AccessCheck

Purva Santosh Dusane
HCI 5840 – Fall 2025

General description of the project (2 pts)

The application will evaluate any public website for a limited number of fundamental accessibility concerns and present the results in a straightforward, beginner-friendly report. It is designed for students, new web designers, and small-business owners who want to enhance accessibility but find technical tools daunting.

The initial version will operate as a command-line interface (CLI): the user inputs a URL and receives a textual summary of the outcomes. A future version will feature a simple web dashboard built with Flask, containing color-coded cards and the option to export results as PDF or CSV files.

The program will retrieve the HTML of the specified page, assess a few essential criteria such as missing alt text for images, incorrect heading hierarchy, and inadequate color contrast, and deliver pass/warn/fail indicators along with brief explanations.

Task Vignettes (User activity "flow") (4 pts)

1) Quick Site Check
Maya, a design student, wants to review her class project site.
She opens the terminal and types:

```
python accesscheck.py https://mayaportfolio.com
```

In a few seconds, the program prints:

- Alt Text – Two images are missing descriptions
- Headings – Good order
- Color Contrast – Three text areas with low contrast

Technical notes

- Input: URL typed in the command line
- The program downloads the page and runs each check
- Returns a simple text report

2) Re-testing After Fixes
Maya adds descriptions to her images and runs the command again.
This time, everything is green except for the color contrast.
She repeats the process until all checks pass.

Technical notes

- The user can run the same command many times
- The program can save each report in a small CSV file to track changes

3) Web Dashboard (future)
Later, Maya opens a small local website made with Flask.
She pastes her URL into a box and clicks Analyze.
The results appear as cards with green, yellow, or red icons and a button to download a PDF.

Technical notes

- Same checking functions as the command line version
- Flask handles the input form and the results page

Here is a straightforward explanation of the events that will take place within AccessCheck once a user submits a website for evaluation. This is not programming code just a general overview of the information flow and key stages.

Here's a simple breakdown of what happens in AccessCheck when someone submits a website for evaluation. This isn't code, just an easy-to-understand overview of the steps involved.

Technical "flow" (3 pts)

Overall data flow
- User Input: The user types or pastes a website URL.
- Page Retrieval: The app reaches out to that URL to fetch the page content.
- Content Reading: The app looks over the page to analyze its layout, including text, images, headings, and colors.

Accessibility Checks:
- It checks if images are missing alt text.
- It makes sure headings (like H1, H2, etc.) are in a logical order.
- It assesses whether the text color stands out enough against the background for easy reading.

- Result Collection: All the findings get summed up in a brief summary showing what worked, what needs changes, and what didn't pass**.**

Output to User:

- In the first version, this will be shown as a text report in the command line.
- Later on, it might be displayed on a simple webpage with color-coded sections.

Main blocks

- Input Block: where the website URL goes.
- Fetch Block: grabs the raw page content.
- Analysis Blocks: run through the three checks mentioned earlier.
- Report Block: puts together the final summary for the user.

Interaction points

The only thing the user needs to do is provide a website URL. The program takes care of delivering the report, either as text on screen or, in a future version, as a webpage.

This gives a clear picture of how the system works, making it easy for anyone to understand before diving into any coding.