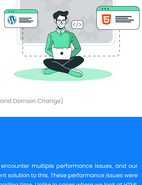
WordPress to Static HTML



Case Study

Industry – **Security**
Year – **2021**
Output – **WordPress to Static** (Migration and Domain Change)

Project Scenario

Sites developed under WordPress usually encounter multiple performance issues, and our team detected the need to find a permanent solution to this. These performance issues were mainly causing hassles in areas like page loading time. Unlike in cases where we look at HTML and Static sites, these issues never really.

There were complications of image optimizations which limited the load time and performance of WordPress sites. Not just that, but there were also instances of security breaches and hacking in the case of WordPress. These issues were of great importance and intensity that they had to be handled with great sensitivity and priority. These required a functional solution that would maintain the ease of developing and maintaining a WordPress site while ensuring an HTML site's performance.

Our Solution

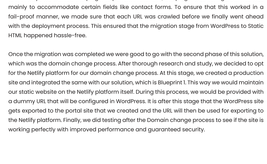
We decided to go with the method of 'Headless WordPress', the benefit of following this system is that WordPress will then serve as the back end to manage the data and content within the website. On the other side, it will be the static HTML setup that will enable to display of the data that is on the front end of the website. We came to the conclusion that this arrangement will effectively improve the security, as well as the scalability aspects in both areas.

The solution we came up with was to maintain a WordPress website for content updations and development and later deploy the same in a Static HTML web format. This was achieved in two phases.

The first phase was Migration from WordPress to Static HTML and the second phase was the Domain Change process. Both of these phases were executed to address performance and security issues. The first phase of WordPress to static conversion was achieved by modifying an existing plugin - static-HTML- output and the second phase was achieved by installing the modified plugin and by configuring our site to a platform called Netlify.

Problem statement & Our inferences

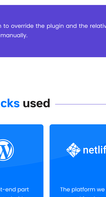
The primary objective of having this solution was based on the need to have a working solution that would effectively meet the convenience of working with a CMS platform like WordPress and at the same time meet the perks of having a static HTML site. This is because HTML enables you to have more control when handling likely scenarios. In the case of content availability also, there are multiple ways like the pre-rendered options and fast CDN options that can be ensured.



This way, the content will be available on the client side within a matter of milliseconds. Not just that, but improved security, reduced maintenance effort and options to scale the application is also much more accessible and comes with different options as well.

When it comes to the performance aspect of a WordPress site, the major drawback comes with loading content and images of varying sizes. In order to maintain the load time and faster performance, one will have to go on optimizing each image that gets uploaded. At the same time, a static website would not have the hassles of getting content and images loaded from the database. This is exactly what we had as a major inference. The same applied to the security part of the website. Therefore it was necessary to have a solution that would not compromise the security of the website at the same time accommodate practicalities that would prevent and protect the site from cyber predators.

The Emvigo Approach



The Agile Approach

All developments and projects under Emvigo Technologies follow an Agile Development Approach. This ensures that the output we come up with is of top-notch quality and at the same time is a fallproof solution. At every stage of development, the output was put through continuous testing to ensure that there are no errors and bugs within the output. After finding the right plugin, it was observed that there are instances where the plugin posted certain limitations and challenges. All of this was resolved through processes of continuous brainstorming and improvements.

The Solutions Approach

Our target population for this solution is majorly corporate sites that are using a limited amount of functionalities. The features that were planned to be incorporated, are to include static pages, an isolated WordPress runtime environment, support for WordPress built-in blogs, a global CDN with Netlify, Yoast SEO support, CF7 support, support for WP Job Openings Plugin, and pre-rendered HTML content served for faster access.

The plugin we decided to use had certain limitations when it came to the execution part. And in order to bridge this, we decided to modify the existing plugin and made certain alterations to it in ways we wanted the plugin to perform. After this, the plugin was configured. This was mainly to accommodate certain fields like contact forms. To ensure that this worked in a fail-proof manner, we made sure that each URL was crawled before we finally went ahead with the deployment process. This ensured that the migration stage from WordPress to Static HTML happened hassle-free.

Once the migration was completed we were good to go with the second phase of this solution, which was the domain change process. After thorough research and study, we decided to opt for the Netlify platform for our domain change process. At this stage, we created a production site and integrated the same with our solution, which is Blueprint 1. This way we would maintain our static website on the Netlify platform itself. During this process, we would be provided with a dummy URL that will be configured in WordPress. It is after this stage that the WordPress site gets exported to the portal site that we created and the URL will then be used for exporting to the Netlify platform. Finally, we did testing after the Domain change process to see if the site is working perfectly with improved performance and guaranteed security.

The Milestones & Challenges we resolved

Challenge 1

Limitations in Contact form submission, content blocks and responses not being fetched rightly after plugin configuration.

Solution

In order to make these options available and to record the responses, we decided to override the plugin and customized it in our desired form.

Challenge 2

Images and Cache settings getting lost

Solution

To fix this issue, we disabled the plugin and made them reflect accurately.

Challenge 3

Failed to detect certain URLs and folders and they were not being visible while crawling the site.

Solution

We went on to override the plugin and the relative was added manually.

Techstacks used

MySQL		netlify
We have used **MySQL** for Database requirements	The Front-end part is executed with **WordPress**	The platform we have used for domain change is **Netlify**

What's next in line?

- **Automated Deployment** options are to be introduced because now every time content updations are made, one will have to manually deploy the changes to be reflected on the live website.
- **Automation of Exporting** process.
- **Search option** to be integrated with WordPress site.