6GEI466 - Applications réseaux et sécurité informatique - Laboratoire #3

Jean-Luc Cyr

Hiver 2025

1 Objectif

Vous familiariser avec le développement en PHP et en javascript coté serveur. Parfaire vos connaissance en application pour le web.

2 Introduction

Le but de ce laboratoire est réviser vos connaissances du prototole HTTP (méthodes, URL, code de résultat, ...) et de développer vos connaissances du langage PHP et JavaScript. Le coté client de votre laboratoire #2 sera réutilisé (c'est-à-dire vos fichiers html, css, js, ...).

Pour ce faire, vous récrire votre serveur web (HTTP) en langage PHP (partie 1) et en JavaScript (partie 2) pour supporter votre client.

3 Installation de l'environnement de travail

Pour réaliser votre laboratoire vous aurez besoin de deux environnements différents.

3.1 Partie 1 - PHP

Vous devrez télécharger l'interpréteur PHP (`https://windows.php.net/download/`)

3.2 Partie 2 - JavaScript

Vous devrez télécharger Node.JS (`https://nodejs.org/en/download/`)

4 Description

En utilisant vos fichiers de front-end (l'interface utilisateur) soit les fichiers html, css, js, images, ... vous devez refaire l'équivalent du code Python de votre serveur dans les langages PHP et JavaScript.

4.1 Partie 1 - PHP

Dans l'environnement PHP, vous pouvez lancer l'interpréteur en mode serveur web avec la commande :

```
php -S 127.0.0.1:8000
```

Prenez le temps de lire son comportement. Par défault, il délivrera automatiquement pour l'URL "/" le fichier "index.html". Vous devrez le forcer à lancer par défaut le fichier "index.php" qui contiendra le code de votre serveur.

Attention, il y a deux stratégies que vous pouvez utiliser pour réaliser le projet :

La première est de créer un fichier index.php pour l'adresse "/" et de créer un second fichier pour l'adresse "/horosocope" qui correspondrait à "horoscope.html". Je ne veux pas que vous utilisiez cette stratégie.

La seconde, celle que je veux que vous réalisiez, toutes les adresses "/", "/static/*", "/horoscope" doivent toutes appeller votre fichier "index.php". Dans ce fichier, votre code devra utiliser les éléments du protocole HTTP (les entêtes des requêtes) contenus dans la variable globale $_SERVER pour décider de l'action à prendre tel qu'il a été montré lors de la séance de cours.

Le comportement de votre code devra donner le même résultat que votre serveur en Python dans le laboratoire #2.

Cette partie du laboratoire a pour but de vous faire apprendre le PHP mais également de voir votre compréhension des entêtes HTTP, comment les récupérer et les interpréter c'est pourquoi vous devez programmer votre propre "framework" et non pas utiliser des outils existants comme Laravel, Symphony, Slim.. ou des outils de "templating" comme twig.

4.2 Partie 2 - JavaScript

Pour la partie JavaScript, vous devez utiliser Node.JS pour réaliser votre serveur, toutjours en utilisant les mêmes fichiers "front-end" (ie html, css, js, ...) que dans le laboratoire #2 et dans la première partie.

Cette fois vous devez refaire le "back-end" soit la partie serveur en utilisant le langage JavaScript et en le faisant exécuter par l'interpréteur node.js.

Dans les powerpoint du cours sur PHP et les autres technologies, vous avez des liens vers des exemples comment faire un serveur web avec Node.JS en Javascript. Vous pouvez utiliser les bibliothèques que vous voulez (par exemple : express, http ou autre).

Cette partie du laboratoire a pour but de vous familiariser avec la réutilisation du langage JavaScript pour la programmation serveur en complément de

la partie client. C'est pourquoi ici vous pouvez utiliser les bibliothèque que vous voulez.

4.3 Vos deux serveur devront

Avoir le même comportement que celui du laboratoire #2 soit :
- Recevoir des requêtes HTTP
- Les appels dont l'adresse commence par /static/* devront retourner le fichier correspondant dans le répertoire static sur le disque
- Les appels à l'adresse / devront être traités par le logiciel et retourner le contenu d'un gabarit nommé accueil.html
- L'appel à l'adresse /horoscope devra retourner des données (texte, xml ou json) contenant pour une date précise reçu le signe du zodiac correspondant, le nom d'un fichier image correspondant et un court texte de prédiction astrale.
- Les appels aux autres pages devront retourner une erreur 404 et le contenu d'un gabarit nommé erreur404.tpl

4.4 Le fichier HTML de la page d'accueil devra contenir les éléments suivants

Doit être le même que pour le laboratoire #2
- **Les éléments d'une page HTML 5 minimale valide**
- **Un titre <h1> (dont la valeur est mise en place par le logiciel)**
- **Un formulaire <form>**
- **Un champ nom et un champ prénom.**
- **un champ date (utiliser l'affichage d'un calendrier "date picker" de jqueriui ou une autre validation)**
- **un bouton voir mon horoscope**
- **une zone (div) qui affichera l'horoscope.**
- **Un logo quelconque <img> (image jpg ou autre)**

4.5 Comportement

Doit être le même que pour le laboratoire #2

Lorsque la page racine (/) est invoquée elle doit afficher le formulaire (sans horoscope) Si les champs sont remplis et que le bouton "voir mon horoscope" est pressé : Vous devez transmettre les information à votre serveur à l'aide d'une requête AJAX (POST) avec les valeurs de nom, prénom et la date. L'URL utilisé pour la requête doit être : /horoscope Vous devez afficher le résultat de la requête AJAX dans la zone destiné à afficher l'horoscope.

Lorsque la page du service "horoscope" (/horoscope) est appelée avec la méthode POST Vous devez valider que vous avez reçu un nom, prénom et une date Si un paramètre est manquant ou vide, vous devez retourner une erreur avec le message paramètre manquant. Vous devez valider la date reçue Si elle est invalide, retourner un message "date invalide" Si elle est valide, retourner

un message contenant "Le nom, prénom, le signe du zodiaque (idéalement accompagné d'un lien <a> vers son image) correspondant à la date accompagné d'un court texte de votre choix prédisant l'horoscope."

Tous les appels non prise en charge par le système (autre que /static/* et /) doivent retourner un erreur 404 avec le contenu du gabarit erreur404.html

Vous pouvez faire votre propre @route et fonction ou utiliser les outils fournit par flask pour gérer les pages inexistantes à votre guise.

5 Politique de remise de travail

Méthodes de remise : via le "Moodle" du cours, Les fichiers devraient être regroupés dans un fichier "zip" ou "tar" afin de préserver l'arborescence du projet.

Remettez seulement les fichiers de votre programme et non pas tout l'environnement virtuel.

Séparer les deux parties dans deux sous-répertoires dans votre fichier ZIP et bien identifier "Partie1" et "Partie2".

Les fichiers

- **.php, .js, ... (vos programmes)**
- **requirement.txt (vos dépendances s'il y a lieu)**
- **readme.txt (les informations sur la version de PHP/NodeJS, les instructions de mise en route de votre programme)**
- **votre répertoire "static" et son contenu (images, css, etc)**
- **votre répertoire "templates" et son contenu (vos pages/templates html)**

Vous pouvez remettre en double vos fichier static et template pour simplifier les instructions pour le démarage de votre projet. (une copie dans /Partie1/ (pour le PHP) et une copie dans le /Partie2/ (Pour le JavaScript).

Attention, la remise doit avoir lieu avant la date inscrite sur le site du cours. Toute remise après cette heure de tombé sera considéré comme remise en retard et soumis à la pénalité de 10%/jour et ne sera pas corrigé après 7 jours de retard. Prévoyez les problèmes et délais possible de transmission internet, n'attendez pas à la dernière minute.

6 Correction

Le laboratoire vaut 10% de la note du cours.

Comme la première partie (PHP) est plus complexe elle vaut 6 points et la seconde (JS) 4 points.

La correction sera effectuée sur les éléments suivants :

- **Suivi des consignes (2 points)**
- **Fonctionnalités (5 points)**
  - **Tout fonctionne avec le serveur PHP (3pts)**
  - **Tout fonctionne avec le serveur NodeJS (2pts)**

- **Validité du code et des documents (3 points)**
  - **PHP (2pts)**
  - **NodeJS (1pt)**