CS 5170: AI-HCI

#4 Mini Class Project

Name: Eshan Wadhwa

Project Idea: HueShift

GitHub Repo: https://github.com/EshanW313/AI-HCI-hw4

Video in GitHub Repo

Idea Justification and Target Population:

Color vision deficiency (CVD) poses a significant challenge to nearly 200 million people (2009) worldwide, often hindering their ability to fully engage with visual information and perform tasks that rely on color perception [3]. Approximately 1 in 12 men and 1 in 200 women have color-blindness [5]. I read about color blindness from [1] and thought of HueShift. It aims to alleviate this challenge by dynamically adjusting website colors, enhancing accessibility and usability for individuals with CVD.

HueShift focuses on being an AI-powered Chrome extension that dynamically adjusts website colors to enhance accessibility for individuals with CVD. This design choice stems from a deep understanding of the challenges faced by this user group and a commitment to leverage AI to create a more inclusive online experience.

To enhance user privacy, the AI model operates solely on the CSS data extracted from the webpage. It does not receive or process any of the website's content, including text, images, or sensitive information. This ensures that user data remains protected and that the AI focuses exclusively on color adjustments without accessing potentially private content. This approach aligns with privacy-preserving AI principles and prioritizes user confidentiality.

The primary target users for this tool are individuals with various types and severities of color blindness. These are the types of color-blindness [2]:

- Deuteranomaly: certain shades of green look redder [2]
- Protanomaly: certain shades of red look greener and dull [2]
- Protanopia and Deuteranopia: can’t differentiate between red and green [2]
- Tritanomaly: can’t differentiate between 1. blue and green, and 2. yellow and red [2]
- Tritanopia: unable to differentiate between 1. blue and green, 2. purple and red, and 3. yellow and pink [2]

- Monochromacy/Achromatopsia: can’t see colors at all (complete color vision deficiency) [2]

People with color-blindness experience varying degrees of color deficiency. So, HueShift acknowledges that and varies different levels of color-blindness within each type. Users can enter the strength of intensity/contrast they want (between 0 and 1). It also has a feedback mechanism where users can enter their choice of color palette in an open textbox. The AI model takes the free text into consideration and decides the color theme accordingly. If the free text contradicts the type of color-blindness selected, the model returns a message explaining the contradiction.

For instance, if a user selects “Protanopia” and enters “preferably red and green” in the textbox, the AI model should not change the color palette of the webpage and return something like “This combo does not match!”

Additionally, a user can also click on the “?” button placed on the top right to read and understand about color-blindness. The button redirects the user to [1].

Gaps in Existing Solutions:

I was not able to find even a few relevant Chrome Extensions in the Chrome Web Store that could assist people with color-blindness. There is a Chrome extension [4] that addresses this issue but has limited customization options. Current color blindness accessibility tools may have limitations:

- Limited Customization: Offers a limited set of pre-defined color transformations, restricting user control and personalization.
- Generic Adjustments: Applies generic color filters that may not be optimal for all types and severities of color blindness.

Possible Harms and Biases, and Addressing them:

AI can offer great potential for accessibility, it's essential to consider potential harms and biases, like:

- Inaccurate Adjustments: AI model might misinterpret certain color combinations, leading to inaccurate or ineffective adjustments

- Reinforcement of Stereotypes: The model might inadvertently favor certain color palettes or aesthetics, potentially reinforcing stereotypes or excluding certain user groups.

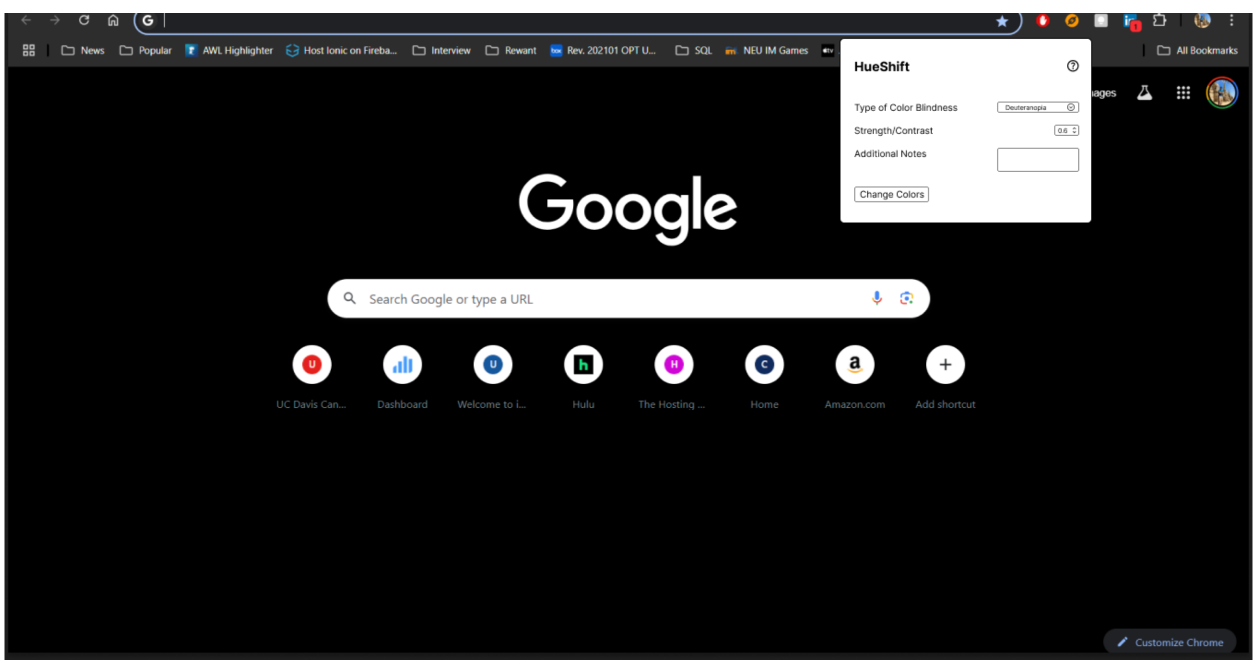
Mitigating these risks:

- User Feedback Mechanisms: User feedback is encouraged to identify and correct inaccurate adjustments or biases.

Mockup (created using Figma):

Image 1: UI of Chrome Extension

Image 2: Design of the Chrome Extension (Browser View)



Design Justification: The minimalistic black font and white background design is for clarity and is accessible for all users, regardless of their color vision abilities. The high-contrast color scheme is readable and aligns with universal design principles, making it easier for people with varying visual abilities to understand and interact with this chrome extension.

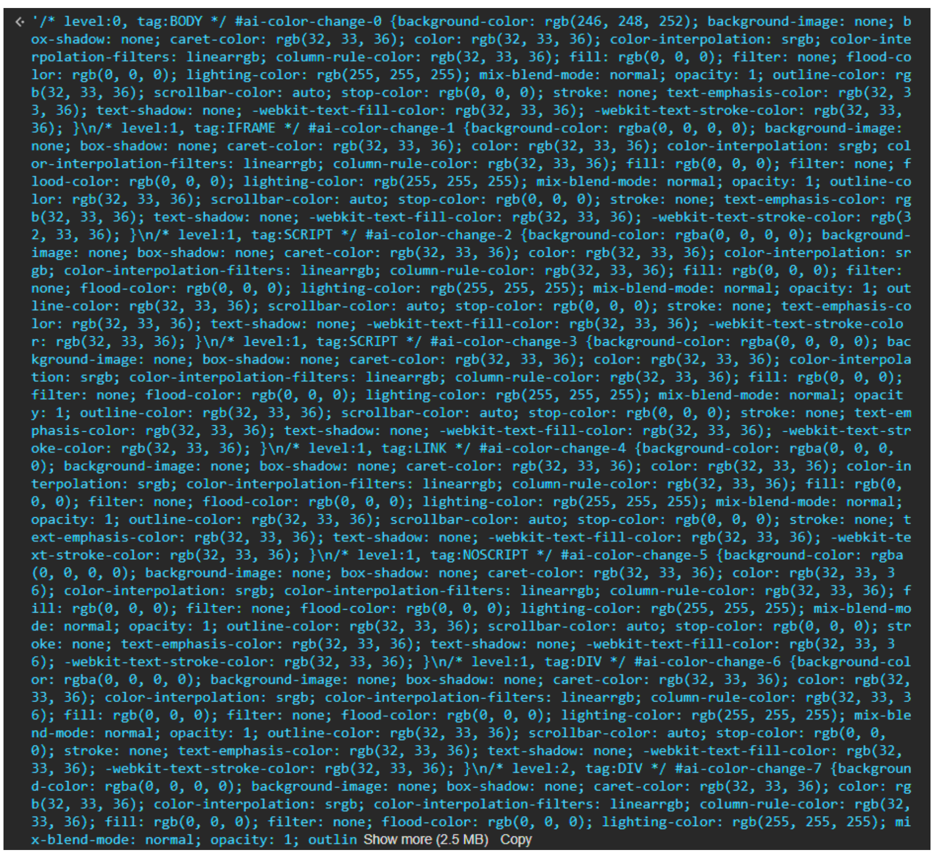
This design adheres to many Web Content Accessibility Guidelines (WCAG) 2.1 [6] sections such as:

- Success Criterion 1.4.1 Use of Color
- Success Criterion 1.4.3 Contrast (Minimum)

Important: Challenges Faced during Development

While significant progress has been made in designing and conceptualizing HueShift, the coding part has presented some unforeseen challenges. Specifically, the volume of HTML and CSS data on modern webpages has proven to be a significant hurdle. Many websites, particularly those with complex layouts or dynamic content, generate HTML and CSS files that exceed several megabytes in size, sometimes containing millions of characters.

Example: this is the output I got from https://mail.google.com/mail/u/0/#inbox




```
'/* level:0, tag:BODY */ #ai-color-change-0 {background-color: rgb(246, 248, 252); background-image: none; b
ox-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32, 33, 36); color-interpolation: srgb; color-inte
rpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 36); fill: rgb(0, 0, 0); filter: none; flood-co
lor: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-blend-mode: normal; opacity: 1; outline-color: rg
b(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0, 0); stroke: none; text-emphasis-color: rgb(32, 3
3, 36); text-shadow: none; -webkit-text-fill-color: rgb(32, 33, 36); -webkit-text-stroke-color: rgb(32, 33,
36); }\n/* level:1, tag:IFRAME */ #ai-color-change-1 {background-color: rgba(0, 0, 0, 0); background-image:
none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32, 33, 36); color-interpolation: srgb; col
or-interpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 36); fill: rgb(0, 0, 0); filter: none; f
lood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-blend-mode: normal; opacity: 1; outline-co
lor: rgb(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0, 0); stroke: none; text-emphasis-color: rg
b(32, 33, 36); text-shadow: none; -webkit-text-fill-color: rgb(32, 33, 36); -webkit-text-stroke-color: rgb(3
2, 33, 36); }\n/* level:1, tag:SCRIPT */ #ai-color-change-2 {background-color: rgba(0, 0, 0, 0); background-
image: none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32, 33, 36); color-interpolation: sr
gb; color-interpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 36); fill: rgb(0, 0, 0); filter:
none; flood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-blend-mode: normal; opacity: 1; out
line-color: rgb(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0, 0); stroke: none; text-emphasis-co
lor: rgb(32, 33, 36); text-shadow: none; -webkit-text-fill-color: rgb(32, 33, 36); -webkit-text-stroke-colo
r: rgb(32, 33, 36); }\n/* level:1, tag:SCRIPT */ #ai-color-change-3 {background-color: rgba(0, 0, 0, 0); bac
kground-image: none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32, 33, 36); color-interpola
tion: srgb; color-interpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 36); fill: rgb(0, 0, 0);
filter: none; flood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-blend-mode: normal; opacit
y: 1; outline-color: rgb(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0, 0); stroke: none; text-em
phasis-color: rgb(32, 33, 36); text-shadow: none; -webkit-text-fill-color: rgb(32, 33, 36); -webkit-text-str
oke-color: rgb(32, 33, 36); }\n/* level:1, tag:LINK */ #ai-color-change-4 {background-color: rgba(0, 0, 0,
0); background-image: none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32, 33, 36); color-in
terpolation: srgb; color-interpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 36); fill: rgb(0,
0, 0); filter: none; flood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-blend-mode: normal;
opacity: 1; outline-color: rgb(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0, 0); stroke: none; t
ext-emphasis-color: rgb(32, 33, 36); text-shadow: none; -webkit-text-fill-color: rgb(32, 33, 36); -webkit-te
xt-stroke-color: rgb(32, 33, 36); }\n/* level:1, tag:NOSCRIPT */ #ai-color-change-5 {background-color: rgba
(0, 0, 0, 0); background-image: none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32, 33, 3
6); color-interpolation: srgb; color-interpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 36); f
ill: rgb(0, 0, 0); filter: none; flood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-blend-mo
de: normal; opacity: 1; outline-color: rgb(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0, 0); str
oke: none; text-emphasis-color: rgb(32, 33, 36); text-shadow: none; -webkit-text-fill-color: rgb(32, 33, 3
6); -webkit-text-stroke-color: rgb(32, 33, 36); }\n/* level:1, tag:DIV */ #ai-color-change-6 {background-col
or: rgba(0, 0, 0, 0); background-image: none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rgb(32,
33, 36); color-interpolation: srgb; color-interpolation-filters: linearrgb; column-rule-color: rgb(32, 33, 3
6); fill: rgb(0, 0, 0); filter: none; flood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mix-ble
nd-mode: normal; opacity: 1; outline-color: rgb(32, 33, 36); scrollbar-color: auto; stop-color: rgb(0, 0,
0); stroke: none; text-emphasis-color: rgb(32, 33, 36); text-shadow: none; -webkit-text-fill-color: rgb(32,
33, 36); -webkit-text-stroke-color: rgb(32, 33, 36); }\n/* level:2, tag:DIV */ #ai-color-change-7 {backgroun
d-color: rgba(0, 0, 0, 0); background-image: none; box-shadow: none; caret-color: rgb(32, 33, 36); color: rg
b(32, 33, 36); color-interpolation: srgb; color-interpolation-filters: linearrgb; column-rule-color: rgb(32,
33, 36); fill: rgb(0, 0, 0); filter: none; flood-color: rgb(0, 0, 0); lighting-color: rgb(255, 255, 255); mi
x-blend-mode: normal; opacity: 1; outlin Show more (2.5 MB)   Copy
```


As the image shows, the size of the total output I received is approximately 2.5 MB.

This output is just a small section of the CSS that contains very few color-related properties of only 10 types of HTML tags:

main, nav, article, footer, header, div, p, span, a, img

Some of the challenges faced are:

- API Limitations: LLMs, while powerful, have limitations on the amount of data they can handle in a single request (input and output) and sending multiple requests is very inefficient (both cost and browser performance). Example: Google Gemini 1.5 Flash can take 1,048,576 tokens and the output token limit is 8,192 [7]
- Performance Bottlenecks: Sending and receiving large amounts of data to the LLM API led to performance delays in the browser and to the webpage.

I am sure that there is a more efficient way to solve this problem like having more robust traversing and parsing algorithm to get the more relevant CSS tags and further reducing the data size, however, I was not able to do it due to time constraint. I have added my code to the GitHub repository. The algorithm I wrote was quite simple:

- It traverses through the webpage's DOM and looks for HTML tags that has specific color properties like "color", "background-color" and "border-color".
- It stores the values of the CSS properties and maps it to the HTML tags and its level in the DOM for the AI model to understand (example: in the above image, the starting characters say "level 0, tag:BODY").
- It also adds custom IDs (eg: "ai-color-change-0" in the image) to those HTML tags so that it is easy for the new CSS (returned by the AI Model) to get mapped and reflect the changes.

Despite these challenges, the core concept of HueShift remains promising. By addressing these data handling issues, the project can move forward towards its goal of providing a valuable accessibility tool for individuals with color blindness.

I genuinely believed my approach could work and it could be a very decent product. So, I kept trying to optimize my algorithm, and hence could not think of any other solution for this assignment.

References:

1. https://www.nei.nih.gov/learn-about-eye-health/eye-conditions-and-diseases/color-blindness
2. https://www.nei.nih.gov/learn-about-eye-health/eye-conditions-and-diseases/color-blindness/types-color-vision-deficiency

3. G. M. Machado, M. M. Oliveira and L. A. F. Fernandes, "A Physiologically-based Model for Simulation of Color Vision Deficiency," in IEEE Transactions on Visualization and Computer Graphics, vol. 15, no. 6, pp. 1291-1298, Nov.-Dec. 2009, doi: 10.1109/TVCG.2009.113.
4. https://chromewebstore.google.com/detail/lets-get-color-blind/bkdgdianpkfahpkmphgehigalpighjck
5. https://www.colourblindawareness.org/colour-blindness/
6. https://www.w3.org/TR/WCAG21/
7. https://ai.google.dev/gemini-api/docs/models/gemini