OPTIMIZED HOSTING PLATFORM FOR VIDEO AND IMAGE DATA

An Open Ended Lab Project Submitted
in Fulfillment of the Requirements
for the Degree of Bachelor of Technology

by

Netra Gupta
(132201017)

&

Alex E Mathew
(132201041)

Under the guidance of **Dr. Abdul Rasheed P**

IIT PALAKKAD

INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD

Contents

Chapter 1

Introduction

When it comes to digital connectivity, one of the most urgent issues is the buffering that websites and applications experience across a variety of networks, from those with bad to medium bandwidth. To address this widespread problem, our effort is focused on creating a complete video/image hosting platform. The purpose of this platform is to standardize several factors related to buffering problems, with the main objectives of improving performance and promoting increased accessibility. In our effort, we will conduct a thorough analysis to measure the platform's effectiveness in reducing buffering issues and assuring a smooth streaming experience, especially when the hosting circumstances are optimal.

1.1 Problem Statement

The challenge of the present concerns buffering issues encountered by websites and applications on various networks, ranging from poor to medium connectivity. In response, we are building a platform that amalgamates the discussed parameters. This platform seeks to enhance performance and accessibility. For our conclusive analysis, we will employ an application to thoroughly assess whether our platform effectively mitigates buffering problems and delivers a seamless streaming experience under optimized hosting conditions.

1.2 Organization of The Report

This section serves as an introduction to the key aspects covered in this report, aligning with the problem statement mentioned earlier. We present a brief overview of our project's functionality, highlighting the main features incorporated, the methodology employed, and the project's current status and future prospects. Chapter 1 delves into the problem statement, outlining the necessity for undertaking this project. Moving to Chapter 2, we delve into the methodology used for project development. Chapter 3 explores the main features implemented, shedding light on the challenges faced during their execution. Finally, Chapter 4 provides a conclusion, summarizing the current project status and offering insights into future endeavors.

Chapter 2

Methodology

2.1 Choosing the Right Tech Stack & Database Storage Server

The first step of our project was to choose the correct technology stack and database storage while considering the speed and cost effectiveness of it as a whole. After much discussion, we decided to go ahead with "Next.js", an open-source, React-based, web development framework, due to its adaptability and effectiveness. For our database, we chose Firebase (Google's database storage server) because it provides easy-to-use real-time database functionalities. Our main goal was to ensure that a cost-effective and scalable project was feasible while following excellent coding practices.

2.2 Designing a good User Interface

A good User Interface is the key to a successful project. We carefully designed a decently attractive interface while keeping it simple to use. We rigorously tested and ensured that the website was suitable and showed optimal performance responsiveness for screens of all dimensions, ranging from the smallest phones to the biggest laptops.

2.3 Developing Dynamic Content Management Features

Since it is a video data hosting platform, developing features to connect the Firebase server to our project for uploading and browsing data, both front-end and back-end, was a necessity. We created an upload page that would take images and videos that were supposed to be uploaded along with optional meta-deta, like file title and description, if necessary. The home page would display all the content stored in the Firebase server. Caching techniques were implemented to ensure that new data was fetched immediately as and when it was uploaded and displayed on the homepage.

2.4 Optimisation of the platform

The most important part of this project was the optimization part. Instead of saving the same file in multiple resolutions and then loading a specific resolution file based on the network quality (an approach which would use a lot of server space), we used FFmpeg, an advanced tool for processing media. FFmpeg encodes and transcodes video files in multiple formats and resolutions. We used FFmpeg to enable video transcoding and modification, which allowed us to help make the user experience even better. We were able to reduce the buffering by successfully deploying FFmpeg algorithms to change file size depending on the internet bandwidth. The above, combined with the Dynamic Adaptive Streaming over HTTP (DASH) method, adjusts the video quality in real time depending on the network. The caching techniques mentioned in the previous section further helped in reducing server requests, as the page would not reload unless the content on the server has been modified.

Chapter 3

Goals and Challenges Faced

3.1 Objectives

3.1.1 Make an optimized Data hosting platform

Our initial objective was to develop a data hosting platform, similar to YouTube with respect to the video optimization technique.

The effort to achieve these goals encountered some challenges and complexities, as outlined below.

3.2 Problems faced and their Solutions

3.2.1 CORS errors in Firebase

CORS(Cross-Origin Resource Sharing) errors occurred in our Next.js project when our web application attempted to interact with Firebase services hosted in different domains. To address this issue, we configured the CORS settings in the Firebase Console, specifying the origins (domains) allowed to whitelist to access Firebase services. Additionally, we used Firebase SDKs, such as the JavaScript SDK, which seamlessly handled CORS issues, simplifying integration, and minimizing errors. By configuring CORS settings and using Firebase SDKs, we successfully mitigated CORS errors in our project, ensuring smooth

interaction between our Webapp and Firebase services.

3.2.2 FFmpeg module

FFmpeg usage guide for rendering and activating it on the Firebase server side is unfortunately not clear. This caused a lot of problems as working with small errors became messy without a proper documentation manual to refer to. However, with the help of stackoverflow and a lot of debugging, we were able to overcome them.

3.2.3 Build issues in Next.js app

We encountered an issue where one of the modules that we attempted to use resulted in conflicts with other dependencies in our project. This conflict led to build errors due to version mismatches, duplicate dependencies, or incompatible configurations. The only possible fix for this was to remove the module and use alternative solutions.

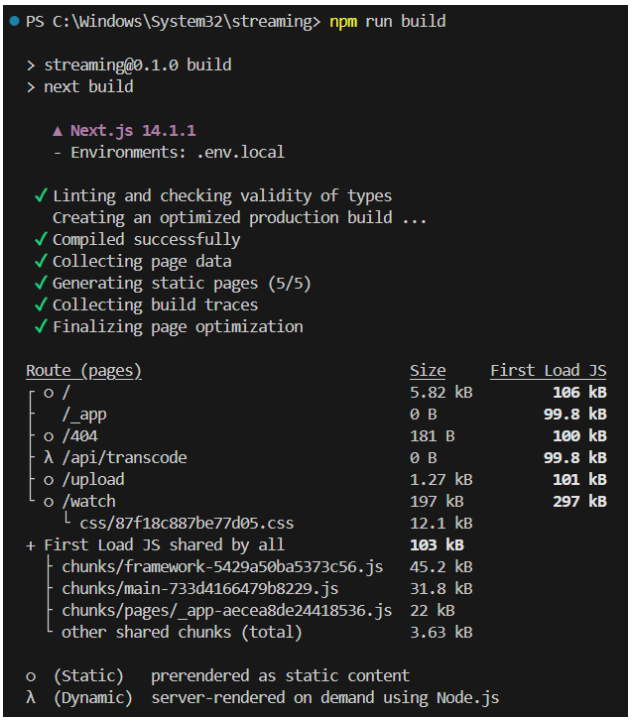

Fig. 3.1 Build successful!

Chapter 4

Conclusion and Future Work

4.1 Current Status

As of May 14, 2024, our scripts have achieved success in accurately identifying the Broadband speeds of the user and changing the resolution of the video accordingly. If the Internet speed is high but the quality of the original video is very poor, then the video streamed will have the same quality as the original. In cases where broadband speeds are not detectable, the quality is 480p.

4.2 Future Work

For further improvements, a thumbnail generating code to add thumbnails to the videos can be added. Also, there are some minor bugs to be fixed like showing the correct duration of a video. As the next step, we plan to convert this video hosting and optimizing platform to a robust API. This will make it possible to seamlessly integrate with various applications and can enhance accessibility and usability across various other platforms.

4.3 Conclusion

We have created an optimized image and video hosting platform. It can efficiently manage and deliver video content based on broadband speed. This platform utilizes various compression techniques and optimized storage strategies. The platform reduces the required storage space while maintaining the quality of the video. Of course, if the connection is too poor, the video quality drops to maintain a steady streaming experience.