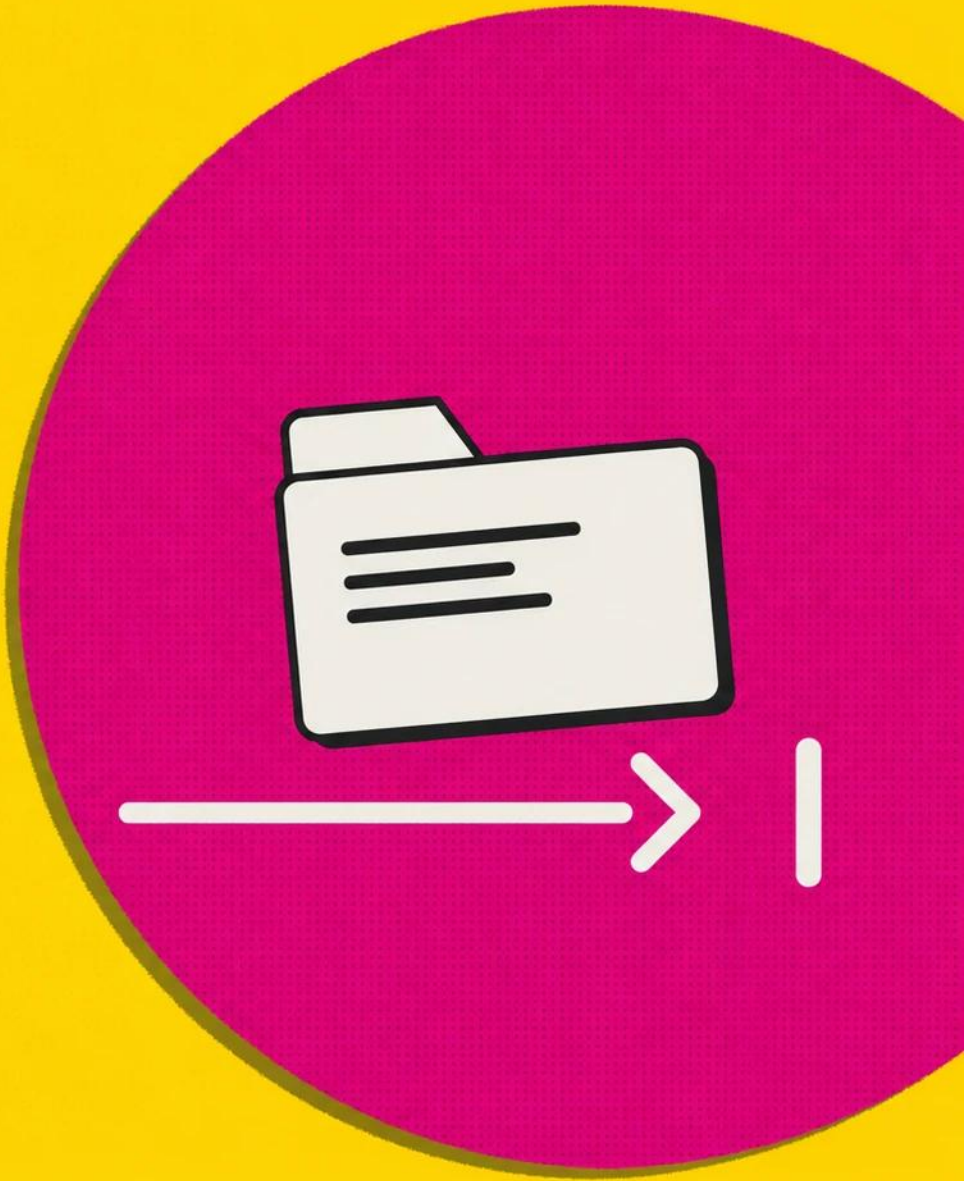
1.22

TEMPS 1 · LE PORTFOLIO · SÉANCE 5

Mettre en ligne chez OVH

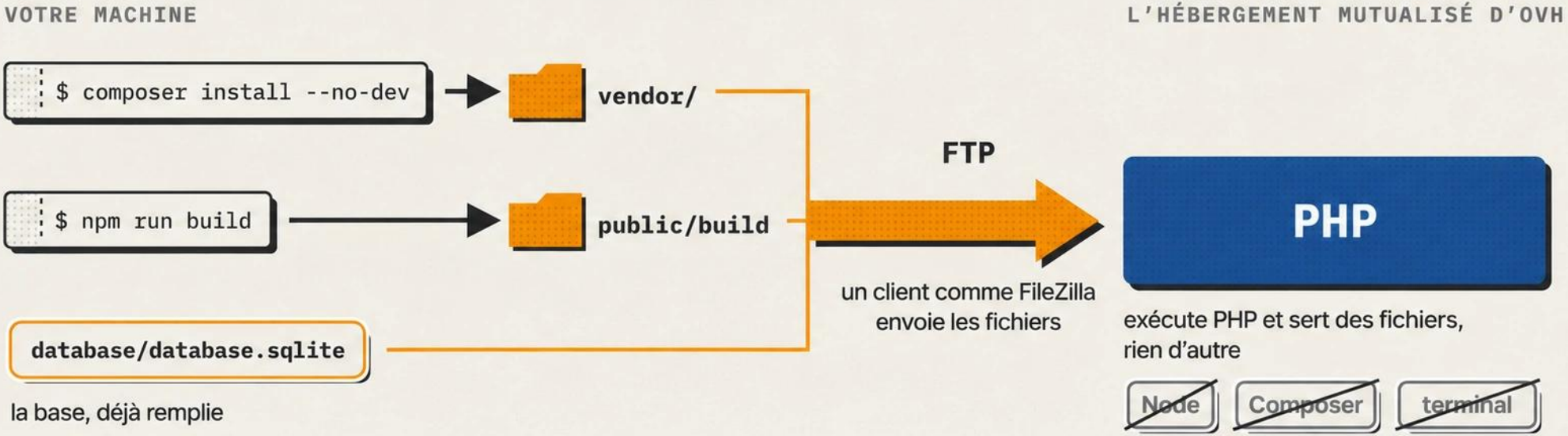


Thibault Six · IFOSUP Wavre · 2026-2027

5XDYN · Projet web dynamique · 1.22

? Chez OVH, il n'y a ni Node, ni Composer, ni terminal. Comment votre portfolio y tourne-t-il ?

OVH n'exécute que PHP : Composer et Vite tournent sur votre machine, et leurs résultats partent par FTP.



À gauche, ce que vous lancez avant l'envoi ; à droite, ce que fait le serveur. Le reste se fait chez vous.

? Vos projets encodés dans le back-office sont dans la base. Comment arrivent-ils chez OVH ?

La base SQLite est un fichier qui part rempli avec le code ; le seeder ne tourne pas en ligne.

database/database.sqlite

- vos projets
- vos étiquettes
- votre compte du back-office
- les messages de contact

depuis le chapitre 1.04

FTP

l'hébergement d'OVH

rien à exporter, rien à importer

PAS AVANT L'ENVOI

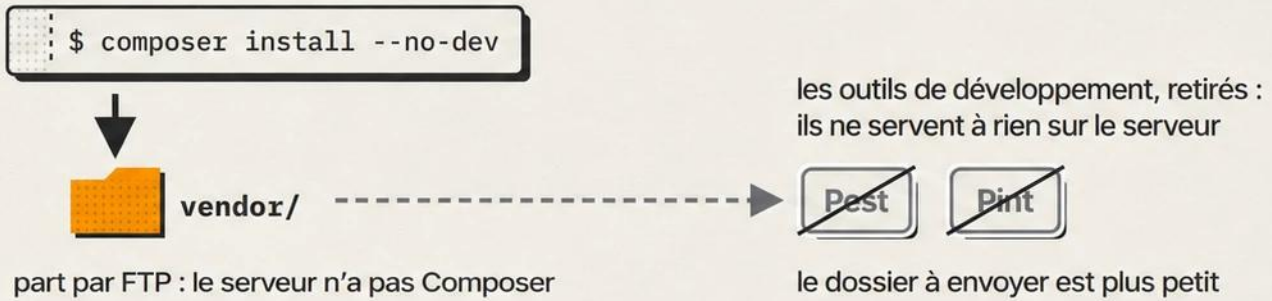
```
$ php artisan migrate:fresh --seed
```

vide toutes les tables, puis les remplit avec le seeder :
vos projets encodés disparaissent

Le fichier de gauche est toute la base ; il part tel quel. La commande du bas l'effacerait avant le départ.

? Le serveur n'a pas Composer. Comment les paquets PHP de votre portfolio y arrivent-ils ?

composer install --no-dev réinstalle vendor/ sans Pest ni Pint, et ce dossier part lui aussi par FTP.



APRÈS LA MISE EN LIGNE, SUR VOTRE MACHINE

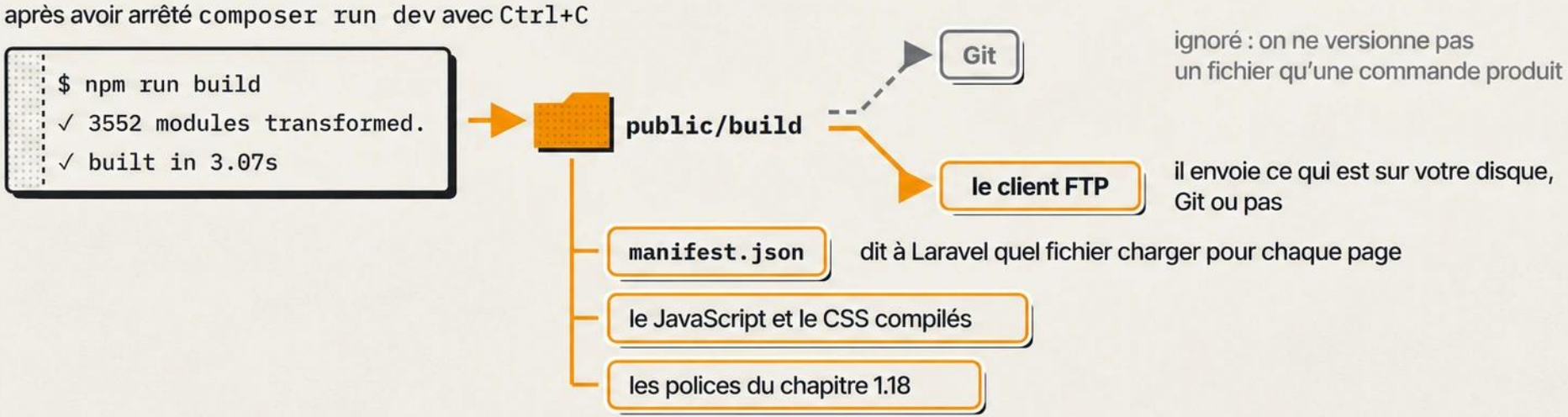
```
$ composer install
```

sans option : vos tests en ont besoin

En haut, ce que vous envoyez ; en bas, ce que vous relancez ensuite pour retrouver vos outils de test.

? Vite compile vos pages Vue sur votre machine. Où va le résultat, et comment arrive-t-il en ligne ?

npm run build compile les pages dans public/build : Git ignore ce dossier, mais le client FTP l'envoie.



une page, un composant ou `app.css` modifié : relancez `npm run build` et renvoyez `public/build`

La commande remplit public/build. Git l'ignore, le client FTP l'envoie : c'est lui qui le met en ligne.

? Que doit-il y avoir sur le serveur après l'envoi, et quel fichier peut rendre la page blanche ?

Tout le projet part par FTP sauf node_modules, et public/hot reste chez vous, sinon la page est blanche.

SUR LE SERVEUR

- ✓ **public/index.php** le point d'entrée de l'application
- ✓ **bootstrap**
- ✓ **config**
- ✓ **storage**
- ✓ **vendor**
- ✓ **.env** certains clients FTP cachent les fichiers dont le nom commence par un point
- ✓ **public/build**
  - ✓ **manifest.json**
  - ✓ **assets**
- ✓ **database/database.sqlite**

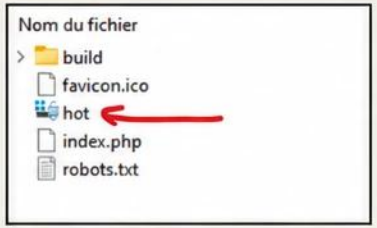
PAS SUR LE SERVEUR

~~**node_modules**~~

les paquets JavaScript qui ont servi à compiler : plusieurs centaines de mégaoctets, aucun rôle sur le serveur

~~**public/hot**~~

créé par `composer run dev` : Laravel demande les assets au serveur de Vite de votre machine, et la page reste blanche



le dossier public dans un client FTP : le fichier hot à supprimer

```
storage/logs/laravel.log
Vite manifest not found at:
```

`public/build` absent : une erreur 500

À gauche, ce que vous vérifiez sur le serveur après le transfert ; à droite, ce qui ne doit pas y être, et pourquoi.

? Tout votre projet est sur le serveur. Pourquoi un visiteur ne peut-il pas lire votre .env ?

Dans Multisite, le domaine pointe vers le dossier public : le visiteur n'atteint ni .env ni database.sqlite.



dans Multisite : le champ dossier racine du domaine, réglé sur le dossier public

.env

database/database.sqlite

vendor/

storage/

hors de portée

le navigateur du visiteur → votre-adresse.be → dossier racine → public/

le seul dossier qu'un visiteur peut atteindre

MULTISITE, DANS L'INTERFACE OVH

chaque domaine et le dossier qu'il sert

Suivez le visiteur : le domaine le mène au dossier public, et rien au-dessus de ce dossier n'est servi.

1.22

? Laravel 13 demande PHP 8.3. Pourquoi l'hébergement doit-il pourtant passer en PHP 8.4 ?

Les paquets du portfolio demandent PHP 8.4.1 ou plus : avec un PHP plus ancien, l'application refuse de démarrer.



vendor/composer/platform_check.php

```
Your Composer dependencies require a PHP version ">= 8.4.1".
```

Composer vérifie la version au démarrage

Le minimum de Laravel ne suffit pas : c'est la version la plus exigeante des paquets qui compte.

08

? Le .env de votre machine part avec le code. Quelles lignes changent sur le serveur ?

Le .env du serveur cache les erreurs, donne l'adresse publique en https et envoie l'e-mail par le serveur d'OVH.



À gauche, ce que voit le visiteur ; à droite, comment part l'e-mail de contact. Les valeurs barrées sont celles de votre machine.

1.22

? Sans certificat, que voit le visiteur, et que devient le message de son formulaire de contact ?

Let's Encrypt, le certificat gratuit d'OVH, chiffre les échanges et met le cadenas devant votre adresse.

SANS CERTIFICAT

le navigateur du visiteur — le message du formulaire de contact → l'hébergement d'OVH

envoyé en clair

Non sécurisé

AVEC LET'S ENCRYPT

le navigateur du visiteur — le message du formulaire de contact → l'hébergement d'OVH

chiffré

le cadenas

l'hébergement, puis sa gestion du SSL

activez le certificat, ou régénérez-le après un nouveau domaine

de quelques minutes à quelques heures

la gestion du SSL, dans l'interface OVH

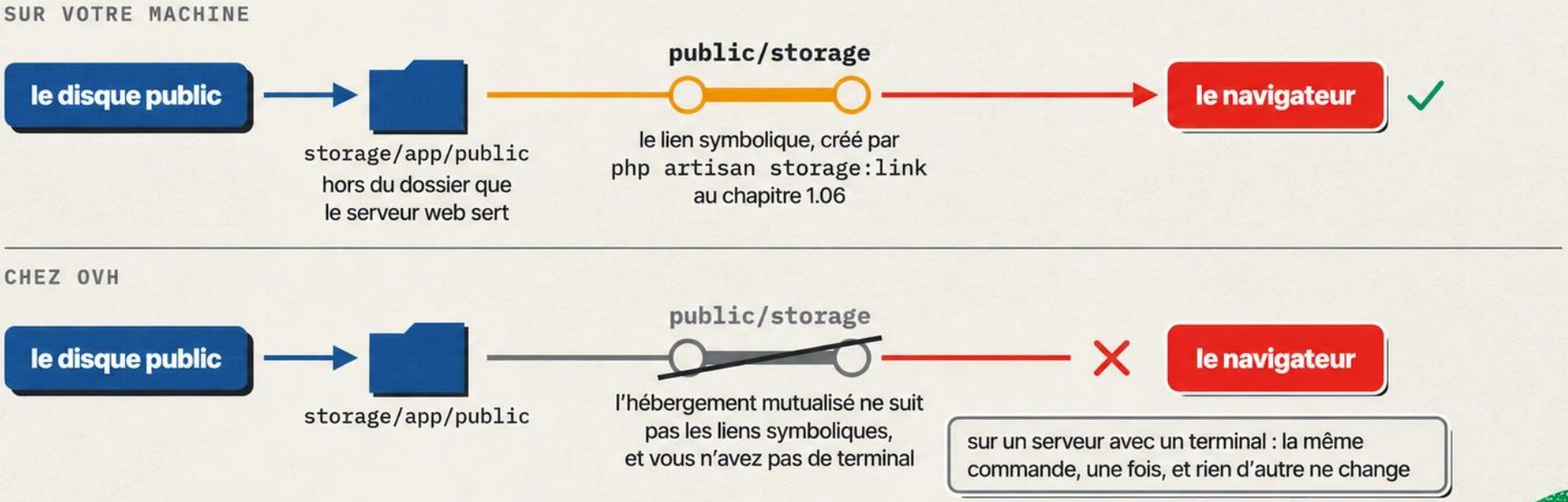


Le même message, deux trajets : en clair sans certificat, chiffré avec. La colonne de droite donne les trois étapes chez OVH.

1.22

? `storage/app/public` est hors du dossier servi. Comment le navigateur voit-il vos images ?

Le lien symbolique public/storage rend visibles les images de storage/app/public, mais OVH ne suit pas ce lien.



En haut, la bonne pratique : le lien relie les deux dossiers. En bas, le même lien chez OVH : il ne mène nulle part.

? OVH ne suit pas le lien. Où le disque doit-il écrire, et qui décide de ce changement ?

FILESYSTEM_PUBLIC_ROOT=public fait écrire le disque dans public/storage, où vous envoyez vos images.

config/filesystems.php — extrait

```
'root' => env('FILESYSTEM_PUBLIC_ROOT') === 'public'
    ? public_path('storage')
    : storage_path('app/public'),
```

seule la ligne `root` du disque public change ;
elle valait `storage_path('app/public')`

.env du serveur

```
FILESYSTEM_PUBLIC_ROOT=public
```

sans la variable, sur votre machine : `storage/app/public`, rien ne change

VOTRE MACHINE

storage/app/public

- **images/** les captures du seeder
- **projects/** les images téléversées depuis le back-office

FTP

LE SERVEUR

public/storage
un vrai dossier, pas un lien

`public/storage/images/meteo-wavre.jpg` → `/storage/images/meteo-wavre.jpg`

l'adresse de l'image ne change pas

En haut, la variable du .env choisit la racine du disque ; en bas, ce que vous envoyez dans cette racine.

1.22

? Qu'est-ce qui doit être vrai à la fin ?

À vous, vingt-cinq minutes : votre portfolio en ligne en https, avec ses images et un vrai e-mail de contact.

À FAIRE

1. En local : `composer install --no-dev`, puis `npm run build`.
2. Le projet par FTP, sans `node_modules` ni `public/hot`.
3. Dans l'interface OVH : Multisite vers `public`, PHP 8.4, certificat SSL.
4. Le `.env` du serveur, avec `FILESYSTEM_PUBLIC_ROOT=public`.
5. Tout `storage/app/public` dans `public/storage`.

une erreur 500 : lisez la fin de `storage/logs/laravel.log` avec votre client FTP

À VÉRIFIER, EN NAVIGATION PRIVÉE

25 min

- [ ] Le dossier `public/build` est sur le serveur, et le fichier `public/hot` n'y est pas.
- [ ] La version de PHP de l'hébergement est 8.4 ou plus.
- [ ] Le `.env` du serveur a `APP_ENV=production`, `APP_DEBUG=false` et l'adresse publique du site dans `APP_URL`.
- [ ] L'accueil s'affiche en https, avec vos couleurs et vos polices.
- [ ] Une fiche de projet montre son image de couverture, servie depuis `/storage/images/` ou `/storage/projects/`.
- [ ] Un message envoyé par le formulaire de contact arrive dans votre boîte e-mail.
- [ ] Le mot de passe du seeder est remplacé.
- [ ] Un outil d'aperçu montre l'image et le résumé d'une fiche.

Cinq étapes à gauche, huit vérifications à droite, dans une fenêtre de navigation privée, à votre adresse publique.