

WordPress analysis

Frontend performance optimization

This is a Google Slides port
from the original document

Please excuse the typos due to incompatible
PDF engines in graphic software



%Client%

Frontend Performance Review

2019/08/13 – 1.0.0



Table of Contents

1. [About the Review](#)
2. [Onpage Performance Review](#)
3. [**Bottlenecks & Challenges**](#)
4. [**Recommendations**](#)
5. [Further Considerations](#)
6. [Terms & Definitions](#)
7. [Resources & Links](#)
8. [Disclaimer](#)

49 slides

1.About the Review



Testing methods & Results focus

How & What we tested

How?

- Combine 12 analytics tools
- Test Desktop & Mobile homepage

What?

- Focus on frontend performance (less on server or backend)
- Check basic HTML structures (SEO)



Preferences & Prioritisation

Evaluating solutions

Client preferences

1. Fast realisation of results
2. Low implementation costs
3. Low recurring maintenance cost
4. (Minimal change to website layout or content needed)

Prioritisation

- Simple & high impact solutions first
- No costly and time intense manual code changes or refactoring
- Custom optimisations only if needed



Scores & Goals

Setting realistic goals &
Measuring results

About scores

- Don't aim for a 100% score just for the perfect number
- Don't correlate a faster page with a higher score^{*2 *3}

Goals

- Set viable speed improvements from the current state; e.g. 25–50% faster initial page load on all devices
- Improve gradually from there, given effort, cost and result are good fits



Reasoning

Prioritisation explained

What to optimise?

- **Frontend:** Offers the largest leverage to optimise big way
- **Backend:** Optimise WordPress + Plugins first (before customising any other code parts or configuration)
- **Server:** Hosting & Server – A shared server offers less speed and options for configuration

Optimisation is a continuous, iterative process.
Track ->Evaluate ->Optimise ->Repeat ...

2. Onpage Performance Review



Facts & Numbers

Average test results

Based on Google PageSpeed &
Google Lighthouse tests

Current status – Homepage

- **6–19 seconds** until content is visible
- **~55seconds** until page is useable
- **17 MB** total page size with images
- **164+files** loaded at once
- **179+total** requests sent
- **17 unnecessary 301** redirects

Exemplaric result. Values may vary depending on time, location, testing methodologies and cache settings.

> 50% visitors bounce if
page loads over 3 seconds

Google benchmark / Mobile user behaviour*¹



...but is it true?

Don't trust all statistics

How to find out

- Implement own tracking solution
- Set custom goals to measure
- Find out how your customers 'tick' (analytics, campaigns, surveys, ...)

Considerations

- Statistics might be too narrow, old or just not fit your users & industry
- Different user behaviour / Stronger internet & faster devices
- Really want the product no matter what (=willing to wait longer for it)



Why optimise?

General benefits

Faster websites...

- Can increase the CR % (your goals)
- Rise user satisfaction (likely more product & brand engagement)
- Impact SEO positively (performance included in ranking algorithms)

Long-term...

- Reduced costs of visiting for users (mobile data), maintenance & editing (staff) and hosting (server, CDN)



Goals

Overarching definitions

Thoughts & Considerations

- What is the desired outcome? (e.g. increased page load speed by x%; reduced file size from x to y MB ...)
- What are we willing to change or `sacrifice` for these goals? (e.g. changing layout; images, animations, ...)

A good start ...

1. Achieve 25% faster pageload.
2. Can we optimise or reduce more?
Go further and aim for $\geq 50\%$.



Our goals – 1/2

Current optimisation goals

1st Stage – Basics

1. Minify images(JPG, PNG, GIF, ...)
2. Minify +Combine source (CSS, JS)
3. Use caching(CSS, JS, HTML)
4. Implement image lazy loading

2nd Stage – Additional

1. Reduce & Remove content parts
 - a. Split content into sub pages
2. Choose new file formats
 - a. Images =WebP?
 - b. GIF =MPEG4 Video?



Our goals – 2/2

Advanced possible optimisations

Tackle if needed

3rd Stage – Future

1. Choose performance-oriented & minimal WordPress theme
 - a. Less complex HTML + faster CSS
 - b. Requires content + layout changes
2. Change hosting solution to dedicated instead shared server (VPS?)
 - a. Better hardware (RAM, CPU, SSD)
 - b. Customisable config (PHP, MySQL)
 - c. Less load peak times and outages
3. Utilise commercial CDN or caching solutions (e.g. Cloudflare)
4. Get custom tailored plugins? (e.g. without jQuery)

3. Bottlenecks & Challenges



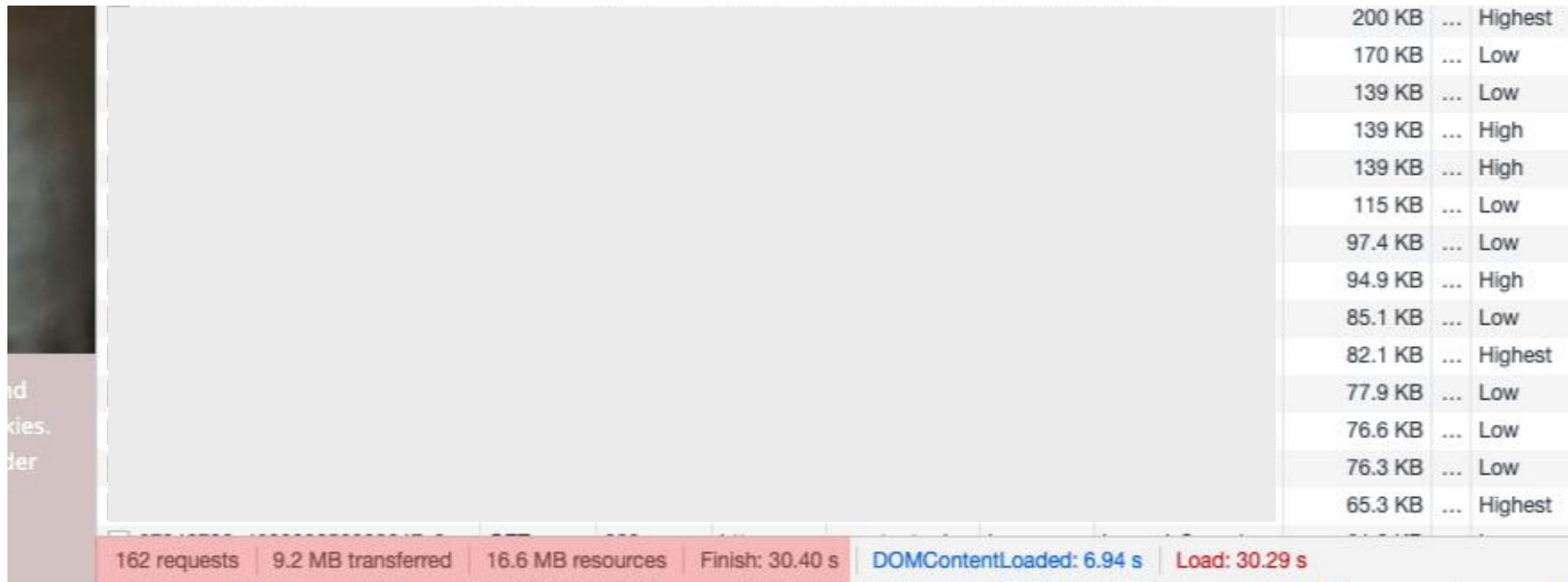
Top challenges

Problems & Solutions

Top 6

1. Oversized +too many image files
2. No image 'lazy loading' enabled
 - a. All files load at once even if invisible
 - b. Too much assets loaded on 1st page
3. Scattered JS/CSS blocks rendering
 - a. Initial page appears partly empty for a while until the main image loads
4. Too many plugins +Scripts
 - a. e.g. YouTube player
5. No file compression(GZIP) enabled
6. Unnecessary 30x/http redirects
7. Partly slow server (lower priority)

1+2. Too many initial requests + Large file sizes



The screenshot displays a network tab in a browser's developer tools. The main area shows a list of 13 requests, each with a size and a priority level. The sizes range from 65.3 KB to 200 KB. The summary bar at the bottom indicates 162 requests, 9.2 MB transferred, 16.6 MB resources, a finish time of 30.40 s, a DOMContentLoaded time of 6.94 s, and a total load time of 30.29 s.

200 KB	...	Highest
170 KB	...	Low
139 KB	...	Low
139 KB	...	High
139 KB	...	High
115 KB	...	Low
97.4 KB	...	Low
94.9 KB	...	High
85.1 KB	...	Low
82.1 KB	...	Highest
77.9 KB	...	Low
76.6 KB	...	Low
76.3 KB	...	Low
65.3 KB	...	Highest

162 requests | 9.2 MB transferred | 16.6 MB resources | Finish: 30.40 s | DOMContentLoaded: 6.94 s | Load: 30.29 s

1.1.GIF too large – Remove/Change/lazyload

▲ Defer offscreen images



Consider lazy-loading offscreen and hidden images after all critical resources have finished loading to lower time to interactive. [Learn more.](#)



Install a [lazy-load WordPress plugin](#) that provides the ability to defer any offscreen images, or switch to a theme that provides that functionality. Also consider using [the AMP plugin](#).

URL	Size	Potential Savings
	1,929 KB	1,929 KB

3.1. Inline JS within HTML (plugins/theme)

```
3666     </div>
3667 </div>
3668 <div id="photoblocks-fancybox-2"></div>
3669 <script>
3670     var $ = jQuery;
3671     $(document).ready(function() {
3672         var $ = jQuery;
3673         $(document).ready(function() {
3674             jQuery(function() {
3675                 var p = new PhotoBlocks({
3676                     selector: "#photoblocks-2",
3677                     columns: 1,
3678                     padding: 10,
3679                     resizer: '
3680                     image_quality: 80,
3681                     disable_below: 320,
3682                     imageFactor: 1.5,
3683                     on: {
3684                         before: function() {},
3685                         after: function() {},
3686                         refresh: function() {}
3687                     },
3688                     mobile_layout: [],
3689                     lazy: false,
3690                     debug: false
```

HTML

JS

3.2. Inline CSS within HTML (plugins/theme)

CSS

HTML

```
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854

#photoblocks-4 .pb-block.pb-type-text .pb-description {
}

#photoblocks-4 .pb-overlay {
  background: rgba(0, 0, 0, 0.75);
}

#photoblocks-4.pb-lift.show-empty-overlay .pb-block.pb-type-image:hover, #photoblock
  box-shadow: rgba(0, 0, 0, 0.75) 0 0 20px;
}

#photoblocks-4 .pb-block {
}

#photoblocks-4 .pb-block .pb-social button {
  color: #ffffff;
  font-size: 14px;
}
</style>
<div class="photoblocks-gallery pb-blur pb-lift pb-effect-fade" data-anim="fade" id="pho
  <div class="pb-blocks">
    <div class="pb-block pb-landscape pb-width pb-type-image" style="" data-colspan:
      
11333   <div class="cookie-notice-container">
11334     <span id="cn-notice-text">Um unsere Webseite für dich optimal zu gestalten und fortlaufend verbessern zu können, ver
11335     <a href="#" id="cn-accept-cookie" data-cookie-set="accept" class="cn-set-cook c-button wp-default button">Verstar
11336     <a href="#" target="_blank" id="cn-more-info" class="cn-more-info cn-button wp-de
11337   </div>
11338 </div>
11339 <script type="text/javascript" defer src="
11340 <script data-cfasync='false'>
11341   !function(t) {
11342     "use strict";
11343     t.loadCSS || (t.loadCSS = function() {}
11344   );
11345     var e = loadCSS.relpreload = {};
11346     if (e.support = function() {
11347       var e;
11348       try {
11349         e = t.document.createElement("link").relList.supports("preload")
11350       } catch (t) {
11351         e = !1
11352       }
11353     }
```

JS

HTML

JS

>11000 code lines

The page is too large

3.4. Theme issue – Exzessive HTML size + Nesting

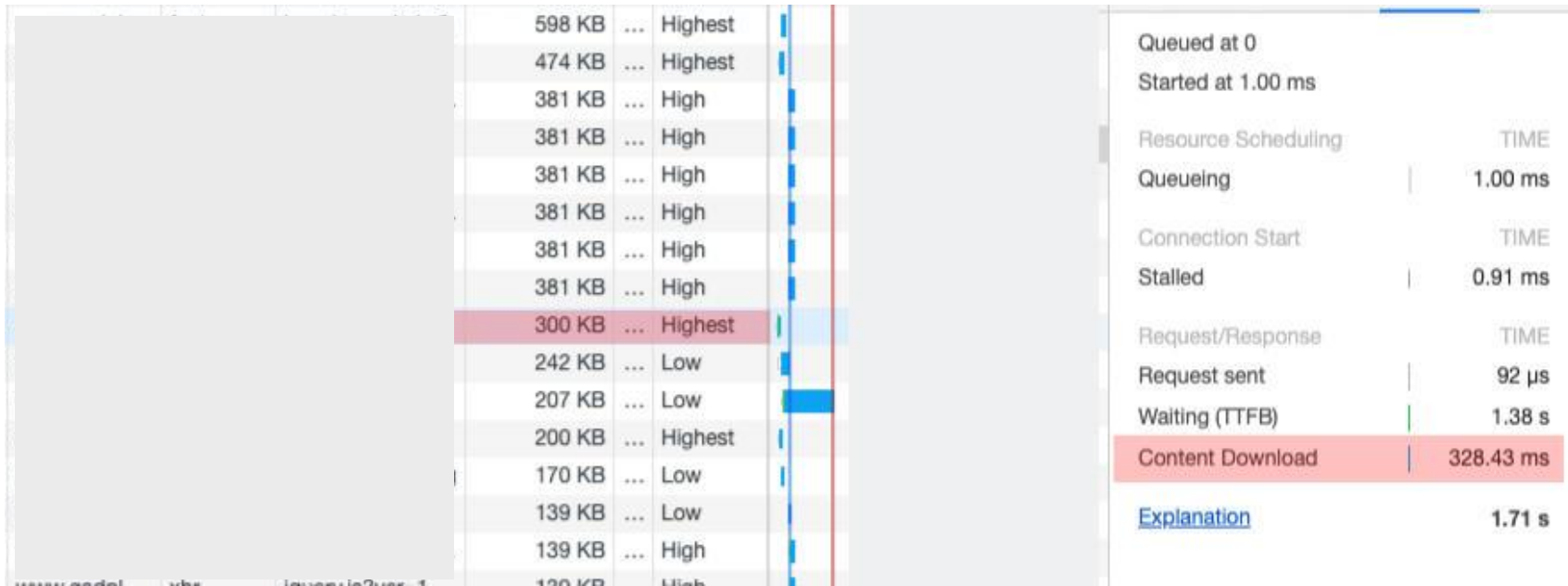
```
<div>
  <div>
    <div>
      <div>
        <div style="font-size: 55px;" align="center">
          <div style="font-size: 55px;">
            <div style="font-size: 55px;">
              <div style="font-size: 55px;">
                <div style="font-size: 55px;">
                  <div style="font-size: 55p
                    <div style="font-size:
                      <div style="font-s
                        <div style="fo
                          <div style="fo
```

```
<div>
  <div style="font-size: 22px; color: rgb(255, 255, 255);">
    <div style="font-align: center; color: rgb(255, 255, 255); font-size: 22
      <span style="" class="">
        <span style="font-size: 22px; line-height: 1.5em;" class="keep-C
          <span data-font-weight="400" style="font-style: normal; font
            <span data-font-weight="400" style="font-style: normal; font
              <span style="color: #252525">
                <span style="" class="">
                  Speedelfie Photobooth Aktion, <br>
                </span>
              </span>
            </span>
          </span>
        </span>
      </span>
    </span>
  </div>
  <div style="font-align: center; color: rgb(255, 255, 255); font-size: 24
    <span style="font-size: 18px;" class="">
      <span style="font-size: 18px; line-height: 1.5em;" class="keep-C
        <span data-font-weight="400" style="font-style: normal; font
          <span data-font-weight="400" style="font-style: normal; font
            <span style="font-size: 18px;" class="">
              <span style="color: rgb(137, 37, 37); font-si
                <span style="" class="">
```

▲ Avoid an excessive DOM size – 4,079 elements

Browser engineers recommend pages contain fewer than ~1,500 DOM elements. The sweet spot is a tree depth < 32 elements and fewer than 60 children/parent element. A large DOM can increase memory usage, cause longer [style calculations](#), and produce costly [layout reflows](#).

3.5. Page document too large (300kbincl. code)



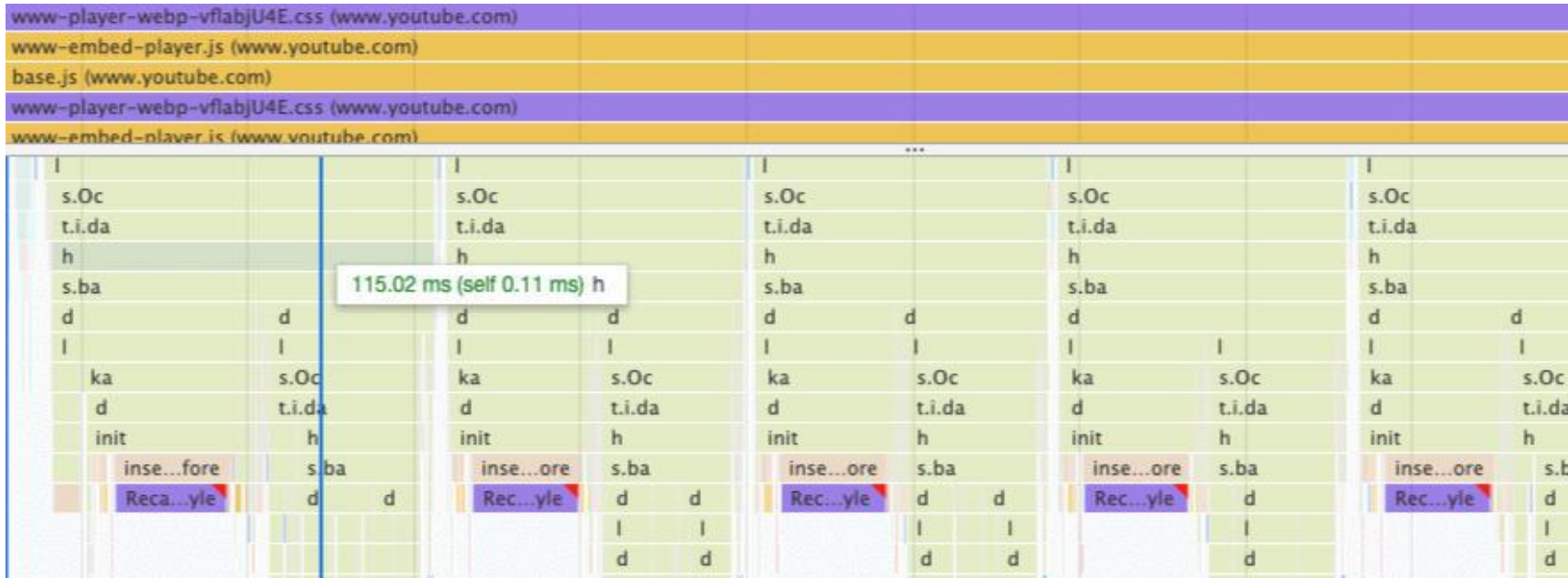


4.1. YouTube plugins – Slow load + Rendering

Consider reducing the time spent parsing, compiling, and executing JS. You may find delivering smaller JS payloads helps with this. [Learn more](#).

URL	Total CPU Time	Script Evaluation	Script Parse
Other	6,022 ms	368 ms	18 ms
...www-embed.../www-embed-player.js (www.youtube.com)	5,539 ms	4,904 ms	100 ms
...jquery/jquery.js?ver=1.12.4-wp (www.gadplan.com)	4,067 ms	2,529 ms	23 ms
...en_US/base.js (www.youtube.com)	1,274 ms	576 ms	698 ms

4.2. YouTube- Loaded multiple times +Slow



5. Enable GZIP – Small text files (HTML, CSS, JS, ...)

Check GZIP compression

Oh No! GZIP is not enabled :(

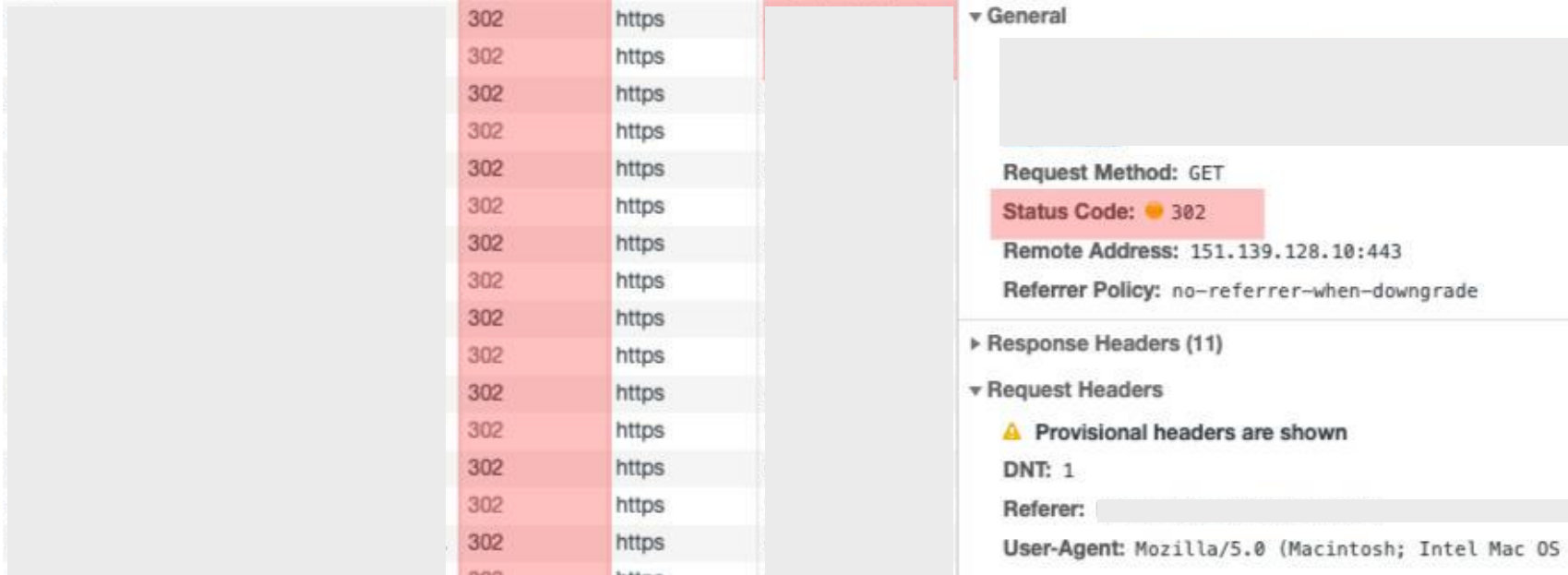
Domain Name	Gadplan.com
Compressed size	33346 bytes (~33 KB)
Uncompressed size	305784 bytes (~299 KB)



Could be saved by
compressing this page with
GZIP.



6.1. Fix 301/302 redirects from CDN



The screenshot displays a network log on the left with 15 entries, each showing a 302 status code and a 'https' protocol. The right pane shows the details for one of these requests:

- General**
 - Request Method: GET
 - Status Code: 302
 - Remote Address: 151.139.128.10:443
 - Referrer Policy: no-referrer-when-downgrade
- Response Headers (11)**
- Request Headers**
 - ⚠ Provisional headers are shown
 - DNT: 1
 - Referer: [Redacted]
 - User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS

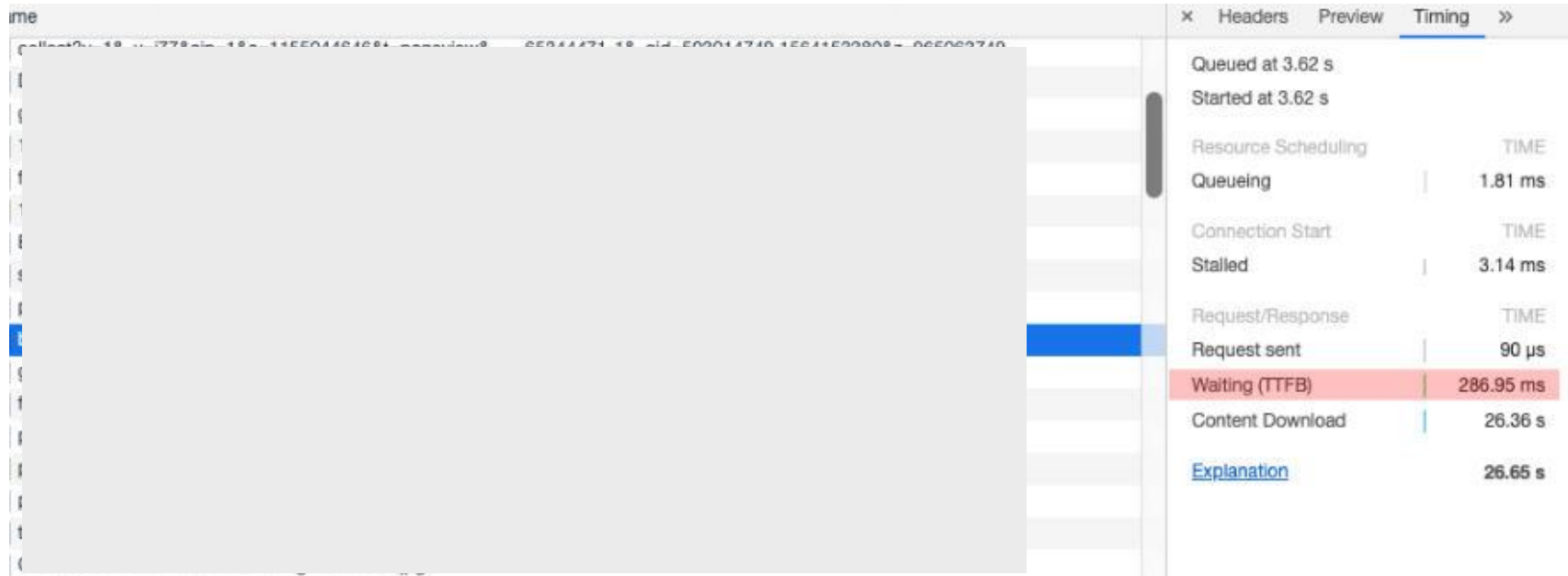
6.2. Ineffective CDN use – Redirects, Few files

GET	302	https	cdn.shortpixel.ai	text/html	lazysizes.min.js:2	126 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	155 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	155 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	164 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	179 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	179 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	188 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	203 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	204 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	204 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	209 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	210 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	217 B	...
GET	302	https	cdn.shortpixel.ai	text/html	(index)	220 B	...
GET	302	https	cdn.shortpixel.ai	text/html	jquery.js?ver=1.1...	228 B	...

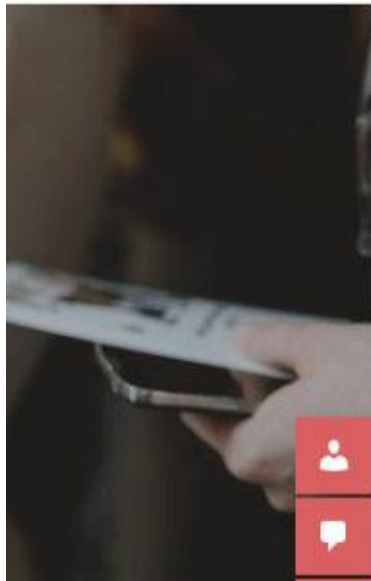
6.3. Fix http to https redirects – Speed + Security

```
 Group similar
⚠ Mixed Content: The page at '!' was loaded over HTTPS, but requested an insecure im
dolan_300px.jpg'. This content should also be served over HTTPS.
✖ ▶ Uncaught ReferenceError: responsive_functions is not defined
  at (index):1768
JQMIGRATE: Migrate is installed, version 1.4.1
3 !
3 !
3 !
3 !
3 !
3 !
⚠ Mixed Content: The page at '!' was loaded over HTTPS, but requested an insecure im
dolan_300px.jpg'. This content should also be served over HTTPS.
✖ ▶ Failed to execute 'postMessage' on 'DOMWindow': The target origin provided ('https://www.youtube.com') does not match the recipi
('!').
>
```


7.1.Slow server – Long delay before page load



7.2.Server – Utilise HTTP/2 protocol for assets



Best Practices

57

- | | | | | |
|---|---|-----------------------------------|---|---|
| 1 | Does not use HTTPS | 2 insecure requests found | ▲ | ▼ |
| 2 | Does not use HTTP/2 for all of its resources | 67 requests not served via HTTP/2 | ▲ | ▼ |
| 3 | Does not use passive listeners to improve scrolling performance | | ▲ | ▼ |
| 4 | Links to cross-origin destinations are unsafe | | ▲ | ▼ |
| 5 | Includes front-end JavaScript libraries with known security vulnerabilities | 2 vulnerabilities detected | ▲ | ▼ |
| 6 | Browser errors were logged to the console | | ▲ | ▼ |

✓ Passed audits

9 audits ▼

Runtime settings

8. Too many font variants (unused alphabets)

```
font-weight: wub;
src: local('Poppins Regular'), local('Poppins-Regular'), url{https://fonts.gstatic.com/s/poppins/v8/pxiEyp8kv8JHgFVrJJbecnFHGPezSQ.woff2} format('woff2');
unicode-range: U+0900-097F, U+1CD0-1CF6, U+1CF8-1CF9, U+200C-200D, U+20A8, U+20B9, U+25CC, U+A830-A839, U+A8E0-A8FB;
}
/* latin-ext */
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 400;
  src: local('Poppins Regular'), local('Poppins-Regular'), url{https://fonts.gstatic.com/s/poppins/v8/pxiEyp8kv8JHgFVrJJbecnFHGPezSQ.woff2} format('woff2');
  unicode-range: U+0100-024F, U+0259, U+1E00-1EFF, U+2020, U+20A0-20AB, U+20AD-20CF, U+2113, U+2C60-2C7F, U+A720-A7FF;
}
/* latin */
@font-face {
  font-family: 'Poppins';
  font-style: normal;
  font-weight: 400;
  src: local('Poppins Regular'), local('Poppins-Regular'), url{https://fonts.gstatic.com/s/poppins/v8/pxiEyp8kv8JHgFVrJJfecnFHGPc.woff2} format('woff2');
  unicode-range: U+0000-00FF, U+0131, U+0152-0153, U+02BB-02BC, U+02C6, U+02DA, U+02DC, U+2000-206F, U+2074, U+20AC, U+2122, U+2191, U+2193, U+2212, U+2215, U+FEFF, U+FFFD;
}
/* cyrillic-ext */
@font-face {
  font-family: 'Roboto';
  font-style: italic;
  font-weight: 100;
  src: local('Roboto Thin Italic'), local('Roboto-ThinItalic'), url{https://fonts.gstatic.com/s/roboto/v20/KFOiCngBu92Fr1Mu51QrEz0dL-vwnYh2eg.woff2} format('woff2');
  unicode-range: U+0460-052F, U+1C80-1C88, U+20B4, U+2D80-2DFF, U+A640-A69F, U+FE2E-FE2F;
}
/* cyrillic */
@font-face {
  font-family: 'Roboto';
  font-style: italic;
  font-weight: 100;
  src: local('Roboto Thin Italic'), local('Roboto-ThinItalic'), url{https://fonts.gstatic.com/s/roboto/v20/KFOiCngBu92Fr1Mu51QrEzQdL-vwnYh2eg.woff2} format('woff2');
  unicode-range: U+0400-045F, U+0490-0491, U+04B0-04B1, U+2116;
}
```

9.1.WP Autooptimize – Emojis, Preconnect, YouTube

Portfolio
Waldere Formulare
Design
Plugins 7
Benutzer
Werkzeuge
WP Bakery Page
Builder
WP BackItUp
Einstellungen
nein
ben

Emojis entfernen

Abfragezeichenfolgen von statischen Ressourcen entfernen

Zu Dritt-Domains vorverbinden (Fortgeschrittene Benutzer)

Asynchrone JavaScript-Dateien (Fortgeschrittene Benutzer)

Optimiere YouTube-Videos

Entfernt WordPress' internes Emoji-Inline-CSS, Inline JavaScript und eine weitere nicht optimierte JavaScript

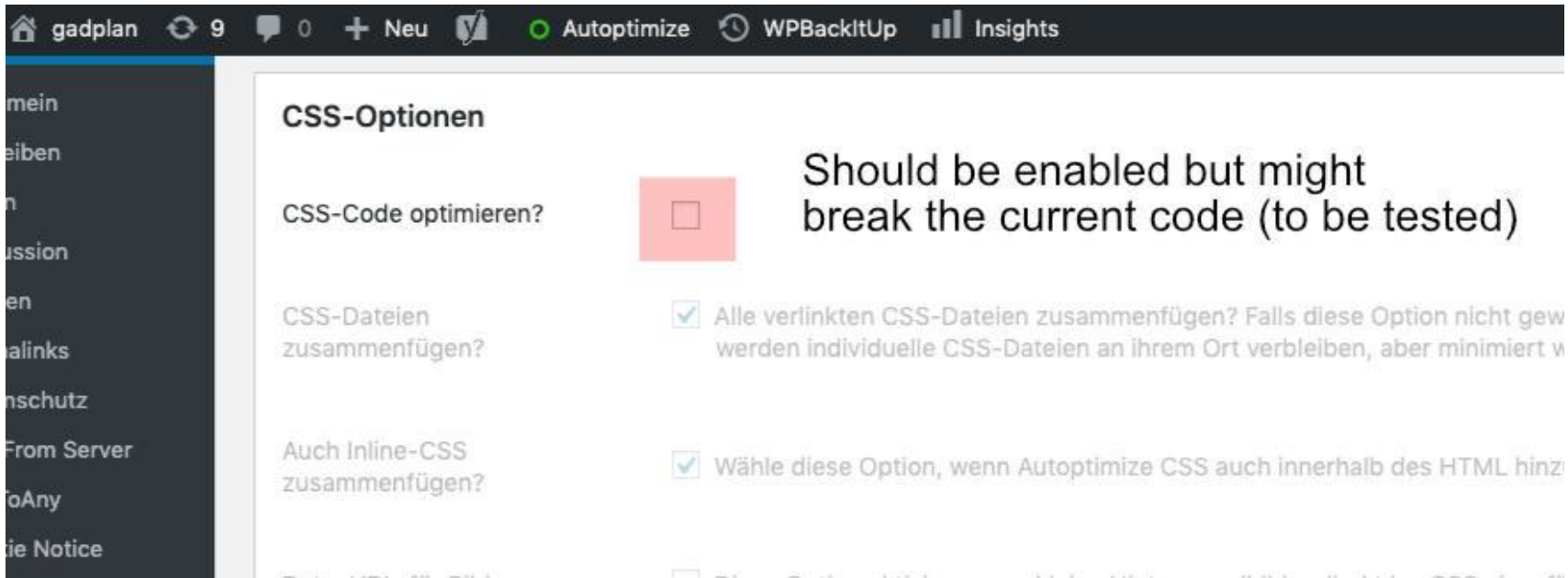
Das Entfernen von Abfrage-Zeichenfolgen (oder genauer gesagt, des `ver`-Parameters) wird die Ladegesc aber Leistungsmessungen verbessern.

Füge Drittanbieter-Domains hinzu, zu denen sich der Browser [vorverbinden](#) soll, durch Komma getrennt. Stelle anzu geben (HTTP oder HTTPS).

Kommagetrennte Liste lokaler oder Drittanbieter-JS-Dateien, die per `async` geladen werden sollen. JS-Datei automatisch ausgeschlossen, falls hier aufgeführt. Die Konfiguration von asynchronem JavaScript ist einfacher [JavaScript](#)". [Hier klicken, um es zu installieren und zu aktivieren.](#)

[WP YouTube Lyte](#) ermöglicht dir, deine Videos durch das Einfügen von responsiven „Lite YouTube Embeds“ ver [installieren und zu aktivieren.](#)

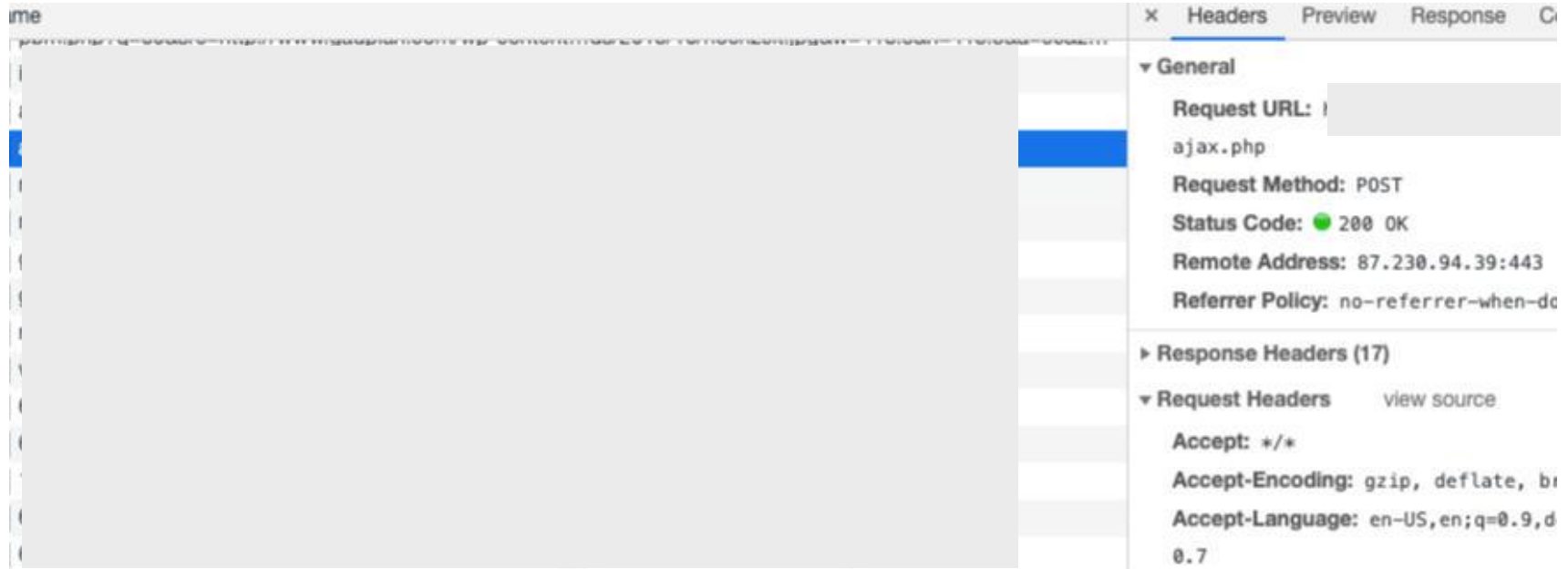
9.2. WordPress Autoptimize – Optimise CSS?



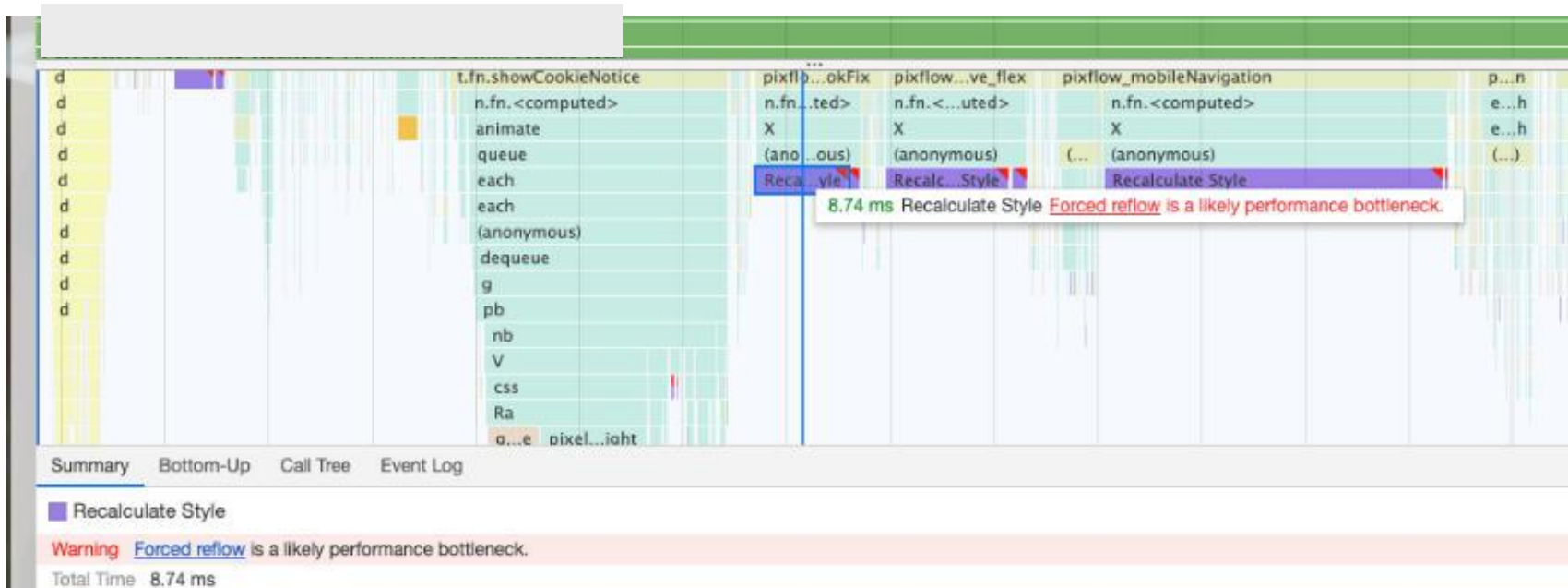
The screenshot shows the WordPress dashboard with the Autoptimize plugin settings for CSS. The top navigation bar includes 'gadplan', '9', '0', '+ Neu', 'Autoptimize', 'WPBackItUp', and 'Insights'. The left sidebar contains menu items like 'mein', 'eiben', 'n', 'ussion', 'en', 'alinks', 'nschutz', 'From Server', 'ToAny', and 'ie Notice'. The main content area is titled 'CSS-Optionen' and contains the following settings:

- CSS-Code optimieren?** Should be enabled but might break the current code (to be tested)
- CSS-Dateien zusammenfügen?** Alle verlinkten CSS-Dateien zusammenfügen? Falls diese Option nicht gew werden individuelle CSS-Dateien an ihrem Ort verbleiben, aber minimiert v
- Auch Inline-CSS zusammenfügen?** Wähle diese Option, wenn Autoptimize CSS auch innerhalb des HTML hinz

10. If possible: Remove 'admin-ajax' calls



11.Style reflows – jQuery (pixflow navigation)



12. UI animation feels slow/laggy (speed up)



4. Recommendations



Solutions – 1/3

Possible solutions

1st priority

High impact and benefits

Varying implementation cost

Problems 1 – 3

- Reduce image files and amount (might result in needed layout changes)
 - Replace some images by text
- Use dedicated WordPress plugin(s) to minify/bundle assets (CSS, JS)
 - e.g. W3 Total Cache
 - Requires extensive browser + functionality testing afterwards
- Implement image lazy load solution
 - Might not work with all themes
 - Requires extensive layout + functionality testing afterwards
 - Don't lazy load first/header image that users will see on page entry



Solutions – 2 / 3

Possible solutions

2nd priority

Less impact, still important

Problems 4 – 6

- Remove or replace animated GIF by Video or static image (<1.9 MB)
- Change all URLs from 'http://' to 'https://' to remove 301 redirects
 - The http-URLs seem to come from a theme / custom CSS editor (plugin?).
 - Fix cdn.shortpixel.ai usage
- Hack: Remove admin_ajax calls from frontend (if possible)^{*10}
 - It can reduce requests and speed up the page a little bit (low impact)
 - It might be needed though, depending on functionality or theme



Solutions – 3 / 3

Possible solutions

3rd priority

High impact and benefits; but also high effort & realisation cost

Problems 4 – 6

Important / High effort & realisation cost

- Change theme to a more performant
 - Drawback: Content + Features will have to change, refactoring needed (plugins, 3rd party integrations)
 - The code is slow and cluttered
 - See chapter [Resources & Links](#)
- Upgrade or move server hosting
 - Slows connection (TTFB^{*8}) delays each file download (CSS, JS, images)
 - High effort & higher maintenance costs; Developer might be needed

5. Further Considerations

Advanced techniques

Additional tips / No in-depth description / Developer needed





WordPress setup

Themes & Optimisations

- Disable unneeded plugins. Check and revise app plugin settings, e.g. 'Heartbeat' or DB/Object caches. ^{*9}
- If choosing a new theme; consider focus on performance first, then features (and prefer to use built in).
 - Choose a theme without jQuery (these are usually slower during load and runtime). Note: It might not be compatible with other widely used plugins!
- Fight spam: Disable Pingbacks / Trackbacks / Reblog feature (if enabled). ^{*4}



Database config

Architecture & Optimisation

- Use a plugin to regularly cleanup and optimise the DB: Remove orphaned data, stored revisions, reduce used disk space, improve memory usage.*⁶
- Change tables to InnoDB instead MyISAM to benefit of different memory and cache settings.*⁶
 - Meaning: Older tables or those created by plugins, if compatible.
 - These settings need to be individually defined for your server environment!



PHP +Apache

Server settings


- Check server +WordPress config: Use PHP 7+ instead of PHP 5.
- Set PHP `memory_limit` to a higher value if possible (e.g. 128/256MB).
 - Might be possible in `.htaccess` (likely not on small/shared host packages)
- Depending on editors needs:
 - Increase upload file limits
 - Increase execution times



Image hotlinking

Prevent & Fight back

- **htaccess:** Prevent hotlinking of images by server config.^{*5}
 - Other websites that embed your content steal your bandwidth and reduce server speed.
 - It helps to prevent content stealing or duplicate content.
 - Black magic trick: You might use a fallback image that is displayed on these sites to link back to your own website or simply display that the content has been stolen. BUT it also costs bandwidth and might not always be displayed in proper context, depending on the website that steals your content.



Layout + Content

Reduce complexity

- Utilise a 'display more' link or page numbers to limit and load content only if requested by users.
- Don't implement image sliders, videos or image-based animations on the most important landing pages.
 - They are 'heavy' in load and runtime performance.
 - Keep it minimal: Prefer smaller static images, text or SVG vector graphics.
- Progressive content enhancement: Start with static elements and exchange later with interactive alternatives (e.g. on second load when cache is 'warm' or out of initial viewport).

6. Terms & Definitions



Terms A–M

- **Asset:** Any files: CSS, JS, Images, Audio/Video, Documents (like PDF) ...
- **Bundle/Bundling:** The technique of combining many files into one to reduce file sizes and speed up load times. Allows for better minification.
 - Bundling might lead to code errors if not done or tested properly (esp. in JS)
- **CSS:** Styling descriptions – Layout, Colors, Fonts, Animations, ...
- **JS:** JavaScript – The ‘interactive logic’ part. Animations, Tracking, Forms, Pop-ups, ...
- **Lazy loading:** The technique of loading only what is in the currently visible user viewport.
- **Minify/Minification:** The technique of reducing file sizes – text, code, images – by stripping irrelevant information: White spaces, comments, unused colors, ... Allows for great size reduction.
 - Some tools can remove unused code (especially when bundling many files).



Terms M–W

- **MPEG4:** Video file format that can offer reduced file sizes. Can be used as replacement for GIF images.
 - Needs a converter tool and coding.
 - Might not work in all themes.
- **SEO:** ‘Search Engine Optimisation’ – nowadays highly related with page speed.
- **SSD:** Solid–statedrive (Flash drive) – Faster than a hard drive (HDD).
- **TTFB:** ‘Time to first Byte’: The time spent waiting for the first server response (for each file).^{*8}
- **Viewport:** The part of the website users currently see on their browser screen.
- **VPS:** ‘Virtual Private Server’ –Virtual server that offers installation and configuration of any desired software.
- **WebP:** Modern imageformat that can offer reduced file sizes compared to JPG/PNG.
 - Needs a plugin to use within WordPress.
 - Might not work with all themes.



Definitions

- **Backend:** The source code that runs server-side 'behind the scenes'. Sometimes ambiguously used for the admin panel.
- **Frontend:** The publicly 'visible' part of the website that anyone can see.
- **Performance:** How fast the page content in the current view is visible and responds to user interaction (e.g. by clicking, touch-and-swipe gestures, scrolling, filling forms, ...)

7. Resources & Links



WordPress – Performance plugins

- [Autoptimize \(CSS, JS, Images\)](#)
- [WP Optimize / Advanced Database Cleaner](#)
- [W3 TotalCache^{*7} / WP Super Cache](#)
- [WP YouTubeLyte –YouTube performance](#)
- [Lazy Load for Comments](#)
- [WP Smushit\(Images\)](#)



WordPress – Performance focused themes 1/2

Review & Comparison of performance focused themes:

- helpiewp.com -5
- wp-rocket.me -10
- hostinger.com -15
- codeinwp.com -20+
- athemes.com -24
- colorlib.com -34
- wpbuffs.com -53



WordPress – Performance focused themes 2/2

Selection of a few themes:

- [WP Astra](#) – VanillaJS (nojQuery)
- [Generate Press](#) – VanillaJS
- [Neve – ThemelSle](#) – VanillaJS
- [The Gem – Codex Themes](#) – Uses jQuery / Offers many tempting ‘slow’ features

Current theme:

- [Massive Dynamic](#)



Testing tools

Speed tests

- [Google Page Speed Insights](#)
- [Google Lighthouse - Chrome DevTool extension](#)

Monitoring

- [Freshping](#)
- [StatusCake](#)
- [ServiceUptime](#)
- [Pingdom](#)



Annotations / Footnotes

1. [Think with Google –Industry benchmarks](#)
2. [MachMetrics –Why Page Speed Scores can be dangerous](#)
3. [WPMUDev Blog –Don't try to get 95 on Google PageSpeed Insights](#)
4. [Trackbacks –Pingbacks –Reblogs](#)
5. [Prevent imagehotlinkingin WordPress / Speed upWordPress –Hotlinking](#)
6. [WordPress Database Optimisation Guide / DB Optimisation –Best Practices](#)
7. [W3 TotalCache –Settings / W3 TotalCache –Ultimate Guide](#)
8. [TTFB –Time to First Byte –SEO impact](#)
9. [Slow WordPress dashboard –13 remedies](#)
10. [Disable admin ajaxcalls –Hack](#)

8. Disclaimer



Warranty & Liability disclaimer

All of the preceding tips and recommendations are understood as informational.

Implementation of any of these is at your own risk.