

2015

Documentation Œuvre éphémère Green IT



Adel BENAMOR
GREEN LAB CENTER
31/08/2015

Table des matières

Contexte globale	3
Présentation de l'entreprise d'accueil	3
Organigramme.....	3
Présentation des partenaires du projet	4
Cahier des charges.....	5
Présentation du projet :	5
Comité projet :	5
Objectif de l'application :	5
Les cibles :.....	6
Fonctionnalités :	6
Plan de l'application :	8
Plan Individuel :	8
Plan Global :.....	9
Fenêtre à remplir :.....	9
Environnement de travail.....	10
Système d'exploitation:.....	10
Installation de Xampp:	10
Utilisation des scripts	12
Script 1: debut_script_recup.py	12
Script 2: url.bash.....	12
Explication du code :	12
Script 3 : script_recup.py.....	13
Explication du code :	14
Script 4 : reseauWea.bash.....	15
Script 5: traceroute.bash.....	16
Base de données	18
Partie PHP	19
Explication du code googleMap.php:	19

Explication du code informations.php :	21
Continuité du projet	23
Bilan et analyse.....	24
Annexes	25
Bibliographie:	25
Résumé	35
Résumé en français:	35
Résumé en anglais:.....	35
Résumé en espagnol:	35

Contexte globale

Présentation de l'entreprise d'accueil

→ Le Green Lab Center

Le Green Lab Center est une association qui a été créée par un ensemble d'entreprises, d'écoles ou encore laboratoires dans le but de travailler tous ensemble dans le domaine green IT.

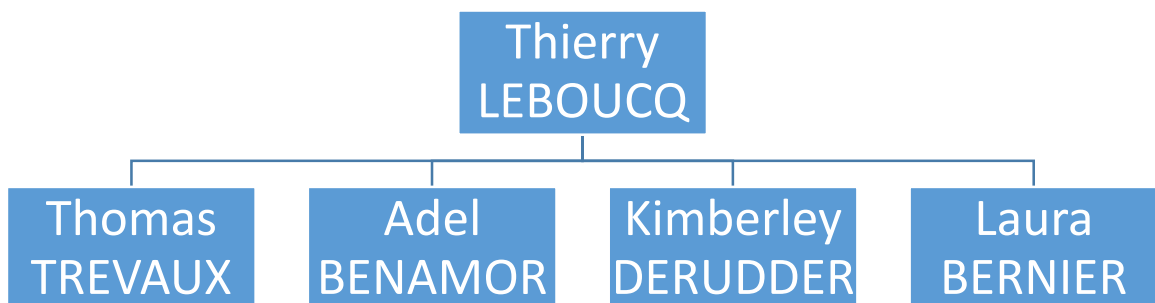
Cette structure a été créée afin de pouvoir centraliser le travail fait par les différents partis et de ce fait avancer plus rapidement. De plus, il permet à un des membres d'être porteur de projets et d'avoir un soutien parmi les différents membres du green lab center.

En effet, celui-ci soutien et diffuse énormément de projets innovants.

Le Green Lab Center est basé au Hub à Nantes, c'est un lieu où se réunissent les différents membres, mais aussi où des cours peuvent avoir lieu. Certaines entreprises louent aussi ces locaux afin d'y travailler.

C'est ici que j'ai été accueilli pendant la durée de mon stage où j'ai été amené à rencontrer plusieurs membres de différentes entreprises.

Organigramme:



→ Sigma

Sigma est un acteur majeur au sein des métiers de l'édition logiciel. C'est une entreprise qui est implantée sur 4 gros sites en France dont celui de Nantes.

J'ai passé 3 semaines dans leurs locaux afin de travailler sur les scripts de mon projet où j'ai été encadré par Guillaume THOMSEN.

C'est une entreprise d'envergure internationale, car en France, elle possède près de 2200 clients et un chiffre d'affaires de 69 millions d'euros. Elle possède aussi près de 850 collaborateurs.

Présentation des partenaires du projet

➔ Kaliterre

C'est une société à responsabilité limitée (SARL) basé à Nantes depuis 2010. Elle est spécialisée dans l'éco-conception des logiciels. En effet, elle travaille essentiellement sur l'optimisation de la consommation électrique des ordinateurs ou mobiles en essayant de développer de manière "green" ou de créer des produits qui permettent d'aider dans ce domaine comme notamment le Greenspector qui permet d'analyser des logiciels informatique et de trouver ses failles pour qu'elles puissent être corrigées.

➔ EasyVirt

C'est une start-up basée à Nantes qui est spécialisée en informatique notamment dans l'efficacité informatique.

EasyVirt propose plusieurs solutions afin de simplifier la maîtrise ainsi que l'optimisation de différentes infrastructures. Pour ce faire, elle fournit des bonnes pratiques, des indicateurs ou encore des actions à mener afin d'optimiser le matériel analysé.

C'est une start-up qui se base sur 4 racines :

- ⇒ La recherche
- ⇒ Le Green IT
- ⇒ La virtualisation
- ⇒ Le Cloud Computing

EasyVirt tente d'innover au maximum et pour ce faire, l'ensemble de ces salariés possèdent une expertise aussi bien en informatique mais aussi une expertise Green IT ce qui leur permet de toujours avoir des solutions différentes de leurs concurrents.

➔ Wouep

C'est une agence basé à Nantes et créer par Pablo PARRAL et Loris CHENNBAULT. Elle a pour objectif de programmer des sites web de façon à limiter leurs consommations.

C'est une entreprise qui a pour objectif de mettre en place de bonnes pratiques dans le développement des sites web tout en choisissant des langages de programmations performantes.

Pablo et Loris font partie du Green Lab Center et viennent aussi au hub afin d'y travailler.

Cahier des charges

Présentation du projet :

On attend de ce projet un résultat qui nous permettra de répondre à un besoin au niveau green IT.

L'internet consommera de plus en plus de ressources. On parle d'une consommation énergétique équivalente en 2030 de ce que consommait l'humanité en 2008. On doit mettre du Green dans cette consommation abusive de ressources pour satisfaire à un web durable. Une page web contient un grand nombre de requêtes qui vont chercher des données, des contenus, des services dans le monde entier. Ces requêtes entraînent des consommations importantes d'énergie à l'échelle de la planète. Le Green Lab center à travers les sociétés et écoles Sigma, Esaip, Easyvirt et KaliTerre souhaitent mettre en évidence et sensibiliser le monde entier à cette problématique en créant une œuvre artistique éphémère en lien avec des événements visibles. Le principe est simple : chaque participant saisit l'adresse internet de son entreprise et voit en direct sur un grand écran quel chemin a parcouru la donnée pour arriver jusqu'au lieu (en kilomètre octet parcouru, en énergie /1000 pages vues, en euros ou dollar / page vue, ... l'œuvre se crée progressivement avec la map monde qui se colore progressivement vers le rouge en fonction de l'intensité des requêtes qui sont échangées avec les points dans le monde. Une interaction avec chaque point est possible en mode tactile pour qualifier le volume de données provenant des sites enregistrées. Cette œuvre pourra être reproduite dans d'autres salons car transposable et pourra être à terme retravaillée pour être mise en ligne.

Comité projet :

Le comité projet est composé de :

- ⇒ THOONSEN Guillaume
- ⇒ DARGENT Martin
- ⇒ GUTOWSKI Nicolas
- ⇒ MORVAN Steven
- ⇒ PHILIPPOT Olivier
- ⇒ PANAU Nicolas
- ⇒ LEBoucQ Thierry

Objectif de l'application :

L'objectif de cette application est de permettre d'analyser les requêtes contenues dans une page d'un site web afin de faire une représentation graphique sur un planisphère de l'origine des données contenues dans une seule page. Ces informations seront stockées dans une base de données qui sera utilisée par la suite afin d'avoir une vue globale sur ce même planisphère avec une intensité des couleurs qui sera d'autant plus forte que le nombre de requête ayant cette origine est important. Dans le cadre d'un événement, il s'agira d'une œuvre éphémère des personnes qui ont voulu poster leur site web pour contribuer à l'œuvre de l'évènement.

Les cibles :

Les cibles sont les différentes personnes qui vont passer dans les évènements suivants :

- La Digital Week => Septembre
 - Caractéristique : Entreprises
 - Importance : Primaire
 - Centre d'intérêt : Recherche d'informations
- Green Week => Octobre
 - Caractéristique : Entreprises
 - Importance : Primaire
 - Centre d'intérêt : Recherche d'informations
- Green Code Lab Challenge => Décembre
 - Caractéristique : Etudiants, Entreprises
 - Importance : Primaire
 - Centre d'intérêt : Recherche d'informations

Ce projet sera publié dans le cadre de l'application web, webenergyarchive.com dont elle doit respecter l'architecture technique et le graphisme.

Fonctionnalités :

L'application permet plusieurs fonctionnalités, voici une liste en fonction de leurs emplacements :

Map Individuelle :

- ⇒ Visualiser sur une carte les chemins empruntés par chaque requête du site entré depuis le lieu jusque le lieu du serveur de la requête via le principe du Traceroute.
- ⇒ Changement de couleur des traits en fonction du poids du volume
 - Plus le volume est important, plus la couleur est rouge
 - Plus le volume est faible, plus la couleur est verte
 - Voici un exemple des proportions utilisées:

Volume de la requête > 30 % du volume total de la page → Rouge

10 % du volume total de la page < Volume de la requête < 30 % du volume total de la page → Orange

5 % du volume total de la page < Volume de la requête < 10 % du volume total de la page → Vert

Volume de la requête < 5 % du volume total de la page → Gris

- ⇒ Récupération et calcul de plusieurs données
 - Nombre de requêtes
 - Km octet parcourus = octets transportés * distance parcourue jusqu'au demandeur
 - KWh : à calculer (pas de récupération)
 - Prix : idem
- ⇒ Après chaque utilisation individuelle, un mail est envoyé à l'utilisateur avec le résultat

Sur la Map Globale :

- ⇒ Permet de visualiser sur une nouvelle carte (map globale) la moyenne de chacune des informations stockées dans la base de données (c'est une moyenne de tous les sites qui ont été enregistrés dans la map individuelle).
 - Nombre de sites = somme des sites ayant accédés à cette région, ville
 - Nombre de requêtes = somme des requêtes ...
 - Nombre de Km total parcourus = somme des
 - Nombre de Km octet parcourus moyen par site
- ⇒ Une liste des sites optimisés et les moins optimisés doit être présente.

Sur Map Globale et individuelle :

- ⇒ Contexte : Chacune des cartes doit pouvoir être filtrée selon un événement
 - Digital Week
 - Green Week
 - Green Code Lab Challenge

Avec plusieurs dates :

- 2016
- 2017
- 2018

Ces filtres ne sont pas modifiables par les utilisateurs mais par l'administrateur seulement.

- ⇒ Un système permettant de switcher entre les deux cartes doit être mis en place.

Base de données :

- ⇒ Stockage des données récupérées dans une base de données (Création d'une base de données)
- ⇒ Base de données : utilisation de mongoDB

Page des renseignements :

- ⇒ La personne qui saisit son site web doit également saisir son mail et la volumétrie en nombre de pages lues par mois

Langues :

- ⇒ Application totalement en Anglais

Aspect technique :

- ⇒ Application en autonomie totale afin de favoriser son utilisation et de limiter le temps d'attente (mais qui utilise au maximum l'architecture du WEA afin de simplifier une éventuelle intégration)
- ⇒ On relie l'application et le WEA via un lien et on intégrera le code par la suite si besoin.
- ⇒ Langage utilisé : PHP
- ⇒ Charte graphique utilisée identique à celle du WEA afin d'être la plus homogène possible.

Gestion des erreurs :

- ⇒ Utilisation du même système de gestion d'erreurs que le WEA.

Autres :

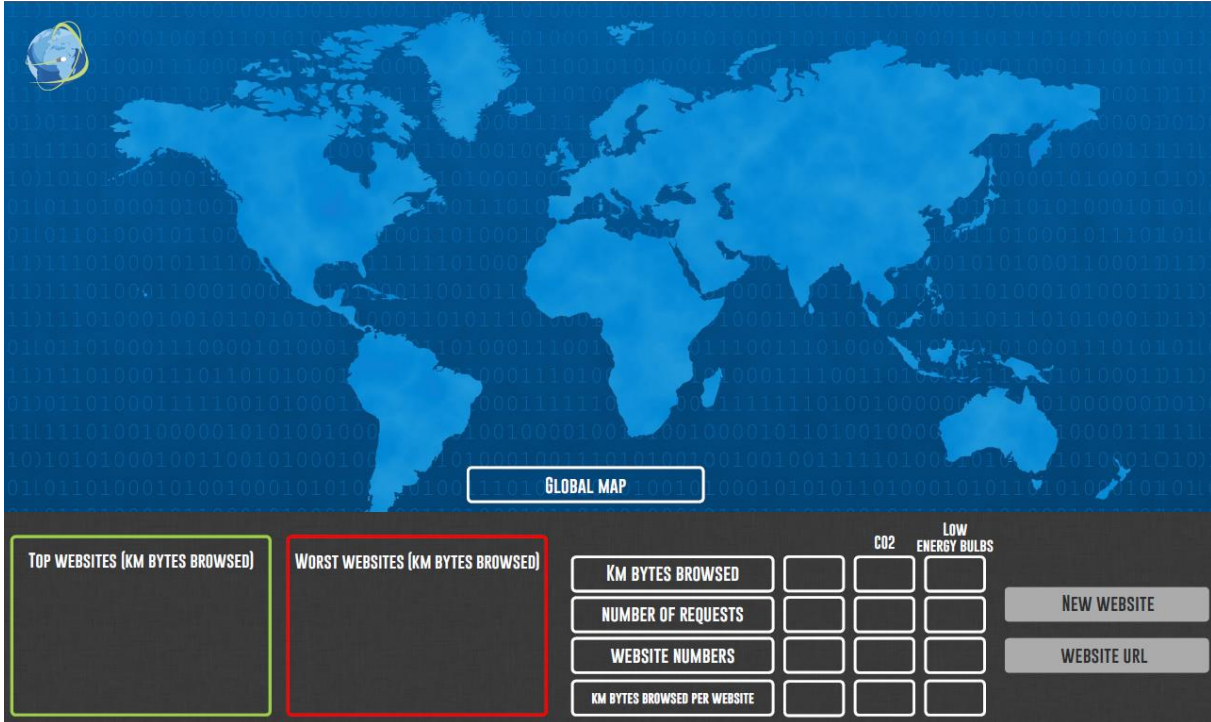
- ⇒ Utilisation d'une carte libre de droit
- ⇒ Récupération des coordonnées des serveurs via un programme.

Plan de l'application :

Plan Individuel :



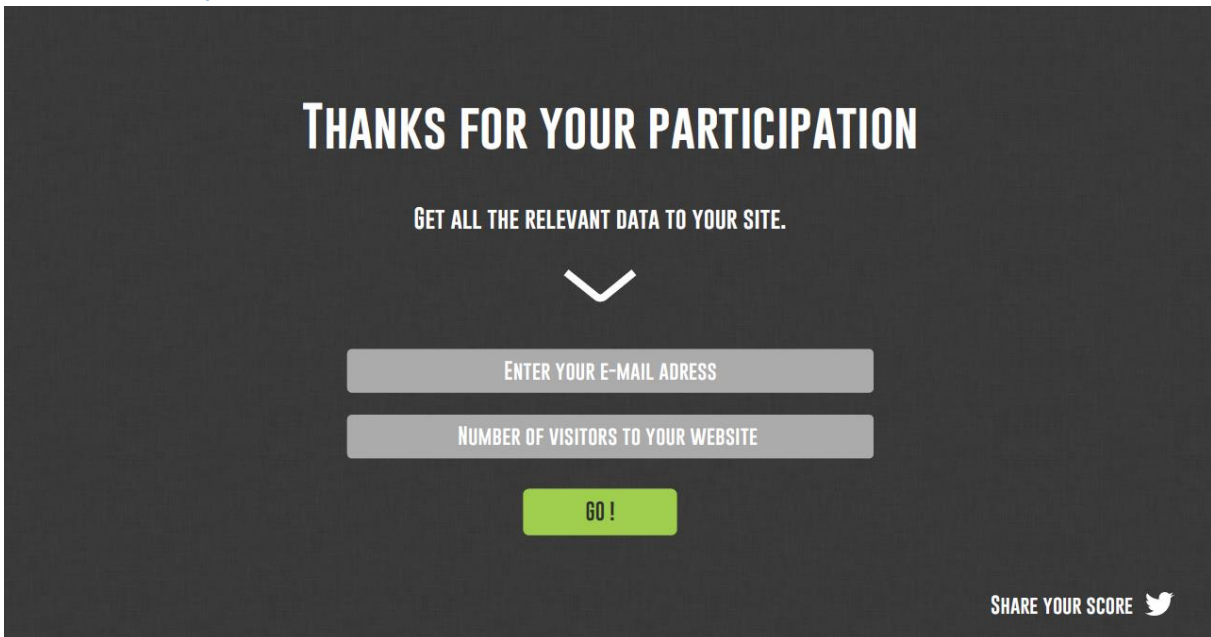
Plan Global :



GLOBAL MAP

TOP WEBSITES (KM BYTES BROWSED)	WORST WEBSITES (KM BYTES BROWSED)	KM BYTES BROWSED	NUMBER OF REQUESTS	WEBSITE NUMBERS	KM BYTES BROWSED PER WEBSITE	CO2	LOW ENERGY BULBS	NEW WEBSITE	WEBSITE URL

Fenêtre à remplir :



THANKS FOR YOUR PARTICIPATION

GET ALL THE RELEVANT DATA TO YOUR SITE.

ENTER YOUR E-MAIL ADDRESS

NUMBER OF VISITORS TO YOUR WEBSITE

GO !

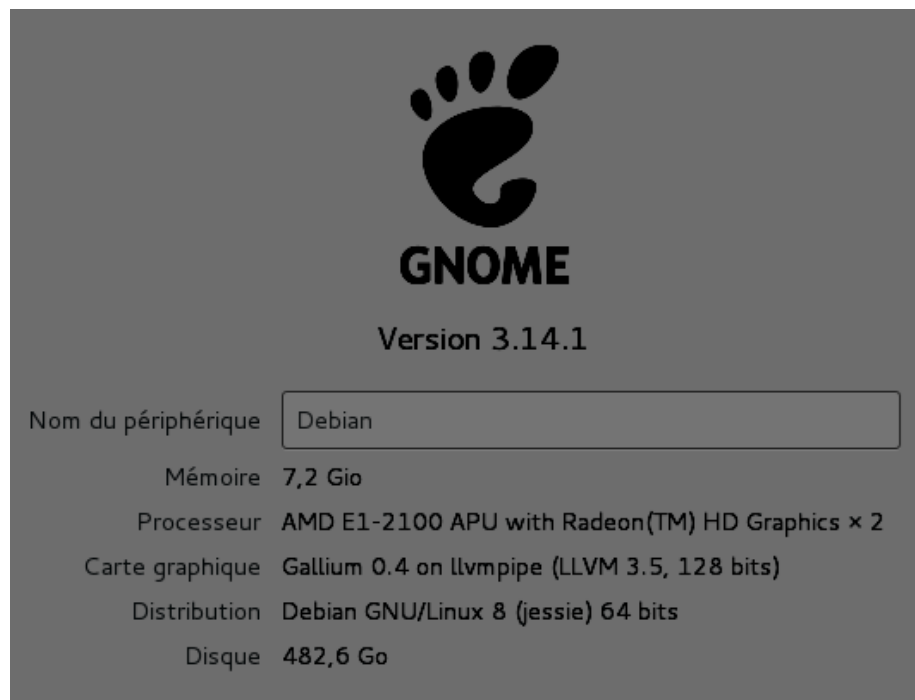
SHARE YOUR SCORE

Environnement de travail

Système d'exploitation:

Afin de réaliser ce projet, j'ai travaillé tout au long de mon stage sur Debian 8. J'ai commencé avec Debian7 mais j'étais dans l'obligation d'utiliser un câble filaire afin d'avoir une connexion internet or avec mon passage de 3 semaines à SIGMA, le filaire n'étant pas possible dû à la politique sécuritaire de l'entreprise, je suis passé à Debian 8 afin de pouvoir installer le firmware-linux-nonfree qui permet d'avoir la wifi.

Voici les caractéristiques de mon ordinateur sur lequel se déroule l'intégralité du projet :



Afin de garder Windows 8.1 sur mon ordinateur j'ai utilisé un dual boot UEFI via une installation par clé usb 8gb.

De cette façon, j'ai pu travailler correctement sans bug tout au long, ce qui n'aurait pas été le cas avec une machine virtuelle car mon processeur est assez faible.

Installation de Xampp:

Afin de réaliser mon projet, j'ai utilisé une base de données, j'ai exécuté des codes PHP etc.

Pour ce faire, j'ai pris l'initiative d'installer la packet xampp afin de posséder php5 ainsi que MySQL.

Pour ce faire je l'ai téléchargé via le lien suivant :

<https://www.apachefriends.org/fr/download.html>

L'installation peut se faire en mode graphique de manière très simple et ne nécessite pas de connaissance particulière.

Voici ci-dessous, la version du xampp installé :

Version	Somme de contrôle	Taille
5.5.28 / PHP 5.5.28 Ce qui est inclus :	md5 sha1	Télécharger (32 bit) 125 Mb
	md5 sha1	Télécharger (64 bit) 129 Mb
5.6.12 / PHP 5.6.12 Ce qui est inclus :	md5 sha1	Télécharger (32 bit) 126 Mb
	md5 sha1	Télécharger (64 bit) 132 Mb
7 / PHP 7 Coming soon	In the mean time, try Bitnami LAMP for PHP 7.0	

Afin de pouvoir travailler, il faut démarrer les services qui composent xampp.

Pour ce faire, voici la commande :

```
/opt/lampp/lampp start
```

Voici un imprime écran du résultat:

```
root@Debian:/opt/lampp/htdocs/projet/Sites/listing/www.leboncoin.fr# /opt/lampp/lampp start
Starting XAMPP for Linux 5.6.11-0...
XAMPP: Starting Apache...ok.
XAMPP: Starting MySQL...ok.
XAMPP: Starting ProFTPD...ok.
```

Outils de programmations :

Le projet contenant une grande partie de codes et avec plusieurs langages différents, j'ai décidé d'utiliser le logiciel Jedit afin de pouvoir coder de manière optimale.

Ce logiciel permet de simplifier grandement la compréhension d'un code sur Linux.

L'installation se fait de la manière très simple via le site web suivant :

<http://jedit.org/index.php?page=download>

On peut aussi utiliser l'outil nano qui est automatiquement installé ou qui se fait via la commande suivante :

```
apt-get install nano
```

Utilisation des scripts

Afin de récupérer les données nécessaires au fonctionnement de la carte et de ces trajectoires ainsi que les données que nous voulons afficher dans notre page web, j'ai utilisé plusieurs scripts qui ont en partie été programmés par une équipe l'année dernière et que j'ai modifié afin de les adapter à mon projet.

Pour le bon fonctionnement, j'ai aussi codé d'autres programmes afin de coordonner facilement l'ensemble des scripts avec la base de données.

L'ensemble des scripts se trouvent en intégralité dans les annexes 1 à 10.

Le premier script utilisé est "debut_script_recup.py".

Script 1: debut_script_recup.py

Ce script est le premier exécuté, en effet, lorsqu'un utilisateur se trouve sur la page web et désire faire une recherche sur un site web donné, lorsqu'il note son site et appui sur valider, ce script est lancé.

debut_script_recup.py a été écrit en langage python, il permet de récupérer le nom du site que l'utilisateur a noté mais aussi d'exécuter le script suivant.

En effet, afin que tout se fasse de manière automatique, chaque script exécute un autre script lorsqu'il a terminé son travail.

Ce premier script exécute url.bash lorsqu'il a terminé la récupération du nom.

Script 2: url.bash

Ce script est très important dans l'organisation de notre application car il permet de créer un dossier avec le nom du site web donné.

Après la création, il copie plusieurs autres scripts afin que l'exécution du prochain script se fasse via le script qui a été copié.

C'est une manière qui nous permet d'éviter que les autres utilisateurs soient bloqués lorsqu'un utilisateur utilise déjà notre application pour un site web donnée.

En effet, le temps mis par l'exécution de certains scripts est très important et nous ne pouvons donc pas nous permettre de faire patienter un utilisateur très longtemps avant que sa requête soit examinée.

Ce script permet également de découper l'adresse du site rencontré afin de la garder sous forme de : www.adresse.fr au lieu de http://www.adresse.fr

Pour finir, certains scripts ne pouvant être exécuté sans certaine permission, celui-ci s'occupe de donner les droits nécessaires à leurs exécutions.

Explication du code :

Cette partie permet le découpage de l'adresse et de l'afficher après sous sa nouvelle forme.

```
nomsite=$(echo $1 | cut -d "/" -f3)
echo $nomsite
```

Création du dossier avec le nom du site web découpé.

```
mkdir /opt/lampp/htdocs/projet/Sites/$nomsite
```

Copie des différents scripts

```
cp /opt/lampp/htdocs/projet/scripts/reseauWea.bash /opt/lampp/htdocs/projet/scripts/geo.sh /opt/lam
cp /opt/lampp/htdocs/projet/scripts/geo.sh /opt/lampp/htdocs/projet/scripts/gestionDonneesGeo.py /o
```

Déplacement dans le dossier qui contient tous les scripts déjà copié.

```
cd /opt/lampp/htdocs/projet/Sites/$nomsite
```

Obtention des droits d'exécution :

```
chmod 755 reseauWea.bash script_recup.py traceroute.bash
```

Exécution du script suivant :

```
./script_recup.py $1
```

Script 3 : script_recup.py

Ce script est le plus important de tous, c'est lui qui nous permet de récupérer le fichier request.csv contenant une grande partie des données qui nous intéresse.

Sans celui-ci, le script suivant ne pourra pas s'exécuter.

script_recup.py lance le script reseauWEA.bash afin de permettre l'analyse des données qui ont été récupérées via le site web "WEB PAGE TEST"

En effet, web page test est un site internet qui permet d'examiner une url afin d'en obtenir plusieurs données, nous allons donc utiliser ce site via nos scripts afin de récupérer ces données sous forme de csv.

Voici le lien du site web de Web Page Test : <http://www.webpagetest.org/>

Ensuite, notre script prend toutes les données et les inserts dans les tables de la base de données que j'ai créée précédemment (je traiterai cette partie dans la suite du rapport)

Voici un aperçu du fichier request.csv que je récupère :

```
GNU nano 2.2.6                                Fichier : requests.csv
Date,Time,Event Name,IP Address,Action,Host,URL,Response Code,Time
08/31/2015,08:08:39,Step 1,193.164.196.82,GET,www.leboncoin.fr/,20
08/31/2015,08:08:39,Step 1,193.164.196.82,GET,www.leboncoin.fr,/css
08/31/2015,08:08:39,Step 1,193.164.196.82,GET,www.leboncoin.fr,/js/
08/31/2015,08:08:39,Step 1,93.184.216.180,GET,tags.tiqcdn.com,/utag
08/31/2015,08:08:39,Step 1,193.164.196.82,GET,www.leboncoin.fr,/js/
08/31/2015,08:08:39,Step 1,193.164.197.60,GET,static.leboncoin.fr,/
08/31/2015,08:08:39,Step 1,193.164.196.82,GET,www.leboncoin.fr,/img
08/31/2015,08:08:39,Step 1,193.164.196.82,GET,www.leboncoin.fr,/img
08/31/2015,08:08:39,Step 1,93.184.216.180,GET,tags.tiqcdn.com,/utag
```

On remarque que ce fichier contient une grande quantité d'informations comme les différents serveurs par lequel il passe afin d'obtenir notre page web, mais aussi la quantité d'énergie utilisée, leurs adresses ip etc.

On peut voir que l'exemple ci-dessus est celui du site web : www.leboncoin.fr
Néanmoins, il arrive que cette étape échoue si le site recherché est trop volumineux.

Explication du code :

Ce code est composé de 3 grandes parties :

La première s'occupe de travailler avec le site WebPageTest :

```
opener= urllib2.build_opener()
opener.addheaders = [('User-Agent', 'Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140610 F$
opener.addheaders.append(('Connection', 'Keep-Alive'))
response= opener.open('http://www.webpagetest.org/runtest.php?url='+url+'&k='+key+'&f=xml&r=12$
html= response.read()
print (html)

temp=re.search(r'<detailCSV>(.*?)</detailCSV>', html)
resCSV= temp.group(1)
print(resCSV)
time.sleep(30)
os.system("wget " +resCSV)
```

La seconde s'occupe de la mise en forme du fichier csv via l'exécution du script `reseauWea.bash` que je décrirai dans la suite du rapport.

```
os.system("./reseauWea.bash requests.csv")
```

La troisième partie s'occupe de tout ce qui a un lien avec la base de données, aussi bien au niveau de la connexion (et déconnexion) que de l'insertion des données.

Voici une partie de ce code, l'intégralité se trouvera dans les annexes.

```
try:
    conn= MySQLdb.connect(db= 'project', user= 'root', passwd= '', host='127.0.0.1$
    print ("connexion reussie")
except:
    print ("La connexion avec la database a fail")
else:
    cursor= conn.cursor()    # creation du curseur

reader= csv.reader(open("reseauWea.csv", "rU"), delimiter=';', dialect= csv.DictReader)
for row in reader:
    #print ("voila le row")
    #print (row)

    if row[0] != 'url':
        url= row[0]
        time= row[1]
        Bodysize= row[2]
        Mime= row[3]
        Trame= row[4]
```

Script 4 : `reseauWea.bash`

Ce script est lancé avant même que le script `script-recup.py` ne se termine, en effet, c'est grâce à ce script que nous pouvons traiter les données du fichier `request.csv` afin qu'elles puissent être utilisées par la suite du script `script_recup.py`.

Je récupère donc les données qui m'intéressent via ce script :

Voici un échantillon des données récupérées et leurs significations:

- ➔ Nombres de sauts: Cette valeur représente le nombre d'équipements qui ont été traversés pour l'affichage du site web.
- ➔ Temps de réponse: Cette valeur correspond au temps de réponse d'une requête.
- ➔ MTU: Cette valeur correspond à la quantité maximale d'octets pouvant être transportés par une trame.
- ➔ Nombre de trame par requête : cette valeurs n'est pas trouvée directement via le fichier `request.csv` mais plutôt grâce au calcul $\text{respBodySize} / \text{MTU}$

Beaucoup d'autres données sont récupérées, afin de les voir, il suffit de regarder le fichier qui est généré après le traitement du fichier `request.csv`.

Ce fichier se nomme `reseauWea.csv`

Voici un aperçu d'une partie de ce fichier:

```
url;time;respBodySize;mineType;nbTrame;nbHops;timeByTrame;TimeByHops;Conso
www.leboncoin.fr;150;8279;text/html;6;12;25.000;2.083;.006749
www.leboncoin.fr;139;1773;max-age=604800;2;12;69.500;5.791;.012327
www.leboncoin.fr;404;35321;max-age=604800;24;12;16.833;1.402;.011540
tags.tiqcdn.com;57;10717;max-age=300;8;13;7.125;.548;.002156
www.leboncoin.fr;180;4589;max-age=604800;4;12;45.000;3.750;.010066
static.leboncoin.fr;345;5824;max-age=172800;4;12;86.250;7.187;.019292
www.leboncoin.fr;139;1205;max-age=604800;1;12;139.000;11.583;.021439
www.leboncoin.fr;131;1438;max-age=604800;1;12;131.000;10.916;.020203
tags.tiqcdn.com;39;1348;max-age=2592000;1;13;39.000;3.000;.005982
static.leboncoin.fr;535;115;max-age=172800;1;12;535.000;44.583;.082522
static.leboncoin.fr;240;5854;max-age=172800;4;12;60.000;5.000;.013425
static.leboncoin.fr;522;133;max-age=172800;1;12;522.000;43.500;.080518
static.leboncoin.fr;255;4277;max-age=172800;3;12;85.000;7.083;.017047
static.leboncoin.fr;259;5621;max-age=172800;4;12;64.750;5.395;.014481
```

Afin de comprendre le fichier, il faut savoir que la première ligne donne le nom de chacune des données traitées, dans les lignes suivantes, les données sont classées dans le même ordre et séparées par un ';'.

Ce fichier est beaucoup plus organisé que le `request.csv` et ce sont ces lignes de données qui sont insérées dans notre base de données.

De plus, ce script exécute le script `traceroute.bash`

Script 5: traceroute.bash

Ce script m'est très utile au niveau de la mesure énergétique car il me permet d'avoir la consommation énergétique d'une requête grâce à la commande traceroute.

Celle-ci permet de savoir combien d'équipements sont traversés depuis le lieu de l'exécution du programme jusqu'au point d'arrivée de la requête.

Elle me permet aussi d'avoir le temps de réponse, je pourrai donc utiliser cette valeur pour en déduire plusieurs autres via certains calculs.

Voici un imprime écran du fichier nbHopsUrl.txt qui est créer afin d'avoir ces informations :

```
56c646cc9f58f35661191763632f428a.leboncoin.fr;13
aimfar.solution.weborama.fr;14
cis.schibsted.com;20
collector.schibsted.io;20
dlnflogr7o23z7.cloudfront.net;12
Host;
logcl53.xiti.com;20
static.leboncoin.fr;12
tags.tiqcdn.com;13
www.leboncoin.fr;12
```

On remarque que cet exemple est toujours sur le site web leboncoin.fr

Après chaque ';', le nombre qui est noté correspond au nombre de saut en fonction de l'hôte noté en début de ligne.

Jusqu'ici, j'ai récupéré plusieurs informations très utiles afin d'avoir les données sur le site web recherché. Néanmoins, il me manque encore leurs coordonnées afin de les représentés sur notre carte.

Pour ce faire, via la page PHP, j'exécute le script geo.sh (Annexe 6)

Ce script est très important car il nous permet de récupérer les coordonnées (latitude et longitude) de chaque serveur utilisé pour l'affichage du site web.

Il me permet également dans une moindre importance le pays ou la région des serveurs.

Ces coordonnées sont stockées dans le fichier cities.csv dont voici un exemple :

```
code,city,country,lat,lon,ip
N/A, N/A, FR, 48.860001, 2.350000,193.164.196.82
''''''
N/A, N/A, FR, 48.860001, 2.350000,193.164.196.120
N/A, N/A, FR, 48.860001, 2.350000,193.164.196.82
N/A, N/A, FR, 48.860001, 2.350000,193.164.197.60
WA, Washington, US, 47.610298, -122.334099,205.251.251.236
N/A, N/A, FR, 48.860001, 2.350000,62.161.94.220
N/A, N/A, FR, 48.860001, 2.350000,91.216.195.3
N/A, N/A, US, 38.000000, -97.000000,93.184.216.180
```

On remarque bien que chacune des requêtes possède ses propres coordonnées ainsi que sa propre adresse ip.

Il peut arriver que certaines informations ne soient pas trouvées d'où le N/A.

Néanmoins, afin d'utiliser ce script, il faut avoir installé au préalable le module geolookup car c'est celui-ci qui permet d'établir la géolocalisation.

Afin d'installer ce module, voici la commande :

apt-get install geolookup

D'ailleurs, nous pouvons utiliser directement cette commande via l'invite de commande afin de voir la confirmation que nos résultats sont bien les bons :































```
root@Debian:/opt/lampp/htdocs/projet/Sites/listing/www.leboncoin.fr# geolookup www.leboncoin.fr
GeoIP Country Edition: FR, France
```

On remarque ici qu'il trouve bien la France pour le site leboncoin.fr comme dans le fichier cities.csv

Après avoir lancé ce script, j'exécute le script gestionDonneesGeo.py afin d'insérer les données récupérées par geo.sh dans la base de données sur la table géolocalisation. (Attention, ce script n'est pas lancé automatiquement via le script traceroute.bash, je l'exécute via une ligne de commande à partir de la partie PHP juste après avoir exécuter le script debut_script_recup.py)

Grace à celui-ci notre base de données contient tout ce dont nous avons besoin, notamment la latitude et la longitude des différents serveurs par lesquels les requêtes passent.

Voici un imprime écran de la base de donnée après l'exécution des deux derniers scripts :

				id	ip	lat	lon
<input type="checkbox"/>		Modifier		Copier		Effacer	36 ip 0 0
<input type="checkbox"/>		Modifier		Copier		Effacer	37 193.164.196.82 48.860001 2.35
<input type="checkbox"/>		Modifier		Copier		Effacer	38 0 0
<input type="checkbox"/>		Modifier		Copier		Effacer	39 193.164.196.120 48.860001 2.35
<input type="checkbox"/>		Modifier		Copier		Effacer	40 193.164.196.82 48.860001 2.35
<input type="checkbox"/>		Modifier		Copier		Effacer	41 193.164.197.60 48.860001 2.35
<input type="checkbox"/>		Modifier		Copier		Effacer	42 205.251.251.236 47.610298 -122.334099
<input type="checkbox"/>		Modifier		Copier		Effacer	43 62.161.94.220 48.860001 2.35
<input type="checkbox"/>		Modifier		Copier		Effacer	44 91.216.195.3 48.860001 2.35
<input type="checkbox"/>		Modifier		Copier		Effacer	45 93.184.216.180 38 -97

Base de données

Dans cette partie, je vais détailler tout ce que j'ai créé afin de pouvoir gérer les données que je récolte via différents scripts.

En effet, le projet, pour fonctionner correctement et de manière autonome, a besoin d'une base de données bien structurée et facilement utilisable.

Pour ce faire, j'ai créé une base de données que j'ai nommée : project

A l'intérieur de celle-ci, j'ai créé plusieurs tables pour y stocker des données bien précises.

Table archive : celle-ci va stocker le donnée suivantes :

- url
- time
- respBodySize
- Conso
- etc.

C'est ici que l'on trouve les données que je pourrai utiliser par la suite afin d'afficher des consommations ou autre mais aussi afin de faire certains calculs pour créer de nouvelle données si besoins.

Ci-dessous un imprime écran d'une partie de la table :

			Id	url	time	respBodySize	mineType	nbTrame	nbHops	timeByTrame	TimeBy	Conso
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	119	www.google.fr	3700372	4032500	4036527	2689	20			
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	120	www.google.fr	43	0	public	1	20	43.000	2.150	.006438
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	121	www.google.fr	73	230	text/html	1	20	73.000	3.650	.010930
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	122	www.google.fr	373	0	max-age=0	1	20	373.000	18.650	.055855
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	123	www.google.fr	81	0	text/html	1	20	81.000	4.050	.012128

Table geolocalisation :

Cette table est très importante au niveau de la carte. En effet, c'est dans celle-ci que je viendrai chercher les coordonnées des différents serveurs qu'aura utilisé le site web.

			Id	code	city	country	lat	lon	ip
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	46	WA	Washington	US	47.610298	-122.334099	205.251.251.236
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	49	N/A	N/A	US	38.000000	-97.000000	93.184.216.180
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	23	CA	California	US	37.419201	-122.057404	173.194.121.15
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	29	CA	California	US	37.419201	-122.057404	74.125.228.255
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	31	CA	California	US	37.419201	-122.057404	74.125.228.255
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	33	CA	California	US	37.419201	-122.057404	74.125.228.255
<input type="checkbox"/>	Modifier	<input type="checkbox"/>	35	CA	California	US	37.419201	-122.057404	74.125.228.255

On remarque qu'on peut y trouver tout ce qui peut nous intéresser au niveau géographique aussi bien au niveau des coordonnées mais aussi au niveau du pays ou de la ville (ou région)

Partie PHP

Afin de pouvoir visualiser les trajectoires, je les ai tracées sur une carte.

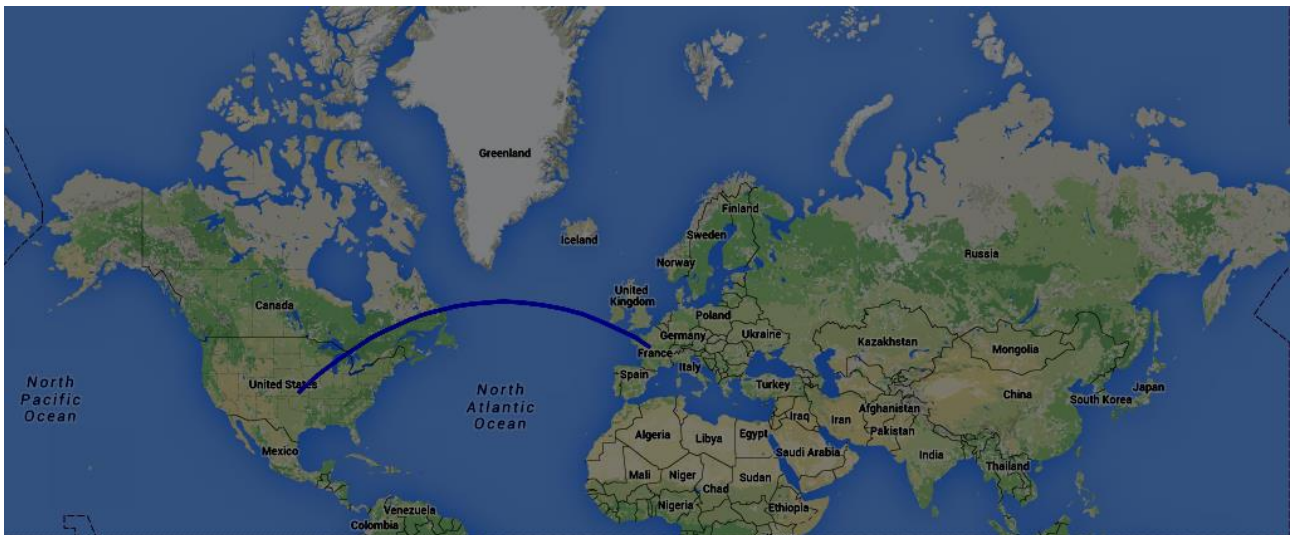
J'ai eu beaucoup de difficulté dans cette partie, en effet, j'ai commencé en essayant de travailler sur du d3.js qui nous permet de manipuler tout ce qui est cartes, graphiques, trajectoires etc.

Néanmoins, à certains moments ma carte ne s'affichait plus correctement à cause de certaines bibliothèques qui n'étaient plus disponible.

J'ai donc décidé d'utiliser l'api Google Map afin d'avoir une carte.

Dans cette carte, je peux afficher une trajectoire dont les coordonnées sont récupérées via la base de données.

Ci-dessous un imprime écran du résultat pour une requête :



Afin d'afficher cette carte, le code est contenu dans le fichier : googleMap.php

[Explication du code googleMap.php:](#)

Ce code est composé de plusieurs parties essentielles.

Attention, pour le bon fonctionnement de l'affichage, tout le code doit être correct, si une erreur apparaît lors de la récupération des données de trajectoires, même la carte ne s'affichera pas.

Première partie :

Cette partie a une importance capitale, car c'est ici que je me connecte à la base de données afin de récupérer les coordonnées que je recherche dans la table 'geolocalisation' et de les stocker dans les variables lon et lat.

```
<?php
//connexion a la base de données
$bdd = new PDO('mysql: host=localhost;dbname=project','root', '') or die ("Connexion à la base SQL impossible");
# $select = mysql_select_db ('project', $connect) ;
echo "connexion à la base de données réussie<br>";

$requete = $bdd -> query('SELECT * FROM geolocalisation2');

while ($data = $requete -> fetch())
{
// echo '<script type="text/javascript">console.log("OK")</script>';
echo '<h2>'. $data['lat']. '</h2>';
$lat = $data['lat'];
echo '<h2>'. $data['lon']. '</h2>';
$lon = $data['lon'];
}
?>
```

Seconde partie :

Dans cette partie, j'ai utilisé du JavaScript afin de matérialiser la carte et les trajectoires.

Dans cette partie, je récupère les valeurs lat et lon que j'ai créé dans la première partie. En effet, dans la première partie, je les ai récupérées dans la base de données via le PHP, or je ne peux pas les utiliser dans la partie du code JavaScript, d'où cette étape. La suite nous permet de déterminer l'emplacement des points via leurs coordonnées. On remarque que le point d'arriver voit ses coordonnées changées via les variable lat et lon.

```
function initialisation () {

    lon = <?php echo $lon; ?>;
    lat = <?php echo $lat; ?>;

    //console.log(lon + " " + lat);

    var pointDepart = new google.maps.LatLng(47.390273,0.688834);
    var pointArrivee = new google.maps.LatLng(lat, lon);
    var limites = new google.maps.LatLngBounds();
    limites.extend(pointDepart);
    limites.extend(pointArrivee);
    limites.extend(new google.maps.LatLng(70.407348,66.796875));
    ...
}
```

Troisième partie :

Dans cette partie, nous dessinons les trajectoires dont les points ont été trouvés précédemment.

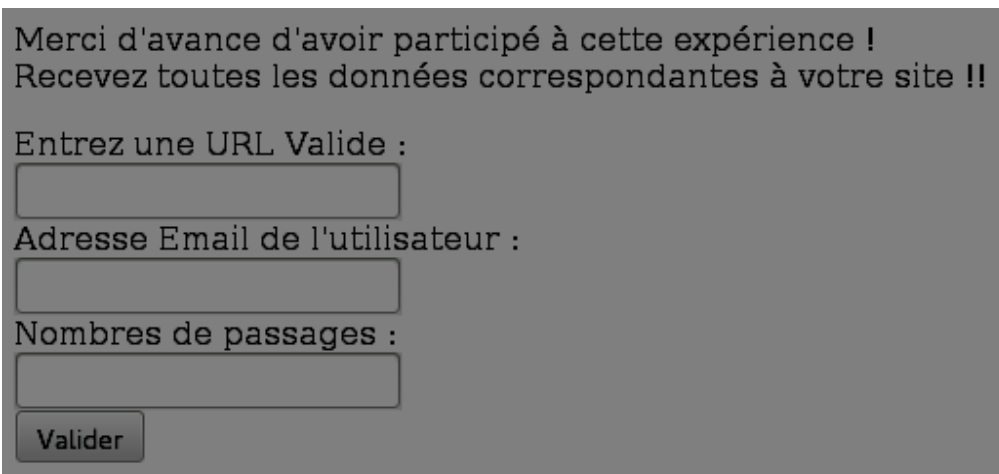
Polyline est le terme qui permet d'avoir les trajets sous forme de parabole.

```
var nordVraiCap = new google.maps.Polyline({map:maCarte,path:[pointDepart,pointArrivee],strokeColor:"#0000FF",geodesic:true});
document.getElementById("origineLat").innerHTML = pointDepart.lat().toFixed(6);
document.getElementById("origineLng").innerHTML = pointDepart.lng().toFixed(6);
document.getElementById("destinationLat").innerHTML = pointArrivee.lat().toFixed(6);
document.getElementById("destinationLng").innerHTML = pointArrivee.lng().toFixed(6);
maCarte.fitBounds(limites);
```

Avec googleMap.php, je peux afficher la carte ainsi que les trajectoires, or j'ai besoin d'une interface afin de donner l'opportunité à l'utilisateur d'écrire le site qu'il désire analyser, la boîte mail où il voudrait recevoir le résultat final ainsi que le nombre de fois il va sur le site en question (ce chiffre me permettra de faire certains calculs).

Pour faire cette étape, j'ai codé la page : informations.php

Voici un imprime écran du résultat :



Vous pouvez remarquer qu'il n'y a pas de partie graphique, je n'ai pas codé le css. En effet, cette partie étant la plus simple, je l'ai laissée de côté pour que ceux qui continueront le projet le fassent par la suite.

De plus, j'ai transmis l'intégralité du code à Pablo BARRAL de l'entreprise WOUEP afin qu'il travaille sur la partie graphique en respect de la charte graphique de site web : Web Energie Archive.

[Explication du code informations.php :](#)

Ce code est composé de trois parties majeures :

Première partie :

C'est le formulaire, c'est celui-ci qui affiche les zones de texte ainsi que le bouton permettant la validation de ce qui a été saisi.

```
<p>
  <form method="post" action="index.php" >
    Entrez une URL Valide : </br>
    <input type="text" name="urlinput" id="urlinput1"/> </br>

    Adresse Email de l'utilisateur : </br>

    <input type="text" name="mail" id="mail" /></br>
    Nombres de passages : </br>
    <input type="text" name="passages" /></br>

    <input type="submit" name="Valider" value="Valider" />

  </form>
</p>
```


C'est après l'appui de ce bouton "valider" que les différents scripts que j'ai expliqué précédemment sont exécutés.

Deuxième partie :

C'est le code permettant de gérer tout ce qui a un lien avec la base de données, en effet, c'est ici que nous faisons la connexion à la base de donnée mais aussi que nous y insérons les données entrées dans les zones de texte.

Cette étape se fait à condition que le bouton « valider » ait été activé.

Ci-dessous un imprime écran de cette partie du code :

```
<?php
//connexion a la base de données
$connect = mysql_connect ('localhost','root', '') or die ("Connexion à la base SQL impossible");
$select = mysql_select_db ('project', $connect) ;

echo "connexion à a base de données réussie<br>";

// si le bouton valider a été activé
if (isset($_POST['Valider']))
{
    //on insert la valeur du champs mail dans la table mail de la base de données
    $sql = 'INSERT INTO mail
            (mail)
            VALUES
            ("'. $_POST['mail'] .'");';
    $result = mysql_query($sql)
    or die('<font color="red">Error :</font> on line <b>'. __LINE__ . '</b><br />'.mysql_error());

    //on insert la valeur du champs passages dans la table passages de la base de données
    $sql = 'INSERT INTO passages
            (passages)
            VALUES
            ("'. $_POST['passages'] .'");';
    //permet de voir les erreurs sur la gestion de base de données
    $result = mysql_query($sql)
    or die('<font color="red">Error :</font> on line <b>'. __LINE__ . '</b><br />'.mysql_error());

    //on insert la valeur du champs url dans la table url de la base de données
    $sql = 'INSERT INTO url
            (url)
            VALUES
            ("'. $_POST['urlinput'] .'");';
    $nomSite = $_POST['urlinput'];
    $result = mysql_query($sql)
    or die('<font color="red">Error :</font> on line <b>'. __LINE__ . '</b><br />'.mysql_error());
```

Troisième partie :

Cette étape est cruciale car elle fait le lien entre nos pages web et les différents scripts que j'ai présentés.

En effet, après l'insertion des données dans notre base, nous exécutons les différents scripts qui nous intéressent.

```
//$debut = exec('python /opt/lampp/htdocs/projet/debut_script_recup2.py www.google.fr');
system("python /opt/lampp/htdocs/projet/debut_script_recup2.py ".$_POST['urlinput']);
system("sh /opt/lampp/htdocs/projet/Site/listinng/$nomSite/geo.sh requests.csv");
system("python /opt/lampp/htdocs/projet/Site/listing/$nomSite/gestionDonneesGeo.py cities.csv");
echo '<br> les scripts sont exécutés';
```

Continuité du projet

Le projet est très ambitieux, c'est la raison pour laquelle Mr LEBOUQCQ m'as demandée un cahier des charges très détaillé.

C'est donc un projet qui va être continué par d'autres personnes, moi-même j'ai continué à partir de certains travaux déjà effectués.

Pour faire cette transaction, j'ai donc réfléchi à un moyen qui permettrait à mon successeur de ne pas perdre de temps.

Le projet n'étant pas à but lucratif et sachant que toutes les personnes ayant travaillé dessus étaient d'accord de le mettre en partage, j'ai décidé de le placer dans un compte Github.

Github est une interface de partage très utilisée dans le milieu du développement.

J'ai donc créé un compte Github afin d'y placer l'intégralité de mon projet.

Voici le lien : <https://github.com/abenamor/oeuvre>

Via ce lien, on peut trouver tous mes exécutables (code PHP, python, JavaScript, bash etc.)

Bilan et analyse

Comme bilan, j'ai effectué dans la première partie de mon stage le cahier des charges du projet dans son intégralité.

J'ai réussi à récupérer les données recherchées à partir de script existant et en développant de nouveaux scripts.

De plus j'ai réussi à mettre en place une carte avec l'api googleMap ainsi qu'à y afficher des trajectoires en fonction de coordonnées précises, celle-ci fonctionnent en statique pour plusieurs trajectoires, mais en dynamique, je n'en affiche qu'une seule à ce jour.

Ensuite j'ai créé la page informations.php qui permet à l'utilisateur d'insérer les données requises pour le bon fonctionnement de l'application.

J'ai fait le lien entre la partie PHP que verront les utilisateurs et les scripts python, bash etc.

J'ai créé la base de données permettant le stockage de toutes les données récupérées via les scripts ainsi que celles écrites par l'utilisateur.

J'ai préparé la continuité du projet en rédigeant une procédure expliquant tout le travail fait, ce qui reste à faire et comment il faudrait le faire.

Pour terminer, j'ai tout placé dans git hub afin de partager l'intégralité de mon travail.

Je pense donc que j'ai effectué une bonne quantité de travail en vue du temps que j'avais néanmoins, ceux qui poursuivront ce projet auront encore du travail notamment dans la finalisation et l'amélioration de la carte ainsi que la gestion des différentes trajectoires afin de toutes les afficher dynamiquement.

Pour bien comprendre ce projet et gagner en temps, il ne faut pas hésiter à me contacter, je répondrai à toutes les questions dans la mesure du possible.

Voici l'adresse mail à utiliser : adel15@hotmail.fr

Annexes

Bibliographie:

Projet "Analyse de la consommation réseaux des sites internet" : Nantes, 2014, Thomas PERPOILE et son équipe.

<http://d3js.org/>: 2015, Mike BOSTOCK.

<http://www.alsacreations.com/tuto/lire/926-geolocalisation-geolocation-html5.html>: Strasbourg, 2011, DEW.

<http://www.html5-css3.fr/html5/tutoriel-geolocalisation-html5>: Septembre 2010, Jonathan VERRECCHIA.

<http://www.html5-css3.fr/demo/exemple-geolocalisation-html5>: Septembre 2010, Jonathan an VERRECCHIA.

<http://dev.maxmind.com/gеоip/legacy/install/city/>: 2012/2015, MaxMind

<https://www.youtube.com/watch?v=H3WsXg622WA>: 2013, d3Vienno.

<http://www.touraineverte.com/google-maps-geometry-library/racine/calculer-position-point-intermediaire-interpolate.html>: 2015, Loire Touraine verte.

<http://www.lije-creative.com/creer-carte-google-maps-personnalisee/>: 25 Septembre 2012, Jérôme.

<https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql>: 2015, Mathieu Nebra.

<http://php.net/manual/fr/function.json-decode.php>: 2001/2015, The PHP Group.

<http://www.commentcamarche.net/faq/9558-sed-introduction-a-sed-part-iii>: Septembre 2015, Jean François PILLOU.

Annexe 1 : Debut script recup.py

```
#!/usr/bin/env python2
import sys, urllib2, re, time, os, csv, MySQLdb
try:
    sys.argv[1]
    url= sys.argv[1]
except:
    print("Need an argument")
    sys.exit(2);
os.system("./scripts/url.bash " +url)
```

Annexe 2 : url.bash

```
#!/bin/bash
nomsite=$(echo $1 | cut -d "/" -f3)
echo $nomsite
mkdir /opt/lampp/htdocs/projet/Sites/$nomsite
cp /opt/lampp/htdocs/projet/scripts/reseauWea.bash /opt/lampp/htdocs/projet/scripts/geo.sh /opt/lampp/htdocs/projet/scripts/script_recup.py /opt/lampp/htdocs/projet/Sites/$nomsite
cp /opt/lampp/htdocs/projet/scripts/traceroute.bash /opt/lampp/htdocs/projet/scripts/gestionDonneesGeo.py /opt/lampp/htdocs/projet/Sites/$nomsite
cp /opt/lampp/htdocs/projet/scripts/geo.sh /opt/lampp/htdocs/projet/scripts/gestionDonneesGeo.py /opt/lampp/htdocs/projet/Sites/$nomsite
cd /opt/lampp/htdocs/projet/Sites/$nomsite

chmod 755 reseauWea.bash script_recup.py traceroute.bash

#chmod u+s,gs reseauWea.bash script_recup.py traceroute.bash
./script_recup.py $1
```

Annexe 3 : script

```
#!/usr/bin/env python2
import sys, urllib2, re, time, os, csv, MySQLdb

key= '32ecef7459594891b8ca53cc08aecc7d'
try:
    sys.argv[1]
    url= sys.argv[1]
except:
    print("Need an argument")
    sys.exit(2);

#### RECUPERATION DEPUIS WEBPAGETEST ####
opener= urllib2.build_opener()
opener.addheaders = [('User-Agent', 'Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140610 Firefox/24.0 Iceweasel/24.6.0')]
opener.addheaders.append(('Connection', 'Keep-Alive'))
response= opener.open('http://www.webpagetest.org/runtest.php?url='+url+'&k='+key+'&f=xml&r=12345')
html= response.read()
print (html)

temp=re.search(r'<detailCSV>(.*?)</detailCSV>', html)
resCSV= temp.group(1)
print(resCSV)
time.sleep(30)
os.system("wget " +resCSV)

#### MISE EN FORME CSV ####
os.system("./reseauWea.bash requests.csv")

#### DATABASE PART ####
try:
    conn= MySQLdb.connect(db= 'project', user= 'root', passwd= '', host='127.0.0.1')
    print ("connexion reussie")
except:
    print ("La connexion avec la database a fail")
else:
    cursor= conn.cursor() # creation du curseur

reader= csv.reader(open("reseauWea.csv", "rU"), delimiter=';', dialect= csv.DictReader)
```

```

for row in reader:
    #print ("voilà le row")
    #print (row)

    if row[0] != 'url':
        url= row[0]
        time= row[1]
        Bodysize= row[2]
        Mime= row[3]
        Trame= row[4]
        Hops= row[5]
        TTrame= row[6]
        TBy= row[7]
        Conso= row[8]

    if row[0] == 'Total':
        requete="UPDATE historique SET consommation=" +row[8] +"' WHERE site=" +sys.argv[1] +""
        cursor.execute(requete)
    else :
        requete="INSERT INTO archive (url, time, respBodySize, mineType, nbTrame, nbHops, timeByTrame, TimeBy, Conso)
        VALUES('"+url+"', '"+time+"', '"+Bodysize+"', '"+Mime+"', '"+Trame+"', '"+Hops+"', '"+TTrame+"', '"+TBy+"', '"+Conso+"')""
        print (requete)
        cursor.execute (requete)

conn.commit()

cursor.execute ("SELECT * FROM archive")
results = cursor.fetchall()
for row in results:
    print (row)

cursor.close ()
conn.close ()
  
```

Annexe 4 : reseauWea.bash

```

#!/bin/bash
rm reseauWea.csv
echo -e "\nINITIALISATION DES VARIABLES"
sed = $1 | sed 'N;s/\n/,/'> modif.csv
#sed -i 's/"//g' modif.csv
mtu=1500
metro="0.000143"
core="0.000251"
box="0.000278"
consoTotal="0.000000"
timeTotal=0
respBodySizeTotal=0
nbTrameTotal='0'
i=0

# RECUPERATION DES NUMEROS DES COLONNES
echo -e "\nAJOUT DES NUMEROS"
echo -e "\n"
awk 'NR==1' modif.csv > ligne.csv
#sed -i 's/"//g' ligne.csv

echo -e "\nLANCEMENT DU TRACEROUTE"
echo -e "\n"
if [ -e reseauWea.csv ];
then
rm reseauWea.csv
fi
./traceroute.bash modif.csv

echo -e "\nINITIALISATION DU FICHIER CSV"
echo -e "\n"

echo "url;time;respBodySize;mineType;nbTrame;nbHops;timeByTrame;TimeByHops;Conso" > reseauWea.csv

echo -e "\nDEBUT DU TRAITEMENT"
echo -e "\n"

while read device
do
  
```

```
do

# recuperation des variable du fichier csv"

url=$(echo $device | cut -d"," -f7 )
echo $url
temp=$(echo $device | cut -d"," -f10)
echo $loadms
respBodySize=$(echo $device | cut -d"," -f15)
echo $respBodySize
typeMine=$(echo $device | cut -d"," -f20)
echo $typeMine
nbTrame=$(echo "($respBodySize/$mtu)+1" | bc)
#urlSimple=$(echo $device | cut -d"," -f$URL | cut -d"/" -f3)
urlSimple=$(echo $device | cut -d"," -f7 )
echo $typeMine

# definition du nombre de sauts"

while read ligne
do

testUrl=$(echo $ligne | cut -d";" -f1)
if [[ $urlSimple == $testUrl ]]
then
    nbHops=$(echo $ligne | cut -d";" -f2)
    #echo $nbHops
    echo ok
fi

done < nbHopsUrl.txt

# calcul des différent temps"
if [[ "$temp" -gt "1000" ]]
then
timeByTrame=$(0)
else
timeByTrame=$(echo "scale=3 ;$temp/$nbTrame" | bc)
fi
timeByHops=$(echo "scale=3 ;$timeByTrame/$nbHops" | bc)
```

```
#calcul de la consommation

if [[ "$timeByTrame" -gt "1000" ]]
#then
#conso="0"
#else
conso=$(echo "scale=6 ;((( $nbHops - 2) * $timeByHops * $metro) + ($timeByHops * $metro) + ($timeByHops * $box) * $nbTrame)" | bc)
#fi

#generation du fichier csv

echo $urlSimple";"$temp";"$respBodySize";"$typeMine";"$nbTrame";"$nbHops";"$timeByTrame";"$timeByHops";"$conso" >> reseauWea.csv

#calcul des totaux"

timeTotal=$(echo "scale=6 ; ($timeTotal + $temp)" | bc)
consoTotal=$(echo "scale=6 ; ($consoTotal + 0$conso)" | bc)

done < modif.csv

# ajout des totaux au fichier csv"
echo -e "\nFIN DU TRAITEMENT"
echo -e "\n"

echo "Total;"$timeTotal";";";";"$consoTotal >> reseauWea.csv
echo $consoTotal > total
sed -i '2d' reseauWea.csv
awk 'NR==2' reseauWea.csv > temp.csv
nomsite=$(cat temp.csv | cut -d";" -f1)
mkdir /opt/lampp/htdocs/projet/Sites/listing/$nomsite
mv -f ligne.csv modif.csv total nbHopsUrl.txt requests.csv /opt/lampp/htdocs/projet/Sites/listing/$nomsite
cp -f reseauWea.csv /opt/lampp/htdocs/projet/Sites/listing/$nomsite
rm temp.csv

exit
```

Annexe 5 : traceroute.bash

```
#!/bin/bash
sed -i 's/"//g' modif.csv
while read device
do
url=$(echo $device | cut -d";" -f7)
echo $url >> url.txt

done < $1

sort -u url.txt > tracert.txt

while read device
do

traceroute -m20 -I $device > temp.txt
nbHops=$(awk 'END {print}' temp.txt | cut -d\ -f1)
if [[ $nbHops == "" ]]
then
nbHops=$(awk 'END {print}' temp.txt | cut -d\ -f2)
fi
echo $device";"$nbHops >> nbHopsUrl.txt

done < tracert.txt

rm tracert.txt
rm url.txt
rm temp.txt
echo "FIN DU TRACEROUTE"

exit
```

Annexe 6 : geo.sh

```
#!/bin/bash
rm cities.csv ip.csv
sed -i 's/"//g' $1
while read ligne
do
    ip=$(echo $ligne | cut -d"," -f4)
    echo $ip >> ip.csv
done < $1
sed -i '1d' ip.csv
echo "code,city,country,lat,lon,ip" > cities.csv
ip_first=$(head -n1 ip.csv)
GeoIP=$(geoiplookup -f /usr/share/GeoIP/GeoLiteCity.dat $ip_first)
code=$(echo $GeoIP | cut -d "," -f3)
city=$(echo $GeoIP | cut -d "," -f4)
country=$(echo $GeoIP | cut -d "," -f2 | cut -d ":" -f2)
lat=$(echo $GeoIP | cut -d "," -f7)
lon=$(echo $GeoIP | cut -d "," -f8)
ip=$ip_first
echo $code,$city,$country,$lat,$lon,$ip >> cities.csv
cat ip.csv | sort | uniq > tmp.csv
sed -i '/0.0.0.0/d' tmp.csv
sed -i '/$ip_first/d' tmp.csv
while read ligne
do
    GeoIP=$(geoiplookup -f /usr/share/GeoIP/GeoLiteCity.dat $ligne)
    code=$(echo $GeoIP | cut -d "," -f3)
    city=$(echo $GeoIP | cut -d "," -f4)
    country=$(echo $GeoIP | cut -d "," -f2 | cut -d ":" -f2)
    lat=$(echo $GeoIP | cut -d "," -f7)
    lon=$(echo $GeoIP | cut -d "," -f8)
    ip=$ligne
    echo $code,$city,$country,$lat,$lon,$ip >> cities.csv
done < tmp.csv
rm tmp.csv
```

Annexe 7 : gestionDonneesGeo.py

```

#!/usr/bin/env python2
import sys, urllib2, re, time, os, csv, MySQLdb
key= '32ecef7459594891b8ca53cc08aecc7d'

try:
    sys.argv[1]
    url= sys.argv[1]
except:
    print("Need an argument")
    sys.exit(2);
#### DATABASE PART ####
try:
    conn= MySQLdb.connect(db= 'project', user= 'root', passwd= '', host='127.0.0.1')
    print ("connexion reussie")
except:
    print ("La connexion avec la database a fail")
else:
    cursor= conn.cursor() # creation du curseur
    reader= csv.reader(open("cities.csv", "rU"), delimiter=',', dialect= csv.DictReader)
    for row in reader:
        print (row)
        if row[0] != 'url':
            code= row[0]
            city= row[1]
            country= row[2]
            lat= row[3]
            lon= row[4]
            ip= row[5]
            if row[0] == 'Total':
                requete="UPDATE historique SET consommation='" +row[8] +"' WHERE site='" +sys.argv[1] +'"
                cursor.execute(requete)
            else :
                requete="INSERT INTO geolocalisation (code, city, country, lat, lon, ip)
                VALUES('" +code +"', '" +city +"', '" +country +"', '" +lat +"', '" +lon +"', '" +ip +'"
                print (requete)
                cursor.execute (requete)
    conn.commit()
    cursor.execute ("SELECT * FROM geolocalisation")
    results = cursor.fetchall()
    for row in results:
        print (row)
    cursor.close ()
    conn.close ()
    print("Deconnexion reussis")

```


Annexe 8 : googleMap.php

```
<!DOCTYPE html>

<head>
<title>Titre de votre page</title>
<meta name="viewport" content="initial-scale=1.0, user-scalable=no">
<meta charset="UTF-8">
<style type="text/css">

    html {
        height: 100%
    }
    body {
        height: 100%;
        margin: 0;
        padding: 0
    }
    #EmplacementDeMaCarte {
        height: 600px
    }
    #info {
        background: #fff;
        padding: 5px;
        font-size: 14px;
        font-family: arial
    }
}

<?php
//connexion a la base de données
$dbdd = new PDO('mysql: host=localhost;dbname=project','root','') or die ("Connexion à la base SQL impossible");
# $select = mysql_select_db ('project', $connect) ;
echo "connexion à la base de données réussie<br>";

$requete = $dbdd -> query('SELECT * FROM geolocalisation2');

while ($data = $requete -> fetch())
{
    // echo '<script type="text/javascript">console.log("OK")</script>';
    echo '<h2>'. $data['lat']. '</h2>';
    $lat = $data['lat'];
    echo '<h2>'. $data['lon']. '</h2>';
    $lon = $data['lon'];
}

?>

</head>
<body>
</style>
<script type="text/javascript" src="http://maps.googleapis.com/maps/api/js?sensor=false&libraries=geometry"></script>
<script type="text/javascript">

function initialisation () {
    lon = <?php echo $lon; ?>;
    lat = <?php echo $lat; ?>;

    var pointDepart = new google.maps.LatLng(47.390273,0.688834);
    var pointArrivee = new google.maps.LatLng(lat, lon);
    var limites = new google.maps.LatLngBounds();
    limites.extend(pointDepart);
    limites.extend(pointArrivee);
    limites.extend(new google.maps.LatLng(70.407348,66.796875));
    ...
    var optionsCarte = {
        zoom: 4,
        center: pointDepart,
        mapTypeId: google.maps.MapTypeId.ROADMAP
    };
    var maCarte = new google.maps.Map(document.getElementById("EmplacementDeMaCarte"), optionsCarte);

    var nordVraiCap = new google.maps.Polyline({map:maCarte,path:[pointDepart,pointArrivee],strokeColor:"#0000FF",geodesic:true});
    document.getElementById("origineLat").innerHTML = pointDepart.lat().toFixed(6);
    document.getElementById("origineLng").innerHTML = pointDepart.lng().toFixed(6);
    document.getElementById("destinationLat").innerHTML = pointArrivee.lat().toFixed(6);
    document.getElementById("destinationLng").innerHTML = pointArrivee.lng().toFixed(6);
    maCarte.fitBounds(limites);
}

google.maps.event.addDomListener( window, "load", initialisation );

</script>
</script>
<div id="EmplacementDeMaCarte"></div>
<div id="info">
    <b>Départ</b> : Latitude : <span id="origineLat"></span> - Longitude : <span id="origineLng"></span> 
    <b>Arrivée</b> : Latitude : <span id="destinationLat"></span> - Longitude : <span id="destinationLng"></span> 
</div>
</body>
```

Annexe 9 informations.php

```

<html>
<p>
    Merci d'avance d'avoir participé à cette expérience !</br>
    Recevez toutes les données correspondantes à votre site !!</br>
</p>
<p>
    <form method="post" action="index.php" >
        Entrez une URL Valide : </br>
        <input type="text" name="urlinput" id="urlinput1"/> </br>

        Adresse Email de l'utilisateur : </br>

        <input type="text" name="mail" id="mail" /></br>
        Nombres de passages : </br>
        <input type="text" name="passages" /></br>

        <input type="submit" name="Valider" value="Valider" />
    </form>
</p>

<?php
//connexion a la base de données
$connect = mysql_connect ('localhost','root', '') or die ("Connexion à la base SQL impossible");
$select = mysql_select_db ('project', $connect) ;

echo "connexion à a base de données réussie<br>";

// si le bouton valider a été activé
if (isset($_POST['Valider']))
{
    //on insert la valeur du champs mail dans la table mail de la base de données
    $sql = 'INSERT INTO mail
            (mail)
            VALUES
            ("'. $_POST['mail']. "')';
    $result = mysql_query($sql)
    or die('<font color="red">Error :</font> on line <b>'. __LINE__ . '</b>:<br />'.mysql_error());

    //on insert la valeur du champs url dans la table url de la base de données
    $sql = 'INSERT INTO url
            (url)
            VALUES
            ("'. $_POST['urlinput']. "')';
    $nomSite = $_POST['urlinput'];
    $result = mysql_query($sql)
    or die('<font color="red">Error :</font> on line <b>'. __LINE__ . '</b>:<br />'.mysql_error());

    // $debut = exec('python /opt/lampp/htdocs/projet/debut_script_recup2.py www.google.fr');
    system("python /opt/lampp/htdocs/projet/debut_script_recup2.py ".$_POST['urlinput']);
    system("sh /opt/lampp/htdocs/projet/Site/listinng/$nomSite/geo.sh requests.csv");
    system("python /opt/lampp/htdocs/projet/Site/listing/$nomSite/gestionDonneesGeo.py cities.csv");
    echo '<br> les scripts sont exécutés';

}
else
{
    echo 'debut_script_recup.py pas encore executer,, pour ce faire, appyuer sur Valider';
}

//deconnexion de la base de données
mysql_close();
echo "insertion effectuée";
?>

<html>

```

Annexe 10 : index.php

```
<?php include("googleMap.php");
#echo $_POST['urlinput'];
?>

<?php
//connexion a la base de données
$connect = mysql_connect ('localhost','root','') or die ("Connexion à la base SQL impossible");
$select = mysql_select_db ('project', $connect) ;
// echo "connexion à a base de données réussie<br>";
?>

<p>
<form method="post" action="index.php" >
    Entrez une URL Valide : </br>

    <input type="text" name="urlinput" id="urlinput1" /></br>
    <input type="submit" name="Search" value ="Search" />

</form>

<form method="post" action="informations.php" >
    <?php // Entrez une URL Valide : </br>

    // <input type="text" name="urlinput" id="urlinput1" /></br>
    ?>
    <input type="submit" name="nouveau site" value ="Nouveau Site" />

</form>

<form method="post" action="mapGlobale.php" >
    <?php // Entrez une URL Valide : </br>

    // <input type="text" name="urlinput" id="urlinput1" /></br>
    ?>
    <input type="submit" name="Map Globale" value ="Map Globale" />

</form>

</p>
```

```
<?php
if (isset($_POST['Search']))
{
    $sql = 'INSERT INTO url
        (url)
        VALUES
        ("'. $_POST['urlinput']. "')';
    $nomSite = $_POST['urlinput'];
    // echo "voici mon echo $". $nomSite;
    $result = mysql_query($sql)
    or die('<font color="red">Error :</font> on line <b>'. __LINE__ . '</b>:<br />'.mysql_error());

    // $debut = exec('python /opt/lampp/htdocs/projet/debut_script_recup2.py www.google.fr');
    system("python /opt/lampp/htdocs/projet/debut_script_recup2.py ".$_POST['urlinput']);
    system("su sh /opt/lampp/htdocs/projet/Site/listinng/$nomSite/geo.sh requests.csv");
    system("python /opt/lampp/htdocs/projet/Site/listing/$nomSite/gestionDonneesGeo.py cities.csv");
    echo '<br> lles scripts sont exécutés';
}
else
{
    echo '<br> aucun script n\'as encore executer, pour ce faire, appyuer sur Valider';
}
?>
```

Résumé

Résumé en français:

Le projet a pour objectif de créer une œuvre éphémère qui permettrait d'analyser les requêtes utilisées pour l'affichage d'une page web que l'utilisateur aura choisi auparavant. Ces requêtes sont affichées sur une carte via Googlemap et un code couleur est utilisé pour aider à sensibiliser les utilisateurs. En parallèle, les données qui sont récoltées via plusieurs scripts sont stockées dans une base de données qui ensuite est affichée sur l'interface de l'utilisateur. Ce projet se fait en plusieurs parties, il a déjà été commencé l'année passée et je l'ai continué, j'ai laissé derrière moi tout le nécessaire pour que mon successeur puisse le terminer.

→ Mot clés : œuvre éphémère, analyse de requêtes, application web, démonstrateur green.

Résumé en anglais:

The project aims to create an ephemeral work that would analyze the queries used for displaying a web page that the user has previously selected. These requests are displayed on a map with Googlemap and a color code is used to help educate users. In parallel, the data store with several scripts are stored in a database. These data are visible in the interface of users. This project involves several parties, it has already been started last year and I continued, I left behind me all the necessary so that my successor can finish it.

→ Keywords: ephemeral work, Analysis query, web application, green demonstrator.

Résumé en espagnol:

El proyecto tiene como objetivo de crear una obra efímera que analizara las consultas utilizadas para la visualización de una página web que el usuario haya seleccionado previamente. Estas solicitudes se muestran en un mapa a través de Googlemap y un código de color se utiliza para ayudar a educar a los usuarios. Paralelamente, los datos que se recopilan a través de varias secuencias de comandos se almacenan en una base de datos que a continuación se muestra en la interfaz de usuario. En este proyecto participan varios partidos, ya se ha comenzado el año pasado y me siguió, me fui detrás de mí todo lo necesario para que mi sucesor puede terminarlo.

→ Palabras claves: obra efímera, aplicación web, demostrador verde.