoins du projet.

**Important :** consacrez cette séance à décrire la *structure* de vos pages, et pas encore leur mise en forme, que l'on abordera plus tard.

##### A l'issue de cette séance :

- vous devez disposer de toutes vos ressources graphiques (images, photos, icônes...)
- la structure de votre site (code HTML) doit être terminée ;
- tous les avertissements et erreurs signalés par le validateur doivent être corrigés ;
- vous veillerez à ce que les balises sémantiques les plus adéquates soient utilisées ;
- la navigation sur le site doit être opérationnelle (le passage correct d'une page à une autre) ;
- les éléments obligatoires du formulaire doivent être fonctionnels ;
- l'en-tête et le pied de page ne sont présents que sur la page d'accueil

💡 Tant que nous n'avons pas abordé la manière de stocker des données, saisissez vos textes "en dur", c'est-à-dire directement dans le code HTML. Il s'agit évidemment d'une pratique déconseillée, et nous verrons au fur et à mesure des cours de meilleures méthodes.

#### Séance 2 : CSS - mise en forme, style

Lors de cette séance, vous allez donner à votre site son *aspect* quasi final. De nombreuses fonctionnalités seront bien sûr inactives ou manquantes à ce stade, mais on doit déjà se faire une bonne idée de ce à quoi ressemblera la version finale.

##### A l'issue de cette séance :

- le style de votre site doit être quasiment terminé (polices de caractères, style des boutons, couleurs...) ;
- les destinations sur la page d'accueil doivent être disposées selon une grille à l'aide d'un **grid layout** comme présenté en cours ;
- les plus rapides peuvent également essayer de rendre leur site *responsive* pour qu'il soit consultable sur un smartphone.

##### 💡 Conseils :

- vous trouverez sur <https://www.w3schools.com/howto/default.asp> de très nombreux tutoriels très simples ("How-to")

- les sites <https://html-css-js.com> et <https://grid.layoutit.com/> proposent de nombreux outils intéressants, comme des générateurs automatiques d'effets, de tableaux ou de dispositions (layouts)
- *Google Fonts* propose de nombreuses polices de caractères que vous pouvez utiliser facilement sur vos sites (voir [https://www.w3schools.com/howto/howto\\_google\\_fonts.asp](https://www.w3schools.com/howto/howto_google_fonts.asp))

## Séance 3 : JavaScript (1/3)

A priori, jusqu'ici tous les contenus de votre site sont écrits « en dur » dans les fichiers HTML. C'est évidemment une très mauvaise pratique car :

- les fichiers HTML devraient être réservés essentiellement à la description de la *structure* de votre site ;
- cela engendre des fichiers HTML longs et difficiles à maintenir ;
- à chaque fois que vous voudrez / devrez modifier le prix ou une illustration d'un produit, vous devrez modifier le fichier HTML et le republier.

L'idée est de rendre votre site web *dynamique* : le contenu des pages doit être lu et ajouté *dynamiquement*, quel que soit le nombre de produits proposés. Dans l'idéal, vous devriez stocker toutes les informations relatives à vos produits dans une **base de données**. L'utilisation d'une base de données nécessitant des connaissances qui dépassent le cadre de ce cours, nous allons *simuler* une base de données en utilisant dans un premier temps des tableaux d'objets JavaScript. Nous ferons évoluer ces tableaux lors de la séance suivante.

### A l'issue de cette séance :

- vous devez disposer d'une **classe** (au sens JavaScript !) **Produit**, avec les attributs caractérisant un produit et toutes les méthodes nécessaires ;
- d'un tableau d'instances / objets de la classe **Produit**
- un clic sur un produit de la page d'accueil doit mener à la page de personnalisation, automatiquement adaptée pour le produit en question (le nom du produit doit apparaître, et le prix de l'objet personnalisé doit correspondre à **ce** produit) ;
- le calcul du prix dans le formulaire de commande doit être effectif, **hormis les éléments qui nécessite un appel à une API externe**.

### Comment savoir sur quel produit l'utilisateur a cliqué ?

Pour retrouver, depuis la page de personnalisation, le produit sur lequel le client a cliqué sur la page d'accueil, vous utiliserez la méthode présentée en cours :

- **sur la page principale**, pour chaque produit, rajoutez dans l'adresse du lien vers la page de personnalisation un identifiant, sous la forme `id=identifiant` précédé par le caractère '?'. L'identifiant doit bien sûr correspondre à l'index correspondant dans votre tableau de produits. Exemple : `<a href="personnalisation.html?id=7">`
- **sur la page Personnalisation**, vous pourrez alors récupérer l'identifiant contenu dans l'URL de la façon suivante :  

```
let produit_id = new URLSearchParams(window.location.search).get("id").
```

## Séance 4 : JavaScript (2/3)

Le but de cette séance est de poursuivre la transformation de votre site « statique » en un site « dynamique ». Normalement, à ce stade, de nombreuses informations sont encore dupliquées : les données de vos produits sont probablement présentes à la fois dans le code HTML et dans le code Javascript, et toute une partie du code HTML est dupliquée pour chacun de vos produits. Là aussi, il s'agit de mauvaises pratiques : si on doit modifier une information sur un produit, on n'aimerait le faire qu'une seule fois ; et si on veut modifier le code HTML associé aux produits, on aimerait là aussi éviter de faire la même tâche plusieurs fois (cela permet de gagner du temps et d'éviter les erreurs!).

A l'issue de cette séance :

- toutes les informations « en dur » concernant vos destinations, figurant dans les fichiers HTML, doivent avoir disparu. A la place, vous devez utiliser un *template* comme présenté en cours ;
- les en-têtes et pieds de page doivent être ajoutés dynamiquement sur chaque page ;
- les appels aux API doivent être fonctionnels, et les éléments qui les utilisent doivent être pris en compte dans le calcul du prix ;
- toutes les **données** concernant les produit doivent être déportées dans un fichier **JSON** (c'est lui qui fait office de « base de données » ), qu'on lit à l'aide d'un appel AJAX ou *fetch* ; il ne doit donc plus subsister aucune donnée de produit en dur, ni dans les fichiers HTML, ni dans les fichiers JavaScript.

## Séance 5 : JavaScript (3/3)

Il est temps de passer à la gestion du panier. Celui-ci permet de conserver les informations quand on passe d'une page à l'autre, ou même si on quitte le site pour y revenir plus tard. Pour cela, vous utiliserez les fonctionnalités de stockage (*local storage*, **session storage**) de votre navigateur.

A l'issue de cette séance :

- le panier doit être fonctionnel en utilisant le **storage** ;
- les filtres sur la page d'accueil doivent être fonctionnels (utilisation du **DOM**) ;
- le bouton « Retour en haut de page » doit être présent et fonctionnel sur toutes les pages.

## Séance 6 : Canvas

Lors de cette séance, vous allez développer le module de personnalisation des objets. Celui-ci peut consister en l'ajout d'une photo, l'écriture d'un texte, la modification d'une couleur, etc. sur une photo du produit à personnaliser.

Vous ajouterez également, dans la page de contact, un *iframe* contenant une carte Google Maps centrée sur l'adresse de l'entreprise.

A l'issue de cette séance :

- le module de personnalisation des objets est fonctionnel ;
- la page de contact contient une carte Google Maps ;
- l'ensemble du site est finalisé.