Website Speed Optimization Case Study

Table of Contents

Project Goal 2

Restrictions 2

Initial Analysis 2

TTFB (was 2.54 seconds) 3

Caching Notes: 3

Test: Configure WP Super Cache 3

Test: Successful. New TTFB: 0.3 seconds 3

Homepage Caching 4

File downloads/page rendering (was 15.7 seconds) 5

HTTP KeepAlive 5

Test: Enable .htaccess KeepAlive 5

Test: Unsuccessful 5

Implement KeepAlive in Apache Server Configuration 6

Test: Successful! New download time: 3.2 seconds 6

Other Page Load Improvements 7

Other Issues 7

Summary 7

Project Goal

Reduce page load time on all pages

Restrictions

1. Cannot use memcached (not available)
2. Cannot cache homepage (plugin requirement)
3. Cannot cache when user logged in (WooCommerce)
4. Cannot cache user area (WooCommerce: /my-account/*)
5. Cannot remove further plugins (plugin audit already complete)

I don't have direct root access, but I can request config changes from the web designer.

Initial Analysis

Using **https://www.example.com/product/example-product/** as the example URL

Load Time	First Byte	Start Render	Visually Complete	Speed Index	DOM Elements	Result (error code)
20.274s	2.548s	18.273s	19.800s	18848	1250	0

RUM First Paint	domContentLoaded	l
19.345s	18.547s - 18.916s (0.369s)	20.282s -

- TTFB: 2.548 seconds
- Start render: 18.273 seconds
- Fully loaded: 19.800 seconds

The aim here is to reduce start render time significantly, try to get to 4 seconds.

TTFB (was 2.54 seconds)

Database calls are the primary reason for slow Time To First Byte in WordPress. Most plugins add several database calls to the page load, so TTFB grows significantly.

1. The first solution to TTFB is to remove plugins. This is a call on whether you want a faster website, or the functionality that the plugins are providing. (Already completed by Web designer& Tim)
2. The second solution to TTFB is to implement caching to remove the need for the server to call the database.

With good caching in place, we should be able to chop an entire 2 seconds from the current ~2.5s.

Caching Notes:

1. WP Super Cache is installed and activated, but caching is currently disabled.
2. Web designernotes in email that homepage can't be cached due to a plugin.
3. WooCommerce requires that the whole /my-account/ section not be cached.
4. WooCommerce requires that logged in users pages not be cached.

Test: Configure WP Super Cache

I've enabled WP Super Cache caching and added a rule to not cache the homepage.

You're running WooCommerce v2.4 - back in v1.4.2 they added a DONOTCACHEPAGE constant on pages that shouldn't be cached so we shouldn't need any special rules for this.

I've set cache timeout to 7200 seconds (2 hours). This means pages will be rebuilt every 2 hours. Given the significantly slow page loads without caching my preference is to serve the cached pages to as many visitors as possible. You could set this to a lower timeout like3600 (1 hour) or if you're happy with the caching performance, eventually bump it up to a larger one expiration timeout like 43200 (12 hours).

Test: Successful. New TTFB: 0.3 seconds

Now seeing TTFB of 0.3s consistently with a Start Render of around 6.5s.

Load Time	First Byte	Start Render	Visually Complete	Speed Index	DOM Elements	Result (error code)
8.367s	0.334s	6.664s	8.100s	7302	1250	0

RUM First Paint	domContentLoaded	loa
7.487s	6.593s - 7.299s (0.706s)	8.357s - 8

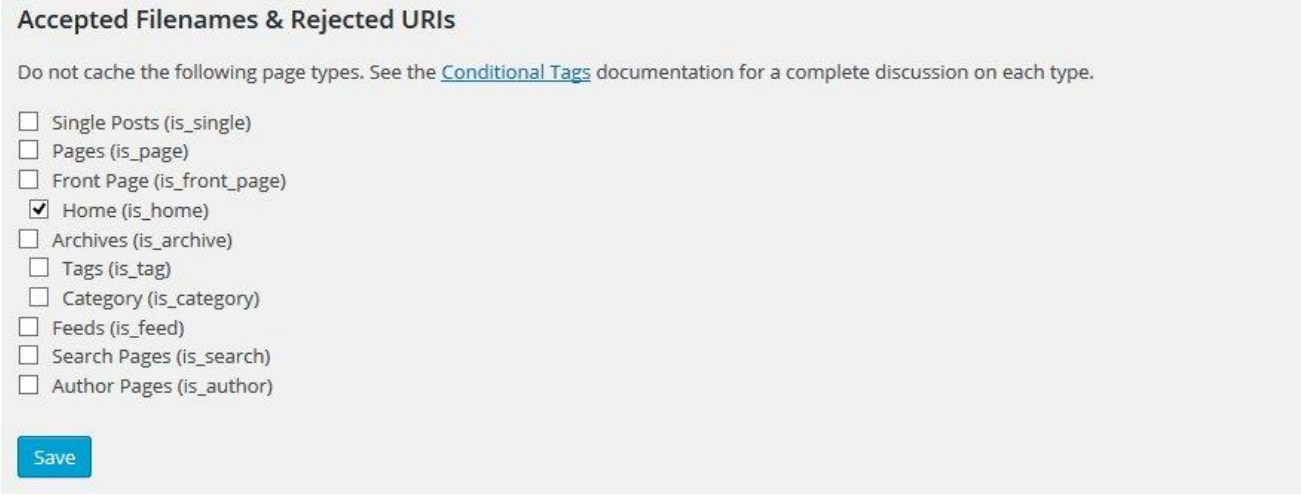
Homepage Caching

Currently this is specifically disabled as you mentioned incompatibility with the Daily Deal widget.

Page load time: starts render in 17.9 seconds and fully loaded in 27 seconds.

						Document Complete			Fully Loaded			
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View	26.999s	2.487s	17.984s	20870	1782	26.999s	174	4,718 KB	26.999s	174	4,892 KB	$$$$$
Repeat View	15.823s	2.329s	7.089s	10086	1782	15.823s	136	4,301 KB	16.439s	140	4,351 KB	

However, I've been looking at the Javascript timer for the Deal widget and it is based on a specific clock time, so I'd suggest that this page could actually be caching enabled. It can be enabled by unticking this and hitting Save:



Home page load with caching enabled has start render of 5.0s and fully loaded in 12.7s.

						Document Complete			Fully Loaded			
	Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In	Cost
First View	11.303s	2.462s	5.077s	8982	1782	11.303s	153	3,916 KB	12.727s	174	5,045 KB	$$$$$
Repeat View	12.483s	2.906s	5.441s	7795	1782	12.483s	135	4,083 KB	13.242s	140	4,573 KB	

I've set home page not to cache for the moment as per your instructions.

Note: a real "gotcha" with WP Super Cache is if you change options in different sections (even down the page) you must hit the specific save button under that one section to have them take effect.

File downloads/page rendering (was 15.7 seconds)

- There are 32 CSS files loading, 41 Javascript files, and ~45 other files (images, fonts, etc).
- The majority of CSS and Javascript is coming from plugins.
- You've already reduced the plugin numbers as much as possible so let's see what else is possible.

HTTP KeepAlive

It looks like HTTP KeepAlive is not on, so this should give us the biggest win in the downloads/page rendering section.

KeepAlive uses RAM and saves CPU, so if RAM is very limited you may want to keep it off. However, from the site profile, I think you'll get a lot of benefit from it.

Test: Enable .htaccess KeepAlive

To turn on HTTP KeepAlive I've added the following code to the site root .htaccess file:

```
<IfModule mod_headers.c>
Header set Connection keep-alive
</IfModule>
```

That should help reduce the 18.2s by 50% or so. Let's implement that in .htaccess:

```
username@CompanyName.com [~/public_html]# grep alive ./.htaccess
username @CompanyName.com [~/public_html]# vi ./.htaccess
username @CompanyName.com [~/public_html]# grep alive ./.htaccess
Header set Connection keep-alive
username @CompanyName.com [~/public_html]#
```

Test: Unsuccessful

Despite the .htaccess rule addition, KeepAlive is still not enabled. Note: GTMetrix and some other tools will show it as enabled, but it's set to "closed".

It may be that there is a line like "BrowserMatch <Browser> nokeepalive" which is preventing it in certain circumstances, or the Apache server config is set to "keep-alive, close". That's the next step here.

Implement KeepAlive in Apache Server Configuration

Since the .htaccess level command isn't working, we need to implement at the Apache server level. Here's the commands we need to add (or uncomment) in the config file:

KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 100

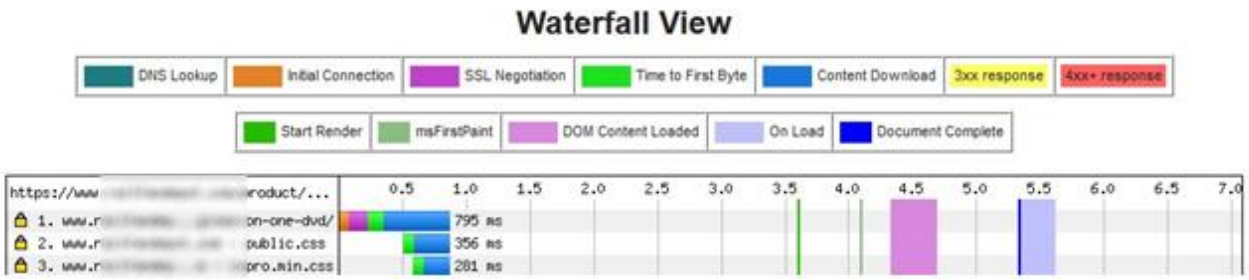
Config probable location: "**/usr/local/apache/conf/httpd.conf**"

Restart command: **service apache2 restart**

Web designerhas implemented this, so here are the results of re-testing now:

							Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	Visually Complete	Speed Index	DOM Elements	Result (error code)	Time	Requests	Bytes In	Time	Requests	Bytes In
5.334s	0.340s	3.602s	5.000s	3976	1250	0	5.334s	115	1,232 KB	7.072s	136	1,588 KB

RUM First Paint	domContentLoaded	loadEvent
4.087s	4.328s - 4.681s (0.353s)	5.342s - 5.618s (0.276s)



Test: Successful! New download time: 3.2 seconds

Big improvements achieved with the KeepAlive setting as it makes download of all of the disparate page elements much more streamlined into less HTTPs connections.

Previous file download time: 15.7 seconds

New file download time: 3.2 seconds

Other Page Load Improvements

There's very little low-hanging fruit here, apart from what's documented here, most other techniques are either already done or unavailable due to project restrictions.

Other Issues

You're probably well aware of these but I have to note them just in case.

1. Home page title is "Home - Sitename", needs work for SEO.
2. Home page is nearly 5MB in total page size. Apart from speed issues, this may be costly or unusable for visitors on mobile connections.
3. Site robots.txt currently turning away indexers ("allow search engines..." setting in WordPress Reading Settings).
4. There are still several import, export, backup and maintenance type plugins enabled. Quite often these can be disabled when not in immediate use.
5. Once the build is complete and site is live, delete all inactive plugins for security reasons.
6. Regarding the 2-4GB memory increase slowing page load, I think that was because something was timing out previously due to lack of memory (maybe an AJAX call), and with more RAM available it then tries to fulfil the request and still is failing but hanging.

Summary

We've got page load time for the example product page down significantly with these two changes:

• Original TTFB: 2.548 seconds • Original Start render: 18.273 seconds • Original Fully loaded: 19.800 seconds	• **New TTFB: 0.340 seconds** • **New Start render: 3.602 seconds** • **New Fully loaded: 7.072 seconds**

The most important improvement is to Start render times as this is when the visitor starts to see the page being drawn on screen and reduces risk of them leaving the site.

Plugins are the primary issue with site load. Due to the amount of database calls that the plugins require, caching is absolutely essential for this site. The KeepAlive setting ensures that all of the plugin necessitated external files are downloaded to the visitor more quickly than opening a new HTTPs connection each time.