PHP de BASE

Modalité :

- Travail en autonomie

Objectif :

- Savoir développer en PHP Procédural

Compétences :

- Programmer des variables, boucles, tableaux, fonctions…
- Envoyer et récupérer des données via un formulaire
- Mettre en œuvre des sessions

Etape 1 – PHP, Qu'est-ce que c'est ? (0.5 journée)

Expliquez l'intérêt des éléments cités ci-dessous et les relations entre-elles :

- Apache
- Navigateur
- PHP
- Langage Client
- Langage Serveur
- Protocole HTTP
- Serveur WEB

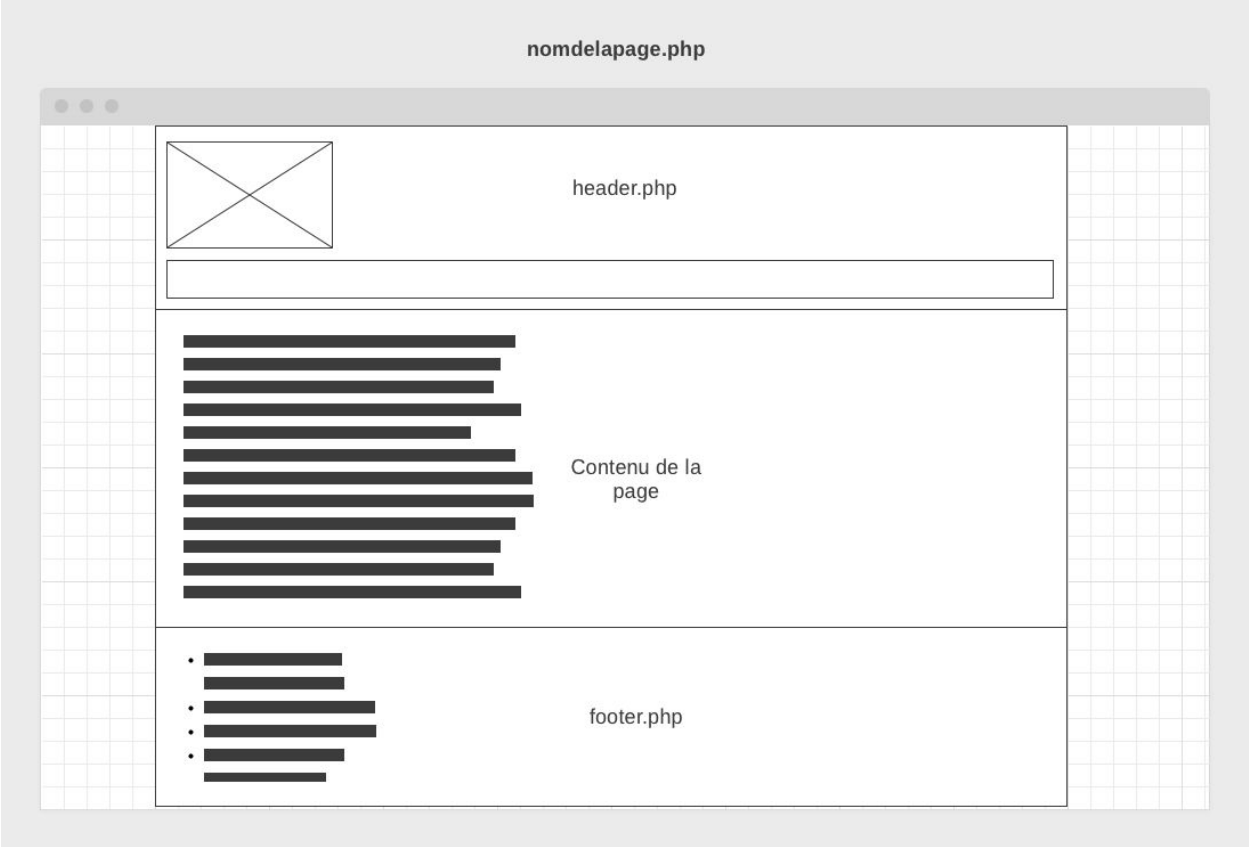
Introduction au PHP - Manual

Etape 2 - Création d’un “Template” (0.5 journée)

Vous allez reprendre vos sites individuels.

Vous découperez en 3 parties vos pages :

- **header.php** :
  Contient le code HTML du début du fichier (balise <html>...), jusqu'à la fin de votre header ou de votre nav.
- **footer.php** :
  Contient le code HTML juste après le contenu (</main>), jusqu’à la fin du fichier (balise </html>)
- ***nomdelapage*.php** :
  La page hobby.html devient hobby.php qui contiendra, dans un premier temps, uniquement le contenu.



Toutes les pages de votre site devront être accessibles via une page “nomdelapage.php” et inclurons les fichiers header.php et footer.php.

Contraintes

- Chaque page doit avoir son propre titre (<title> le contenu de la balise) et une meta description différente.
- Aucun doublon de contenu n'est toléré !

Ressources

- include - Manual
- require - Manual

Etape 3 - Concept de Front-Controller (1,5 journées)

Le Front Controller est un design pattern qui réduit à **un seul point** d'entré l'accès à vos pages pour des raisons de sécurité, maintenance… bref c'est bien 👍.

ℹ Pour les courageux qui veulent en savoir plus.

Voyez par vous même la question posé par un débutant PHP sur le site StackOverflow qui se demande ce que c'est, car il a rien compris, le pauvre 😢 :
https://stackoverflow.com/questions/6890200/what-is-a-front-controller-and-how-is-it-implemented-in-php
L'exemple donnée est un peu complexe, mais vous pouvez vous en inspirer. Resté sur une structure de IF / ELSEIF / ELSE.

Donc tout votre code devra passer par un nouveau fichier **index.php** qui sera votre Front Controller et utilisera un **paramètre GET** avec le nom "page".
Index.php appellera les 3 fichiers en fonction du paramètre GET de nom "page".

Livrables

- Toutes vos pages sont accessibles via : index.php?page=**lenomdelapage**

Ressources

- $_GET - OpenClassRooms

Etape 4 - Formulaire (2,5 journées)

Dans votre page contact, vous avez un formulaire et vous allez le rendre fonctionnel.

Le formulaire est à envoyer avec la méthode POST sur la même URI (index.php?page=contact)

Pour rappel ou pas votre formulaire doit contenir :

- Un champ civilité <select>
- Un champ nom <input type="text">
- Un champ prénom <input type="text">
- Un champ email <input type="email">
- Un champ raison du contact <input type="radio"> avec plusieurs raisons (service comptable, bref vous inventerez...)
- Un champ message <textearea>

***i** Évitez l'usage de <input type="checkbox"> et n'utilisez pas la validation HTML 5 "required" sur vos champs.*

- Vérifier en **PHP** les données reçus :
- Vérifier que le champ message à un contenu d'au moins 5 lettres.
- L'email saisi doit être dans un format valide.
- Un choix de raison de contact parmi la liste définie dans votre formulaire.
- Le nom et prénom remplis.

Bref, que l'utilisateur ne puisse pas oublier ou tricher, via l'inspecteur et en injectant des donnés avec des caractères spéciaux à votre formulaire.

- Indiquez sur les champs l'erreur de saisie à l'utilisateur.

- Enregistrer l'ensemble des champs du formulaire dans un fichier .txt avec : https://www.php.net/manual/fr/function.file-put-contents.php

***i** Pour information les formateurs ont faire le choix, de pas vous faire envoyer de mail. Les restrictions d'envoi d'email sont devenu d'année en année de plus en plus complexe, merci le SPAM et les filtres anti-SPAM. Evitons de brûler arbres.*

- Lors d'une erreur de saisie d'un utilisateur et après la soumission du formulaire, les champs ne doivent pas se vider. $_SESSION - Manual

Etape 5 : Bonus

5.1 :
Faite l'étape 4 avec les Filter Input, technique de Warrior :

- Ajouter des filtres pour éviter les champs vides avec PHP filter_has_var() Function.
- Indiquez un message d'erreur comme quoi le champ est vide au-dessus de chaque champ.
-

5.2 :
Maintenant qu'on sait que tous les champs sont présents, nous allons valider les champs avec PHP filter_input() Function avec les filtres de validation Filtres de validation - Manual .

Etape 6 - Super Bonus

Vous rajouterez à votre formulaire de contact, la possibilité d'envoyer un fichier.
Les fichiers reçu seront stocké dans un dossier "storage" que vous créez.

C'est un gros morceau 😵, courage.
https://www.php.net/manual/fr/features.file-upload.post-method.php