T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

HTML ANALYZER

MUHAMMED OĞUZ

SUPERVISOR
DR. ÖĞR. ÜYESI ALP ARSLAN BAYRAKÇİ

GEBZE
2023

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

HTML ANALYZER

MUHAMMED OĞUZ

SUPERVISOR
DR. ÖĞR. ÜYESI ALP ARSLAN BAYRAKÇİ

2023
GEBZE

GEBZE TECHNICAL UNIVERSITY	GRADUATION PROJECT JURY APPROVAL FORM

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 15/01/2023 by the following jury.

JURY

Member (Supervisor)	:	Dr. Öğr. Üyesi Alp Arslan BAYRAKÇİ
Member	:	Prof. Dr. Erkan ZERGEROĞLU

ABSTRACT

In this project, a tool was developed to analyze and improve the quality of HTML code. The tool was developed using C and React. The tool checks for common HTML errors such as missing tags, duplicate attributes, and invalid DOCTYPE declaration. It also includes functionality to automatically fix certain errors and provide solutions for others. Additionally, an error level system was implemented to classify errors as warnings, errors, or SEO/accessibility related. In order to improve the code, the tool also includes a feature to convert tables to proper divs and add inline styles to images for better accessibility. Overall, this tool can be used to improve the quality and accessibility of HTML code, making it more compliant with web standards.

ACKNOWLEDGEMENT

I am greatly indebted to my professor, Alp Arslan Bayrakci, for his invaluable guidance and support throughout this project. He suggested that I use React and C to implement the project, which proved to be an extremely informative experience. I am deeply grateful for his insightful suggestions and support.

Muhammed Oğuz

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol or Abbreviation	:	**Explanation**
React	:	JavaScript library for building user interfaces
Vite	:	JavaScript build tool and development server
Azure	:	Microsoft’s cloud computing platform
Github Pages	:	Platform for hosting static websites
HTML	:	Hypertext Markup Language
CSS	:	Cascading Style Sheets
JS	:	JavaScript
C#	:	Programming language
.Net	:	Framework for developing web applications using C#
API	:	Application Programming Interface
HTML5	:	Latest version of HTML
DOM	:	Document Object Model
JSON	:	JavaScript Object Notation
SEO	:	Search Engine Optimization
WCAG	:	Web Content Accessibility Guidelines
ACCESSIBILITY	:	Accessible Rich Internet Applications
URL	:	Uniform Resource Locator
URI	:	Uniform Resource Identifier
HTTP	:	Hypertext Transfer Protocol

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

This chapter provides an overview of the project and its objectives. The main focus of the project is to develop a tool for analyzing and improving the quality of HTML code. The tool is designed to be easy to use and understand, with a user-friendly interface that allows users to quickly identify and fix common HTML errors. The project is implemented using React, a popular JavaScript library for building user interfaces, and .Net framework, a C framework for creating backend functionality. The project is deployed on Azure and Github pages. The project also includes a detailed roadmap for the development process. The chapter also includes a list of symbols and abbreviations used in the report.

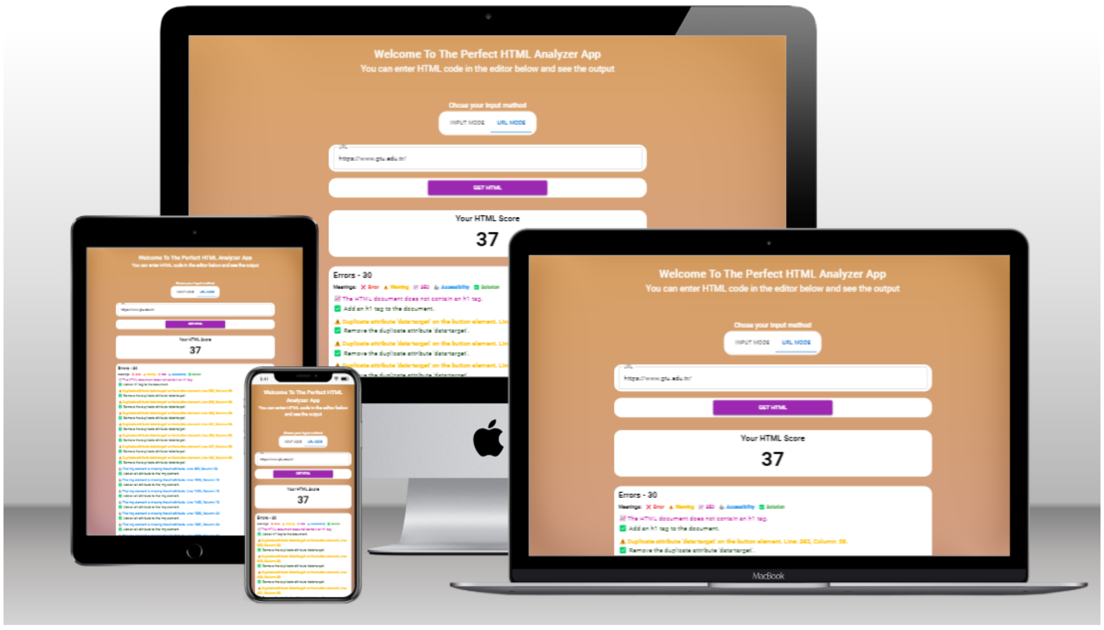

Welcome To The Perfect HTML Analyzer App
You can enter HTML code in the editor below and see the output
Choose your input method
INPUT MODE
URL MODE
GET HTML
Your HTML Score
37
Errors - 30

Figure 1.1: Mockup of the project.

1.1. Objectives

The main objective of the project is to create a tool that can analyze and improve the quality of HTML code. The tool should be able to identify common HTML errors and provide solutions to fix them. The tool should also be able to give a score to the user indicating the overall quality of the HTML code. The tool should be easy to use, with a user-friendly interface that allows users to quickly identify and fix errors.

1.2. Scope

The scope of the project is to develop a tool for analyzing and improving the quality of HTML code. The tool should be able to identify common HTML errors, such as missing or duplicate tags, invalid attribute values, and invalid DOCTYPE declarations. The tool should also be able to provide solutions to fix the identified errors. The tool should be implemented using React and .Net framework, and deployed on Azure and Github pages.

1.3. Methodology

The project is developed using an Agile methodology, with a focus on iterative and incremental development. The development process includes several phases, including requirements gathering, design, implementation, testing, and deployment. The project uses Vite to initialize the project, and C and .Net framework for the backend. The project is deployed on Azure and Github pages.

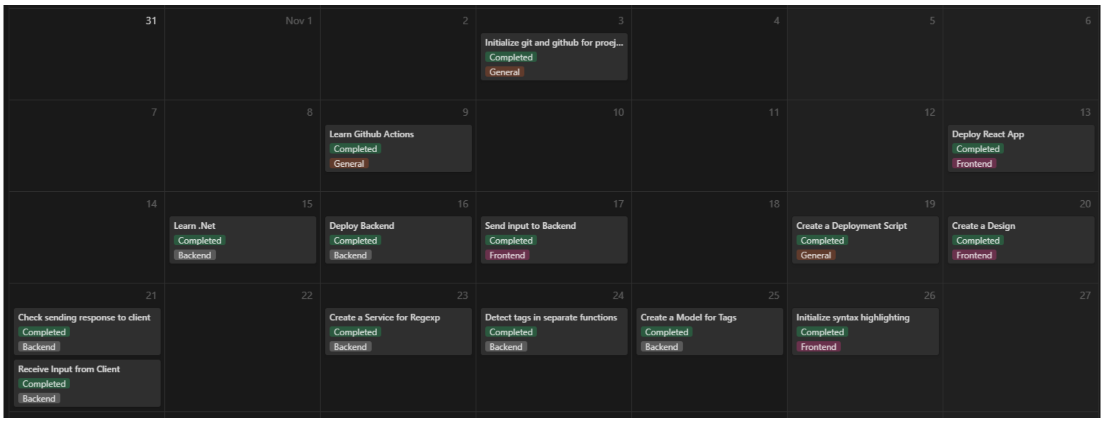
1.4. Limitations

The tool is limited to analyzing and improving the quality of HTML code. It does not include support for other web technologies such as CSS or JavaScript. It also does not include support for more advanced features such as semantic HTML or accessibility.

2. ROADMAP - AGILE

I divide the development process into 3 phases

2.1. Phase One - Initial Setup



31	Nov 1	2	3 Initialize git and github for proej... Completed General	4	5	6
7	8	9 Learn Github Actions Completed General	10	11	12	13 Deploy React App Completed Frontend
14	15 Learn .Net Completed Backend	16 Deploy Backend Completed Backend	17 Send input to Backend Completed Frontend	18	19 Create a Deployment Script Completed General	20 Create a Design Completed Frontend
21 Check sending response to client Completed Backend Receive Input from Client Completed Backend	22	23 Create a Service for Regexp Completed Backend	24 Detect tags in separate functions Completed Backend	25 Create a Model for Tags Completed Backend	26 Initialize syntax highlighting Completed Frontend	27

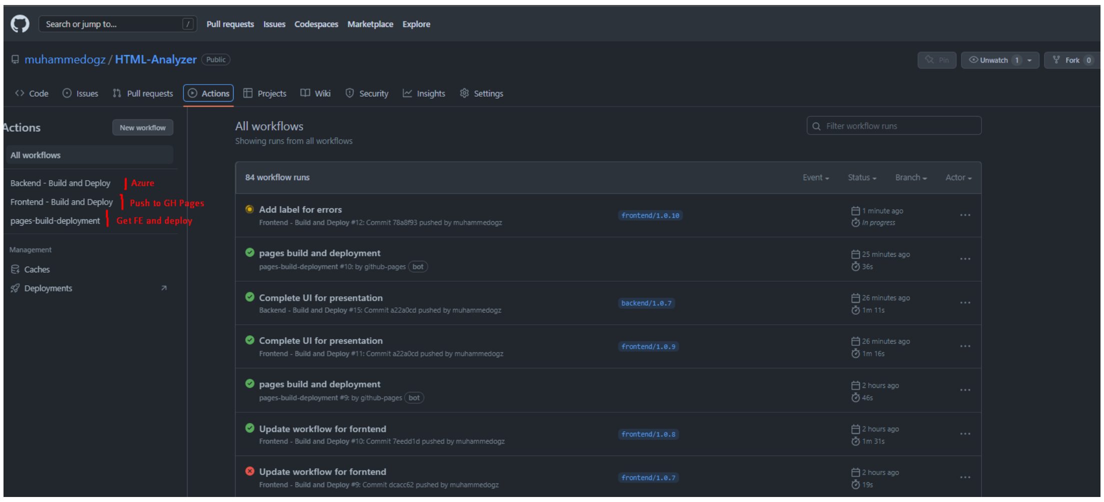
Figure 2.1: Phase One Roadmap

2.1.1. infrastructure

n the first phase of development, the focus was on setting up the necessary infrastructure for the project. This included initializing git and github for version control and collaboration, learning how to use Github Actions for continuous integration and deployment, and deploying the React frontend to a hosting platform such as Azure. Additionally, the .Net framework was learned and used to create the backend of the application, and the process for sending input from the frontend to the backend was established.

2.1.2. Deployment

To streamline the deployment process, a deployment script was created to automate the process of updating the application on the hosting platform. A design was also created to guide the visual appearance of the application. Furthermore, the application was tested to ensure that the response from the backend was correctly being sent to the



All workflows
Showing runs from all workflows

Backend - Build and Deploy | Azure
Frontend - Build and Deploy | Push to GH Pages
pages-build-deployment | Get FE and deploy

84 workflow runs

Run	Branch	Time
Add label for errors Frontend - Build and Deploy #12: Commit 78a8f93 pushed by muhammedogz	frontend/1.0.10	1 minute ago *In progress*
pages build and deployment pages-build-deployment #10: by github-pages bot		25 minutes ago 36s
Complete UI for presentation Backend - Build and Deploy #15: Commit a22a0cd pushed by muhammedogz	backend/1.0.7	26 minutes ago 1m 11s
Complete UI for presentation Frontend - Build and Deploy #11: Commit a22a0cd pushed by muhammedogz	frontend/1.0.9	26 minutes ago 1m 16s
pages build and deployment pages-build-deployment #9: by github-pages bot		2 hours ago 46s
Update workflow for forntend Frontend - Build and Deploy #10: Commit 7eedd1d pushed by muhammedogz	frontend/1.0.8	2 hours ago 1m 31s
Update workflow for forntend Frontend - Build and Deploy #9: Commit dcacc62 pushed by muhammedogz	frontend/1.0.7	2 hours ago 19s

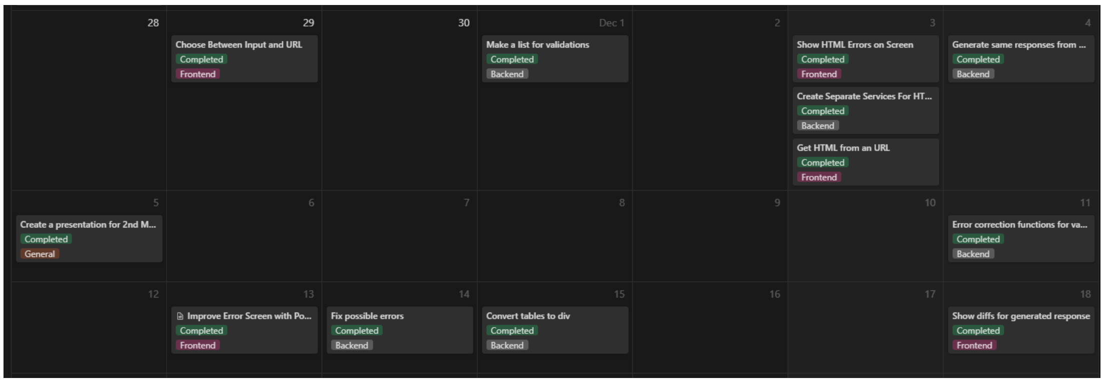
Figure 2.2: Github Actions

client.

2.1.3. Improve Functionality

To improve the functionality of the application, a service for regular expressions was implemented to detect specific tags in the HTML code. The tags were then separated into individual functions for better organization and maintainability. A model was also created to store information about the tags and their properties. Finally, syntax highlighting was implemented to improve the readability of the code for the user.

2.2. Phase Two - User Input and Error Correction



28	29 Choose Between Input and URL Completed Frontend	30	Dec 1 Make a list for validations Completed Backend	2	3 Show HTML Errors on Screen Completed Frontend Create Separate Services For HT... Completed Backend Get HTML from an URL Completed Frontend	4 Generate same responses from ... Completed Backend
5 Create a presentation for 2nd M... Completed General	6	7	8	9	10	11 Error correction functions for va... Completed Backend
12	13 Improve Error Screen with Po... Completed Frontend	14 Fix possible errors Completed Backend	15 Convert tables to div Completed Backend	16	17	18 Show diffs for generated response Completed Frontend

Figure 2.3: Phase Two Roadmap

2.2.1. User Input

In the second phase of development, the focus was on providing more options for input and implementing a list of validations to ensure that the input is in the correct format. One of the key features developed in this phase was the ability to show HTML errors on the screen and provide possible solutions for these errors.

To achieve this, separate services were created for the HTML document package, allowing for more flexibility and ease of use. The application was also able to get the HTML from a given URL and generate the same responses as if the input was entered manually. Additionally, a presentation was created for the second meeting to showcase the progress and achievements of the project.

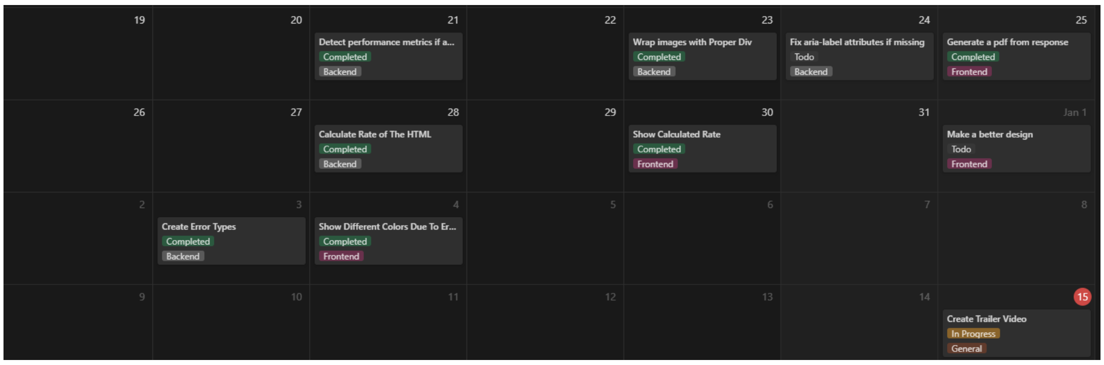
2.2.2. Error Correction

Another important aspect of this phase was the implementation of error correction functions for various HTML tags. These functions were designed to improve the overall functionality and usability of the application. Furthermore, the error screen was enhanced with possible solutions to make it more user-friendly.

In order to further improve the application, a feature was added to fix possible errors that may occur during the validation process. Additionally, tables were converted to div elements to ensure that the HTML is more accessible and can be displayed correctly on different devices.

2.2.3. Overall Functionality

Finally, a feature was added to show the differences between the original HTML and the generated response. This allows users to easily identify any changes that have been made and understand the impact of the validations on their HTML code. Overall, the second phase of development aimed to provide more options for input, improve the overall functionality and usability of the application, and make the HTML more accessible.



19	20	21 Detect performance metrics if a... Completed Backend	22	23 Wrap images with Proper Div Completed Backend	24 Fix aria-label attributes if missing Todo Backend	25 Generate a pdf from response Completed Frontend
26	27	28 Calculate Rate of The HTML Completed Backend	29	30 Show Calculated Rate Completed Frontend	31	Jan 1 Make a better design Todo Frontend
2	3 Create Error Types Completed Backend	4 Show Different Colors Due To Er... Completed Frontend	5	6	7	8
9	10	11	12	13	14	15 Create Trailer Video In Progress General

Figure 2.4: Phase Three Roadmap

2.3. Phase Three - Final Touches

2.3.1. Accessibility

In phase three, various improvements were made to the application. One of the main focuses was on improving the accessibility of the generated HTML code. This included detecting and fixing missing aria-label attributes, as well as wrapping images with proper divs to ensure they were properly accessible to screen readers. Additionally, performance metrics were detected and displayed if available on the server.

2.3.2. Generating PDF and Improve Design

Another important aspect of this phase was the ability to generate a PDF from the generated HTML code. This allowed for easy sharing and printing of the code. Additionally, a calculation was implemented to rate the overall quality of the HTML code, and this rating was displayed to the user.

To improve the overall design of the application, various design changes were made. This included implementing different colors based on error types, which made it easier for the user to identify and fix specific issues.

2.3.3. Last Touches

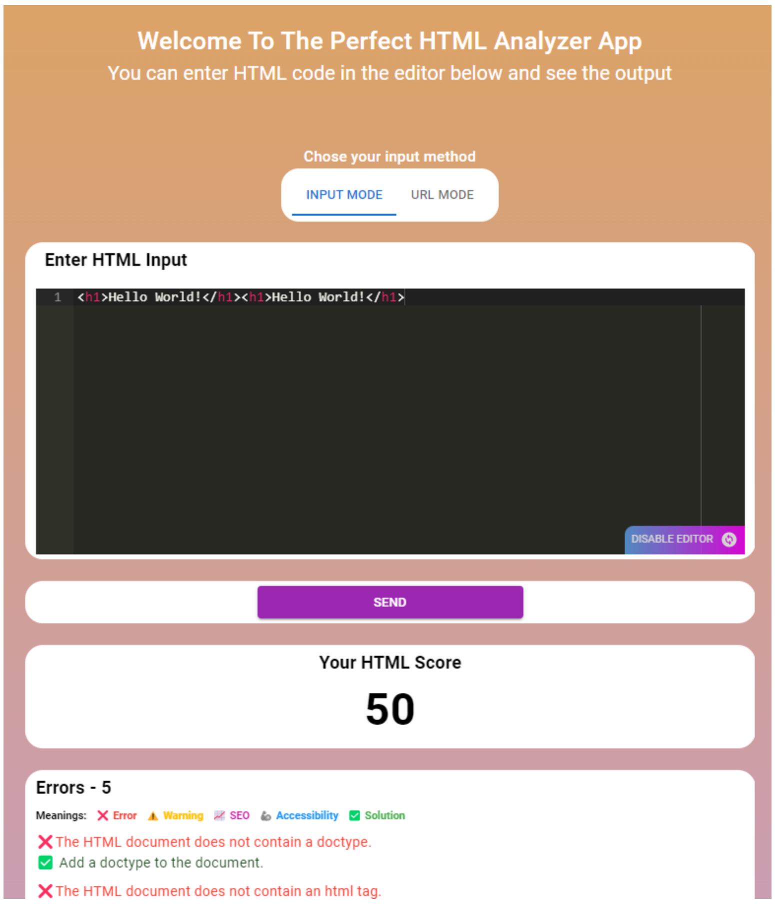
Finally, the application was tested to ensure that all features were working correctly. The application was also tested on different devices to ensure that it was responsive and displayed correctly on all platforms. Overall, the third phase of development aimed to improve the accessibility of the generated HTML code, generate a PDF from the response, and calculate the rate of the HTML code.

3. DEVELOPMENT

I divide the development process into 3 part. Frontend - Backend - Devops

3.1. Frontend

The User Interface of the project was developed using React, a popular JavaScript library for building user interfaces. Vite, a lightweight development tool, was used to initialize the project. The application was designed to provide a user-friendly experience with a simple and elegant layout.


Welcome To The Perfect HTML Analyzer App
You can enter HTML code in the editor below and see the output
Chose your input method
INPUT MODE
URL MODE
Enter HTML Input
DISABLE EDITOR
SEND
Your HTML Score
50
Errors - 5
Meanings: Error Warning SEO Accessibility Solution

```html
<h1>Hello World!</h1><h1>Hello World!</h1>
```

✗ The HTML document does not contain a doctype.
✓ Add a doctype to the document.

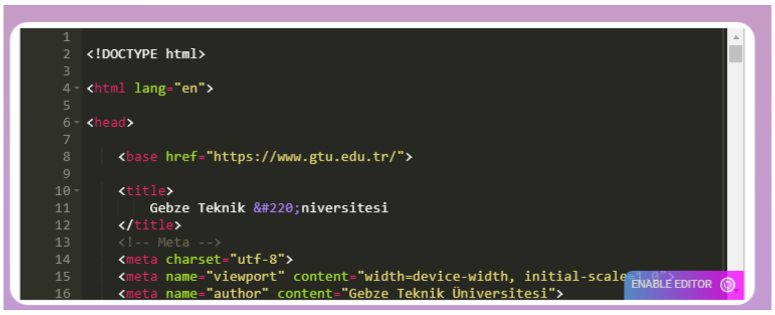
✗ The HTML document does not contain an html tag.

Figure 3.1: Frontend UI

3.1.1. Text Editor

A text editor, built using the ace-editor library, was implemented in the project to allow users to input and edit their HTML code. This feature enables users to make changes to their code directly within the application, providing a more seamless experience.

This text editor has a disable feature. User could disable the text editor and use the application as a validator. This feature is useful for users who want to validate their HTML code without having to edit it.



```

<!DOCTYPE html>

<html lang="en">

<head>

    <base href="https://www.gtu.edu.tr/">

    <title>
        Gebze Teknik &#220;niversitesi
    </title>
    <!-- Meta -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale
    <meta name="author" content="Gebze Teknik Üniversitesi">
```

Figure 3.2: Editor UI

3.1.2. Deployment

The frontend of the application was deployed on GitHub Pages, a web hosting service that allows easy deployment of static sites. This service was chosen because it is free and easy to use. The application was also designed to be responsive, ensuring optimal performance on different devices and screen sizes.

3.1.3. Testing and Validation

The application was thoroughly tested to ensure its functionality and usability. This included testing the text editor, sending requests to the backend, and validating the responses. Additionally, the application was designed to be responsive, ensuring optimal performance on different devices and screen sizes.

3.2. Backend

In the backend of the project, the primary technology used was the .NET framework. This allowed for the creation of a robust and efficient server to handle the incoming HTML requests and perform the necessary validation and error correction. In order to accomplish this, various libraries and frameworks such as ASP.NET Core were utilized.

3.2.1. API

The backend of the application was designed to be modular and extensible, allowing for easy expansion and maintenance in the future. This was achieved through the use of controllers and routing, allowing for easy communication between the two parts of the application.

One of the key features implemented in the backend was the ability to receive and process requests from the frontend. This was achieved through the use of controllers and routing, allowing for easy communication between the two parts of the application.

Additionally, the backend also included a service for performing regular expressions on the incoming HTML, allowing for the detection and correction of various errors. This service was built to be modular and extensible, allowing for easy expansion and maintenance in the future.

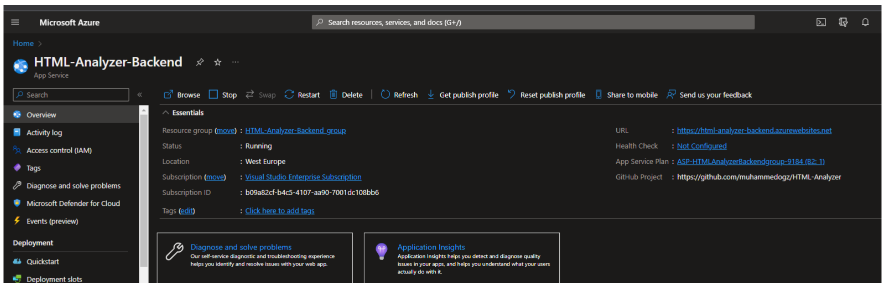
To streamline the deployment process, a deployment script was created to automate the process of updating the application on the hosting platform. A design was also created to guide the visual appearance of the application. Furthermore, the application was tested to ensure that the response from the backend was correctly being sent to the client.

3.2.2. Deployment and Azure Portal

The backend of the application was deployed on Azure, a cloud platform that allows easy deployment of web applications. This service was chosen because it is free and easy to use.

3.2.3. Performance

Lastly, in order to improve the performance and scalability of the backend, various performance metrics were monitored and optimized where necessary to reduce the load on the server and improve the overall user experience.



HTML-Analyzer-Backend
App Service

Resource group (move) : HTML-Analyzer-Backend_group
Status : Running
Location : West Europe
Subscription (move) : Visual Studio Enterprise Subscription
Subscription ID : b09a82cf-b4c5-4107-aa90-7001dc108bb6
Tags (edit) : Click here to add tags

URL : https://html-analyzer-backend.azurewebsites.net
Health Check : Not Configured
App Service Plan : ASP-HTMLAnalyzerBackendgroup-9184 (B2: 1)
GitHub Project : https://github.com/muhammedogz/HTML-Analyzer

Diagnose and solve problems
Our self-service diagnostic and troubleshooting experience helps you identify and resolve issues with your web app.

Application Insights
Application Insights helps you detect and diagnose quality issues in your apps, and helps you understand what your users actually do with it.

Figure 3.3: Azure Portal

3.2.4. Swagger

Swagger is a tool that allows developers to design, build, document, and consume RESTful web services. It was used to create a user-friendly interface for the backend of the application. This allowed for easy testing and debugging of the application.

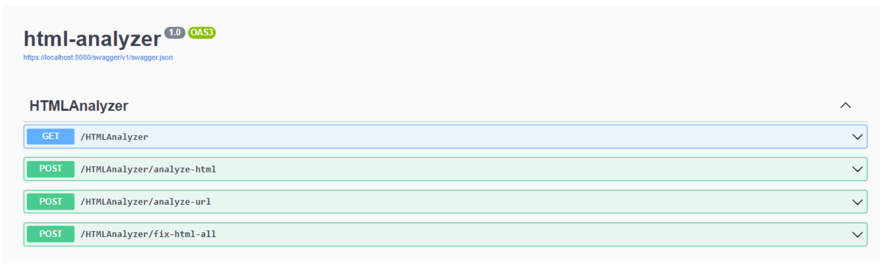

html-analyzer
1.0
OAS3
https://localhost:8080/swagger/v1/swagger.json
HTMLAnalyzer
GET
/HTMLAnalyzer
POST
/HTMLAnalyzer/analyze-html
POST
/HTMLAnalyzer/analyze-url
POST
/HTMLAnalyzer/fix-html-all

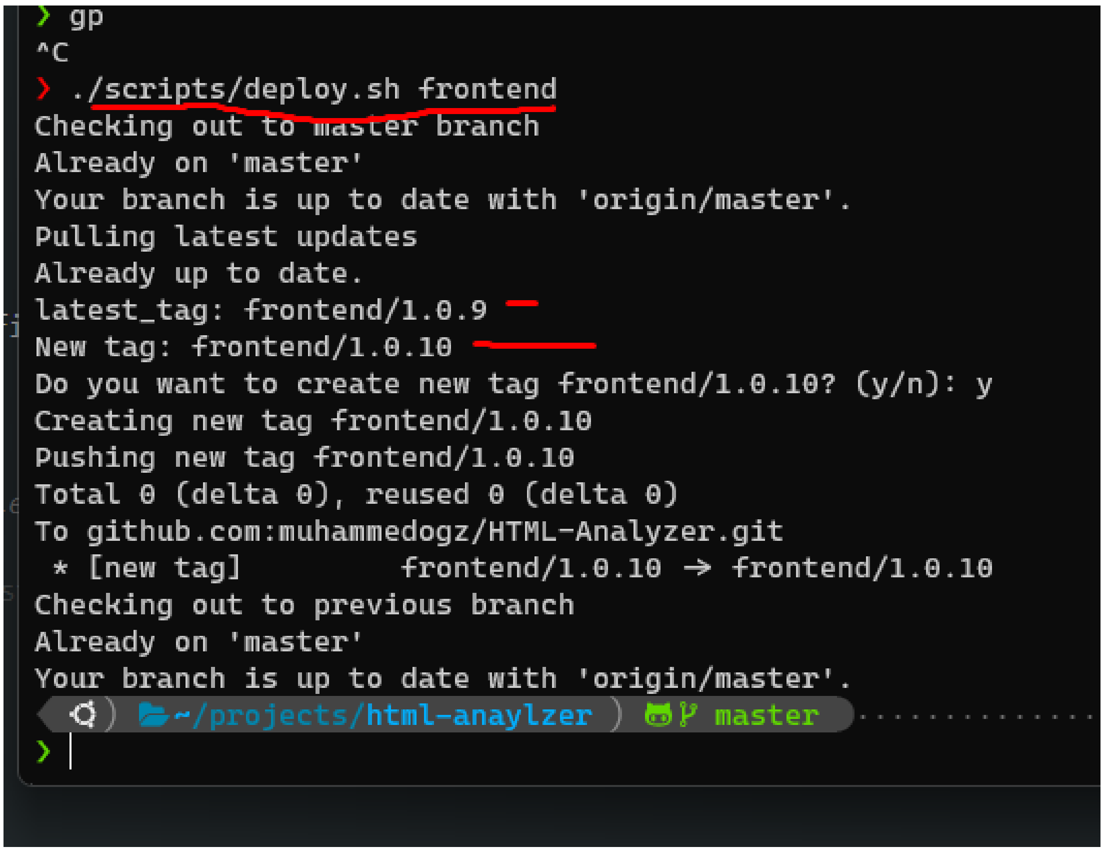
Figure 3.4: Swagger UI

3.3. Devops

In the development of this project, a focus was placed on implementing best practices for DevOps. This included utilizing git tags for releasing both the frontend and backend portions of the application.

3.3.1. Script for Tags

A script was also created to automate the process of creating git tags, streamlining the release process and ensuring consistency.



```
❯ gp
^C
❯ ./scripts/deploy.sh frontend
Checking out to master branch
Already on 'master'
Your branch is up to date with 'origin/master'.
Pulling latest updates
Already up to date.
latest_tag: frontend/1.0.9
New tag: frontend/1.0.10
Do you want to create new tag frontend/1.0.10? (y/n): y
Creating new tag frontend/1.0.10
Pushing new tag frontend/1.0.10
Total 0 (delta 0), reused 0 (delta 0)
To github.com:muhammedogz/HTML-Analyzer.git
 * [new tag]         frontend/1.0.10 -> frontend/1.0.10
Checking out to previous branch
Already on 'master'
Your branch is up to date with 'origin/master'.
  ~/projects/html-anaylzer    master
❯
```

Figure 3.5: A script for tags

3.3.2. CI/CD

Additionally, the project was integrated with Github Actions to automate the build, test and deployment process, further promoting efficiency and reliability.

Overall, the implementation of these DevOps practices greatly improved the project's development and release process, allowing for a more seamless and efficient workflow.

4. CONCLUSION

In this project, the main aim was to create a web application that can parse HTML over an URL, detect attributes and their usages with Regex, and provide a responsive design that works on all devices. Additionally, the application should be able to generate a report containing at least five suggestions and give a good rate to the HTML.

4.1. Achievements

The project was a success as all of the proposed goals were achieved. The application can parse HTML over an URL, detect attributes and their usages with Regex, provide a responsive design that works on all devices. Furthermore, the application can detect invalid images, provide syntax highlighting when copy-pasting HTML to the site, and generate a PDF report. Additionally, the application can detect performance metrics, show the diff if an error is fixed, check accessibility labels such as ”aria-label,” check SEO labels such as meta tags, and give a point to the site within a category.

4.2. Success Criteria

The success criteria for this project were met as the application can parse HTML over an URL, detect attributes and their usages with Regex, provide a responsive design that works on all devices. Additionally, the application can generate a report containing at least five suggestions and give a good rate to the HTML. Furthermore, the application can detect invalid images, provide syntax highlighting when copy-pasting HTML to the site, and generate a PDF report. Additionally, the application can detect performance metrics, show the diff if an error is fixed, check accessibility labels such as ”aria-label,” check SEO labels such as meta tags, and give a point to the site within a category.