

Nicolas Vero

But Métiers du multimédia et de l'Internet

Aniwaa

Alban Sagot

Rapport de stage

Année 2023 – 2024

Développement web

Développement sous WordPress



Evaluation du stagiaire



INSTITUT UNIVERSITAIRE DE TECHNOLOGIE DE ROUEN
BUT3 MMI

Année 2023-2024

Les enseignants doivent pouvoir disposer de l'évaluation du stagiaire et du rapport de stage pour le au plus tard le 12 JUIN 2024

EVALUATION DU STAGE EN ENTREPRISE

<i>Nom du stagiaire :</i> Nicolas Vero <i>Sujet de stage :</i> Développement WordPress d'une marketplace	très bon	bon	moyen	faible	insuffisant
	A	B	C	D	E
<u>Utilisation des connaissances (théoriques et pratiques)</u>					
A. A très bien su adapter ses connaissances au sujet. E. N'a pas su mettre en oeuvre ses connaissances.	X				
<u>Ponctualité - Assiduité</u>					
A. Stagiaire très ponctuel et assidu. E. Retards/absences fréquents et/ou non motivés.	X				
<u>Organisation du travail - ordre et méthode</u>					
A. A su très bien organiser son travail pour être efficace. E. Absence de méthode et de travail cohérent.	X				
<u>Faculté d'adaptation</u>					
A. Adaptation facile et rapide au travail demandé. E. N'a pas su ou pu s'adapter.	X				
<u>Dynamisme - Esprit d'initiative</u>					
A. Très dynamique. A su prendre de très bonnes initiatives dans les limites de ses responsabilités. E. Très passif. Manque total d'initiative		X			
<u>Ouverture d'esprit</u>					
A. A cherché à connaître et comprendre l'environnement (scientifique, industriel, économique, social et humain) E. Non concerné par cet environnement.	X				
<u>Motivation</u>					
A. A manifesté un intérêt vif et permanent pour son travail. E. n'a manifesté aucun intérêt pour son travail.	X				
<u>Comportement social - Relations humaines</u>					
A. Intégration très facile et rapide. Esprit d'équipe remarquable. E. A su se faire apprécier par son entourage. Communications défectueuses (en tant qu'émetteur et récepteur de messages) Contacts humains difficiles. Trop renfermé sur lui-même. Etc...		X			
REMARQUES COMPLEMENTAIRES (FORME ET FOND) - SIGNATURE					
Nicolas est un très bon élément. Il s'est bien intégré à l'équipe et mène ses projets à terme. Il sait faire preuve d'autonomie et apprend tous les jours à se perfectionner dans sa méthodologie.					
					

Institut Universitaire de Technologie de Rouen, Département MMI,
24 Cours Gambetta -76500 Elbeuf-sur-Seine
Tel : (33) 02 32 96 10 38

Remerciements

Je tiens à remercier les personnes ayant contribué au succès de ce stage et qui m'ont aidé lors de la rédaction de ce rapport.

Je tiens tout d'abord à exprimer ma profonde gratitude envers l'entreprise Aniwaa, pour m'avoir offert l'opportunité d'effectuer mon stage à leur côté.

Je tiens également à exprimer ma reconnaissance envers mon maître de stage, M. Alban Sagot, pour son soutien constant et ses conseils avisés tout au long de cette expérience enrichissante. Il m'a permis de développer mes compétences et de relever avec succès les défis rencontrés.

Je le remercie également pour le temps qu'il a pris à lire ce rapport afin de m'aider à l'améliorer.

Je souhaite adresser mes remerciements les plus sincères à tous mes collègues chez Aniwaa, dont la collaboration, le professionnalisme et la convivialité ont grandement contribué à rendre mon stage aussi agréable qu'instructif.

Je remercie aussi mon tuteur de stage, Monsieur Kerjean, pour m'avoir aidé à la rédaction de ce rapport de stage.

Sommaire

L'entreprise	5
Présentation de l'entreprise	5
Présentation du site	5
Le stage	7
Mes outils de travail	7
Présentation des missions effectuées	7
PROJET ANALYTICS	9
Lancement du projet	9
Maquettage de la page	10
Utilisation de l'API	11
Données envoyées	12
Données reçues	13
Problème de taille d'URL	14
Création progress-bar et chart	15
Utilisation d'ApexChart.js	15
Finalisation du Front	16
Problème de crédit	17
Mise en base de données	18
Création de la base de données	18
Stockage des données	19
Tâches de Cron	20
Création de variables pour les mails	21
Création des vues catalogues	22
AJAX	22
Utilisation des postmeta	23
Déploiement du projet	25
Apport du stage	27
Connaissances acquises	27
Bilan humain	27
Poursuite d'études	28
Glossaire	30
Illustrations	31

L'entreprise

Présentation de l'entreprise

Aniwaa est une entreprise B2B (Business to Business) spécialisée dans la mise en relation entre entreprises dans le secteur de la fabrication additive.

Ses différents pôles d'activités sont situés à plusieurs endroits en France, mais son "équipe produit" (équipe chargée du développement de la marketplace) est située à Rouen.

Les bureaux sont situés rive droite dans le coworking l'Opensèn, dans le quartier Luciline, à côté des Docks 76, au bord de la Seine. Étant un espace de travail ouvert, les locaux sont partagés avec plusieurs autres entreprises.

Dans le bureau dans lequel j'étais, nous étions trois :

Alban, mon maître de stage, qui s'occupait notamment du back du site et de la gestion de WordPress et Clément, qui s'occupait de la partie front-end et de l'UX / UI.

En termes de matériel, l'entreprise m'a fourni un ordinateur, deux écrans, ainsi que tous les abonnements nécessaires aux différents services dont j'ai eu besoin.

Présentation du site

Le site d'Aniwaa s'adresse à deux publics distincts mais complémentaires. D'une part, les marques et de l'autre, les acheteurs.

Les marques sont sur la plateforme pour se faire connaître. Elles peuvent payer des abonnements leur permettant de gagner en visibilité, et de débloquer des fonctionnalités pouvant les aider à se démarquer des concurrents.

De l'autre côté, les acheteurs. Sur Aniwaa, les acheteurs ne paient rien. Ils ne sont pas monétisés. Ces clients ont pour but d'attirer des marques sur la plateforme.

Le site d'Aniwaa permet à chaque type d'audience d'avoir un espace utilisateur dédié.

Les comptes de marque (MA - Manufacturers Accounts), donnent la possibilité de soumettre des produits sur la plateforme, de gérer sa page de marque, de voir ses statistiques, ses leads (acheteurs potentiels), etc.

Les comptes acheteurs (BA - Buyers Accounts), donnent la possibilité aux clients de mettre en favoris des produits, de contacter les marques, de télécharger leurs brochures, etc.

Chaque action réalisée par l'utilisateur montrant un intérêt pour une machine (demande de devis, demande de contact, ...) générera un lead. Comme expliqué brièvement plus haut, un lead est un acheteur potentiel.

C'est le principal apport d'Aniwaa aux marques : leur apporter passivement des clients.

Dans le forfait gratuit, Aniwaa ne fournit pas les informations des leads, seulement le nombre de personnes intéressées.

Le stage

Outils de travail

Durant mon stage, j'ai beaucoup utilisé la suite Google. Elle nous permettait de communiquer via Gmail, mais surtout de planifier nos emplois du temps grâce à Calendar.

Cet outil me permettait de savoir quand une personne était disponible, et ainsi éviter de lui proposer des appels quand elle ne l'était pas.

[Voir Google calendar](#)

Pour la communication plus informelle, nous utilisions Slack. Ce logiciel équivalent à un Discord professionnel nous permettait d'échanger et de faire des appels courts.

En ce qui concerne la partie technique, j'ai utilisé le logiciel Figma pour réaliser des maquettes, ainsi que le logiciel Filezilla pour me permettre de déposer des fichiers sur le serveur.

Présentation des missions effectuées

En ce qui concerne mes missions, j'en ai eu un total de trois.

Le premier projet est celui que je vais vous présenter dans ce rapport. Il consistait en la création d'une section dans les comptes MA permettant aux marques sur le site d'avoir une page permettant de consulter leurs statistiques et globalement leur impact au niveau du public.

Le deuxième gros projet a été de créer une section dans les comptes BA permettant de visualiser les documents ayant été sauvegardés. Ces documents peuvent ensuite être téléchargés par l'utilisateur depuis son compte.

Enfin, mon troisième et plus gros projet, a été de repenser complètement la page d'accueil et les éléments de navigation (menus, header, footer). Ce projet s'est également propagé à d'autres pages importantes du site.

A ces trois projets sont venus s'ajouter des miscs (abréviation du mot anglais miscellaneous). Il s'agit de petites tâches annexes à réaliser, ne demandant pas suffisamment de temps pour en constituer un projet. Il s'agit généralement d'un bug ou de petits changements de style.

Pour la gestion des miscs, nous utilisions l'outil en ligne Trello qui permet de créer des tâches et de les assigner à un groupe. Cela permet de voir rapidement les tâches qu'il reste à traiter, ainsi que celles en cours, ou terminées.

Analytics

Le travail que je vais vous présenter dans ce rapport concerne l'ajout d'une section permettant aux vendeurs présents sur la plateforme d'avoir accès à un certain nombre de données.

Cette section sera donc accessible pour les comptes des manufacturers - MA. Pour réaliser cette section, nous nous sommes appuyés sur les données fournies par Plausible Analytics, une alternative à Google Analytics.

Ce service permet entre autres de donner le nombre de visites qu'il y a eues sur les différentes pages d'un site, de suivre l'évolution du trafic, ou encore de fournir des données sur le taux de conversion/rebond d'un site.

Bien que ce projet n'ait pas été le plus long ou le plus complexe, il est celui qui a été le plus formateur pour moi, car il m'a permis de toucher à de nombreux domaines. Il a également été formateur du fait que certaines des options envisagées au départ se sont révélées infructueuses, et qu'il a donc fallu faire un travail de réflexion sur la façon de contourner le problème en question.

Lancement du projet

A l'origine, le projet devait être beaucoup moins ambitieux : récupérer une vue créée par Plausible et l'afficher pour l'utilisateur, et seulement intégrer certains éléments visuels déjà en grande partie existante.

Seulement, nous nous sommes rendus compte que la vue donnée par Plausible avait un problème : elle permettait d'appliquer des filtres directement sur la vue, sans que nous ayons la possibilité de les désactiver.

Pour que vous compreniez bien à quel point ceci aurait pu être problématique, je dois vous expliquer comment fonctionnent les données reçues de Plausible. Ces données sont filtrées à l'aide d'un certain nombre de paramètres que nous lui fournissons. L'un de ces paramètres concerne les pages sur lesquelles nous souhaitons récupérer des informations. Ces pages sont données à Plausible sous la forme d'une liste d'URLs.

Or, étant donné que la solution proposée par Plausible permettait de gérer les filtres, les URLs auraient été accessible et auraient pu être changées par un utilisateur. N'importe qu'elle marque sur le site aurait ainsi pu obtenir des informations sur le trafic du site en général, ou sur ses concurrents.

La solution initialement simple, qui consistait à envoyer des données à Plausible et à seulement afficher ce qu'il donnait n'était plus viable. Il en restait deux :

- Abandonner le projet
- Devoir créer nous mêmes une page créée de zéro, pour donner aux utilisateurs les données fournies par Plausible sans passer par leur solution clé en main.

Après plusieurs discussions, nous avons décidé de partir sur la deuxième option et d'utiliser l'API de Plausible pour créer à la main un outil de visualisation de données.



Maquettage

Pour commencer, j'ai dû réaliser les maquettes de la section sur Figma.

Pour le style général, j'ai repris les éléments déjà existant sur Aniwaa, comme le header, les couleurs, les ombres et les bordures des éléments.

En ce qui concerne les éléments que je n'avais pas, je me suis inspiré de la manière dont Plausible présentait ses données.

[Voir interface de Plausible](#)

[Voir maquettes Figma](#)

Utilisation de l'API

Une fois l'étape de maquettage terminée, je me suis penché sur l'utilisation de l'API de Plausible, grâce à laquelle j'allais pouvoir récupérer les données.

Une API (application programming interface) est un programme permettant à deux services différents de communiquer.

L'un des services va envoyer une requête contenant en paramètre les données qu'il souhaite récupérer, l'autre service va lui retourner l'ensemble des informations demandées.

Dans cet exemple, nous créons une URL que nous allons envoyer au service. Cette URL est composée de plusieurs parties :

- Un lien vers l'API de Plausible ;
- Le nom du service appelant (ici Aniwa) ;
- Les paramètres définissant les données que l'on souhaite récupérer.

En plus de cette URL, la requête doit fournir un "token" pour accéder au service.

Ces tokens sont comme un identifiant, qui permet au service recevant la requête d'avoir la certitude que l'utilisateur a la permission d'accéder aux données. Ce token permet également d'identifier à qui sont envoyées les données et permet donc de bloquer l'envoi de réponses au-delà d'un certain quota.

Voir envoi de données à Plausible

Avant de commencer à utiliser l'API, j'ai lu la documentation pour comprendre comment elle fonctionnait, découvrir les données que j'allais récupérer et comment les exploiter.

L'API nous permet d'agir sur quatre champs principaux :

- Une métrique, ce que l'on veut mesurer ;
- Une période ;
- Des propriétés permettant d'affiner les données ;
- Des filtres permettant de ne récupérer que certaines données.

Documentation de l'API : <https://plausible.io/docs/stats-api>

Données envoyées

Pour mon projet, j'avais besoin de plusieurs types de données :

- Le nombre de visiteurs sur une page précise ;
- Le pays des visiteurs ;
- Le nombre de visiteurs en fonction du temps.

Voir récupération des données via Plausible

Chaque donnée est récupérée de cette manière :

- On appelle la fonction `get_plausible_datas()` qui prend en paramètre :
 - Les URLs sur lesquels nous voulons nos données ;
 - La durée ;
 - Ce que l'on veut obtenir ;
 - La section de l'API dans laquelle nous allons chercher nos données.

Comme dit précédemment, nous voulons uniquement donner aux marques les informations les concernant. Pour ce faire, nous devons fournir à Plausible l'ensemble des URLs sur lesquelles nous voulions récupérer nos données. Pour une marque, il s'agira des URLs vers :

- La page de la marque ;
- L'ensemble des pages des produits de cette marque ;
- L'ensemble des articles que la marque a publiés.

Etant donné que la marketplace propose deux langues (français et anglais), nous devons récupérer chacune des pages sur les deux sites.

Je commence donc par récupérer l'ensemble des ids de ces pages à l'aide de la fonction `get_brands_elements()`.

Voir la récupération des pages dans les deux langues

Une fois l'entièreté des ids récupérés, ceux-ci sont retournés et seront ensuite convertis en URL par la fonction `get_brands_URLs()`.

Ce sont ces mêmes URLs que nous donnerons à Plausible en tant que filtres.

Voir la récupération des URLs de marques

Voir le tableau des URLs récupérées

Données reçues

Intéressons nous maintenant aux données que Plausible va nous renvoyer.

[Voir réponse de Plausible](#)

Une fois notre requête envoyée, le serveur va nous retourner un objet contenant toutes les données dont nous aurons besoin.

Par exemple, le code de retour, nous permettant de savoir comment s'est déroulé le processus (200 est par exemple le code d'une requête ayant retourné une réponse valide).

Une fois ces données reçues, il ne me reste plus qu'à les mettre en forme et à stocker uniquement les données qui m'intéressent.

[Voir données de Plausible mises en forme - 1](#)

Cette capture d'écran montre les données mises en forme (pour chaque élément du tableau, j'ai un élément page contenant l'URL de la page et le nombre de visiteurs).

Seulement, ces données peuvent encore être optimisées. En effet, chaque objet contient une URL unique et une donnée chiffrée associée. Pour simplifier l'utilisation des données et leur lecture, j'ai décidé de les réorganiser pour donner le résultat suivant :

[Voir données de Plausible mises en forme - 2](#)

Ici, les données sont organisées comme telles :

- Chaque URL fait office de clé;
- Chaque nombre de visiteurs fait office de valeur.

Cela permet également de trier le tableau beaucoup plus facilement.

[Voir fonction de formatage des données reçues par Plausible](#)

Problème de taille d'URL

Une fois le système de mise au propre des données effectué, j'ai voulu le tester sur plusieurs marques. Ces tests n'étaient à la base que de la curiosité : je voulais voir le trafic qu'engendraient les différentes marques, quelles étaient les plus populaires. Je voulais aussi voir concrètement la différence de trafic entre une marque en compte gratuit et une autre payante.

Seulement, lors de mes essais, il m'est arrivé régulièrement de faire face à un problème : Plausible ne me renvoyait pas de données.

Après plusieurs recherches et discussions avec mes collègues, nous nous sommes rendu compte d'un point bloquant : les navigateurs limitent les URLs à une certaine taille. Si une URL dépasse la limite imposée, elle ne sera pas envoyée au service. Les URLs étaient trop longues du fait que je doive donner à Plausible les pages sur lesquelles je voulais ses données. Or, si une marque a beaucoup de produits ou beaucoup d'articles, la limite est vite atteinte.

Browser	Maximum URL Length
Chrome	2,083 characters
Firefox	65,536 characters
Safari	80,000 characters
Internet Explorer	2,083 characters

Je devais donc changer de méthode pour récupérer mes informations, étant donné que certaines de mes URLs allaient poser problème.

Après réflexion, la solution la plus simple serait de découper mes URLs. En effet, comme celle-ci sont construites d'après un modèle, je n'aurais qu'à envoyer plusieurs fois une requête avec des URLs de pages différentes pour avoir au final mes données. J'ai donc procédé comme ceci :

Si une URL est détectée comme étant trop longue, celle-ci sera découpée à la fin de l'une des URLs de pages, un autre envoi sera fait et ainsi de suite.

[Voir la fonction de séparation d'urls](#)

[Voir l'amélioration de la fonction get_plausible_datas](#)

Comme le montre les deux captures d'écran ci-dessus, les URLs sont maintenant découpées quand celles-ci sont jugées trop longues (pour ne pas avoir d'erreur, nous avons choisi une limite plus basse, pour laisser suffisamment de caractères pour le début de l'URL).

Création progress bar et chart

Une fois mes données récupérées, je n'avais plus qu'à les mettre en forme. Pour cette page, nous avions besoin de deux types de représentations :

- Des progress bars, pour mesurer l'impact de la marque (pays, pages) ;
- Un chart, pour mesurer l'impact de la marque en fonction du temps.

Je reviendrai sur le chart un petit peu plus tard, intéressons-nous tout d'abord aux progress bars. Pour les réaliser, je suis parti sur quelque chose de très simple : mettre un background aux différentes barres en fonction de leur avancement.

[Voir la fonction de création des graphes](#)

Le code ci-dessus va récupérer l'ensemble des pages vues par pays, et va construire les barres avec. Pour ce faire, nous prenons la plus grande valeur (qui sera notre valeur de référence, la barre qui sera complétée à 100%).

Ensuite je change la taille de mes éléments, en fonction de la plus haute valeur de manière proportionnelle.

Utilisation d'ApexCharts.js

En ce qui concerne le chart, j'ai eu l'occasion d'utiliser la bibliothèque ApexCharts.js.

Celle-ci permet de créer facilement différents types de graphiques.

Le type de graphique que je devais créer était un area chart (une ligne représentant le nombre de visiteurs en fonction du temps).



Le code ci-dessous montre une configuration d'un chart avec Apex. Ces options sont stockées dans un objet, qui sera ensuite transmis à la bibliothèque pour lui permettre d'afficher les données. Cet objet représentant les options contient les données, les différents paramètres d'affichage et de style.

[Voir options d'ApexCharts.js](#)

Finalisation du Front

Une fois les informations mises en place, la page ressemblait à ceci.

[Voir page intégrée - 2 images](#)

Seulement, la plateforme possède un système de compte avec différents niveaux d'abonnements, je devais donc permettre à certains utilisateurs d'accéder à l'ensemble des données et les restreindre pour d'autres.

[Voir attribution des privilèges](#)

J'ai donc défini quelles sections pourraient être consultées par quels types de compte. Cette approche a l'avantage d'être facilement modifiable.

[Voir page plan gratuit](#)

Pour faire en sorte que les données soient réellement inaccessibles pour l'utilisateur n'ayant pas un niveau de compte assez élevé, celles-ci ont été hachées.

[Voir fonction de hash d'informations](#)

De cette manière, les chaînes de caractères gardent la même taille. Il était important de garder cela, pour créer une cohérence entre les versions gratuite et payante. De cette façon, si une personne passe en compte payant, les données qu'elle aura auront le même aspect que celles qui lui ont été cachées.

De même pour les nombres, ceux-ci sont hachés, pour garder les données intactes (pour les progress bar par exemple), mais celles-ci n'ont plus de sens pour l'utilisateur.

Problème de crédit

Arrivé à cette étape, le projet était terminé.

La dernière étape était la mise en production du projet (déploiement du code du site de développement vers le site live).

Seulement, pendant les dernières phases de tests, nous nous sommes rendu compte que la limite des crédits API pouvait être rapidement atteinte si beaucoup d'utilisateurs étaient connectés en même temps sur cette page et faisaient beaucoup d'actions, ou s'ils rafraîchissaient de manière abusive leur page Analytics. Une fois ce quota de crédit dépassé, l'API de Plausible ne nous renvoie plus de données, seulement des codes d'erreurs.

Le fait de se retrouver bloqué après avoir dépensé tous ses crédits était particulièrement problématique, car la plupart des utilisateurs visitaient cette page et y effectuaient des actions en réponse à certains signaux (publication sur LinkedIn, envoi d'un récapitulatif mensuel aux marques).

Il fallait donc trouver une solution permettant aux utilisateurs de ne pas pouvoir faire face à ce genre de problèmes.

La solution qui a très vite été mise en avant, a été de stocker les informations fournies par Plausible et de les transmettre aux utilisateurs. Cela permettrait de garder une trace des données des marques et de pouvoir leur proposer un service qui ne dépend plus des crédits dépensés.

Mise en base de données

Une fois le problème des crédits API identifiés, j'ai donc commencé à travailler sur la structure d'une base de données permettant de stocker les données de l'ensemble des utilisateurs.

Création de la base de données

La base de données était à l'origine très simple (3 champs) et je lui en ai rajouté au fur et à mesure du développement pour permettre d'avoir davantage d'informations.

id	brand_id	brand_name	update_time	call_weight	analytics_datas	is_update
50	88013	HoloAM	2024-04-04 17:04:26	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
51	83159	Titomic	2024-04-04 17:18:26	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
52	82467	ARBURGadditive	2024-04-04 16:54:25	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
54	83237	Xian Bright Laser Technologies	2024-04-04 17:40:32	36	{"yesterday": {"pages": {"Vbrands\Xian-Bright-Lase...": 1}}	1
55	82666	Exaddon	2024-04-04 17:00:28	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
56	82398	5AXISWORKS	2024-04-04 16:41:31	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
57	82372	3D Systems	2024-04-05 09:31:22	108	{"yesterday": {"pages": {"Vproduct\3d-printers\V3d...": 1}}	1
58	82464	APS - Tech Solutions	2024-04-04 16:20:25	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
59	137073	4D Pioneers	2024-04-04 17:42:25	18	{"yesterday": {"pages": {"Vfr\produit\Vimprimantes...": 1}}	1
60	107420	D-Shape	2024-04-04 16:42:30	18	{"yesterday": {"pages": [], "countries": []}, "pageviews...": 1}	1
61	82460	AON3D	2024-04-04 17:25:31	18	{"yesterday": {"pages": {"Vbrands\Aon3d\V": 1}, "Vpr...": 1}}	1

La base de données est composée de sept colonnes :

- Id : numéro d'identifiant unique à chaque tuple (ligne dans la base) ;
- brand_id : numéro d'identifiant de chaque marque sur le site ;
- brand_name : nom de la marque ;
- update_time : date et heure à laquelle a eu lieu la dernière mise à jour des données ;
- call_weight : nombre d'appels API nécessaires au remplissage des données de la marque ;
- analytics_data : contient l'ensemble des données pour tous les filtres proposés à l'utilisateur ;
- is_update : permet de savoir si les données stockées sont à jour par rapport aux autres.

J'expliquerai certains de ces champs plus en détails par la suite, mais cette base de données est ce sur quoi repose la section Analytics.

Stockage des données

Cette image montre comment est structurée l'information qui sera donnée par la suite aux marques dans leur MA.

Il s'agit d'une chaîne de caractères, encodée à l'aide de la fonction `json_encode()`.

Cette chaîne est structurée comme ceci : pour chaque période que l'utilisateur peut sélectionner, l'ensemble des données est sauvegardé.

Je n'aurai donc plus qu'à venir récupérer ces données et à les réutiliser pour donner les informations à l'utilisateur.

35	83073 HBD (Shanghai Hanbang United 3D Tech)	2024-04-04 17:30:29	36	hanbang-united-3d-tech-hbd-150\":3,"\\product\\3d-printers\\hbd-shanghai-hanbang-united-3d-tech-hbd-1000\\":2,"\\product\\3d-printers\\hbd-shanghai-hanbang-united-3d-tech-hbd-1000pro\\":1,"\\product\\3d-printers\\hbd-shanghai-hanbang-united-3d-tech-hbd-200\\":1,"\\product\\3d-printers\\hbd-shanghai-hanbang-united-3d-tech-hbd-500\\":1],"countries":
36	82656 Ermakian	2024-04-04 18:05:29	37	"\\Ch\\":1,"\\CZ\\":1,"\\DE\\":1,"\\FR\\":1,"\\MX\\":1),"\\pageviews\\":{\\\"2024-04-03 00:00:00\\":0,\"2024-04-03 01:00:00\\":0,\"2024-04-03 02:00:00\\":0,\"2024-04-03 03:00:00\\":0,\"2024-04-03 04:00:00\\":0,\"2024-04-03 05:00:00\\":0,\"2024-04-03 06:00:00\\":0,\"2024-04-03 07:00:00\\":0,\"2024-04-03 08:00:00\\":0,\"2024-04-03 09:00:00\\":0,\"2024-04-03 10:00:00\\":0,\"2024-04-03 11:00:00\\":0,\"2024-04-03 12:00:00\\":0,\"2024-04-03 13:00:00\\":0,\"2024-04-03 14:00:00\\":0,\"2024-04-03 15:00:00\\":0,\"2024-04-03 16:00:00\\":0,\"2024-04-03 17:00:00\\":10,\"2024-04-03 18:00:00\\":0,\"2024-04-03 19:00:00\\":0,\"2024-04-03 20:00:00\\":0,\"2024-04-03 21:00:00\\":0,\"2024-04-03 22:00:00\\":0,\"2024-04-03 23:00:00\\":1},\"\\total_pageviews\\":15},\"_days\\":{\\\"pages\\\":{\\\"\\product\\3d-printers\\hbd-shanghai-hanbang-united-3d-tech-hbd-200\\\":14,\"\\brands\\hbd-shanghai-hanbang-united-3d-tech\\\":9,\"\\product\\3d-printers\\hbd-shanghai-hanbang-united-3d-tech-hbd-150\\\":7,\"\\product\\imprimantes-3d\\hbd-shanghai-hanbang-united-3d-tech-hbd-150\\\":1,\"\\products\\3d-
37	82998 Prismlab	2024-04-04 18:12:32	38	
38	124049 GROB	2024-04-05 10:40:16	39	
39	142221 AMFREE	2024-04-04 15:52:26	40	
40	84226 Namma	2024-04-04 17:51:30	41	
41	88009 CDFI 3D	2024-04-04 16:21:59	42	
42	82646 eMotion Tech	2024-04-04 17:45:29	43	
43	82849 Lynxter	2024-04-05 10:55:10	44	
44	82976 Peopoly	2024-04-04 17:33:26	45	
45	82740 HeyGears	2024-04-05 10:45:17	46	
46	83089 Sizer Stream	2024-04-05 09:15:19		

Voir la fonction `get_plausible_datas` pour remplir la base de données

Cette fonction boucle donc en trois temps :

Pour chaque plage de temps filtrable proposée à l'utilisateur, la fonction récupère toutes les requêtes nécessaires pour remplir tous les champs, qui, à leur tour, parcourront l'ensemble des URLs à envoyer à l'API.

Elle permet également de faire deux choses en plus :

- Si l'ensemble des marques ont été mises à jour (toutes les colonnes `is_update` sont à 1), elle remet à tout le monde la valeur de `is_update` à 0 ;
- Si une marque est créée entre-temps, la fonction lui génère un tuple pour sa marque dans la base de données.

La fonction est donc globalement la même qu'avant, à la différence qu'il nous faut maintenant aller chercher les informations pour l'ensemble des scénarios d'un seul coup.

Une fois ces données récupérées, elles seront fournies à la fonction `update_bdd()`, qui s'occupera de les enregistrer dans la base de données présentée plus tôt.

Voir la fonction de mise à jour de la base de données

Nous n'aurons plus qu'à récupérer les données et à les décomposer grâce aux différents attributs de l'objet que nous retourne la base de données.

Tâche de Cron

Une fois toutes ces étapes terminées, il nous fallait rendre le tout automatique, pour que les données aillent se chercher toutes seules, sans avoir besoin d'intervenir.

Pour réaliser ceci, il existe un programme, nommé Cron - Chrono table, table de planification en français. Cron permet d'exécuter automatiquement des tâches à un moment précis. Ces tâches peuvent également être répétées dans le temps à intervalles réguliers.

De manière plus générale, Cron fait partie de la famille des daemon, mot qui désigne en informatique un processus s'exécutant en arrière-plan, sans que celui-ci soit sous le contrôle direct d'un utilisateur.

WordPress fournit directement des tâches de Cron, nous permettant ainsi d'aller régulièrement chercher des données.



```
add_action('cron_get_plausible_datas', 'get_plausible_datas');

function get_plausible_datas() {...}
```

Nous retrouvons donc notre fonction `get_plausible_datas()`, à qui nous allons ajouter un événement grâce à la fonction `add_action()`, qui prend en premier paramètre le nom de la tâche de Cron dans WordPress et la fonction à appeler quand cette tâche de Cron se déclenche.

[Voir le back-end WordPress de gestion des tâches de Cron](#)

Pour être le plus efficient possible, la tâche de Cron exécute la fonction `get_plausible_datas()` toutes les minutes. Cette approche a deux avantages :

- Elle permet de ne pas envoyer trop de requêtes d'un coup, ce qui peut être problématique au niveau de la bande passante ;
- Cela permet également de ne pas dépenser tous les crédits API d'un coup et de laisser des crédits si une marque venait à être créée.

Donc, chaque minute, nous remplissons le tuple d'une marque.

Pour chaque marque étant dans la base de données et dont la valeur dans la colonne `is_update` est à 0 (non mis à jour), celle-ci à une chance d'être sélectionnée pour la prochaine minute.

```
$query = $wpdb->prepare(
    "SELECT brand_id
     FROM $table_name
    WHERE is_update = 0
    ORDER BY CASE WHEN analytics_datas IS NULL OR analytics_datas = '' THEN 0 ELSE 1 END ASC, analytics_datas ASC
    LIMIT 1"
);
```

Cette requête SQL marche de la manière suivante :

Elle va chercher l'identifiant d'une marque non mise à jour (is_update à 0).

Ces données sont triées de sorte à ce qu'une marque sans donnée (une marque nouvellement créée par exemple), se retrouve devant les marques ayant seulement des données non mises à jour.

Il ne nous reste plus qu'à utiliser l'id récupéré pour générer les appels API vers Plausible et ainsi, mettre à jour les données de la marque.

Ce système permet à toutes les marques d'accéder rapidement à des données presque instantanément.

Création de variables pour les mails

Sur Aniwaa, la quasi-totalité des mails est envoyée de manière automatique. Ceux-ci sont construits selon des templates écrits manuellement.

[Voir template de mail automatique](#)

Ce template par exemple est le mail envoyé aux marques afin de leur faire part de leurs résultats sur la plateforme le mois dernier.

Chaque mot écrit entre pourcentage représente une variable. Ces variables sont des chaînes de caractères qui seront remplacées par le texte final avant l'envoi du mail au client grâce à la fonction PHP str_replace().

[Voir la gestion de variables des mails automatiques – 2 images](#)

Voici les fonctions que j'ai créées pour récupérer les données d'Analytics et pour les mettre en forme. Grâce à ces fonctions, il ne reste plus qu'à écrire %top_pages% pour avoir une liste des meilleures pages de la marque.

Plus tôt dans ce rapport, j'avais parlé du fait que certains types de comptes ne pouvaient pas avoir accès à certaines informations. Dans le cas des meilleures pages par exemple, une marque en compte gratuit n'a pas accès à l'information. Il ne faut donc pas l'afficher. Si une marque est en compte gratuit, nous retournons donc une chaîne vide à la place de la liste.

Création des vues catalogues

A ce stade, le projet est presque terminé, nous récupérons l'ensemble des données des pages vues avec plusieurs filtres pour les marques.

Après plusieurs discussions, nous avons néanmoins décidé de rajouter un type de donnée supplémentaire ne pouvant être fourni par Plausible : les vues de produits depuis le catalogue.

Le catalogue est l'outil permettant aux acheteurs d'accéder facilement aux produits disponibles sur Aniwaa. Il peut être trié à l'aide de filtre, et c'est généralement via ce catalogue que des produits sont consultés. Le catalogue est divisé en univers (imprimantes 3D, scanners 3D, ...) qui représentent les grandes familles de produits que propose Aniwaa.

AJAX

Afin de réaliser cette dernière partie de projet, je vais avoir besoin de détecter quand un utilisateur voit un produit sur le catalogue.

[Voir la détection d'un produit vu](#)

J'ai d'abord créé une fonction jQuery permettant de détecter quand un élément entre dans le champ de vision de l'utilisateur (un élément ne sera pas compté 2 fois même si l'utilisateur le fait redéfiler).

Une fois cette structure mise en place, il ne reste plus qu'à mettre en place la sauvegarde du fait que ce produit ait été vu. Pour cela, je vais avoir besoin d'une base de données.

Seulement, il y a un problème : la base de données se gère à l'aide de PHP (côté serveur), et la détection des vues catalogues se fait en JavaScript (côté client).

Comment communiquer les informations récupérées d'un côté pour les fournir à l'autre ?

Pour cela, il existe une méthode, appelée l'AJAX (Asynchronous JavaScript And XML).. C'est une méthode permettant d'envoyer des requêtes vers le serveur du site depuis le client.

[Voir l'utilisation d'AJAX](#)

Ce code montre l'intégration d'AJAX au processus de détection de vues des cartes. Nous utilisons la fonction `ajax()`, fournie avec jQuery pour cette tâche. La fonction prend en paramètre un certain nombre de données, comme par exemple le type de requête (POST, GET, ..). Il lui faut également une url, c'est le fichier vers lequel elle doit envoyer ses données. L'attribut `data`, permet lui de fournir les données que l'on souhaite transmettre.

La fonction prend également en paramètre la gestion du retour de la requête. L'attribut `complete` permet par exemple de gérer le cas de réussite. Dans ce cas là, nous n'avons pas besoin de retourner d'informations depuis PHP, mais c'est de cette manière qu'il faudrait procéder si nous en avions besoin.

De l'autre côté, nous avons la fonction `view_product()`, qui recevra les données en PHP depuis le JavaScript.

Les données sont donc extraites du tableau `$_POST`. Les clés du tableaux correspondent aux attributs que nous avions donnés précédemment à la fonction `ajax()` de jQuery.

Cette fonction se contente de récupérer les données et de les mettre à jour dans la base de données.

Finalement, la fonction `wp_send_json()` permet de retourner une réponse à JavaScript (comme le bon déroulé du processus par exemple).

Utilisation des Postmeta

Maintenant que nous avons vu comment faire transiter nos données et comment les sauvegarder, il nous reste un problème : où les sauvegarder ?

WordPress met à disposition une table de sa base de données nommée 'wp_postmeta', qui permet de stocker des informations liées aux posts, aux pages. Cette table marche avec une combinaison clé / valeur, et avec l'id du post en question.

[**Voir la mise à jour des compteurs de produits vus depuis le catalogue**](#)

[**Voir la récupération des vues catalogue depuis MA Analytics**](#)

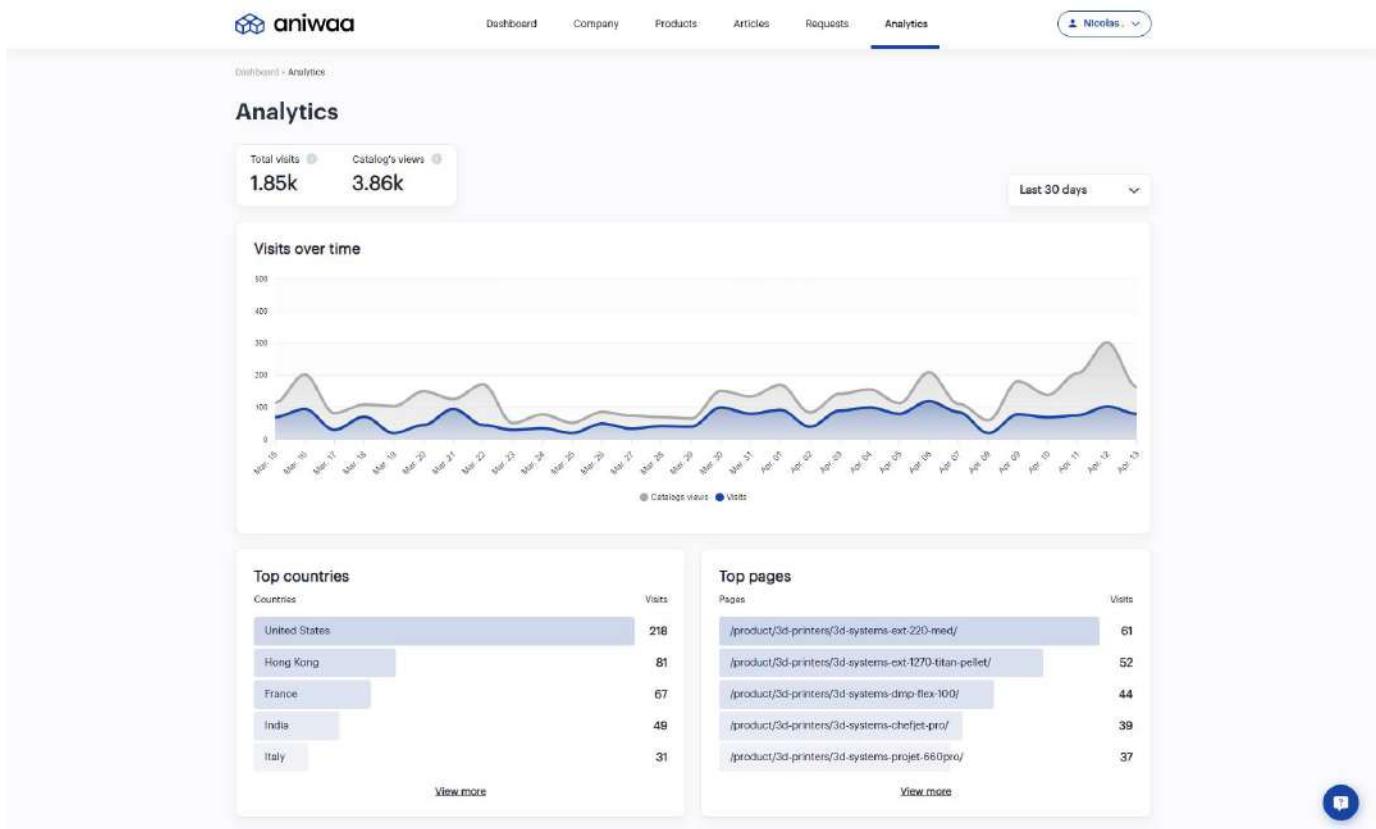
meta_id	post_id	meta_key	meta_value	Actions
5272767	82372	catalog_view	s:73:"a:7:{i:14;i:5;i:16;i:4;i:12;i:2;i:13;i:4;s:2..."	Supprimer Copier Éditer □
5272768	88203	catalog_view	s:60:"a:6:{i:14;i:6;i:16;i:4;i:12;i:2;i:11;i:2;i:1..."	Supprimer Copier Éditer □
5272769	82785	catalog_view	s:60:"a:6:{i:14;i:3;i:16;i:6;i:12;i:3;i:11;i:5;i:1..."	Supprimer Copier Éditer □
5272770	82869	catalog_view	s:60:"a:6:{i:14;i:3;i:16;i:2;i:12;i:1;i:11;i:1;i:1..."	Supprimer Copier Éditer □
5272771	82454	catalog_view	s:60:"a:6:{i:14;i:1;i:16;i:2;i:12;i:1;i:11;i:2;i:1..."	Supprimer Copier Éditer □
5272772	83018	catalog_view	s:84:"a:8:{i:14;i:12;i:16;i:9;i:12;i:3;i:15;i:9;i:1..."	Supprimer Copier Éditer □

Grâce à ceci, nous pouvons donc facilement lier un fabricant avec son nombre de vues.

Les données dans la colonne meta_value correspondent aux données de vues. Il s'agit de données qui sont sérialisées avant d'être enregistrées dans la base.

Ces données sont enregistrées puis stockées dans la table wp_analytics (table où sont stockées les informations retournées par Plausible). Ces données sont enregistrées heure par heure, cela me permet de connaître précisément l'activité et donc de proposer une courbe de données sur l'intervalle d'une journée (courbe qui ne pourrait pas être réalisée sans cette granularité).

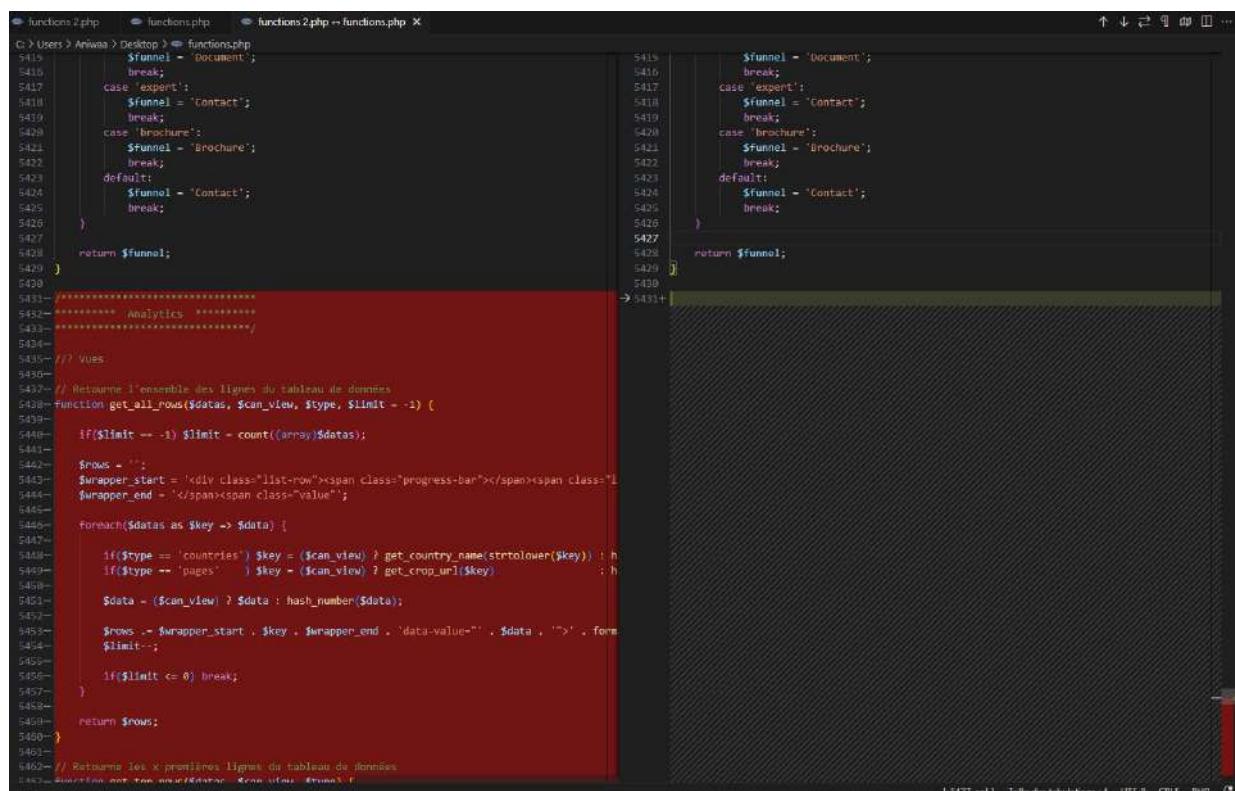
Pour finir, il me suffit de lier ces données à ma page Analytics. Pour cela, j'ai créé une fonction me permettant d'aller chercher ces données en fonction de la date choisie par l'utilisateur.



Déploiement du projet

Une fois le projet terminé, il ne reste plus qu'à le faire passer du site de développement (abc.aniwaa.com), au site live (www.aniwaa.com).

Pour ce faire, je dois reprendre tous les fichiers que j'ai modifiés un à un et faire le comparatif de ces fichiers avec ceux présents sur le site live.



The screenshot shows a code editor comparing two versions of a PHP file, likely named functions.php. The left pane contains the original code, and the right pane contains the modified code. The modifications are highlighted in red. The code includes logic for determining a funnel type ('Document', 'Contact', 'Brochure') based on input, and a function to get all rows from a database table. The code uses variables like \$funnel, \$rows, \$wrapper_start, \$wrapper_end, and \$data. The interface includes tabs for 'functions 2.php' and 'functions.php', and status bars at the bottom indicating file number, tab count, and encoding.

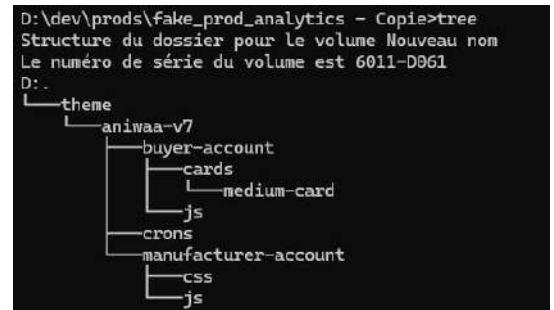
```
functions 2.php  functions.php  functions 2.php -- functions.php X
C:\>Users>Aniwaa>Desktop>functions.php
5415     $funnel = 'Document';
5416     break;
5417   case 'expert':
5418     $funnel = 'Contact';
5419     break;
5420   case 'brochure':
5421     $funnel = 'Brochure';
5422     break;
5423   default:
5424     $funnel = 'Contact';
5425     break;
5426   }
5427 
5428   return $funnel;
5429 }
5430 
5431 //----- Analytics -----
5432 //----- VUES -----
5433 // Retourne l'ensemble des lignes du tableau de données
5434 function get_all_rows($datas, $can_view, $type, $limit = -1) {
5435 
5436   if($limit == -1) $limit = count((array)$datas);
5437 
5438   $rows = '';
5439   $wrapper_start = '<div class="list-row"><span class="progress-bar"></span><span class="l';
5440   $wrapper_end = '</span><span class="value">';
5441 
5442   foreach($datas as $key => $data) {
5443 
5444     if($type == 'countries') $key = ($can_view) ? get_country_name(strtolower($key)) : h;
5445     if($type == 'pages')    $key = ($can_view) ? get_page_url($key) : h;
5446 
5447     $data = ($can_view) ? $data : hash_number($data);
5448 
5449     $rows .= $wrapper_start . $key . $wrapper_end . 'data-value="' . $data . '">' . form;
5450     $limit--;
5451 
5452     if($limit <= 0) break;
5453   }
5454 
5455   return $rows;
5456 }
5457 
5458 // Retourne les X premières lignes du tableau de données
5459 function get_top_n_rows($datas, $can_view, $rows) {
5460 }
```

La capture d'écran ci-dessus me permet de comparer les deux fichiers et de facilement pouvoir fusionner mon travail avec le fichier existant.

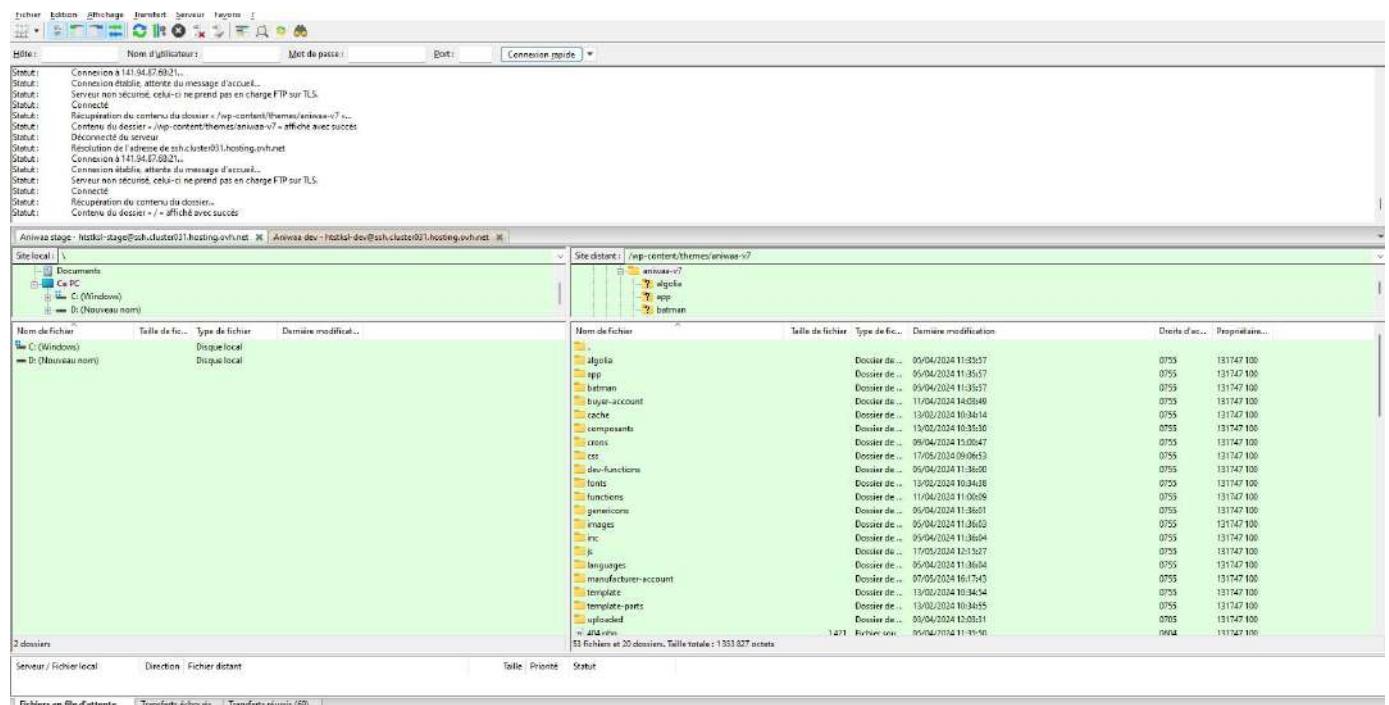
Dans cet exemple, les modifications sont faciles à transférer, mais il arrive de devoir travailler sur du code qui a été touché depuis notre version et de devoir comparer certaines lignes entre elles pour décider lesquelles garder.

Pour que le déploiement soit le plus efficace possible, je dois recréer une structure semblable à celle sur le serveur, comprenant les fichiers que j'ai modifiés.

Cette image du terminal montre l'arborescence de mon dossier de passage en production. Il faudra par la suite simplement le drag and drop (glisser-déposer) sur le FTP du site.



Le FTP (File Transfert Protocol) est un protocole de communication destiné au partage de fichiers entre un utilisateur et une machine distante.



Ceci est l'interface du logiciel FileZilla, un client FTP me permettant d'interagir avec les fichiers du serveur.

C'est ici que je vais déposer mes fichiers pour remplacer les anciens, et ainsi donner accès aux utilisateurs aux nouvelles fonctionnalités.

Appports du stage

Connaissances acquises

Ce stage m'a permis de développer énormément de compétences.

Tout d'abord, au niveau technique, j'ai beaucoup progressé sur la compréhension poussée d'un thème WordPress aussi important en termes de taille.

J'ai également beaucoup appris grâce aux conseils et aide de mes collègues Alban et Clément, que ce soit sur un aspect de WordPress, ou sur les langages que nous utilisions (JS, PHP, SQL, CSS).

J'ai aussi appris à utiliser une API de A à Z : de la lecture de la documentation jusqu'à la récupération des données dont j'avais besoin. Cet exercice a été très formateur, car les documentations sont parfois compliquées à comprendre et il faut donc, au même titre que de faire pour apprendre, s'entraîner à lire une documentation technique.

Enfin, j'ai pu progresser sur ma maîtrise de l'anglais. En effet, le site était d'abord conçu pour un public anglophone, et il fallait donc travailler en anglais (rédaction de templates de mail, création de textes pour le site).

J'ai également pu m'améliorer grâce au fait que la documentation technique que j'avais était en anglais.

Bilan humain

Mais le stage m'a davantage permis d'apprendre sur la posture professionnelle.

J'ai appris à mieux communiquer, à me faire comprendre plus facilement. J'avais par exemple tendance en début de stage à faire part à Alban et Clément de mes problèmes techniques de cette façon :

"Il y a quelque chose qui ne marche pas quand j'essaie de récupérer les données de Plausible".

Je n'y faisais pas attention, mais ces phrases ne pouvaient pas me permettre d'être facilement aidé par mes collègues, qui m'ont dit plusieurs fois d'essayer de ne plus utiliser de mot fourre-tout ('truc') et de bien expliquer mon problème, le contexte dans lequel cela se produisait, etc. Car ils n'étaient pas sur le même projet que moi et qu'une demande d'aide sans plus de détail les obligeait à me poser des questions pour comprendre, alors que j'aurais pu leur fournir toutes ces informations directement dans ma question.

"J'ai une fonction get_plausible_datas qui me permet de récupérer les données de Plausible, mais j'ai une erreur 500 quand j'appelle l'API pour récupérer les données sur les pages les plus vues".

Il y a également le fait de savoir s'adapter aux emplois du temps des autres personnes. Par exemple, quand j'avais besoin de faire une réunion avec Martin (directeur d'Aniwaa), j'ai appris à d'abord regarder ses disponibilités sur Google Calendar et lui proposer directement une réunion avec un horaire et une durée estimée plutôt que de simplement lui demander sur Slack.

Poursuite d'études

Ce stage m'a permis de découvrir le monde de l'entreprise dans une plus grosse structure que l'année dernière, où j'étais dans une agence web.

Ce stage m'a conforté dans l'idée de poursuivre dans ce domaine (développement web). Je compte donc poursuivre dans le monde du travail après mon BUT MMI.

Glossaire

Logiciels :

Slack : plateforme de communication collaborative (équivalent à un Discord professionnel).

Plausible : plate-forme qui collecte des données à partir de sites Web (équivalent à Google Analytics).

Technique :

FTP : protocole de communication destiné au partage de fichiers sur un réseau TCP/IP.

Cron : programme permettant d'exécuter automatiquement des tâches à un moment précis.

AJAX : méthode permettant d'envoyer des requêtes vers le serveur du site depuis le client.

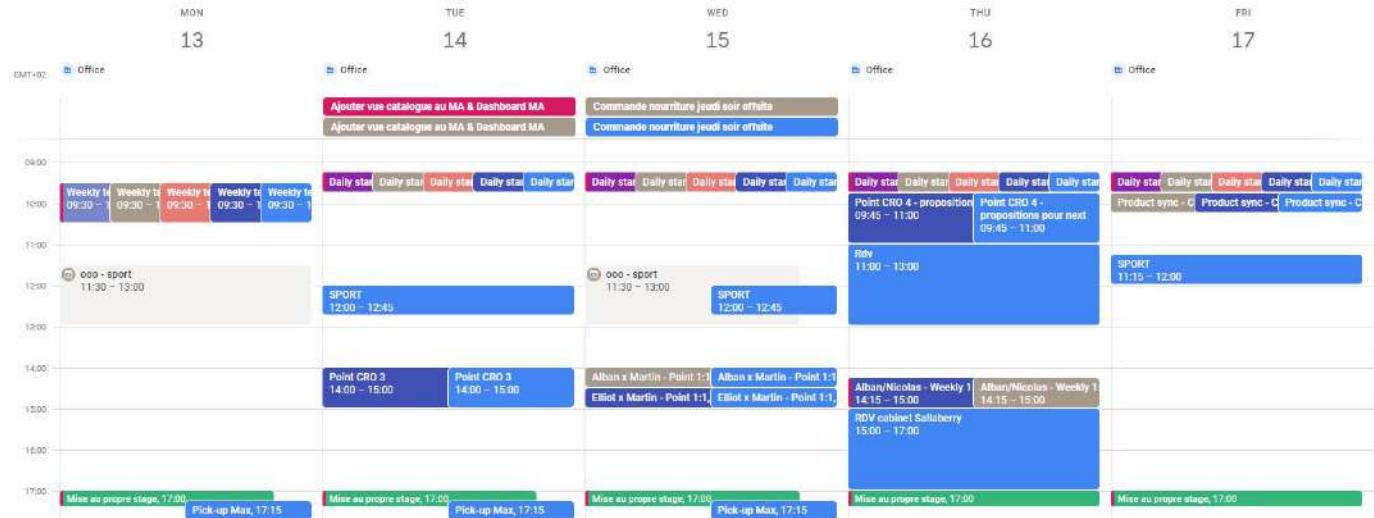
Tuple : ligne de donnée dans une base de données.

Aniwaa :

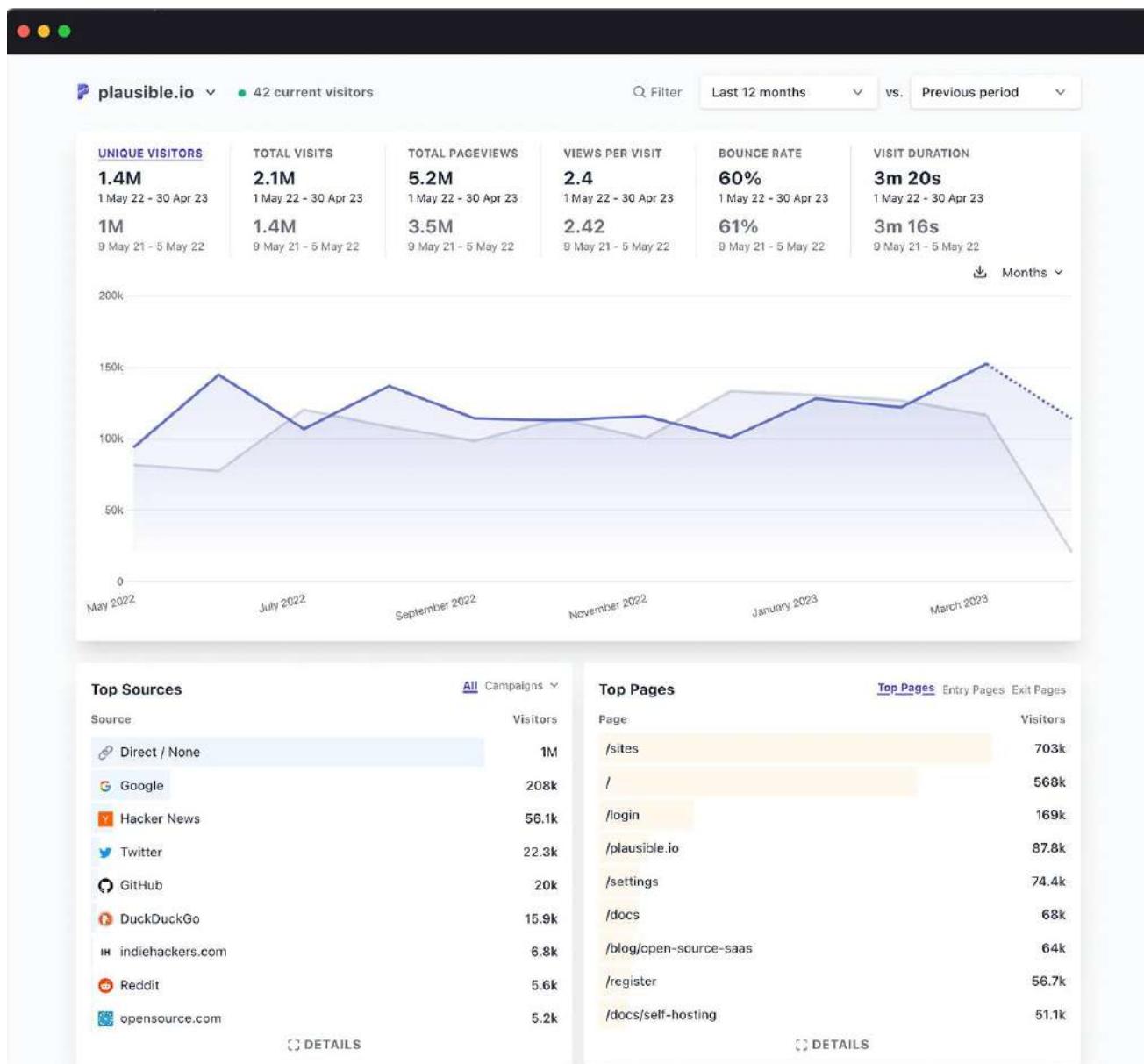
BA - Buyer account : compte sur Aniwaa pour les acheteurs.

MA - Manufacturer account : compte sur Aniwaa pour les fabricants.

Illustrations

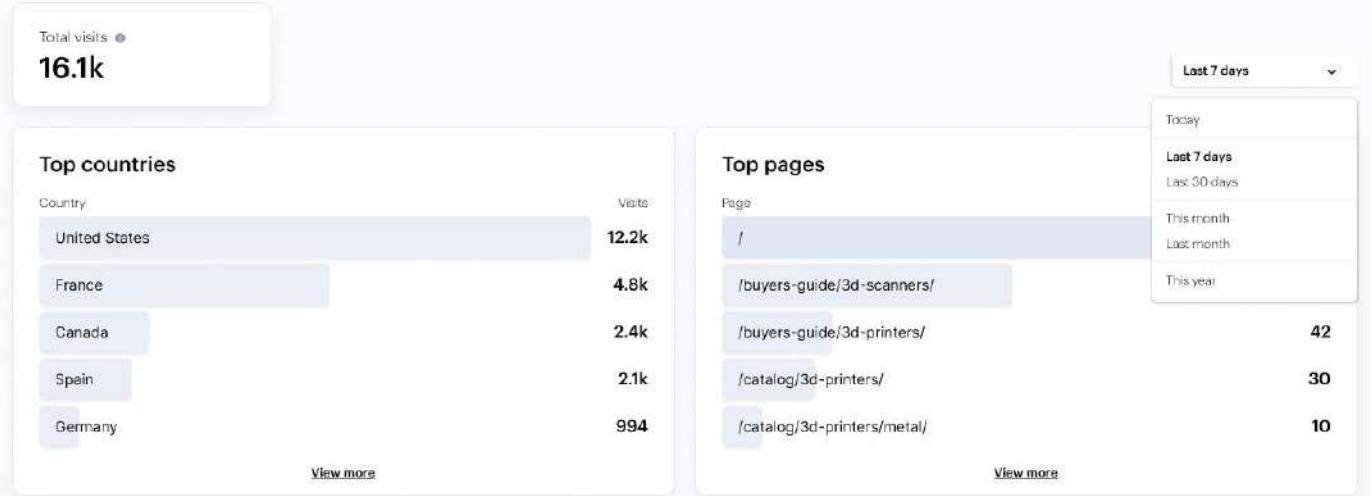


Google calendar



Interface de Plausible

Analytics



Maquettes Figma

```

function get_plausible_datas($url, $link = 'breakdown') {

    $site_id = 'aniwaa.com';
    $access_token = "REDACTED";
    $headers = [
        'Authorization' => 'Bearer ' . $access_token,
        'Content-Type' => 'application/json'
    ];

    $url = "https://plausible.io/api/v1/stats/$link?site_id=" . $site_id . $url;

    $response = wp_remote_get($url, ['headers' => $headers]);
    return json_decode($response['body'])->results;
}

```

[Voir envoi de données à Plausible](#)

```

$pageview_datas = get_plausible_datas(
    $filter_url,
    $current_duration_query . get_pageview_timeline_request(),
    'pageview',
    'timeseries'
);

```

[Voir récupération des données via Plausible](#)

```

// Retourne l'ensemble des ids des posts concernants la marque
function get_brand_elements($brand_id) {

    $brand_products_en = get_brand_account_products_by_brand_id(array('publish'), $brand_id);
    $brand_products_fr = get_post_translation($brand_products_en, 'fr');
    $brand_products     = array_merge($brand_products_en, $brand_products_fr);

    $brand_articles_en = extract_articles_ids(get_brand_articles_by_brand_id(array('publish'), $brand_id));
    $brand_articles_fr = get_post_translation($brand_articles_en, 'fr');
    $brand_articles     = array_merge($brand_articles_en, $brand_articles_fr);

    $brand_page_en     = $brand_id;
    $brand_page_fr     = get_post_translation($brand_id, 'fr');
    $brand_pages        = array($brand_page_en, $brand_page_fr);

    return [
        'brand_products' => array_values(array_filter($brand_products, "is_not_equal_to_zero")),
        'brand_articles' => array_values(array_filter($brand_articles, "is_not_equal_to_zero")),
        'brand_pages'      => array_values(array_filter($brand_pages, "is_not_equal_to_zero"))
    ];
}

```

Récupération des pages dans les deux langues

```

// Permet d'obtenir l'ensemble des URL filtrées des pages concernant la marque
function get_brands_urls($ID_lists) {
    $site_name = 'https://' . $_SERVER['SERVER_NAME'];
    $urls = array();

    foreach($ID_lists as $ID_list) {
        foreach($ID_list as $element) {
            $urls[] .= str_replace($site_name, '', get_permalink($element));
        }
    }

    return (!empty($urls)) ? $urls : false;
}

```

Récupération des URLs de marques

```

Array
(
    [0] => /product/3d-scanners/shining-3d-autoscan-inspec/
    [1] => /product/3d-scanners/shining-3d-autoscan-sparkle/
    [2] => /product/3d-scanners/shining-3d-autoscan-ds100/
    [3] => /product/3d-scanners/shining-3d-autoscan-ds200/
    [4] => /product/3d-scanners/shining-3d-autoscan-ds300/
    [5] => /product/3d-scanners/shining-3d-autoscan-ha/
    [6] => /product/3d-scanners/shining-3d-autoscan-it/
    [7] => /product/3d-scanners/shining-3d-digimetric/
    [8] => /product/3d-scanners/shining-3d-eascan-2/
    [9] => /product/3d-scanners/shining-3d-eascan-5m/
    [10] => /product/3d-scanners/shining-3d-eascan-d/
    [11] => /product/3d-scanners/shining-3d-eascan-q/
    [12] => /product/3d-scanners/shining-3d-eascan-t/
    [13] => /product/3d-scanners/shining-3d-einscan/
    [14] => /product/3d-scanners/shining-3d-einscan-h/
)

```

Tableau des URLs récupérées

```

[body] -> {"results":{"visitors":{"value":451}}}
[response] -> Array
    (
        [code] -> 200
        [message] -> OK
    )

[cookies] -> Array
    (
    )

[filename] ->
[http_response] -> WP_HTTP_Requests_Response Object
    (
        [response:protected] -> Requests_Response Object
            (
                [body] -> {"results":{"visitors":{"value":451}}}
                [raw] -> HTTP/1.1 200 OK
            )
    )

```

Réponse de Plausible

```

Array
(
    [0] => stdClass Object
        (
            [page] => /product/3d-printers/eos-eosint-m-100/
            [visitors] => 102
        )

    [1] => stdClass Object
        (
            [page] => /product/3d-printers/eos-eosint-m-280/
            [visitors] => 88
        )

    [2] => stdClass Object
        (
            [page] => /product/3d-printers/eos-eosint-m-290/
            [visitors] => 35
        )
)

```

Données de Plausible mises en forme (1)

```

(
    [/product/3d-printers/3d-systems-sla-750/] => 25
    [/product/3d-printers/3d-systems-dmp-flex-100/] => 20
    [/brands/3d-systems/] => 17
    [/product/3d-printers/3d-systems-sls-380/] => 15
    [/product/3d-printers/3d-systems-dmp-factory-500/] => 15
    [/product/3d-printers/3d-systems-projet-660pro/] => 13
    [/product/3d-printers/3d-systems-sls-300/] => 13
    [/product/3d-printers/3d-systems-ext-220-med/] => 13
    [/product/3d-printers/3d-systems-ext-1270-titan-pellet/] => 9
    [/product/3d-printers/3d-systems-projet-6000-hd/] => 8
    [/product/3d-printers/3d-systems-dmp-flex-350-triple/] => 8
    [/product/3d-printers/3d-systems-mjp-300w/] => 7
    [/product/3d-printers/3d-systems-ext-titan-pellet/] => 7
    [/fr/product/imprimantes-3d/3d-systems-projet-660pro/] => 7
    [/product/3d-printers/3d-systems-prox-dmp-200/] => 7
    [/product/3d-printers/3d-systems-dmp-flex-350/] => 7
    [/product/3d-printers/3d-systems-ext-1070-titan-pellet/] => 7
    [/product/3d-printers/3d-systems-projet-3510-hd/] => 7
)

```

Données de Plausible mises en forme (2)

```

// Tri les données récupérées par l'API
function format_plausible_datas($responses, $type) {

    foreach($responses as $response)
        $datas = array_merge($datas ?? array(), json_decode($response['body']))->results;

    foreach($datas as $data) {

        if($type == 'page')      $key = get_crop_url($data->$type);
        if($type == 'country')   $key = get_country_name(strtolower($data->$type));

        $visitors = $data->visitors;
        $sort_responses[$key] = $visitors;
    }

    arsort($sort_responses);

    return $sort_responses;
}

```

Fonction de formatage des données reçues par Plausible

```

// Sépare les url en plusieurs parties
function split_url($url, $max_size = 1950) {

    if(strlen($url) < $max_size) return [$url];

    $url_parts = explode('|', $url);
    $cpt = 0;

    for($i = 0; $i < count($url_parts); $i++) {
        if(strlen($urls[$cpt]) + strlen($url_parts[$i]) > $max_size)
            $cpt++;

        $urls[$cpt] .= $url_parts[$i] . '|';
    }

    $urls[$cpt] = substr($urls[$cpt], 0, -1);

    return $urls;
}

```

Fonction de séparation d'urls

```

// Permet de se connecter à l'API et de récupérer les données
function get_plausible_datas($urls, $parameters, $type, $link = 'breakdown') {

    $site_id = 'aniwaa.com';
    $access_token = 'REDACTED';
    $headers = [
        'Authorization' => 'Bearer ' . $access_token,
        'Content-Type' => 'application/json'
    ];

    $urls = split_url($urls);

    foreach($urls as $url) {
        $final_url = "https://plausible.io/api/v1/stats/$link?site_id=" . $site_id . $parameters . $url;
        $responses[] = wp_remote_get($final_url, ['headers' => $headers]);
    }

    return format_plausible_datas($responses, $type);
}

```

Amélioration de la fonction get_plausible_datas

```

function get_analytics_top_countries_progress_bar() {

    const progress_bars = jQuery('.analytics-top-countries .progress-bar');
    const max_number = progress_bars.first().siblings('.value').data('value');

    progress_bars.each(function() {
        const value = jQuery(this).siblings('.value').data('value');
        jQuery(this).css('width', get_completion_pourcentage(max_number, value));
    });
}

function get_completion_pourcentage(max_number, value) {
    return ((value / max_number) * 100) + '%';
}

```

Fonction de création des graphes

```

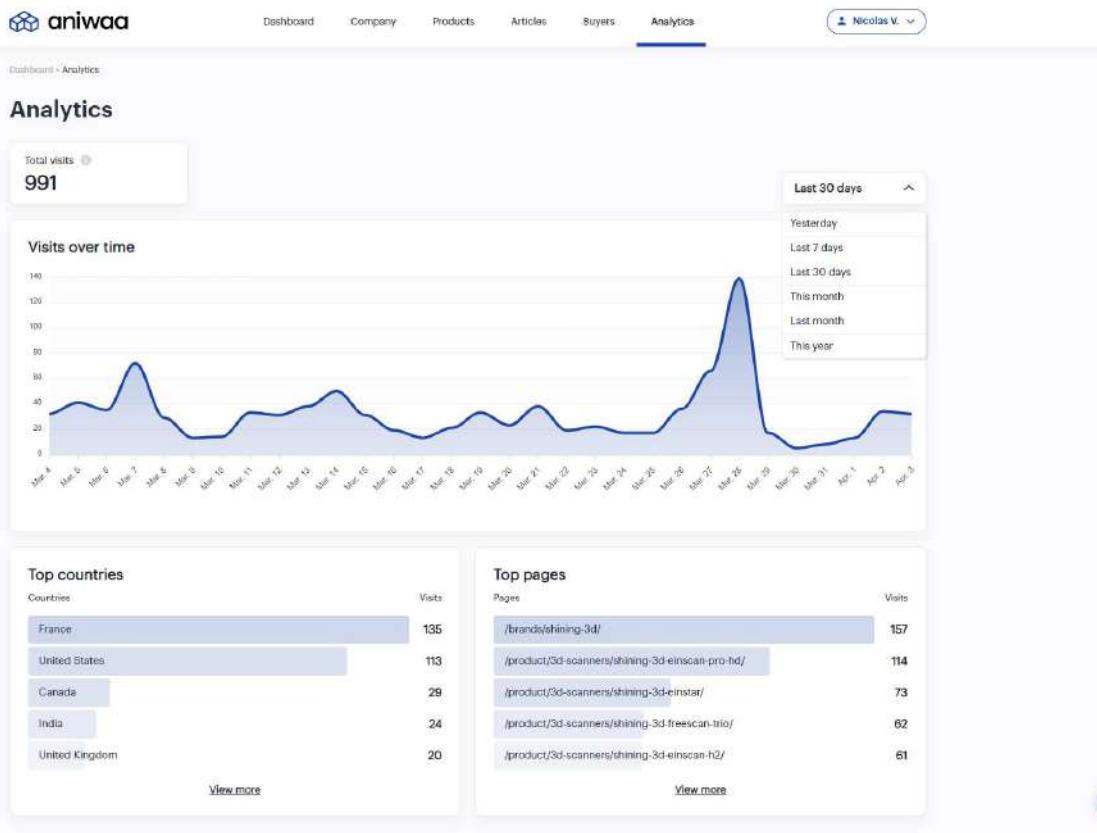
const options = {
    series: [
        {
            name: 'Visits',
            data: <?= $chart_datas['values']; ?>
        }],
    chart: {
        height: 350,
        width: 1260,
        type: 'area',
        zoom: {
            enabled: false
        },
        toolbar: {
            show: false
        }
    },
    dataLabels: {
        enabled: false
    },
    colors: ['#1b45a5'],

    title: {
        text: '',
        align: 'left'
    },
    grid: {
        borderColor: '#e9ecef',
        row: {
            colors: ['#f3f3f3', 'transparent'],
            opacity: 0.2
        },
    },
    yaxis: {
        min: 0,
        stepSize: <?= ($current_duration == 'yesterday') ? 1 : "undefined"; ?>
    },
    xaxis: {
        labels : {
            rotateAlways: true,
            rotate: -45,
            offsetY: 10,
        },
        categories: <?= $chart_datas['keys']; ?>
    }
};

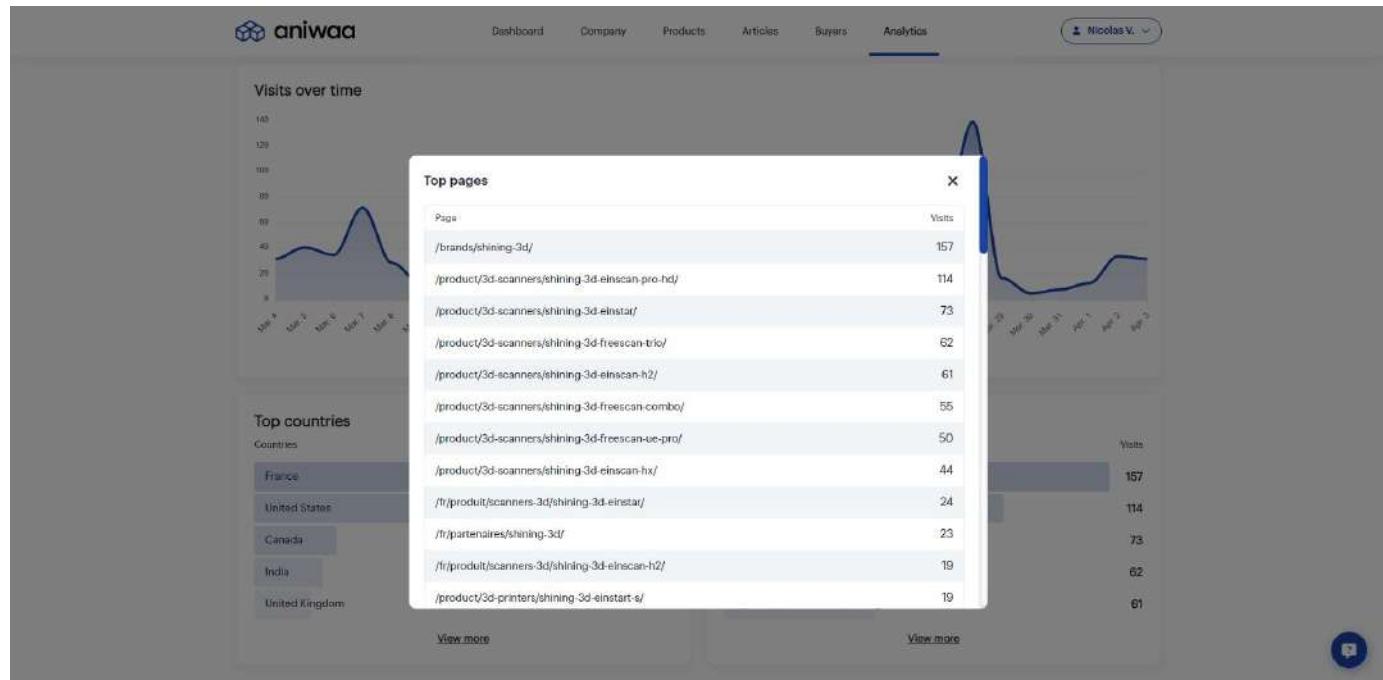
const chart = new ApexCharts(document.querySelector(".chart-pageview"), options);
chart.render();

```

Options d'ApexCharts.js



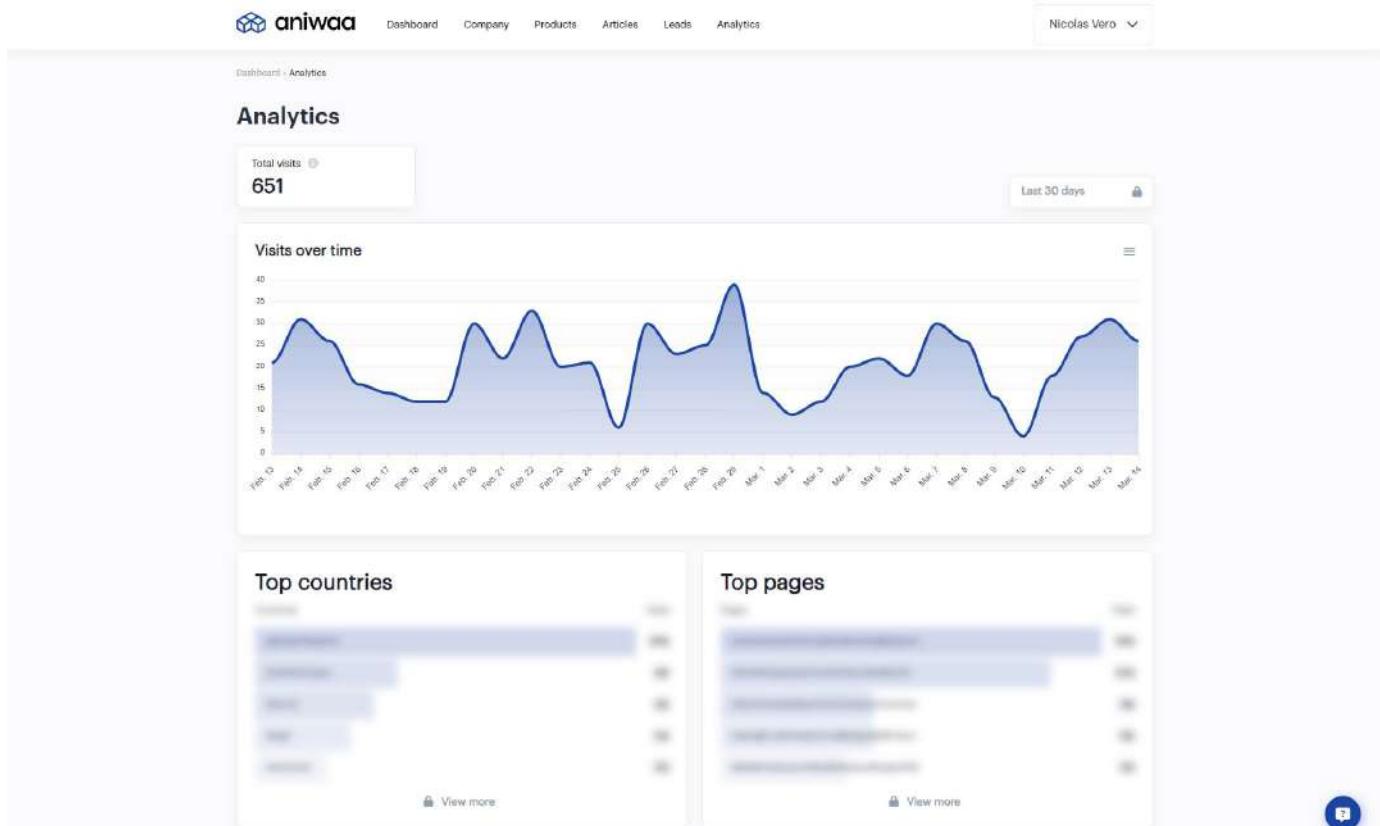
Page intégrée - 1



Page intégrée - 2

```
analytics_date_picker'      => ['starter','growth'],
'analytics_total_visits'    => ['free','starter','growth'],
'analytics_total_visits_chart' => ['free','starter','growth'],
'analytics_top_countries'   => ['starter','growth'],
'analytics_top_pages'       => ['starter','growth'],
'analytics_view_more'       => ['starter','growth']
```

Attribution des priviléges



Page plan gratuit

```
// Retourne une fausse chaîne de caractères
function hash_string($string) {
    for($i = 0; $i < strlen($string); $i++)
        $fake_string .= chr(rand(97,122));

    return $fake_string;
}

// Retourne un faux nombre
function hash_number($number) {
    return str_pad(ceil($number * 0.12), strlen($number), '0', STR_PAD_LEFT);
}
```

Fonction de hash d'informations

```

// Permet d'aller chercher les données de Plausible pour les mettre en base de données
function get_plausible_datas() {

    $site_id = 'aniwaa.com';
    $access_token = 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX';
    $headers = [
        'Authorization' => 'Bearer ' . $access_token,
        'Content-Type' => 'application/json'
    ];

    if(are_all_updates())
        reset_update_counter();

    insert_unindexed_brands(get_all_brand_ids());
    $brand_id = get_first_updateless_brand_id();

    $brand_datas = get_brand_elements($brand_id);

    $brand_products = $brand_datas['brand_products'];
    $brand_articles = $brand_datas['brand_articles'];
    $brand_pages     = $brand_datas['brand_pages'];

    $urls = get_brands_urls([$brand_products, $brand_articles, $brand_pages]);
    $urls = implode_urls($urls);
    $urls = split_url($urls);

    $duration_queries = get_time_parameters();
    $requests = get_all_requests();

    $datas = array();
    $call_weight = 0;

    foreach($duration_queries as $duration_key => $duration_query) {
        foreach($requests as $request_key => $request) {
            foreach($urls as $url) {
                $final_url = "https://plausible.io/api/v1/stats/" . $request['slug'] . "?site_id=" . $site_id . $duration_query . $request['request'] . $url;
                $response = wp_remote_get($final_url, ['headers' => $headers]);

                if($response['response']['code'] != 200) {
                    return;
                }

                $data = json_decode($response['body'])->results;

                if(!isset($datas[$duration_key][$request_key])) {
                    $datas[$duration_key][$request_key] = $data;
                } else {
                    $datas[$duration_key][$request_key] = array_merge($datas[$duration_key][$request_key], $data);
                }

                $call_weight++;
            }
        }
    }

    update_bdd(merge_datas($datas), $call_weight, $brand_id);
}
?>

```

Fonction get_plausible_datas pour remplir la base de données

```

// Mets à jour la base de données avec les nouvelles informations
function update_bdd($datas, $call_weight, $brand_id) {

    global $wpdb;
    $table_name = $wpdb->prefix . 'analytics';

    date_default_timezone_set('Europe/Paris');

    $query = $wpdb->prepare(
        "UPDATE $table_name SET analytics_datas = %s, update_time = %s, call_weight = %d, is_update = 1 WHERE brand_id = %d",
        json_encode($datas),
        current_time('mysql'),
        $call_weight,
        $brand_id
    );

    $wpdb->query($query);
}

```

Fonction de mise à jour de la base de données

All events (41) Events with no action (0) WordPress core events (11) Custom events (30)				
Bulk actions	Apply	Export	Search Hook Names	
<input type="checkbox"/> Hook	Arguments	Next Run (UTC+2)	Action	Recurrence
<input type="checkbox"/> rocket_preload_process_pending	None	2024-05-07 16:25:48 now	WP_Rocket\Engine\Preload\Cron\Subscriber->process_pending_uris() WP_Rocket\Engine\Preload\Cron\Subscriber->clean_reload_jobs()	WP Rocket Preload pending jobs
<input type="checkbox"/> action_scheduler_run_queue	["nIP_Cron"]	2024-05-07 16:26:11 now	ActionScheduler\Runner->run()	Every minute
<input type="checkbox"/> wordfence_itc_ntp_cron	None	2024-05-07 16:24:10 7 minutes 42 seconds	Wordfence\LS\Controller_Time->wordfence_itc_ntp_cron()	Once Hourly
<input type="checkbox"/> wordfence_hourly_cron	None	2024-05-07 16:34:23 7 minutes 47 seconds	wordfence::hourlyCron()	Once Hourly
<input checked="" type="checkbox"/> wp_privacy_delete_old_export_files	None	2024-05-07 16:39:33 12 minutes 57 seconds	wp_privacy_delete_old_export_files()	Once Hourly
<input type="checkbox"/> wpseo_indexable_index_batch	None	2024-05-07 16:44:49 16 minutes 13 seconds	YoothemeSEO\Integrations\Admin\Background_Indexing_Integration->index()	Every fifteen minutes

Back-end WordPress de gestion des tâches de Cron

Email

Hello,

We're very happy to share the %last_month% monthly recap of %brand%'s activity on Aniwa with you.

%product_listings%

%company_page%

%buyers_interactions%

%separator%

To complete your profile and access buyers' information:

@Sign into your account@

Template de mail automatique

```
$variables = array(
    "%brand%",

    ...
    "%pageviews_last_30_days%",
    "%pageviews_last_month%",
    "%catalog_brandviews_last_month%",
    "%top_countries%",
    "%top_pages%"
);

$values = array(
    $brand_name,
    ...
    $pageviews_last_30_days,
    $pageviews_last_month,
    $catalog_brandviews_last_month,
    $top_countries,
    $top_pages
);
```

Gestion de variables des mails automatiques - 1



```
// Variable %pageviews_last_xyz%
function get_pageview_variable($brand_id, $period) {
    return formate_big_numbers(get_analytics_datas($brand_id)->{$period}->total_pageviews);
}

// Structure des variables %top-%%
function get_top_variable($brand_id, $period, $limit, $type) {

    if(!in_array(get_field('tier', $brand_id), ['starter', 'growth']))
        return '';

    $datas = (array) get_analytics_datas($brand_id)->{$period}->{$type};

    $structure = '<ul>';

    foreach($datas as $key => $value) {
        if($limit <= 0) break;
        $limit--;

        if($type == 'countries') $header = get_country_name(strtolower($key));
        if($type == 'pages')    $header = get_crop_url($key);

        $structure .= '<li>' . $header . ' : ' . formate_big_numbers($value) . '</li>';
    }

    $structure .= '</ul>';

    return $structure;
}

// Variable %top_countries%
function get_top_countries_variable($brand_id, $period = 'last_month', $limit = 5) {
    return get_top_variable($brand_id, $period, $limit, 'countries');
}

// Variable %top_pages%
function get_top_pages_variable($brand_id, $period = 'last_month', $limit = 5) {
    return get_top_variable($brand_id, $period, $limit, 'pages');
}
```

Gestion de variables des mails automatiques - 2



```
function checkViewport() {
    jQuery('.item.cart').each(function() {

        const $element      = jQuery(this);
        const elementTop   = $element.offset().top + 150;
        const elementBottom = elementTop + $element.outerHeight();
        const viewportTop  = jQuery(window).scrollTop();
        const viewportBottom = viewportTop + jQuery(window).height();

        if(elementBottom > viewportTop && elementTop < viewportBottom) {
            if(!$element.data('offset-displayed')) {

                $element.data('offset-displayed', true);

                const text = $element.find('.product a')[0].innerText;
                const brand_id = $element.data('brand-id');

                jQuery.ajax({
                    type: "POST",
                    url: "<?= admin_url('admin-ajax.php'); ?>",
                    data: {
                        action: 'view_product',
                        product_name: text,
                        brand_id: brand_id
                    },
                    complete: function(jqXHR, textStatus, errorThrown) {
                        const response = JSON.parse(jqXHR.responseText);
                    }
                });
            }
        }
    });
}

if( jQuery('body').hasClass('page-template-archive-product') ){
    jQuery(window).on('scroll', checkViewport);
    checkViewport();
}
});
```

Détection d'un produit vu



```
// Met à jour les post meta dès qu'un produit est vu dans le catalogue
add_action( 'wp_ajax_view_product', 'view_product' );
add_action( 'wp_ajax_nopriv_view_product', 'view_product' );
function view_product() {

    $product_name = $_POST['product_name'];
    $brand_id = $_POST['brand_id'];
    $meta_key = 'catalog_view';

    $meta_value = get_post_meta($brand_id, $meta_key)[0];

    if($meta_value == '' || $meta_value == NULL) {
        $meta_value = array();
    } else {
        $meta_value = unserialize($meta_value);
    }

    date_default_timezone_set('Europe/Paris');
    $index = date("H");
    $meta_value[$index] = $meta_value[$index] + 1;

    update_post_meta($brand_id, $meta_key, serialize($meta_value));

    $response = array(
        'success' => true,
        'response' => $product_name,
        'meta' => $meta_value,
    );

    wp_send_json($response);
    wp_die();
}
```

Utilisation d'AJAX



```
// Update les post meta dès qu'un produit est vu dans le catalogue
add_action( 'wp_ajax_view_product', 'view_product' );
add_action( 'wp_ajax_nopriv_view_product', 'view_product' );
function view_product() {

    $product_name = $_POST['product_name'];
    $brand_id = $_POST['brand_id'];
    $meta_key = 'view_catalog';

    $meta_value = get_post_meta($brand_id, $meta_key)[0];

    if($meta_value == '' || $meta_value == NULL) {
        $meta_value = array();
    } else {
        $meta_value = unserialize($meta_value);
    }

    date_default_timezone_set('Europe/Paris');
    $index = date("H");
    $meta_value[$index] = $meta_value[$index] + 1;

    update_post_meta($brand_id, $meta_key, serialize($meta_value));

    $response = array(
        'success' => true,
        'response' => $product_name,
        'meta' => $meta_value,
    );

    wp_send_json($response);
    wp_die();
}
```

Mise à jour des compteurs de produits vus depuis le catalogue



```
// Permet de récupérer les vues catalogue pour une période donnée
function get_views_catalogs_data($brand_id, $period) {
    $views_catalog = json_decode(get_post_meta($brand_id, 'views_catalog')[0], true);

    $total_views_catalog = array(
        'yesterday' => 0,
        '7_days' => 0,
        '30_days' => 0,
        'this_month' => 0,
        'last_month' => 0,
        'year' => 0,
    );

    $yesterday           = date('Y-m-d', strtotime('-1 day'));
    $last_7_days         = date('Y-m-d', strtotime('-7 days'));
    $last_30_days        = date('Y-m-d', strtotime('-30 days'));
    $start_of_this_month = date('Y-m-01');
    $start_of_last_month = date('Y-m-01', strtotime('last month'));
    $start_of_year        = date('Y-01-01');

    foreach($views_catalog as $date => $views) {

        if($date == $yesterday)
            $total_views_catalog['yesterday'] += $views;

        if(strtotime($date) >= strtotime($last_7_days))
            $total_views_catalog['7_days'] += $views;

        if(strtotime($date) >= strtotime($last_30_days))
            $total_views_catalog['30_days'] += $views;

        if(strtotime($date) >= strtotime($start_of_this_month))
            $total_views_catalog['this_month'] += $views;

        if(strtotime($date) >= strtotime($start_of_last_month) && strtotime($date) < strtotime($start_of_this_month))
            $total_views_catalog['last_month'] += $views;

        if(strtotime($date) >= strtotime($start_of_year))
            $total_views_catalog['year'] += $views;
    }

    return $total_views_catalog[$period];
}
```

Récupération des vues catalogue depuis MA Analytics