

Cégep de Sainte-Foy – Hiver 2023
Programmation Web Dynamique – 420-W21-SF

Travail Pratique 2

17 avril 2023

Préparé par
Benjamin Lemelin

1 Résumé

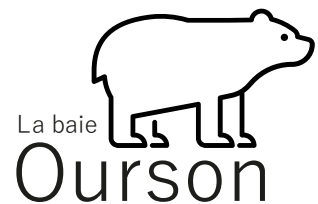
Créer un site web dynamique en PHP pour une entreprise spécialisée dans la vente de vêtements. Le site doit contenir un catalogue de produits, le détail de chaque produit et permettre des achats en ligne.

2 Conditions de réalisation

Valeur de la note finale	Contexte	Durée	Nombre de remises
30%	En équipe	4 semaines	2 remises

3 Mise en contexte

La compagnie de *La Baie d'Ourson* se spécialise dans la vente de vêtements pour enfants de 0 à 16 ans. Leur clientèle est surtout composée de parents ayant à cœur l'achat local, les vêtements étant fabriqués au Québec.



Durant ses premières années, *La Baie d'Ourson* ne prenait que des commandes par catalogue, façon *Sears*. Depuis, l'entreprise a énormément grossi et désire offrir un site de ventes en lignes. Ce site doit permettre à un client consulter des produits, de créer un compte, un panier et d'effectuer des commandes.

4 Tâches à réaliser

4.1. Création du projet

Copiez le projet fourni avec cet énoncé dans un sous-dossier nommé **tp2** à la racine de votre serveur. Il contient déjà les images dont vous aurez besoin ainsi que la configuration pour *Visual Studio Code*.

Vous trouverez aussi des scripts SQL pour la création de la base de données, dont voici le schéma :

user			product			order			order_item		
PK	id	INT	PK	sku	INT	PK	id	INT	PK	id	INT
U	email password first_name last_name shipping_address	VARCHAR(255) TEXT VARCHAR(255) VARCHAR(255) TEXT		name description price	VARCHAR(255) TEXT DECIMAL(15,2)	FK	user_id creation_date	INT DATETIME	FK FK	order_id product_sku quantity	INT INT INT

Ce script ajoute aussi quelques produits. Pour éviter des problèmes lors de la correction, assurez-vous de ne pas les modifier. Cela dit, vous pouvez en ajouter si vous le souhaitez.

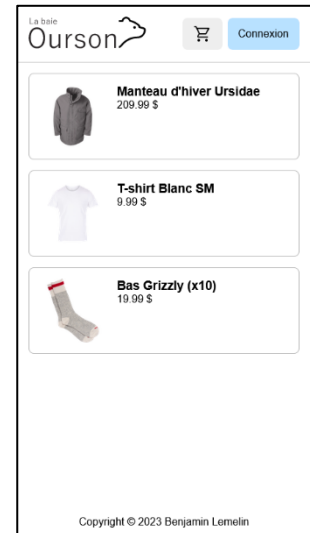
Sku	Nom	Description	Prix
612458	T-shirt Blanc SM	100% polyester. Facile à laver. Format Small.	9.99
154789	Manteau d'hiver Ursidae	Manteau pour l'hiver très chaud avec doublure.	209.99
887414	Bas Grizzly (x10)	Dix bas de laine gris. Très résistants.	19.99

4.2. Page d'accueil

Recréez la page d'accueil (**index.php**) en vous fiant à la maquette ci-dessous. La liste de produits doit être obtenue à partir de la base de données. Cliquer sur un produit amène à la page du produit.



Large (> 576px)



Mobile

i Information

Vous trouverez les couleurs et les dimensions des éléments de cette maquette dans la section [Couleurs et dimensions](#) de ce document. Ce seront les mêmes au travers de tout le site. Vous pouvez altérer le style visuel du site, pour autant que la structure reste la même.

4.3. Page de consultation d'un produit

Recréez la page de consultation d'un produit (**product.php**) en vous fiant à la maquette ci-dessous. Le formulaire de cette page permet l'ajout du produit au panier, avec la quantité indiquée.



Large (> 576px)



Mobile

💡 Astuce

Votre formulaire devrait contenir deux champs : un champ pour le *sku* du produit et un champ pour la quantité. Or, vous aurez rapidement remarqué que la maquette ne montre qu'un seul champ. En fait, le champ du *sku* est [un champ caché](#) prérempli par le serveur.

Vous aurez à faire usage de ce genre de tactique tout le long de ce travail.

Vous devez avoir une classe **Cart** représentant un panier. Cette classe doit être testée unitairement. Afin de simplifier, il s'agit de la seule classe à tester pour tout le travail. Enfin, n'oubliez pas de faire des [Form Objects](#) pour représenter et valider les données de vos formulaires.

⚠ Prudence

Les tests unitaires sont importants dans la mesure **où ils sont utiles**. Par exemple, il est inutile de tester unitairement vos *Form Objects*, car il n'y a pour ainsi dire aucune logique qui mérite d'être testée : le potentiel à l'erreur est pratiquement inexistant.

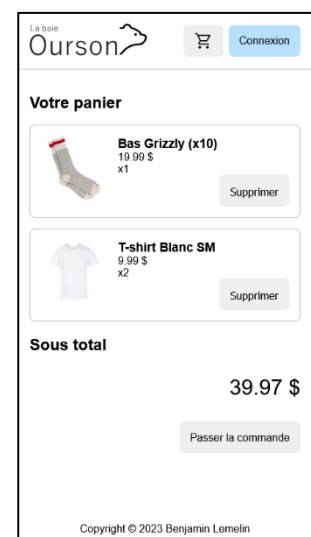
À l'inverse, le potentiel à l'erreur de la classe **Cart** est beaucoup plus élevé. Par exemple, avez-vous songé au cas où un utilisateur ajouterait un produit déjà présent dans son panier ? Dans cette situation, il faudrait simplement additionner les quantités au lieu de l'ajouter une seconde fois. C'est le genre de situation complexe qui mérite d'être bien testée.

4.4. Page de consultation du panier

Recréez la page de consultation du panier (**cart.php**) en vous fiant à la maquette ci-dessous. Cette page permet de retirer un produit du panier et de finaliser la commande. Le sous-total prend en considération la quantité de chaque produit, mais fait omission des taxes pour simplifier.

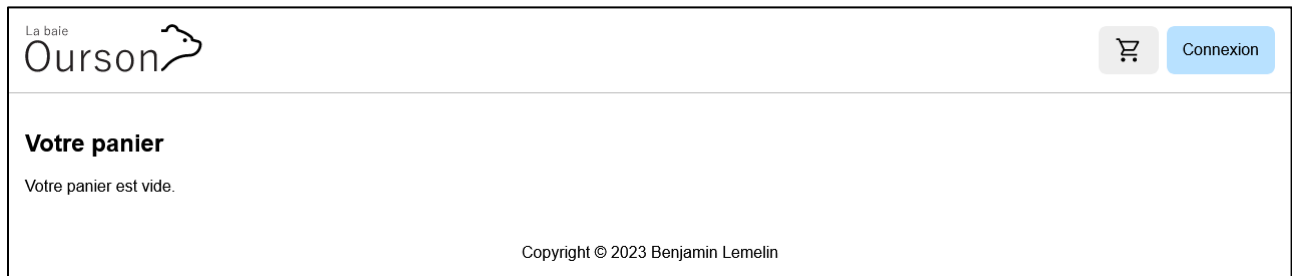


Large (> 576px)



Mobile

Lorsque le panier est vide, cette page n'affiche qu'un court message et rien de plus.



Le panier devrait être conservé dans la session, et non pas dans la base de données. Aussi, il ne devrait pas changer lorsque l'utilisateur se connecte à son compte. Cela dit, vous pouvez le vider lorsqu'il se déconnecte, mais cette décision reste à votre discrétion.

💡 Astuce

Le bouton « *Supprimer* » peut facilement être implémenté avec un formulaire contenant des champs cachés, tout comme sur la page de consultation d'un produit. Chaque produit aura donc son formulaire à lui.

4.5. Page de connexion

Recréez la page de connexion (**sign-in.php**) en vous fiant à la maquette ci-dessous. Pour se créer un compte, l'utilisateur doit cliquer sur l'hyperlien « *Créer un compte* » en bas du formulaire, menant à une autre page. Enfin, remarquez l'absence de barre de navigation.

Large (> 576px)

Mobile

Conservez l'utilisateur connecté dans la session. Pour [le déconnecter](#), vous n'aurez qu'à détruire la session (fonction `session_destroy()`) ou à l'enlever de la session (fonction `unset()`).

Voici les détails de ce formulaire :

Champ	Détails
Email <i>Texte</i>	<ul style="list-style-type: none">• Obligatoire.• Doit être une adresse courriel valide (voir filter_var).
Mot de passe <i>Texte masqué</i>	<ul style="list-style-type: none">• Obligatoire.• Non vide (pas seulement des espaces).

Le compte utilisateur, lorsque valide, doit être ajouté à la base de données. Une [saine gestion des mots de passe](#) est requise : n'oubliez pas de les chiffrer.

Si l'utilisateur n'existe pas, ou si le mot de passe est incorrect, affichez un message d'erreur, tel que montré ci-contre.

Prudence

Tous les formulaires doivent être validés, mais il n'est pas toujours nécessaire d'afficher un message d'erreur. Par exemple, bien que le bouton « Supprimer » du panier fasse partie d'un formulaire, tous les champs sont préremplis. Un utilisateur bien intentionné ne devrait donc pas causer d'erreur.

Pour ce travail, vous pouvez omettre l'affichage des messages d'erreur dans les situations où un utilisateur normal ne peut pas en provoquer. Cela dit, dans un véritable projet, ces erreurs seraient tout de même enregistrées dans un fichier journal, au cas où.

4.6. Page de création de compte

Recréez la page de création de compte (**sign-up.php**) en vous fiant à la maquette ci-dessous.

Large (> 576px)

Mobile

Voici les détails de ce formulaire :

Champ	Détails
Email <i>Texte</i>	<ul style="list-style-type: none">• Obligatoire.• Doit être une adresse courriel valide (voir filter_var).
Mot de passe <i>Texte masqué</i>	<ul style="list-style-type: none">• Obligatoire.• Non vide (pas seulement des espaces).
Confirmation du mot de passe <i>Texte masqué</i>	<ul style="list-style-type: none">• Obligatoire.• Doit être identique au mot de passe.
Prénom <i>Texte</i>	<ul style="list-style-type: none">• Obligatoire.• Non vide (pas seulement des espaces).
Nom <i>Texte</i>	<ul style="list-style-type: none">• Obligatoire.• Non vide (pas seulement des espaces).
Adresse de livraison <i>Texte</i>	<ul style="list-style-type: none">• Obligatoire.• Non vide (pas seulement des espaces).

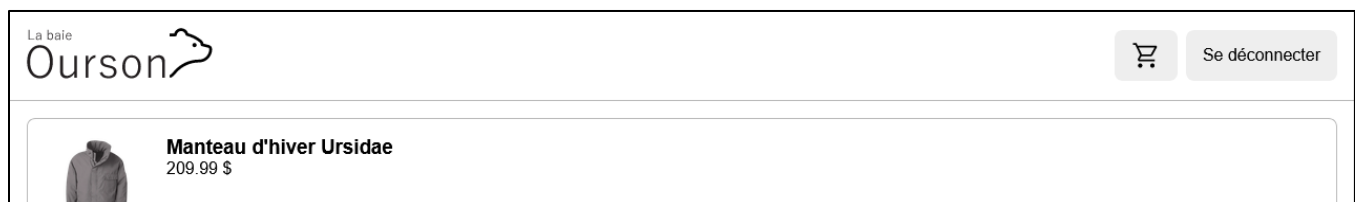
Lorsque le compte est créé, redirigez l'utilisateur vers la page de connexion.

i Information

Dans un contexte réel, les validations seraient beaucoup plus poussées. Par exemple, le nom de domaine du courriel serait comparé à une base de données de domaines reconnus. Aussi, l'adresse de livraison serait validée par un fournisseur de services de livraison, tel que [Poste Canada](#).

4.7. Page de déconnexion

Créez un fichier PHP nommé **sign-out.php** dont le seul but est de déconnecter l'utilisateur. Il n'y a pas de vue : elle redirige toujours vers la page d'accueil. L'utilisateur peut y accéder via le bouton « Se déconnecter » dans la barre de navigation. Ce bouton n'est accessible que si l'utilisateur est connecté.



4.8. Page de confirmation de commande

Lorsque l'utilisateur clique sur le bouton « Passer la commande » dans la page du panier (**cart.php**), sa commande doit être enregistrée dans la base de données (**checkout.php**) et une page de confirmation (**confirm.php**) doit être affichée. Le panier doit aussi être vidé. Tout comme **sign-out.php**, le fichier **checkout.php** n'a aucune vue et ne fait que rediriger vers d'autres pages.

Voici la maquette de la page de confirmation :












4.9. Informations supplémentaires

Voici quelques informations supplémentaires sur ce projet :

- Séparez la logique applicative de l'interface graphique [en faisant des vues](#).
- Utilisez le patron [Post/Redirect/Get](#) pour la soumission des formulaires.
- Utilisez le patron [Form Object](#) pour représenter les données de chaque formulaire.
- Utilisez le patron [Data Access Object](#) pour vos accès à la base de données.
- Lorsqu'il y a erreur, l'utilisateur ne devrait pas avoir à recommencer son formulaire au complet.
- Certaines pages ne devraient être accessibles que si l'utilisateur est connecté à son compte. Empêchez tout accès à ces pages et redirigez l'utilisateur vers une [page 404](#) ou la page d'accueil.
- Chaque produit possède une image. Le nom de l'image est composé du **sku** suivi de **.png**. Par exemple, **154789.png** ou **887414.png**. Ces images sont déjà fournies.

4.10. Couleurs et dimensions

Couleur	
Principale	 #b8e1ff  #99d5ff  #0075c9  #005a9b
Secondaire	 #eee  #ddd
Danger	 #dc3545
Bordure	 #bbb  #777
Dimension	
Marge et rembourrage	1rem ou 0.5rem
Coins arrondis	0.5rem
Boutons et champs de texte	48px de haut
Images de produits	300px de large
Vignettes de produits	100px de large par 96px de haut
Formulaire de connexion	576px de large au maximum
Formulaire de création de compte	1024px de large au maximum

5 Collaboration et plagiat

Ce travail ne peut être le produit d'une collaboration entre équipes ou de partage de code¹. Si de tels comportements sont observés, le professeur attribuera automatiquement la note de 0 %. Aussi, tout étudiant peut être convoqué à une rencontre afin de s'assurer de sa compréhension du travail remis.

Tout acte de plagiat, de tricherie et de fraude sera sanctionné. Constitue notamment un plagiat, une tricherie ou une fraude tout acte de copier, de fournir ou de recevoir volontairement de l'information lors d'un examen, de reproduire en tout ou en partie le travail d'une autre personne, qu'il s'agisse d'un document imprimé, multimédia ou électronique, sans y faire expressément référence, de remplacer un étudiant ou de se faire remplacer lors d'un examen, de remettre un travail réalisé par une autre personne, d'obtenir, posséder ou utiliser frauduleusement des questions ou réponses d'examen, d'utiliser du matériel, des applications, des sites Web ou des ressources non autorisés, de se faire aider par une autre personne lorsqu'il est demandé de réaliser l'évaluation seul, de falsifier les résultats de travaux ou d'examens.

Source : Politique d'évaluation des apprentissages du collège

6 Modalités de remise

6.1. Remise 1

Remettez sur LÉA une archive ZIP contenant l'entièreté de votre projet pour la page d'accueil, la page de consultation de produit et le panier (points 4.1 à 4.4). Incluez aussi un fichier texte vos noms et matricules.

N'incluez pas l'énoncé ni la grille de correction avec la remise : une pénalité de 10% sera appliqué si ces fichiers sont remis. Une pénalité de 15 % est appliqué en cas de retard de moins d'une journée. Au-delà de ce délai, le travail est refusé et la note de 0 % est automatiquement attribuée.

6.2. Remise 2

Remettez sur LÉA un fichier ZIP contenant l'entièreté de votre projet ainsi que toutes les fonctionnalités. Comme toujours, incluez aussi un fichier texte avec vos noms et matricules.

N'incluez pas l'énoncé ni la grille de correction avec la remise : une pénalité de 10% sera appliqué si ces fichiers sont remis. Une pénalité de 15 % est appliqué en cas de retard de moins d'une journée. Au-delà de ce délai, le travail est refusé et la note de 0 % est automatiquement attribuée.

¹ Du code identique avec des variables renommées et/ou des espaces introduits est considéré comme du plagiat.

7 Évaluation

Fonctionnalités – Le Quoi
Page d'accueil : <ul style="list-style-type: none">• Respect de la maquette (personnalisation du style permise).• Grille de produits (nom et prix) provenant de la base de données.
Page de consultation de produit : <ul style="list-style-type: none">• Respect de la maquette (personnalisation du style permise).• Numéro du produit consulté obtenu à partir de l'url.• Détails sur le produit (nom, description et prix) provenant de la base de données.• Possibilité d'ajouter le produit au panier (quantité incluse).
Page du panier : <ul style="list-style-type: none">• Respect de la maquette (personnalisation du style permise).• Liste des produits dans le panier (nom et quantité) provenant de la session courante.• Détails sur les produits (nom et prix) provenant de la base de données.• Calcul du sous-total impliquant le prix de chaque produit et leur quantité.• Possibilité de retirer un produit du panier.• Affichage d'un message informatif en cas de panier vide.
Page de création de compte : <ul style="list-style-type: none">• Respect de la maquette (personnalisation du style permise).• Champ de texte pour l'adresse courriel (obligatoire, non vide et du bon format).• Champ pour le mot de passe (obligatoire et non vide).• Champ pour la confirmation du mot de passe (obligatoire et égal au mot de passe).• Champ pour le prénom (obligatoire et non vide).• Champ pour le nom (obligatoire et non vide).• Champ pour l'adresse de livraison (obligatoire et non vide).• Création du compte dans la base de données.• Redirection vers la page de connexion après création du compte.
Page de connexion à un compte : <ul style="list-style-type: none">• Respect de la maquette (personnalisation du style permise).• Champ de texte pour l'adresse courriel (obligatoire, non vide et du bon format).• Champ pour le mot de passe (obligatoire et non vide).• Vérification du compte à partir de la base de données.• Redirection vers la page d'accueil après connexion au compte.• Connexion au compte ajoute le compte à la session.• Déconnexion du compte retire le compte de la session.
Page de confirmation de commande : <ul style="list-style-type: none">• Respect de la maquette (personnalisation du style permise).• L'utilisateur ne peut passer une commande s'il n'est pas connecté.• Possibilité de passer la commande (enregistré dans la base de données, vide le panier).
Autres : <ul style="list-style-type: none">• Barre de navigation s'adaptant à la connexion et la déconnexion de l'utilisateur (toutes pages).• Liste de messages d'erreur de validation en haut des pages.• Redirection vers la page d'accueil ou une page 404 en cas de ressource non trouvée.

Compétences – Le Comment
<p>Préparer la base de données (00SU-3) (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Rédaction de requêtes SQL (syntaxe générale). • Utilisation de PDO (syntaxe générale, connexion et gestion des erreurs). • Utilisation du patron Data Access Object. • Utilisation de transaction lors des requêtes en écriture (si nécessaire). • Utilisation de requêtes préparées.
<p>Programmer la logique applicative coté serveur (00SU-5) (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Rédaction de code en PHP (syntaxe générale). • Séparation suffisante de la logique application et du visuel. • Utilisation du bon verbe Http (Get, Post) pour l'envoi et la réception de formulaires. • Utilisation du patron Post/Redirect/Get pour les soumissions de formulaires. • Utilisation du patron Form Object pour les données des formulaires et leur validation. • Usage de sessions.
<p>Programmer la logique applicative coté client (00SU-6) (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Rédaction de code en HTML (syntaxe générale). • Rédaction de code en Css (syntaxe générale).
<p>Contrôler la qualité de l'application (00SU-7) (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Rédaction de tests unitaires avec PHPUnit. • Tous les tests unitaires passent avec succès.
<p>Rédiger la documentation (00SU-9) (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Emplacement et nature des commentaires. • Réusinage du code.

Pénalités – La rigueur
<p>Qualité générale du code (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Erreur à l'interprétation du code. • Avertissement de l'interpréteur ou de l'IDE ignoré sans raison valable. • Duplication de code ou découpage en fonctions déficient. • Code malpropre, illisible ou comportant du code en commentaire. • Code « patché » visant à corriger une erreur se produisant ailleurs dans le code. • Classement malpropre des fichiers et des dossiers. • Formatage incorrect ou déficient. • Nommage ambiguë ou paresseux. • Autres pénalités.
<p>Remise du travail (incluant, mais sans s'y limiter) :</p> <ul style="list-style-type: none"> • Non-respect des consignes de remise. • Usage du mauvais type d'archive pour la remise du projet. • Fichier ou dossier inutile remis avec le projet. • Autres pénalités.

La qualité de la langue française fait partie de l'évaluation. Chaque faute de français retire 0,5 % à la note finale jusqu'à concurrence de 20 %.