

Lycée Turgot

CTRL SHIFT

69 rue de Turbigo,

75003 Paris

RAPPORT DE STAGE DEUXIÈME ANNÉE BTS SIO(SLAM)



Du 28 Novembre 2022 au 21 Janvier 2023

PAGLIARA NEO

Sommaire

1. Présentation de l'Entreprise
2. Environnement de Travail et Outils Utilisés
3. Rôle en tant que stagiaire
4. Les Différents Projets
5. Conclusion
6. Remerciement

Présentation de l'Entreprise

CTRL SHIFT est une entreprise créée en 2022 par 2 amis (Abdelaziz El Mansari et Elie Laloum).

Leur travail principal est de réaliser des solutions web pour différentes entreprises dans différents domaines.

Les tâches de travail sont réparties en 2, d'un côté Abdelaziz s'occupe de faire la partie frontend, et de l'autre côté, Elie la partie backend. Le peu d'effectif contraint donc à une entraide, et une communication primordiale.

Leur site était prévu pour être réalisé dans les environs de septembre/octobre 2022, mais malheureusement à cause de leur grosse charge de travail et les différents projets en cours le site n'est pas encore abouti, il devrait être cependant disponible pour l'année 2023.

Le site sera public et accessible à tous, il proposera les services de CTRL SHIFT et les différentes réalisations de cette même société pour d'autres clients.

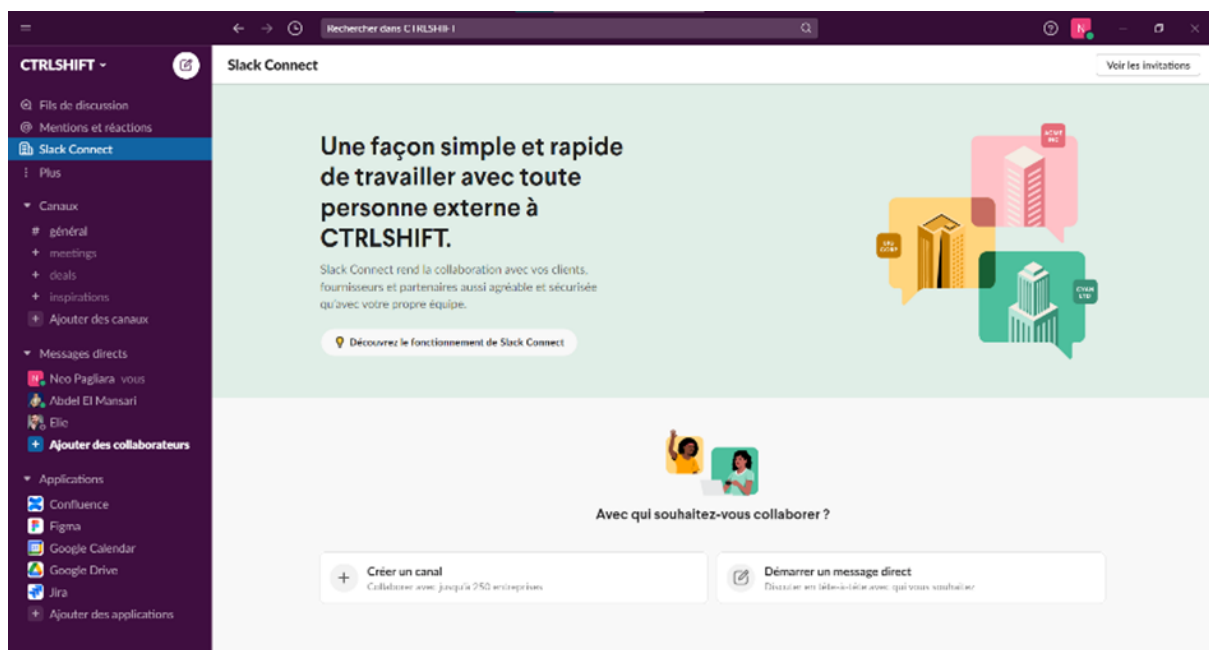
Environnement de Travail et Outils Utilisés

Durant ce stage, l'environnement de travail était le même qu'au précédent stage effectué avec eux, nous travaillions dans des espaces de coworking sur Paris et utilisions le bureau situé dans le 18^{ème} arrondissement.

Cependant, les outils utilisés par l'entreprise et leur mode de travail ont partiellement changés, des outils ont été gardés et d'autres remplacés.

Outils Utilisés :

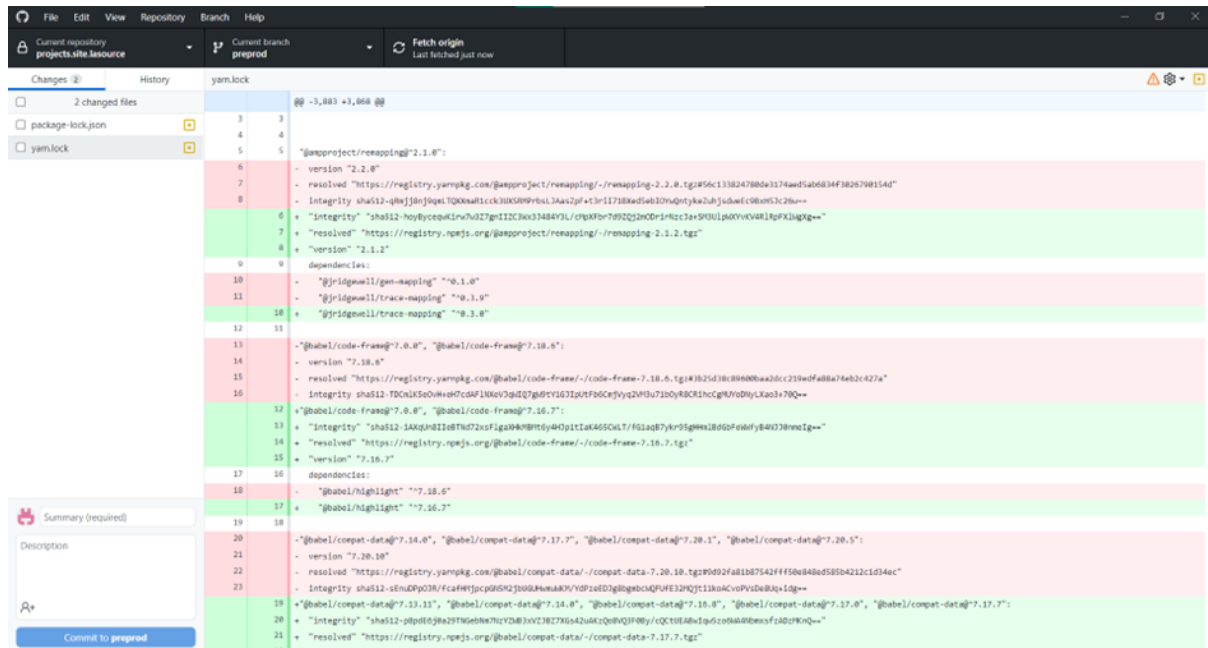
Slack (logiciel de communication) a été gardé et est primordial à l'entreprise pour la communication et le partage de fichiers



Figma (logiciel de maquettes) a été gardé, le logiciel adhère aux différents clients et donne lieu à un partage en temps réel sur le même fichier.

(+) : Ajout/Nouveauté

La grosse nouveauté ajoutée est (+) GitHub Desktop :



GitHub Desktop est une application qui permet d'interagir avec GitHub à l'aide d'une interface graphique au lieu des lignes de commandes ou d'un navigateur Web. L'application favorise donc un gain de temps et une compréhension plus simple de Git.

(+) Prismic (CMS dédié à la gestion des contenus)



Prismic est un CMS qui fournit un back-office isolé du front-office et propose des appels API intégrés permettant de récupérer des champs sur la partie front afin de les dynamiser.

(+) **Airtable** (logiciel de gestion de projets) :



Airtable vous permet de créer des bases de données collaboratives et complètes avec la simplicité d'un tableur pour les présentations. De plus, Airtable est très flexible, permettant à tous les types d'entreprises de le personnaliser en fonction de leurs besoins.

(+) **Google Cloud** (hébergeur, cloud) :



Google Cloud est une plateforme de cloud computing fournie par Google, proposant un hébergement sur la même infrastructure que celle que Google utilise en interne pour des produits tels que son moteur de recherche.

(+) **Flutter** (framework cart) :



Flutter est un kit de développement logiciel d'interface utilisateur open-source créé par Google. Il est utilisé pour développer des applications pour Android, iOS, Linux, Mac, Windows, Google Fuchsia et le web à partir d'une seule base de code.

(+) NuxtJs (framework Vuejs, NodeJS) :



NuxtJS permet de créer des applications web rendues côté serveur de manière simple et rapide.

Vuejs permet de réduire le temps de chargement et des compiles et facilite la lecture dans le code

Environnement de Travail :

Gestion du projet : La gestion du projet à toujours été la même, elle s'organise par étapes :

- *Etude du besoin, propositions fonctionnelles et technique, livrables*
 - *1er call pour comprendre le projet du client d'une manière générale*
 - *Rendez-vous avec le client pour établir ensemble le besoin de son projet*
 - *Établissement d'un cahier des charges et également une estimation de prix*
 - *Validation avec le client et démarrage du projet*
 - *Création et présentation du rétroplanning sur projet*
- *Wireframe et propositions graphiques, UI & UX*
 - *Création et présentation de l'arborescence du projet*
 - *Création structurelles du projet*
 - *Création de la charte graphique et des wireframes*
 - *Validation des maquettes*
- *Développement spécifique/générique*
 - *Découpage des tâches, estimation et attribution*
 - *Mise en place des outils propre au processus de développement*
 - *Développement du projet*
 - *Revue du code*
 - *Revue du client*
 - *Itération*
- *Infrastructure et mise en production*

- *Selon le projet mettre en place des pipelines Continuous Deployment*
- *Mise en place d'une pré production*
- *Mise en place d'une production*
- *Documentation et formation*
 - *Création et transmission au client d'une documentation d'utilisation de son produit*
 - *Formation aux outils et au produit*
- *Maintenance*
 - *Correction des bugs spécifiques à notre développement*

Mise en production : La mise en production est renouvelée :

CTRL SHIFT utilise maintenant des pipelines d'intégration continue et des déploiements continus.

Cela consiste à push son code sur GitHub, ce qui va activer les pipelines, faire que le code se build automatiquement, et générer un test du code (affichage d'erreurs), elles offrent donc une meilleure qualité de code.

Cela sert à optimiser les ressources et privilégie un gain de temps.

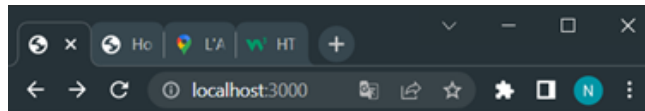
Rôle en tant que stagiaire

Mon rôle en tant que stagiaire était d'apporter mon aide à l'entreprise sur les différents travaux et missions effectuées auprès des clients, ce qui m'a permis d'approfondir mes connaissances dans différents langages mais également sur l'utilisation de CMS.

J'ai pu les aider en donnant mon avis personnel sur des modifications de projets et également pu assister à des réunions avec des clients ce qui m'a permis d'apprendre leur façon de s'organiser pour un projet.

Les Différents Projets

Tout d'abord, mon tout premier projet a été de gérer le responsive d'un site d'un client.



N'ayant jamais été confronté au codage de responsive, j'ai été directement plongé dedans, ce qui permettra par la suite la réalisation de différents projets.

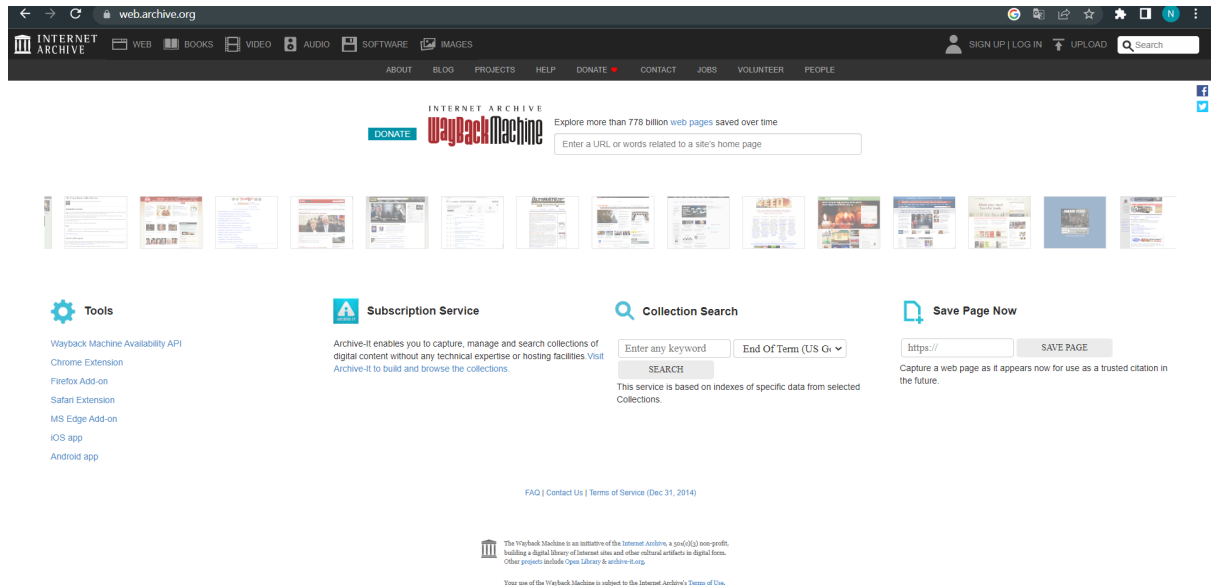
Par la suite, j'eus dû créer les différentes pages de ce même site, codage en html, css et js.



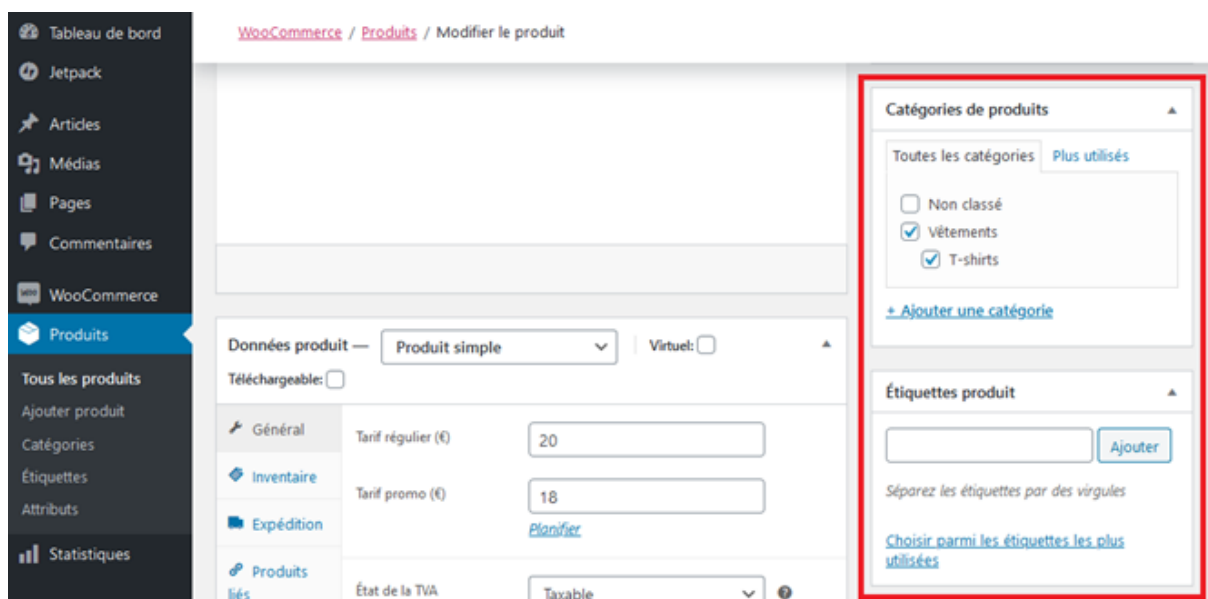
Ce projet ne m'aura pas pris beaucoup de temps, j'ai continué à faire des fixations dessus en fin de stage.

Ensuite j'ai travaillé sur un commerce en ligne avec Wordpress.

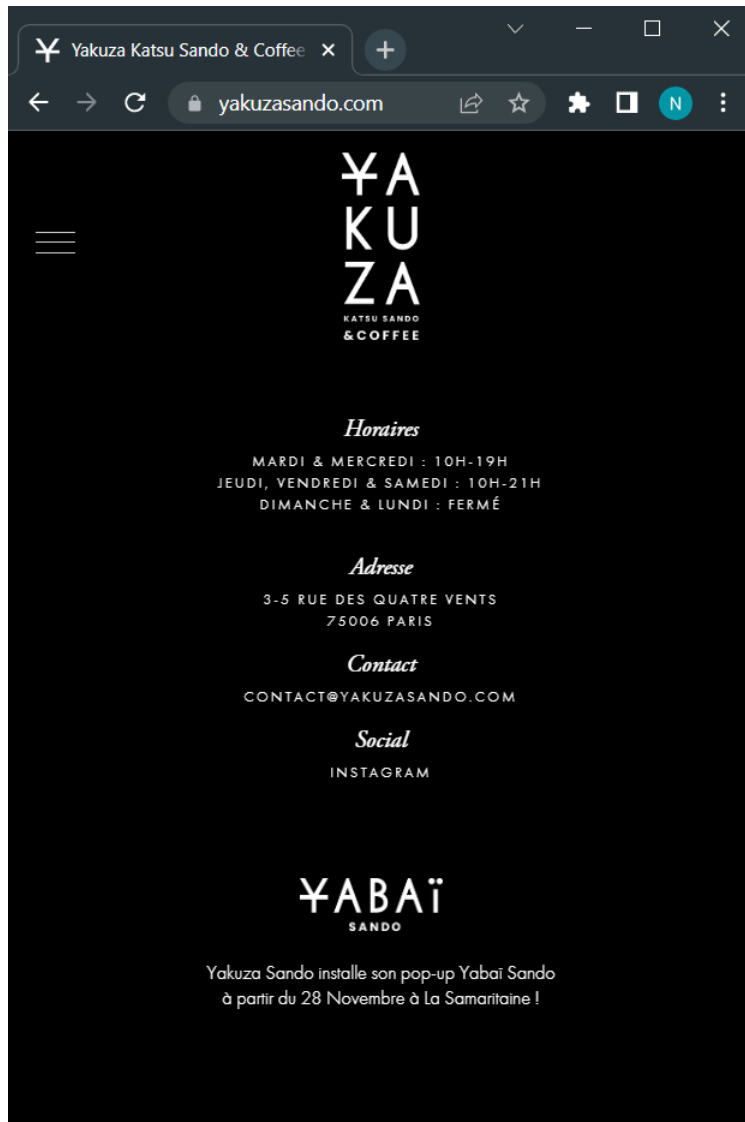
Les produits du commerce avaient été supprimés et j'ai eu accès à un outil retraçant tous les sites mis en ligne sur internet mêmes ceux supprimés : Web Archive



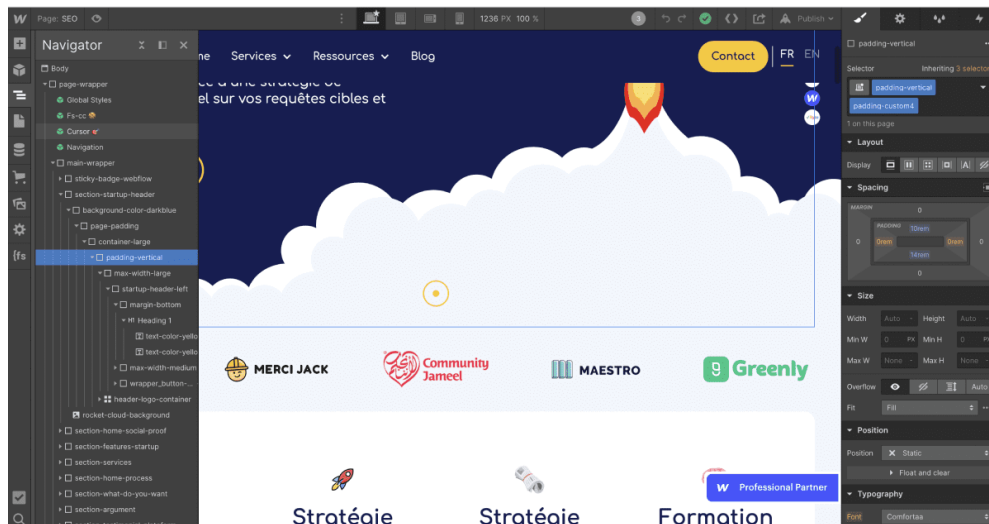
Cela m'a permis de faire la comparaison des produits mis en ligne et de pouvoir récupérer leur description, prix ... afin de les mettre à jour sur Wordpress



J'ai ensuite, comme pour le premier projet, dû mettre à jour des infos et gérer le responsive d'un autre site de client.

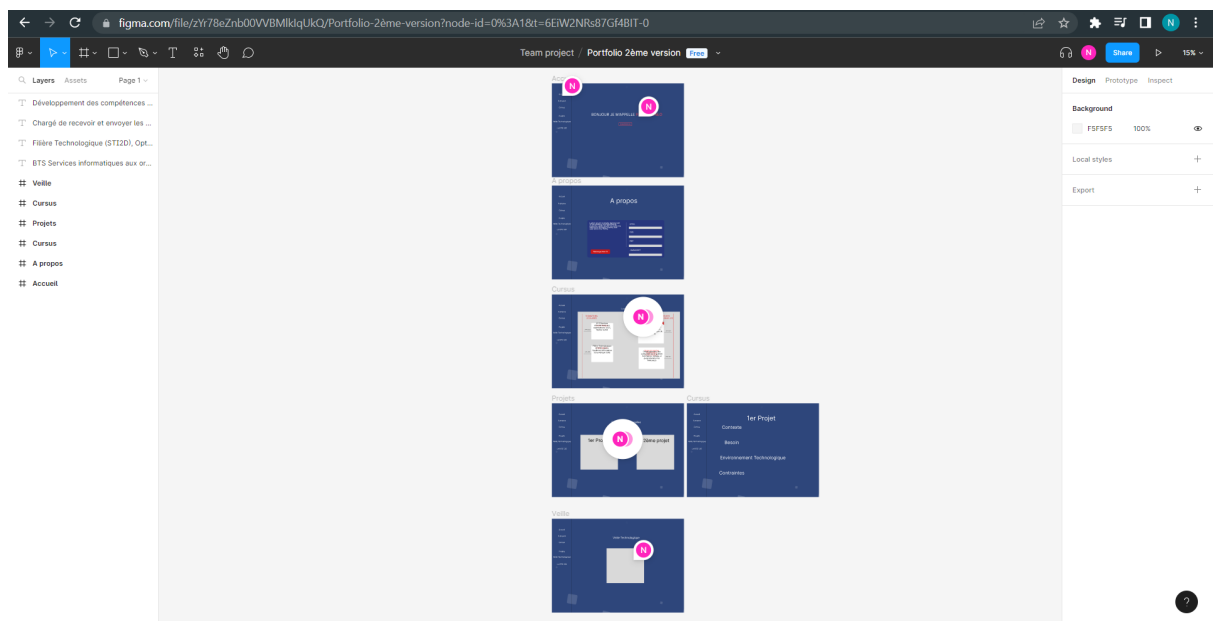


J'ai par la suite découvert WebFlow (CMS), j'y ai passé un peu plus de temps que les précédents projets, et ai dû créer et dynamiser des champs et des conditions pour des pages d'un site client. Cela m'a permis d'apprendre l'utilisation de CMS et l'apprentissage de dynamisation des champs, j'ai pu par la suite gérer également le responsive des pages.



Pour terminer, j'ai réalisé le projet qui m'aura pris le plus de temps, la réalisation de mon portfolio.

Tout d'abord, j'ai réalisé les maquettes du portfolio avec Figma, pour avoir un aperçu au brouillon.



Pour mon portfolio, j'ai utilisé le framework symfony (framework php), et Gitpod (environnement de développement). J'ai donc d'abord mis en place mon environnement de travail et ai appris à utiliser symfony.

J'ai eu beaucoup de mal à comprendre l'utilisation dans mes débuts, mais au fur et à mesure tout était clair.

Des problèmes ont été rencontrés à l'installation et au démarrage du framework, mauvaise version de node, variables d'environnement.

Cela m'a quand même appris à me documenter et rechercher les solutions sur internet et à mettre en place un environnement de travail.

J'ai donc utilisé npm, yarn, composer pour le lancement de mon projet et l'ai codé en html, css/scss, js et php pour la liaison des pages et des routes.

Une fois le projet mis en place sur VsCode, on le met en place et le configure sur Gitpod.

L'utilisation de Gitpod peut se faire de 2 façons différentes.

Dans un premier temps, les développeurs peuvent reprendre l'URL de leur projet GitHub par exemple, puis ajouter devant celle-ci :

<https://gitpod.io#>.

L'URL finale devrait donc être semblable à ceci :


<https://gitpod.io#github.monexemple.com>.

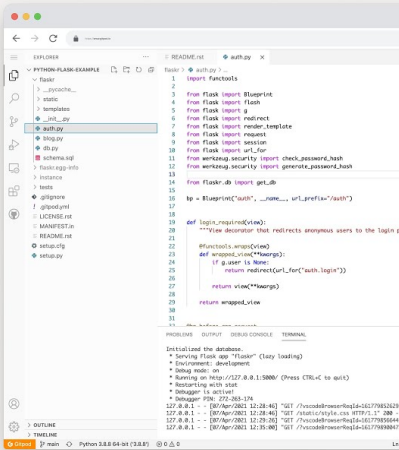
Dans un second temps, les développeurs peuvent installer sur leur navigateur une extension qui leur permettra d'ouvrir leur projet grâce à celle-ci directement dans Gitpod.

Qu'est ce que Gitpod ?

Screencast 001

Getting Started With Gitpod





```
1 from flask import Flask
2
3 from flask import Blueprint
4 from flask import flash
5 from flask import g
6 from flask import redirect
7 from flask import render_template
8 from flask import request
9 from flask import session
10 from flask import url_for
11 from werkzeug.security import check_password_hash
12 from werkzeug.security import generate_password_hash
13
14 from flask_db import get_db
15
16 bp = Blueprint("auth", __name__, url_prefix="/auth")
17
18
19 def login_required(f):
20     """View decorator that redirects anonymous users to the login page"""
21
22     @functools.wraps(f)
23     def wrapped(*args, **kwargs):
24         if 'user' in session:
25             return redirect(url_for("auth.login"))
26         return f(*args, **kwargs)
27     return wrapped
28
29
30 @bp.route("/login", methods=["GET", "POST"])
31 def login():
32     if request.method == "POST":
33         username = request.form.get("username")
34         password = request.form.get("password")
35         db = get_db()
36         user = db.execute('SELECT * FROM users WHERE username = :username').fetchone()
37         if user and password == user['password']:
38             session['user'] = username
39             return redirect(url_for("index"))
40         flash("Invalid username or password")
41         return render_template("auth/login.html"), 401
42     return render_template("auth/login.html")
```

Python 3.8.5 (tags/v3.8.5:4282b16, Jun 7 2020) [AMD64]

Detected the database.

- Serving Flask app "flask" (lazy loading)
- Environment: development
- Debug mode: on
- Running on http://127.0.0.1:3000 (Press CTRL-C to quit)
- Restarting with stat
- Debugger is active

127.0.0.1 - [07/Apr/2021 12:28:46] "GET /auth/login HTTP/1.1" 200

127.0.0.1 - [07/Apr/2021 12:28:46] "GET /auth/login HTTP/1.1" 200

127.0.0.1 - [07/Apr/2021 12:28:46] "GET /auth/login HTTP/1.1" 200

Gitpod est un Environnement de Développement Intégrés (EDI) qui se tourne vers le cloud.

Il donne lieu à de la simplicité, de la rapidité, mais surtout de l'efficacité !

Gitpod peut être utilisé avec tous les logiciels de gestion de projet GIT et son EDI est compatible avec des outils open source très utilisés comme VsCode. Il fournit également un accès avec une clé ssh au code.

Il offre 50 heures d'utilisations gratuites, et c'est donc pour cela que j'ai utilisé Github pour mon projet.

Une fois que tout était configuré, j'ai pu commencé à coder et à faire mes premiers envois de mon code sur GitHub.

Tant que les crédits de Gitpod n'étaient pas terminés, le projet pouvait être récupéré facilement et rapidement dessus, mais une fois les 50 heures atteintes le seul moyen de récupérer le projet était via GitHub. Le projet se lance en local via les commandes :

```
yarn run watch et php -S 0.0.0.0:8000 -t public/
```

Une fois que le projet était relativement bien avancé, je l'ai hébergé avec AWS.

J'ai donc d'abord créer une instance EC2 à laquelle j'ai attribué une IP statique

The screenshot shows the AWS Management Console interface. On the left, there's a navigation menu with options like 'New EC2 Experience', 'Tableau de bord EC2', 'Vue globale EC2', 'Événements', 'Balises', 'Limites', 'Instances', 'Types d'instances', 'Modèles de lancement', 'Demandes Spot', 'Savings Plans', 'Instances réservées', 'Hôtes dédiés', 'Instances planifiées', 'Réservations de capacité', 'Images', 'AMI', 'Catalogue des AMI', 'Elastic Block Store', 'Volumes', 'Instantanés', 'Gestionnaire de cycle de vie', and 'Réseau et sécurité'. The main area displays a table of instances. The instance 'pagliara-neo-portfolio' (ID: i-04d1c7b109754a806) is selected. Below the table, the 'Details' tab is active, showing a summary of the instance's information, including its ID, name, state, type, and various network and security settings.

Name	ID d'instance	État de l'instance	Type d'instance	Contrôle des st...	Statut d'alar...	Zone de dispon...	DNS IPv4 public	Adresse IPv4...	IP élastique
LAMP	i-05a8f4d8a64290fa3	En cours d'exécution	t2.small	Initialisation en co	Aucune al...	us-east-1b	ec2-44-212-68-227.co...	44.212.68.227	-
tpDocker	i-0272e6b2859f15d4b	En cours d'exécution	t2.micro	Initialisation en co	Aucune al...	us-east-1b	ec2-44-204-85-100.co...	44.204.85.100	-
pagliara-neo-portfolio	i-04d1c7b109754a806	En cours d'exécution	t2.micro	Initialisation en co	Aucune al...	us-east-1b	ec2-23-23-131-32.com...	23.23.131.32	23.23.131.32

Instance : i-04d1c7b109754a806 (pagliara-neo-portfolio)

Détails | Sécurité | Mise en réseau | Stockage | Vérifications de statut | Surveillance | Balises

Résumé de l'instance Informations

ID d'instance: i-04d1c7b109754a806 (pagliara-neo-portfolio)

Adresse IPv6: -

Type de nom d'hôte: -

Nom de l'adresse IP: ip-172-31-94-198.ec2.internal

Réponse à un nom DNS de ressource privée IPv4 (A): -

Adresse IP attribuée automatiquement: -

Rôle IAM: -

Plateforme: Ubuntu (défaut)

Adresse IP publique: 23.23.131.32 | [adresse ouverte](#)

État de l'instance: En cours d'exécution

Nom DNS de l'IP privé (IPv4 uniquement): ip-172-31-94-198.ec2.internal

Type d'instance: t2.micro

ID de VPC: vpc-0f94529fb774d3a01 |

ID de sous-réseau: subnet-09047870fbbbe481 |

ID AMI: ami-0574da719dca65348

Adresses IPv4 privées: 172.31.94.198

DNS IPv4 public: ec2-23-23-131-32.compute-1.amazonaws.com | [adresse ouverte](#)

Adresses IP élastiques: 23.23.131.32 [IP publique]

Recherche d'AWS Compute Optimizer: Inscrivez-vous à AWS Compute Optimizer pour obtenir des recommandations. | [En savoir plus](#)

Nom du groupe Auto Scaling: -

Surveillance: désactivé

Ensuite on se connecte à l'instance : Actions → Se connecter

The screenshot shows the 'Connect to instance' dialog box in the AWS Management Console. The dialog is titled 'Connectez-vous à l'instance Informations' and provides options to connect to the instance 'i-04d1c7b109754a806 (pagliara-neo-portfolio)'. The 'EC2 Instance Connect' tab is selected, showing the instance ID, public IP address (23.23.131.32), and the default user 'ubuntu'. A note indicates that the default user name is 'ubuntu' and that users should check the AMI owner's instructions if they have changed the default user name. The dialog includes buttons for 'Annuler' (Cancel) and 'Se connecter' (Connect).

Connectez-vous à l'instance Informations

Connectez-vous à votre instance i-04d1c7b109754a806 (pagliara-neo-portfolio) à l'aide de l'une de ces options

EC2 Instance Connect | Session Manager | Client SSH | EC2 Serial Console

ID d'instance: i-04d1c7b109754a806 (pagliara-neo-portfolio)

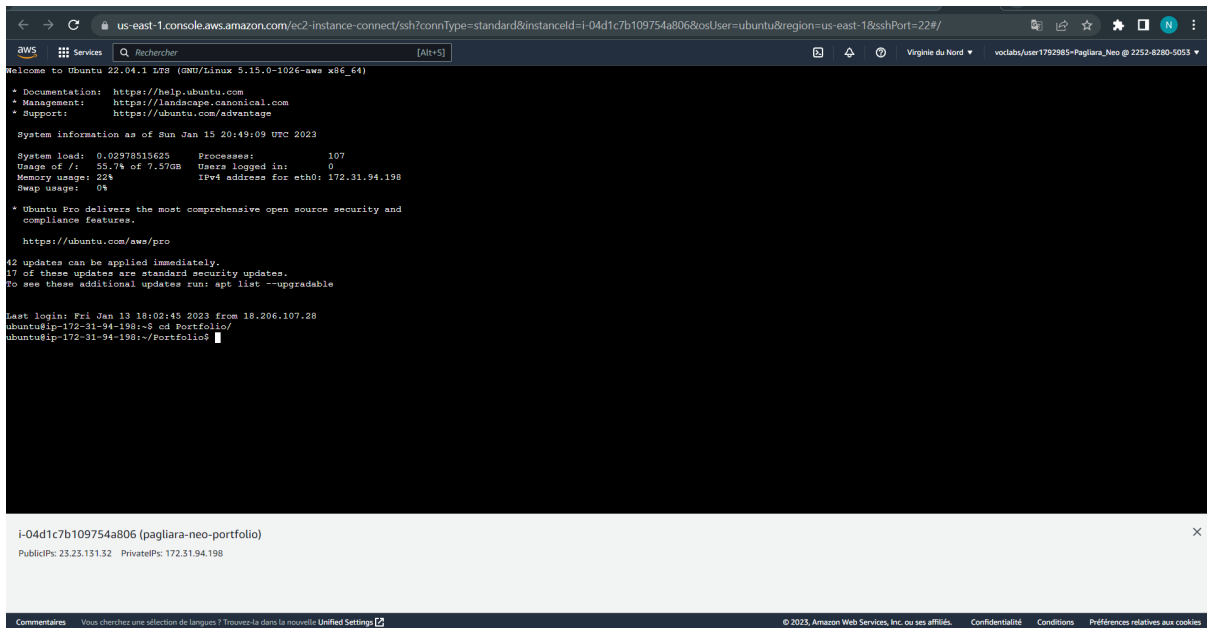
Adresse IP publique: 23.23.131.32

Nom utilisateur: ubuntu

Note: In most cases, the default user name, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

[Annuler](#) [Se connecter](#)

On a ensuite donc accès à un terminal ssh



```
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-04d1c7b109754a806&osUser=ubuntu&region=us-east-1&sshPort=22#/  
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1026-aws x86_64)  
+ Documentation: https://help.ubuntu.com  
+ Management: https://landscape.canonical.com  
+ Support: https://ubuntu.com/advantage  
System information as of Sun Jan 15 20:49:09 UTC 2023  
System load: 0.02978515625 Processes: 107  
Usage of /: 55.7% of 7.5GB Users logged in: 0  
Memory usage: 22% IPv4 address for eth0: 172.31.94.198  
Swap usage: 0%  
+ Ubuntu Pro delivers the most comprehensive open source security and compliance features.  
https://ubuntu.com/awx/pro  
42 updates can be applied immediately.  
17 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
Last login: Fri Jan 13 18:02:45 2023 from 18.206.107.28  
ubuntu@ip-172-31-94-198:~$ cd portfolio/  
ubuntu@ip-172-31-94-198:~/portfolio$
```

i-04d1c7b109754a806 (pagliara-neo-portfolio)
PublicIP: 23.23.131.32 PrivateIP: 172.31.94.198

Commentaires Vous cherchez une sélection de langues ? Trouvez-la dans la nouvelle Unified Settings
© 2023, Amazon Web Services, Inc. ou ses affiliés. Confidentialité Conditions Préférences relatives aux cookies

Pour configurer le projet:

- On clone le projet dans le serveur :

```
git clone https://github.com/name/repository
```

- On modifie le fichier .gitconfig et on active des extensions :

```
extension=sqlite3.so
```

```
extension=pdo_sqlite.so
```

- On installe composer via php :

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
php -r "if (hash_file('sha384', 'composer-setup.php') ===  
'55ce33d7678c5a611085589f1f3ddf8b3c52d662cd01d4ba75c0ee0459970c2200a51f492d5  
57530c71c15d8dba01eae') { echo 'Installer verified'; } else { echo  
'Installer corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"
php composer-setup.php
php -r "unlink('composer-setup.php');"
```

- On ajoute ensuite le webpack symfony pour que le terminal puisse assimiler le projet :

```
yarn add --dev @symfony/webpack-encore  
./composer.phar require symfony/webpack-encore-bundle
```

- On met le projet en production :

```
nano .env
```

APP_ENV=production

APP_ENV=production permet aussi de faire disparaître la toolbar de symfony.

Et voilà le projet est enfin hébergé et disponible sur internet.

Le projet se lance via les commandes :

yarn run build et php -S 0.0.0.0:8000 -t public/

Si des modifications veulent être ou sont réalisés sur le projet en local alors il y a juste à faire :

git add .

git commit -m "message"

git push

Les code est alors mis à jour sur GitHub et pour récupérer les modifications on a alors juste à faire sur le terminal ssh de l'instance ec2 :

git pull

./composer.phar update

Conclusion

Plongé dans le cœur du métier de développeur pendant 8 semaines, j'ai pu m'épanouir et acquérir de nombreuses et de nouvelles compétences. Je me suis habitué à de nouvelles méthodes de travail et j'ai su m'adapter aux différentes situations. J'ai pu comprendre des notions avancées en php grâce au framework.

J'ai appris à me renseigner, me documenter, résoudre des problèmes, trouver des solutions.

J'ai beaucoup aimé faire ce stage qui m'a énormément apporté sur différents points, que ce soit en théorie ou en pratique.

De mon côté, je pense et j'espère avoir apporté mon aide si ce n'est que le minimum sur des projets, et une proposition intéressante sur l'avenir de l'entreprise.

Je voudrais et j'espère pouvoir les retrouver par la suite que ce soit au cours d'un stage ou une alternance ou pourquoi pas être employé chez eux un jour.

Remerciement

Je tiens à remercier Abdelaziz et Elie qui m'ont porté un excellent accueil et qui m'ont beaucoup apporté durant ces deux stages.

Je les remercie en grande partie de m'avoir accepté en tant que stagiaire, c'était un choix compliqué pour eux sachant qu'ils sont en pleine création de leur entreprise et ne leur souhaite que du bien à l'avenir.