API Navigo

Résumé

Ce document a pour objectif de fournir aux futurs développeurs une documentation technique détaillée des différentes parties du projet NavigoApi,

La première partie concerne la configuration des environnements nécessaires au bon fonctionnement du site web.

La seconde partie contient une documentation de l'API et décrit le fonctionnement du site web.

1 - Configuration de l'environnement

Ce projet est composé des éléments suivants :

- Un environnement LAMP (Linux Apache MySQL PHP5.6)
- Une base de données MySQL servant à stocker les utilisateurs et les cartes
- Le site web symfony

1.1 - Importation des projets

Le site web utilise le gestionnaire de versions git et est hébergé sur la plateforme Github. Pour récupérer ce projet il faut avoir git d'installé sur son environnement et le récupérer à l'aide des lignes de commandes sur le terminal ou bien à l'aide du module intégré à un IDE comme PHPStorm.

Localisation du projet :

- Site web : https://github.com/Mapsred/NavigoProject

1.2 - Installation des outils

Afin de pouvoir faire tourner le projet, il suffit d'avoir un environnement LAMP et git d'installé.

1.2.1 - Configuration Apache

```
<VirtualHost *:80>
	ServerName navigo.mindgame.ovh

	DocumentRoot /var/www/sites/cours/Navigo/web
	<Directory />
		Options FollowSymLinks
		AllowOverride None
	</Directory>

	<Directory /var/www/sites/cours/Navigo/web>
		Options Indexes FollowSymLinks MultiViews
		AllowOverride all
		Order allow,deny
		allow from all
	</Directory>

	ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
	<Directory "/usr/lib/cgi-bin">
		AllowOverride None
		Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
		Order allow,deny
		Allow from all
	</Directory>

	ErrorLog /var/log/apache2/error.log

	# Possible values include: debug, info, notice, warn, error, crit,
	# alert, emerg.
	LogLevel warn

	CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

1.2.2 - Configuration PHP

Il est aussi nécessaire de modifier le fichier de configuration PHP PHP.INI.
memory_limit = 256M

1.2.3 - Configuration Symfony

Il faut installer composer à l'aide de la commande "composer install" puis remplir les informations demandées
En cas de problème, on peut vider le cache à l'aide de la commande "php bin/console cache:clear --env=prod" pour la production ou "php bin/console cache:clear" pour le développement.

Pour la configuration de la base de données il faut utiliser les commandes suivantes dans l'ordre.
php bin/console doctrine:database:create
php bin/console doctrine:schema:update --force
php bin/console doctrine:fixtures:load

Le projet devrait donc être accessible au lien suivant : https://navigo.mindgame.ovh/

2 - API

l'API est organisée avec une architecture REST. Notre API est prévue pour avoir des url semblables aux données échangées.

Nous utilisons les fonctions HTTP et retournons toutes les données sous forme de JSON

2.1 - Routes

Route	Type	Paramètres	Retour
/api/card/{apiKey}	GET	apiKey : clé d'api unique à l'utilisateur	type: Json Données sur la carte
/api/user/{apiKey}	GET	apiKey : clé d'api unique à l'utilisateur	type: Json Données sur l'utilisateur
/api/update/card/{apiKey}	GET	apiKey : clé d'api unique à l'utilisateur	type: Json Lien de paiement paypal pour renouvellement
/api/update/card/{apiKey}/confirmation	GET	apiKey : clé d'api unique à l'utilisateur	Confirmation du renouvellement
/api/card/validation/{apiKey}	GET	apiKey : clé d'api unique à l'utilisateur	type: Json vérification de la validité de la carte

2.2 - Explication du processus

Cette API permet à l'utilisateur possédant sa clé d'api de récupérer des informations comme ses informations d'utilisateur (hors mot de passe), ses informations de carte ou encore renouveler sa carte (par tranches de 2 mois) et tester sa validité.

3 - Le site

3.1 - La couche logicielle

La couche logicielle est composée de :

- PHP 5.6
- Apache 2.4.10
- MySQL 5.5.49-0+deb8u1
- Linux Debian 8 (Prod) ou Ubuntu 16 (Dev)

3.2 - Le framework

Le site a été développé sous le framework Symfony car il permet une gestion facile des urls avec son routing, notamment pour l'API.
La version utilisée est le 3. Développé entre la version 3.1.3 et 3.1.7

C'est un framework de type MVC (Modèle Vue Contrôleur).

3.3 - Bundles

En plus des bundles venant à l'installation de Symfony, j'ai inclus au projet :

- friendsofsymfony/jsrouting-bundle
  - Utilisation des routes de Symfony en JavaScript
- stof/doctrine-extensions-bundle
  - Ajout de fonctionnalités à l'ORM Doctrine
- beberlei/DoctrineExtensions
  - Ajout de fonctionnalités à l'ORM Doctrine
- knplabs/doctrine-behaviors
  - Ajout de fonctionnalités à l'ORM Doctrine
- sonata-project/admin-bundle
  - Interface administrateur
- sonata-project/doctrine-orm-admin-bundle
  - Extension au bundle précédent
- paypal/rest-api-sdk-php
  - SDK Paypal en PHP
- kmj/paypalbridgebundle
  - Service Symfony pour le SDK PHP Paypal
- symfony/assetic-bundle
  - Gestionnaire de ressources (css, js, img, etc …)
- jms/serializer-bundle
  - Sérialiseur de données
- doctrine/doctrine-fixtures-bundle
  - Ajout de fonctionnalités à l'ORM Doctrine

3.4 - Partie publique du site

Toute la partie publique du site est contenue dans les bundles situés dans :

- src/UserBundle
- src/AppBundle

La partie Admin est située dans

- src/AppBundle/Admin

La liste des routes de la partie publique est récupérable via la console Symfony :
php bin/console router:debug

Le site est géré par plusieurs contrôleurs situés dans le dossier Controller de chaque Bundle.

Les vues en twig sont disponibles dans le dossier Resources/views de chaque Bundle.

Toutes les pages héritent de *index.html.twig*

4 - Informations

Accès à la base de données :
http://phpmyadmin.francois-mathieu.fr
Username : navigo
Password : NavigoTesting

Utilisateur admin :
Nom :Lala POIRIER
UUID : N33002345679D
Password: NavigoTesting
Username : LalaPoirier