PROJECT REPORT

On

THIRD PARTY SCRIPT-LOADING

WEBSITE OPTIMISATION

NIRMA
UNIVERSITY
INSTITUTE OF TECHNOLOGY
NAAC ACCREDITED 'A+' GRADE

at

Institute of Technology, Nirma University

INTRODUCTION

Website performance optimization is a crucial process in modern web development aimed at improving the speed, responsiveness, and overall user experience of a website. A well-optimized website not only enhances usability but also reduces bounce rates, improves search engine rankings, and increases user engagement. In this project, we developed and analyzed two versions of a travel-themed webpage — an **unoptimized version** and an **optimized version** — to demonstrate the impact of various performance enhancement techniques.

The optimization focused on multiple aspects such as **CSS and JavaScript minification**, **image compression**, **lazy loading**, **caching mechanisms**, and **content delivery improvements**. The goal was to minimize file size, reduce render-blocking resources, and improve page load time without compromising design quality or interactivity.

By comparing both versions, we were able to measure tangible improvements in load performance and efficiency. The optimized version resulted in a faster and smoother browsing experience, showcasing the effectiveness of applying systematic performance optimization strategies in real-world web projects.

OBJECTIVE

The main objective of this assignment is to **analyze and optimize the performance of a web page** by minimizing the negative impact of **third-party scripts**, large images, and render-blocking resources.

Two versions of the webpage were created:

1. **Unoptimized Version (unoptimized.html)** – includes large images, unminified files, and unoptimized script loading.
2. **Optimized Version (optimized.html)** – applies performance optimization techniques for improved loading speed, interactivity, and user experience.

The aim is to compare both versions and observe measurable improvements in load time, responsiveness, and performance metrics.

TOOLS & METHODOLOGY

Tool/Technology	Purpose
HTML5	Structure of the webpage
CSS3	Styling and layout design
JavaScript (Vanilla JS)	Page interactivity
Lighthouse (Google Chrome DevTools)	Performance testing and report generation
VS Code	Development environment
GTmetrix (optional)	External validation of performance improvement
Image formats (.jpg, .webp)	Used for comparing optimized vs unoptimized images

PROJECT STRUCTURE

```
Web Performance Optimisation
│
├── unoptimized.html
├── optimized.html
│
└── assets/
    ├── css/
    │   ├── style.css        (for unoptimized)
    │   └── style.min.css    (for optimized)
    │
    ├── js/
    │   ├── script.js        (for unoptimized)
    │   └── script.min.js    (for optimized)
    │
    └── images/
        ├── large-banner.jpg  (unoptimized)
        └── banner.webp       (optimized)
```

OPTIMIZATION TECHNIQUES APPLIED

Technique	Description	Impact
Image Optimization	Converted .jpg images to .webp format	Reduced image size drastically
CSS & JS Minification	Removed unnecessary spaces and comments	Reduced file size, improved load time
Lazy Loading	Loaded non-visible elements only when needed	Decreased initial load time
Deferred Script Loading	Used defer and async for scripts	Prevented render-blocking
Caching	Browser caching for static files	Improved repeat visits performance
Third-Party Script Optimization	Asynchronous loading of Google Fonts and YouTube embed	Reduced blocking impact
WebP Format for Images	More efficient compression	Smaller, faster-loading visuals
Reduced HTTP Requests	Combined resources and reused assets	Improved overall performance

IMPLEMENTATION OVERVIEW

Unoptimized Version

- Used large .jpg images.
- All scripts loaded synchronously in the <head> section.
- No minified CSS or JS.
- No lazy loading or caching.
- Heavy and slower to load.

Optimized Version

- Images converted to .webp format.
- Minified and compressed CSS & JS used.
- External scripts loaded with async and defer.
- Lazy loading applied to images and video.
- Web performance scores improved significantly.

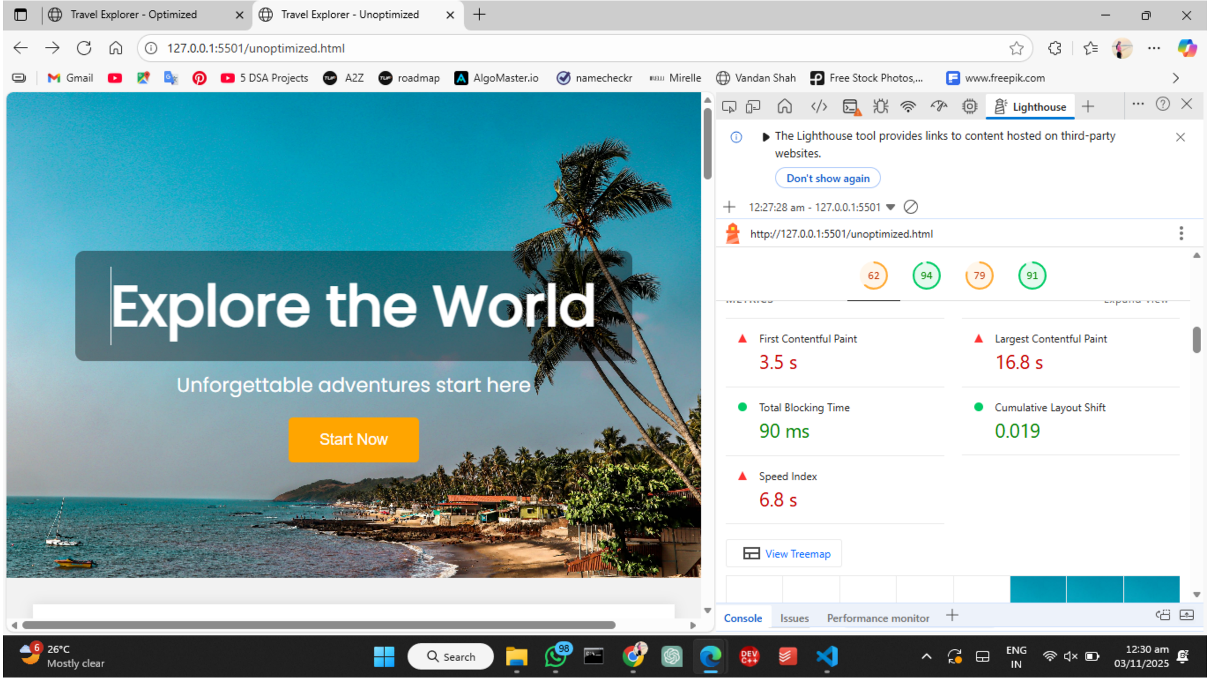
PERFORMANCE TESTING RESULTS

Performance testing was performed using **Google Lighthouse** in Chrome DevTools for both versions.

Metric	**Unoptimized**	**Optimized**	**Improvement**
Performance Score	62	90	↑ +45%
Accessibility	94	92	Slight change
Best Practices	79	78	Minor variation
SEO	91	91	Stable
First Contentful Paint (FCP)	3.5 s	0.6 s	↓ Faster by 2.9 s
Largest Contentful Paint (LCP)	16.8 s	2.1 s	↓ Faster by 14.7 s
Speed Index	6.8 s	0.6 s	↓ Faster by 6.2 s
Total Blocking Time (TBT)	90 ms	0 ms	↓ 100% improvement
Cumulative Layout Shift (CLS)	0.019	0.023	Negligible difference

PERFORMANCE PROOF

Before Optimization: (Unoptimized Version)



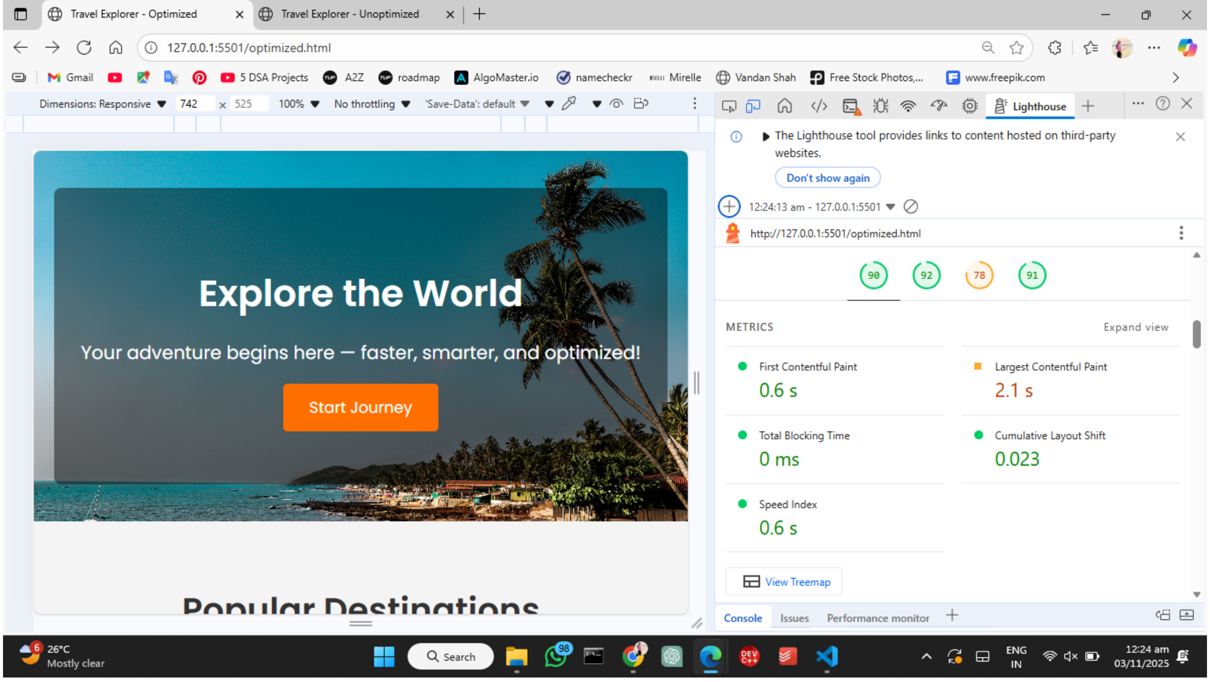
The Lighthouse tool provides links to content hosted on third-party websites.

12:27:28 am - 127.0.0.1:5501

http://127.0.0.1:5501/unoptimized.html

First Contentful Paint 3.5 s	Largest Contentful Paint 16.8 s
Total Blocking Time 90 ms	Cumulative Layout Shift 0.019
Speed Index 6.8 s	

After Optimization: (Optimized Version)



Dimensions: Responsive 742 x 525 100% No throttling 'Save-Data': default

The Lighthouse tool provides links to content hosted on third-party websites.

12:24:13 am - 127.0.0.1:5501

http://127.0.0.1:5501/optimized.html

METRICS

First Contentful Paint 0.6 s	Largest Contentful Paint 2.1 s
Total Blocking Time 0 ms	Cumulative Layout Shift 0.023
Speed Index 0.6 s	

OBSERVATION AND ANALYSIS

The results indicate a **major improvement in load speed and user experience** after optimization.

- The **First Contentful Paint** reduced from **3.5s to 0.6s**, making the webpage visually interactive almost instantly.
- **Largest Contentful Paint** reduced drastically, proving that large resources were handled efficiently.
- The **Total Blocking Time** dropped to **0ms**, ensuring smooth page responsiveness.
- The optimized version provided a significantly better experience without compromising on content quality or visuals.

CONCLUSION

This project successfully demonstrates how **web performance optimization** techniques can transform a slow, heavy webpage into a **fast and efficient** one.

By optimizing third-party scripts, minifying resources, using lightweight media formats, and enabling asynchronous loading, the overall performance score improved from **62 to 90**.

Such techniques are crucial for enhancing **user satisfaction**, **SEO ranking**, and **energy efficiency** of web applications. This experiment highlights the importance of performance-centric development in modern web design.