

PROBLEM STATEMENT 3

By : Data Dynamo



Goals and Objectives

- **Preserving Formatting and Styles:** Maintaining the original formatting, including fonts, colors, styles, and layouts, is crucial for the converted document to accurately represent the original HTML content.
- **Dealing with Hyperlinks and References:** Ensuring that hyperlinks, references, and cross-references within the HTML document are retained and functional in the resulting DOCX file.
- **Supporting Complex HTML Structures:** Robust handling of complex HTML structures, including nested elements, tables, lists, and CSS styles, is necessary for a reliable conversion process.
- **Minimizing Data Loss:** Minimizing the loss of information during the conversion process, such as special characters, non-standard fonts, and custom CSS styles.
- **Scalability and Performance:** The conversion tool should be capable of handling large or multiple HTML files efficiently without compromising on accuracy or speed.



TEAM MEMBERS



Maanak Findal

21105065

TECH



Adarsh Mandloi

21105089

TECH



Harsh Kumar

21105022

TECH

TEAM MEMBERS



Om Vaish

21105087

TECH



Anjali

21105113

NON-TECH



Upload
HTML File



Submit
Conversion
request



Display
Conversion
Progress



Download
Docx file

USER INTERFACE



○ **Handle
File Upload**

○ **Validate HTML**

○ **Convert HTML
to DOCX**

○ **using HTML to
DOC js library**

○ **Apply Styling**

○ **Generate
DOCX File**

○ **Send
Conversion
Progress
Updates**



○ **Send DOC
File to User**

SERVER-SIDE

CHOICE OF TECHNOLOGIES

We have chosen Node.js, Express.js, Python as the core technologies for our project, and here's why:

Node.js for Real-Time Conversion

Non-blocking, event-driven architecture, that makes it exceptionally well-suited for real-time operations like document conversion.

Its speed and scalability are key advantages, ensuring our tool can handle large HTML docs efficiently.

Express.js for Web Server Functionality

Express.js offers features for routing, and handling HTTP requests.

It enables us to provide a user-friendly web interface for our converter.

Cross-Platform Compatibility

Node.js is inherently cross-platform ensuring that our converter can be used on different operating systems, including Windows, macOS, and Linux.,

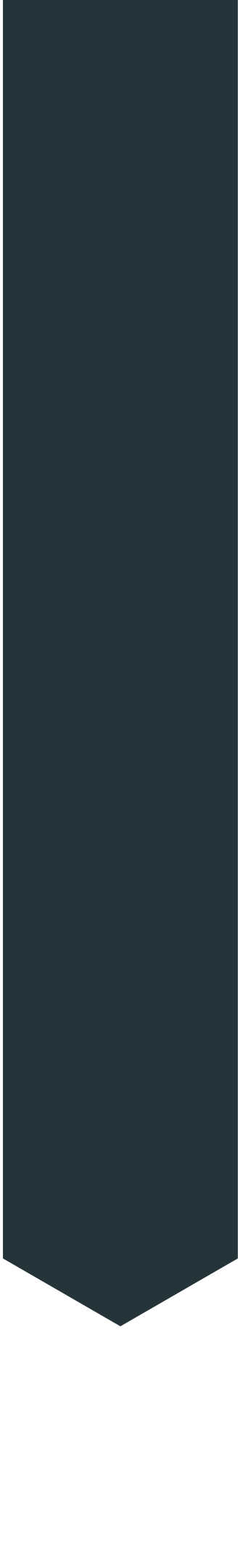
Rich Libraries and Frameworks:

Both Node.js and Python offer a wide array of libraries and frameworks that aid in HTML parsing, CSS handling, and DOCX generation.



APPROACH I

using inbuilt library

- Frontend (index.html and script.js)
 - Backend(server.js)
 - HTML to DOCX Conversion (docxConverter.js)
 - This only worked for small piece of HTML code.
- 



APPROACH 2

- Flask:
Python Framework
- HTML_TO_DOCX:
Javascript Library
- OS:
Generates Path for conversion
- Request:
URL input is provided and we get o/p with
`request.get(URL)`
- Logging:
Have `console.log` and displays error in terminal
or power shell



FLASK

MODULES:

- request
- render_template
- redirect
- url_for
- send_file

Home Route:

- Basic Rendering

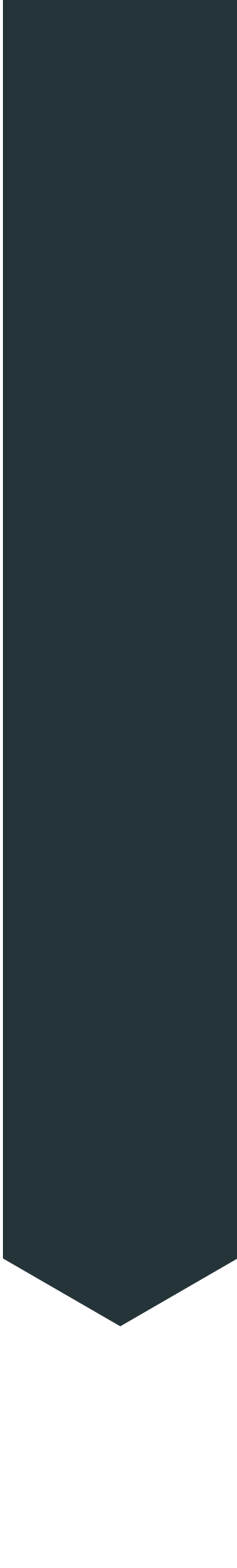
Complete Route:

- Rendering after
the Conversion
to docx



APPROACH 3

user defined Python library

- By using BeautifulSoup- a python library used for web scrapping
 - We tried tackling all the HTML and CSS components individually , writing code for each component and tried converting into DOC format
- 



CHALLENGES

- Image Conversions
- Table Conversions
- Handling Flex and Grid Layouts

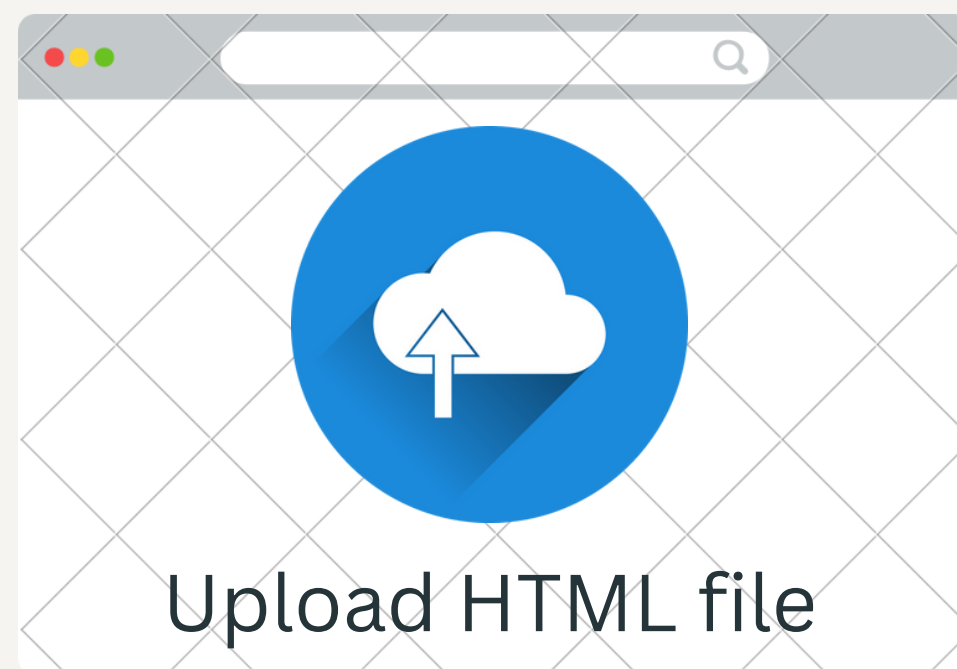


FILE UPLOAD AND CONVERSION

Flask-based web application

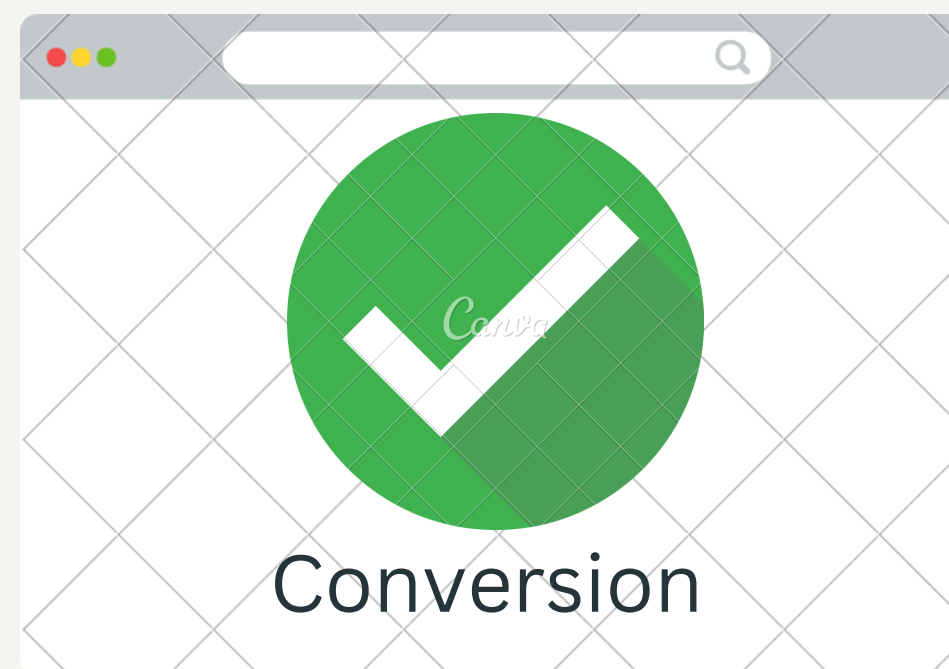
1. File Upload Functionality:

- Users can navigate to the "File Upload" section on the web application.
- The /upload route handles file uploads.
- Conversion is made possible through the convert html to docx function.



2. Completed Page:

- After conversion, the resulting DOCX document is made available for download.
- Users are then redirected to the "**completed**" page to indicate the successful conversion



3. Document Download:

- On clicking on Download button the Docx starts downloading.
- The /download route allows users to download the generated DOCX file.



A top-down view of a workspace featuring a silver laptop with a black keyboard, an orange spiral notebook, and a white cup of coffee on a light-colored wooden desk. The image is overlaid with a large yellow shape on the left and green shapes in the corners, containing white dotted patterns.

THANKYOU

By: Data Dynamo