ICT XXX

DÉVELOPPEMENT D'UN SITE DE E-COMMERCE

Nicolas Crausaz & Mathias Brügger
EPSIC | ICT XXX

1 Table des matières

2 E-Commerce

2.1 Objectif

Nous avons décidé de faire ce projet avec des technologies performantes et peu utilisés qui nous permettront de gagner en performance et en expérience. La méthode utilisée est propre et permet une gestion avancée des sécurités. Il permet également le développement d'une application mobile si souhaité.

En ne générant pas les pages de front end avec PHP nous gagnons en performance, car elles sont générées sur côté client. Ceci évite la fragmentation du code en différents langages qui rajoutent des incompréhensions et des difficultés à la maintenance.

L'api sera faite en suivant le concept du **RestFul *(cf : source)*** (séparation client-serveur, flexibilité des supports d'accès, organisation, maintenance et entretiens facilités). Nous tenons à suivre ce concept qui est comme un mode de vie, une directive d'organisation.

2.2 Servitudes

Interface client :

- Liste des articles
- Système de panier et liste de souhaits
- Recherche et navigation vers les articles (nom, catégories etc.)
- Création de compte et authentification
- Modification des informations du client par lui-même

Interface administrateur :

- Possibilité d'éditer les articles, les commandes et les infos des clients.

2.3 Cahier des charges

Développement d'un site de vente d'habits avec différences technologie selon un schéma précis

- Développement d'un Front End en JavaScript
  Ce Front End sera fait à l'aide d'un Framework d'interface exécutée du côté client en JavaScript *qui enverra des requêtes sur l'API PHP*
- Développement d'une API en PHP
  Cette API prendra en charge les entrée en base de donnée, l'authentification et la gestion des erreurs

2.4 Choix des techniques et technologie utilisée

2.4.1 Framework Front End – Vue.js (JavaScript)

Description :

Vue.js est un Framework JavaScript permettant de réaliser des interfaces client. Il est très performant et également très léger (20ko de base). Ce Framework gère uniquement la « vue », il est possible d'inclure d'autre librairies, par exemple pour faire des requêtes http. Il permet de créer des composants (voir ci-dessous).

Utilisation :

Nous allons utiliser Vue.js pour construire toute notre interface client (vue), accompagné d'un routeur natif qui permet d'envoyer des composants à la vue en fonction de l'URL, une librairie CSS et une librairie pour les requêtes http.

2.4.2 Framework Back End (API) - Laravel (PHP)

Description :
Laravel est un framework php qui permet la gestion d'url, d'erreur. Le but de ce framework est de ne pas avoir à rappeler des commandes déjà créées et de simplifier la structure du code
Utilisation :
Nous allons faire une API à l'aide de Laravel en php. Le but est de recevoir des requêtes HTTP de différents types (GET, POST, PATCH, PUT, DELETE, HEADER), de les interpréter et de faire des retours en JSON[1]. Le tout sans conserver de données côté serveur utile à la connexion.
Chaque requête génère une authentification. Décrite dans la description du projet
Chaque requête http sera loguée, les erreurs d'accès également. Les erreurs liées à l'utilisation seront renvoyées en JSON avec un code de retour http définie et interprétés(affichées) par le client.

2.4.3 Librairie Back End – Dingo (php)

Description :
Le package d'api Dingo fournit un set de différents outils utiles pour créer rapidement une api. Elle contient des librairies de routage et d'authentification
Utilisation :
Le but de cette librairie est de pouvoir monter une api avec les fonctions classiques utile à une api ainsi que les possibilités d'authentification http. Ces fonctions sont intégrés de base dans laravel mais Dingo simplifie au maximum leur utilisation.

2.4.4 Framework Front End – CSS

Description : Non choisi
Utilisation :

2.5 Infrastructure utilisée

2.5.1 Serveur

VPS[2] : web01.domain.tld
Utilisation : *2 Vhosts séparés pour l'API et le Front End. L'api fera les requêtes sur le serveur de DB directement*

Paramètre	Valeur
Système d'exploitation	*Debian 9 (stretch)*
RAM	*2 Go*
Puissance	*1 Core*
Services Installés	*Apache, php*
Storage	*40 Go*
IP privée	*10.x.x.x*

[1] JSON : JavaScript Object Notation est un format de donnée en tableau
[2] VPS : Virtual Private Server est une machine virtuelle située sur un hôte physique contenant possiblement plusieurs machines virtuelles.

IP public	*x.x.x.x*

VPS : mysql01.domain.tld

Utilisation : *1 base de données accessible depuis web01.domain.tld exclusivement.*

Paramètre	Valeur
Système d'exploitation	*Debian 9 (stretch)*
RAM	*2 Go*
Puissance	*1 Core*
Services Installés	*MariaDB*
Storage	*25 Go*
IP privée	*10.x.x.x*

2.5.2 Configuration des services

Apache2 :

VirtualHost[3] - api.domain.tld

Paramètre	Valeur
ServerName	*api.domain.tld*
ServerAlias	-
DocumentRoot	*/var/www/vhost/api.domain.tld/web*
ErrorLog	*/var/www/vhost/api.domain.tld/logs/error.log*
CustomLog	*/var/www/vhost/api.domain.tld/logs/access.log combined*
Port d'accès	*443*
Ip d'accès autorisés	*
SSLEngine	*ON*
SSLCertificateFile	*/var/www/vhost/api.domain.tld/.conf/ssl/api.domain.tld.crt*
SSLCertificateKeyFile	*/var/www/vhost/api.domain.tld/.conf/ssl/api.domain.tld.key*
SSLCACertificateFile	*/var/www/vhost/api.domain.tld/.conf/ssl/ca.crt*
SSLProtocol	*all -SSLv2 -SSLv3*
SSLCipherSuite	*ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS*
SSLHonorCipherOrder	*ON*
Docroot::Options	*-Indexes +FollowSymLinks +MultiViews*

[3] VirtualHost : hôte virtuel est utilisé afin de diriger les requêtes http dans un répertoires selon des paramètres définis (ports, domaines d'accès).

VirtualHost - domain.tld

Paramètre	Valeur
ServerName	*domain.tld*
ServerAlias	*www.domain.tld*
DocumentRoot	*/var/www/vhost/domain.tld/web*
ErrorLog	*/var/www/vhost/domain.tld/logs/error.log*
CustomLog	*/var/www/vhost/domain.tld/logs/access.log combined*
Port d'accès	*443*
Ip d'accès autorisés	*
SSLEngine	*ON*
SSLCertificateFile	*/var/www/vhost/domain.tld/.conf/ssl/domain.tld.crt*
SSLCertificateKeyFile	*/var/www/vhost/domain.tld/.conf/ssl/domain.tld.key*
SSLCACertificateFile	*/var/www/vhost/domain.tld/.conf/ssl/ca.crt*
SSLProtocol	*all -SSLv2 -SSLv3*
SSLCipherSuite	*ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS*
SSLHonorCipherOrder	*ON*
Docroot::Options	*-Indexes +FollowSymLinks +MultiViews*

3 API - Back End Documentation

3.1 Requêtes

3.1.1 GET

Path (url)	Utilité	Paramètre (postfield)	Description

3.1.2 POST

Path (url)	Utilité	Paramètre (postfield)	Description

3.1.3 PATCH

Path (url)	Utilité	Paramètre (postfield)	Description

3.1.4 PUT

Path (url)	Utilité	Paramètre (postfield)	Description

3.2 Authentification

3.3 Fonctions

4 Journal de travail

09.11.2017	Création du rapport – première édition
xx.xx.xx	

5 Sources de documentations utilisés

Source (URL, Livre, documents)	Auteur	Utilisation
https://fr.wikipedia.org/wiki/Representational_state_transfer	Collaboratif	Représentation et description du fonctionnement de RESTFUL
https://github.com/dingo/api	Thilanga Pitigala	Librairie laravel pour le développement d'api
https://fr.vuejs.org/v2/guide/	Vue.js	Guide Vue.js